IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Mathematical Modelling of Engineering Problems
Vol. 12, No. 6, June, 2025, pp. 1893-1910
Journal homepage: http://iieta.org/journals/mmep

Improving Voice Recognition in Iraqi Dialect Using Deep CNN-LSTM Architecture: A Comprehensive Deep Learning Approach

Mithal Khaleel Ismael[1*], Goh Chin Hock[1], Hazem Noori Abdulrazzak[2]

[1] Institute of Power Engineering, Universiti Tenaga Nasional, Kajang 43000, Malaysia
[2] Computer Communication Engineering Department, Al-Rafidain University College, Baghdad 10014, Iraq

Corresponding Author Email: pe21029@student.uniten.edu.my

https://doi.org/10.18280/mmep.120607

Received: 15 February 2025
Revised: 3 May 2025
Accepted: 12 May 2025
Available online: 30 June 2025

Keywords:
voice recognition, Iraqi dialects, CNN, LSTM, data augmentation

ABSTRACT

Arabic dialect is the variety of Arabic used in daily communication in the Arab world. Each Arab country has its own dialect. Due to the difficulty of recognizing spoken Arabic dialects, such as the large variations in Arabic dialects and the lack of a standard structure, few research attempts in the scientific literature have addressed this problem. The comprehensive deep learning approach offers a promising approach to improving the performance of speech recognition systems. However, overfitting remains a significant problem in light of limited data. In this paper, we propose a deep learning-based framework for improved Iraqi dialect voice recognition (DIVR) using a hybrid model that combines convolutional neural networks (CNNs) and long-term recurrent networks (LSTMs), supported by an enhanced attention mechanism and a series of data augmentation techniques geared toward rare dialect variations. Careful preprocessing was necessary to remove noise and standardize the audio input, while the integrated attention mechanism helped the model dynamically focus on the most important spatiotemporal features, contributing to improved performance and overall interpretability. The proposed model relied on the use of four CNN-based model architectures: Base CNN, CNN-LSTM, Base CNN-LSTM, and Deep CNN-LSTM. Among the tested models, the Deep CNN-LSTM architecture outperformed with a test accuracy of 98.43%. Based on training efficiency, generalization ability, and classification performance across 22 different classes, the model demonstrated a significant improvement in performance.

1. INTRODUCTION

Automatic speech recognition (ASR) is one of the most important innovations in the field of artificial intelligence and human-machine interaction. With the continuous development of audio signal processing and deep learning technologies, this technology has become an essential part of modern operating systems and smart devices. Many applications, such as intelligent personal assistants (Siri/Google Assistant) and voice control systems in cars and home appliances, rely on the effectiveness of voice recognition to understand voice commands issued by users, and it can also help people with disabilities and the elderly interact with society with ease [1, 2]. Despite the great progress in ASR systems when dealing with Modern Standard Arabic or some dialects supported by sufficient data, there are still significant challenges when trying to recognize less common or more diverse dialects. These dialects suffer from the poor performance of current models due to the scarcity of training data and the multiplicity of phonetic patterns in them [3]. Among these dialects, the Iraqi dialect stands out as one of the most complex Arabic dialects, as it is characterized by its wide diversity between different regions such as the Baghdad, Mosul, Basra, and Southern dialects [4]. This great diversity poses a challenge to speech recognition systems, as models face difficulty in accommodating differences in pronunciation and vocabulary, which affects the accuracy of performance and limits the actual use of these techniques [5]. ASR systems rely on huge amounts of audio data and annotated texts to train models and improve their performance. However, Iraqi dialects lack extensive and documented datasets, which limits the ability of models to learn and adapt to different phonetic variations [6]. In recent years, deep learning has emerged as an effective approach to improving speech recognition, allowing models to learn from real-time data and improve their decisions based on the extracted acoustic features [7]. Among the most prominent approaches used, convolutional neural networks (CNNs) have proven effective in extracting acoustic features and improving recognition accuracy, especially when sufficient training data is available [8]. However, data size is not the only critical factor; data quality and distribution play a fundamental role in achieving optimal performance. If the data is not sufficiently representative of the target dialects or suffers from poor structure, the model's performance may remain substandard, regardless of its size [9]. Moreover, increasing data size poses challenges such as higher computational costs and increased resource consumption, which necessitates striking a balance between data size and computational efficiency. In the context of Arabic dialects, which require a thorough understanding of linguistic, social, and cultural characteristics, improving speech recognition systems must go beyond simply increasing the data size. It is also necessary to improve data quality and enhance the algorithms used to ensure higher accuracy and reliability [10, 11]. Given the pressing need to address the unique acoustic variability of the Iraqi dialect, this study demonstrates its significance by developing a deep learning model tailored to the specific phonetic characteristics of this dialect. The proposed approach incorporates specialized data preprocessing and augmentation techniques, setting it apart from previous work that primarily targeted more uniform speech patterns. The study aims to enhance speech recognition accuracy for the Iraqi dialect through the implementation of a CNN-LSTM model augmented with an attention mechanism. This model effectively combines spatial and temporal learning of acoustic features and benefits from meticulous preprocessing steps that help mitigate bias and improve generalization. As such, this research represents a pivotal step toward building more accurate and robust intelligent systems capable of handling the complexity of non-standard spoken Arabic varieties.

The main objectives of this research can be summarized as follows:

1. Analyse voice recognition features using Iraqi dialect datasets from different geographical regions, with the aim of evaluating their efficiency compared to traditional systems and identifying the factors affecting performance.
2. Compare the proposed Deep CNN-LSTM model with other models such as Base CNN and CNN-LSTM, where the proposed model highlights its superiority by effectively integrating spatial and temporal learning of audio features. It also enhances performance by incorporating attention mechanisms, which help focus on the most important features, leading to improved model accuracy.
3. Data preprocessing: The study emphasized the importance of improving the quality of data and not just increasing it, as preprocessing contributed to reducing bias and enhancing the generalization of the model.
4. Evaluate the performance of the classified models: using standard metrics including Accuracy, Recall, F1 score, and Precision, to ensure a comprehensive and objective analysis of performance.

The paper is organized as follows. Literature Review section, Methodology section which focused on the details of the dataset as well as the general structure of the model, Results section, and Conclusion section.

2. RELATED WORKS

ASR has become an essential component of virtual assistants, enabling systems to analyze human speech and identify specific words within a dialogue. This capability is widely applied in various fields, including voice-activated devices, executing commands in video games, and interacting with Internet of Things (IoT) devices. Over the past five decades, ASR has emerged as an important area of scientific research, facilitating communication between individuals and enhancing interactions between humans and machines [10]. ASR systems use voice interfaces to detect keywords through single-word recognition, making them highly suitable for portable and embedded devices. Initially, the development of ASR relied heavily on machine learning techniques, such as hidden Markov models (HMMs) and Gaussian mixture models (GMM-HMMs) [11]. However, these approaches have faced limitations in dealing with modern requirements, prompting researchers to adopt advanced techniques such as deep learning. Deep learning algorithms, including CNNs, have shown remarkable potential due to their multi-layered architecture, which enables them to handle complex tasks with high accuracy. CNNs have been successfully implemented in various fields, such as image and video recognition, text processing, and speech [12, 13]. Prominent technology companies, including Google, Facebook, Microsoft, and IBM, have taken advantage of these developments to improve their software applications [14]. A study on speech recognition, especially speaker-independent speech, using a dataset of 50 speakers demonstrated its effectiveness using dense and convolutional neural network (DNN and CNN) algorithms, achieving average recognition accuracy of 75% to 80% [15]. However, another study found that higher accuracy was achieved by increasing the amount of data by combining CNN algorithms with a recurrent neural network (RNN), which proved effective in classifying voice commands such as "yes", "no", "up", "down", "left", "right", "stop", and "go". Using a dataset of 65,000 audio files developed by Google's TensorFlow and AIY teams, the model achieved an impressive classification accuracy of 96.66% [16]. Since a large amount of audio data is needed to train convolutional neural networks, data augmentation methods have been considered to increase the sizes of audio datasets [17]. In order to train CNNs to achieve greater accuracy, methods to increase the sizes of audio datasets are required [18]. This research demonstrated the efficiency of combining Mel-Frequency Cepstral Coefficient (MFCC) for feature extraction, with a CNN model for learning and classifying additional features using data from Google Speech to recognize voice commands containing 65,000 1-second audio recordings with a test accuracy of 94.8% [19]. To improve the accuracy of isolated speech recognition by extracting audio features through spectrograms, the research used CNN-based models such as Alex Net and Google Net, and due to overfitting, the training data, the system failed to achieve the desired accuracy of 72% and 66%, respectively [20]. Another study used CNN with MFCC features to recognize isolated words recorded in noisy environments. Although the system outperformed single-word recognition, it faced challenges with multi-syllabic words [21]. Although data augmentation has a direct impact on the accuracy of the model, as it has been shown to be effective in data augmentation for visual tasks, unlike noisy audio samples, the proposed Attention RNN with Keras untrainable layer has been shown to capture both short- and long-term dependencies, achieving an accuracy rate of 94.5% [22]. Additionally, the methodology of transforming 1D audio samples into 2D representations using MFCCs yielded significant results. Three-layer CNN The architecture model applied to the TIDIGITS dataset achieved a classification accuracy of 97.46%, proving the effectiveness of this approach in speech recognition [23]. Another study combining HMM and GMM algorithms focused on discrete speech recognition in the Amazigh language. Using a 43-word dataset and the HTK toolkit, the system achieved an accuracy of 91.31% [24]. The combination of traditional HMMs and neural networks also showed promise in improving speech recognition performance using data from Google and Pocket Sphinx. The researchers achieved a recognition accuracy of 79.2% and observed an additional improvement of 84.4% when applying the model to the TIMIT dataset. These results underscore the importance of dataset selection in improving model performance [25]. To address the challenge of noise in audio signal processing, the researchers combined feature extraction techniques, including discrete cosine transform (DCT) and MFCC features, in a framework that combines bidirectional long short-term memory (BiLSTM), CNN, and traditional HMM classifiers. This approach has significantly improved recognition accuracy and reduced loss in both noisy and clean environments [26].

Speech recognition systems for Arabic dialects present unique challenges due to the lack of standardized spelling and pronunciation rules. This linguistic complexity requires dedicated research to improve the performance of speech recognition systems for Arabic, taking into account the characteristics of diverse dialects [27-29]. One study developed an Arabic speech recognition system using a dataset of Arabic numerals spoken by ten Tunisian dialect speakers. Acoustic feature extraction techniques combined with a feed-forward backpropagation neural network (FFBPNN) yielded a performance rate of 98.54%. Additionally, the Linde-Bozo-Gray (LBG) vector classification technique outperformed principal component analysis (PCA) in both accuracy and computational efficiency [30]. Another study proposed an isolated speech recognition system for the Yemeni dialect, using a support vector machine (SVM) algorithm for audio classification. Features such as MFCC, perceptual model-inspired audio spectral coefficients (PNCC), and modified gradient frequency derivatives (ModGDF) were used for feature extraction. PNCC with SVM achieved the best performance, with an accuracy rate of 93.9% [31]. Similarly, a Moroccan dialect search using HMM with a dataset of 20 isolated words achieved a recognition accuracy of 90% by combining MFCC and its derivatives (delta and delta-delta) [32]. Finally, the combination of spectrograms and CNNs was proven to be effective in distinguishing Sundanese dialects. Using a dataset containing 50 audio samples per dialect and a 32-32-64-64 filter configuration, the model achieved a test accuracy of 95%, contributing to a better understanding of the differences in Sundanese language and reducing miscommunication between ethnic groups in Indonesia [33]. In this study, the effect of data size on the test accuracy in recognizing different dialects was investigated using convolutional neural network using 12 different vocal commands, where the test accuracy rate for a large dataset was 94.64%, while the accuracy for a small dataset was 64.81% [34].

3. METHODOLOGY

The methodology for this study involved regularizing audio files corresponding to 22 predefined labels, ensuring balanced representation across all classes. Data preprocessing included augmentation techniques such as time stretching, pitch transformation, and noise addition to enhance model robustness, followed by feature extraction using Mel spectrograms. Labels were encoded with a one-hot encoding to align with the loss function, and the dataset was split into 95% training and 5% testing sets. Four CNN-based model architectures were developed: a basic CNN, a regularized CNN with dropout and regularization, a deep-LSTM CNN combining convolutional and LSTM layers, and a wide-LSTM CNN designed to capture complex patterns. Models were trained using the Adam optimizer with a validation loss checkpoint. This methodology provided a robust framework for effectively classifying audio data. Figure 1 illustrates the methodology flowchart.

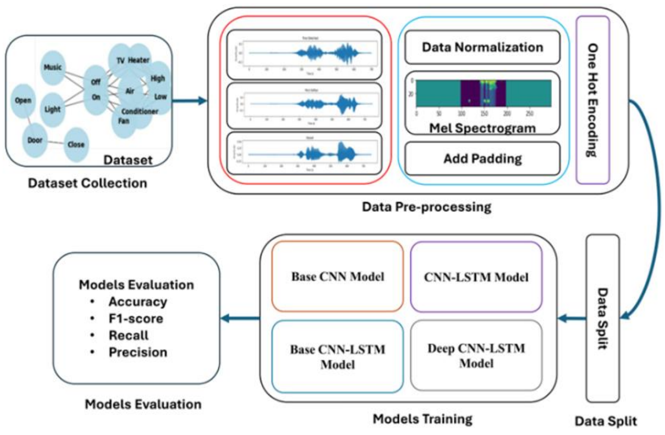


Figure 1. Flow diagram of proposed methodology

Table 1. Details of classes in dataset

No.	Class Name in Iraqi Dialect	Class Name in English	Number of Samples
1	سد الباب	Close Door	268
2	علي المكيف	High Air Conditioner	270
3	علي المروحة	High Fan	261
4	علي التدفئة	High Heater	292
5	علي التلفزيون	High TV	284
6	نصي المكيف	Low Air Conditioner	267
7	نصي المروحة	Low Fan	290
8	نصي التدفئة	Low Heater	267
9	نصي التلفزيون	Low TV	260
10	طفي المكيف	Off Air Conditioner	272
11	طفي المروحة	Off Fan	277
12	طفي التدفئة	Off Heater	296
13	طفي الاضاءة	Off Light	244
14	طفي الموسيقى	Off Music	244
15	طفي التلفزيون	Off TV	244
16	شغل المكيف	On Air Conditioner	289
17	شغل المروحة	On Fan	286
18	شغل التدفئة	On Heater	264
19	شغل الإضاءة	On Light	255
20	شغل الموسيقى	On Music	241
21	شغل التلفزيون	On TV	273
22	افتح الباب	Open Door	265

3.1 Data collection

The dataset for this study consists of audio recordings of speech commands, collected from various regions in Iraq to include a diversity of dialects, age, and gender. The sample included speakers with a variety of dialects, including Mosuli, Baghdadi, Southern, and Central. The recordings were obtained from four main locations: streets, universities, social media, and shopping malls, ensuring a wide range of linguistic characteristics were captured. Before each recording was included in the dataset, the sample underwent a rigorous quality control process, which included manual verification of comments and audio purity checks to exclude poor recordings and ensure data quality. These procedures helped reduce bias and maintain a balanced representation of the 22 pre-defined categories (as shown in Table 1), which represent distinct states or actions. All audio files were stored in WAV format, sampled at 92,800 kHz to ensure data accuracy and quality. The files are also organized into separate directories according to their categories, which facilitates preprocessing and efficient model training, while ensuring a balanced distribution of all categories, as shown in Figure 2.

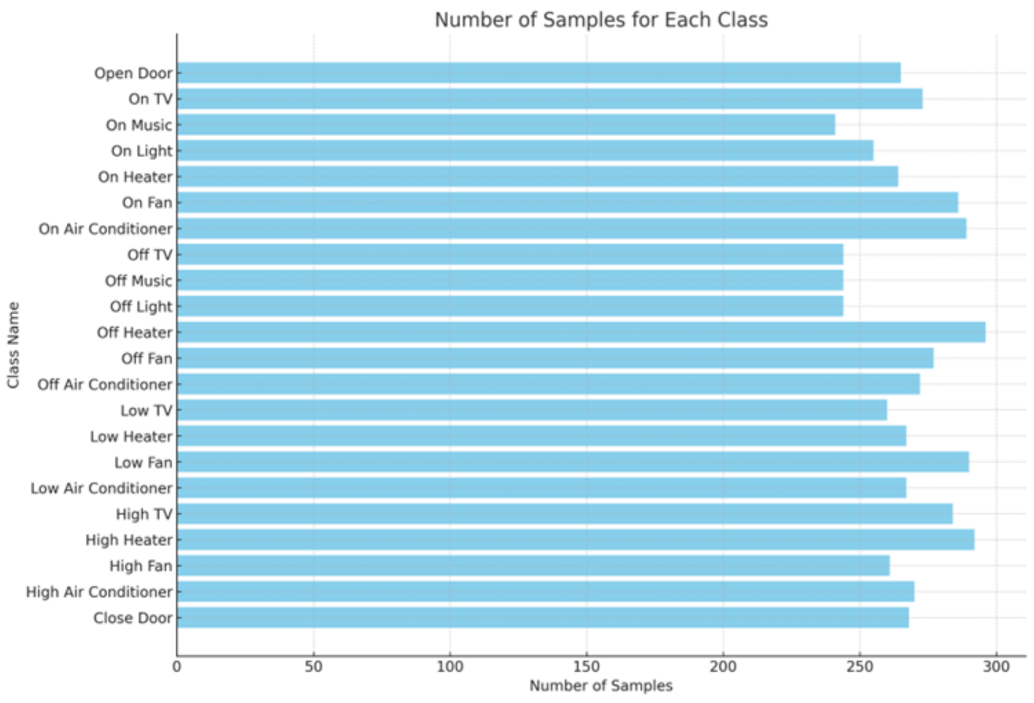


Figure 2. Details of samples of dataset for each class

3.2 Data preprocessing

3.2.1 Data augmentation

To expand the dataset and improve model robustness, three augmentation techniques were applied to the raw audio data [35]:

(1) Time stretching

Playback speed was adjusted using rates of 0.81 (slower) and 1.07 (faster) without altering the pitch as shown in Figure 3. This was implemented using Librosa's `time_stretch` function, ensuring pitch invariance.

(2) Pitch shifting

To expand the audio dataset and improve the generalization of speech recognition models, the pitch of the audio signals was shifted by -2, -1, +1, and +2 semitones using the `pitch_shift` function in Librosa as shown in Figure 4. This modification mimics the natural differences between speakers at different pitches without losing speech clarity, helping the model focus on the intrinsic features of the audio signal and reduce overlearning

(3) Noise addition

Gaussian noise was added to simulate environmental disturbances. Random samples of noise amplitudes between 0.005 and 0.008 were taken, as shown in Figure 5. This step was implemented using custom Python functions with NumPy to generate random noise. This is based on previous studies demonstrating that these parameters preserve speech characteristics while providing realistic contrast [36, 37].

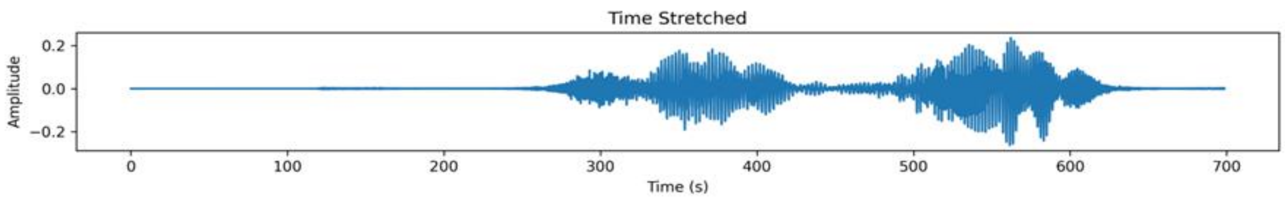


Figure 3. Sample of time stretched

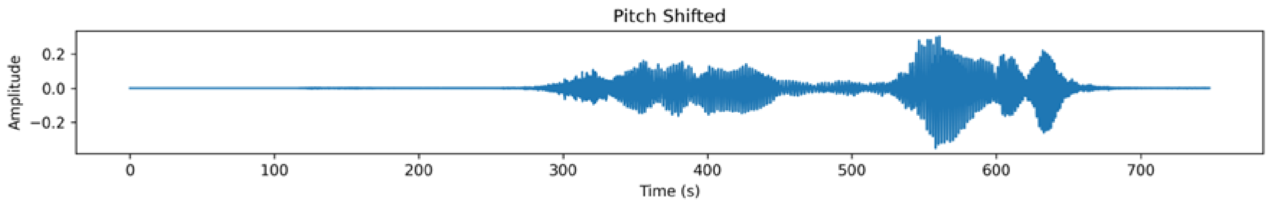


Figure 4. Sample of pitch shifted

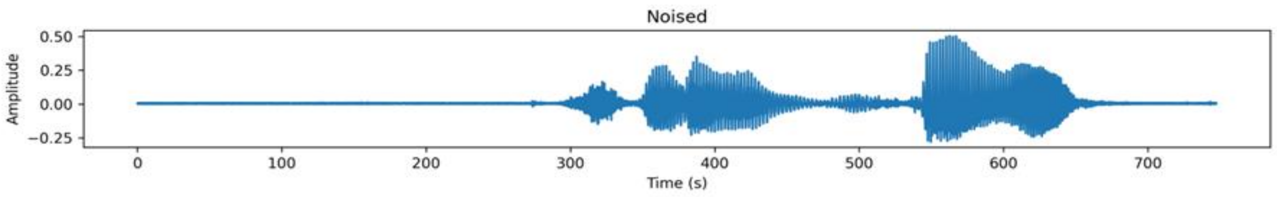


Figure 5. Sample of added noise

3.3 Mel spectrogram calculation

Mel spectrograms were generated to convert raw audio signals into a suitable format for deep learning models. This involved several steps:

3.3.1 Loading and normalization

Audio files were loaded using Librosa, with a sampling rate of 22,050 Hz. Signals were normalized to a range of -1 to 1 for consistent amplitude scaling.

3.3.2 Spectrogram generation

Mel spectrograms were calculated using specific parameters to extract detailed frequency information from the audio signal. The number of Mel bands was set to 40, which determines the resolution of the frequency axis in the spectrogram. The Fast Fourier Transform (FFT) size was chosen to be 2048, ensuring sufficient frequency resolution for analyzing the signal's spectral content. Additionally, the hop length was set to 512, which refers to the step size between successive frames of the signal, affecting the time resolution of the spectrogram. These parameters were selected to balance the trade-off between time and frequency resolution, allowing for effective analysis of the audio. The spectrograms were converted to the decibel scale using Librosa's `amplitude_to_db` function making the features suitable for machine learning.

(1) Framing

The signal is divided into frames of 2048 samples with a hop length of 512 samples. At a sampling rate of 22,050 Hz, each frame is approximately 93 milliseconds, with a 50% overlap. The hop length of 512 samples equates to 23 milliseconds. This overlap ensures shared information between frames, preserving temporal dependencies and minimizing boundary detail loss.

(2) Windowing

A Hanning window is applied to each frame to smooth edges and reduce spectral leakage caused by framing. The window function (1) is defined as:

$$\omega(n) = 0.5\left(1 - \cos\left(\frac{2\pi n}{N-1}\right)\right) \tag{1}$$

Here, $\omega(n)$ is the window function, and N is the number of samples in a frame. This bell-shaped curve tapers signal edges, improving frequency representation for subsequent FFT processing.

(3) FFT

FFT converts each frame into the frequency domain, extracting the magnitude spectrum. The power spectrum of the m-th frame is given by Eq. (2):

$$P_m(f) = |\mathrm{FFT}(x_m)|^2 \tag{2}$$

where, $P_m(f)$ represents the power spectrum of the m-th frame at frequency f.

(4) Mel filter bank

The Mel filter bank simulates human auditory perception by mapping the power spectrum to the Mel scale. It applies a matrix, H_{mel}, to transform the power spectrum into the Mel spectrogram as shown in Eq. (3):

$$M_m = H_{mel} \cdot P_m \tag{3}$$

Here, M_m is the Mel spectrogram, H_{mel} is the filter bank, and P_m is the power spectrum.

(5) Logarithmic compression

Logarithmic compression mimics the human ear's perception of loudness by applying.

Where L_m is the log-compressed Mel spectrogram, and ϵ (e.g., $1{\times}10^{-6}$) prevents taking the log of zero. Compression reduces dynamic range and enhances feature representation as shown in Eq (4):

$$L_m = \log(M_m + \epsilon) \tag{4}$$

(6) DCT

DCT decorrelates features and compresses information, yielding MFCCs. It is expressed as shown in Eq (5):

$$c_n = \sum_{m=0}^{M-1} \log(L_m)\, \cos\left[\frac{\pi n}{M}\left(m + \frac{1}{2}\right)\right] \tag{5}$$

where, c_n are MFCCs, n is the coefficient index, and M is the number of Mel filters. After DCT, MFCC features are obtained as shown in Figure 6.

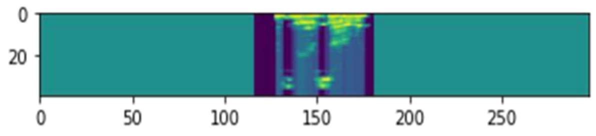


Figure 6. Mel spectrogram of time series signal

3.3.3 Padding

To maintain consistency in input dimensions across all samples, spectrograms with fewer frames than the maximum sequence length were zero-padded. Padding ensures that all spectrograms within the dataset have uniform dimensions, preventing shape mismatches during model training. Without padding, models would struggle to process variable-length inputs, leading to inefficiencies in batch processing. The zero-padding technique was applied at the end of shorter spectrograms, preserving the temporal integrity of existing frames while ensuring compatibility with deep learning frameworks. This method allows the model to handle sequences of different lengths while maintaining computational efficiency.

3.4 Label encoding

To make the class labels suitable for training a deep learning model, integer encoding was first applied. Each unique label in the dataset was mapped to a corresponding integer identifier using a predefined dictionary (e.g., 'on_tv' → 0, 'off_tv' → 1, etc.). However, directly using integer labels in classification tasks can lead to unintended ordinal relationships between classes, which may negatively affect model learning. To address this, one-hot encoding was employed, transforming each integer label into a binary vector representation. In this encoding scheme, each class was represented as a vector where only one position was marked as 1 (active), while all others were 0. For example, if there were 10 total classes, the 'on_tv' label (mapped to 0) would be converted into [1, 0, 0, 0, 0, 0, 0, 0, 0, 0]. This categorical representation ensures that the deep learning model, which utilizes the categorical cross-entropy loss function, can effectively differentiate between different class labels without misinterpreting ordinal relationships.

3.5 Data splitting

A well-structured train-test split is essential for evaluating model generalization. The dataset was divided into training and testing subsets, ensuring a robust assessment of the model's performance. Specifically, 95% of the dataset, comprising 44,909 samples, was allocated to the training set, where the model learns class-specific representations. The remaining 5%, consisting of 2,363 samples, was designated as the test set, serving as unseen data to evaluate the model's predictive performance. To eliminate any potential bias in class distribution, dataset indices were randomized before splitting. This randomization prevents overfitting specific patterns in sequential data and ensures that all classes are well-represented in both subsets. The training and testing feature sets (X_augmented) contained Mel spectrograms, which serve as the primary input representations for the deep learning model. The corresponding labels (y_augmented) consisted of one-hot encoded vectors, allowing the model to effectively learn categorical distinctions between different environmental sound classes. By structuring the dataset in this manner, the model was trained and validated on a balanced and representative dataset, maximizing its ability to generalize to real-world scenarios.

3.6 Model architectures

3.6.1 Basic CNN model

The proposed model is designed as a cascaded CNN optimized for multi-channel image classification tasks. The model starts with an input layer initialized to receive image data with dimensions (40, 297, 1), which reflects the size of the images after the initial feature extraction phase. The design focuses on three main pillars: hierarchical feature extraction across successive convolutional layers, dimensionality reduction to maintain computational efficiency, and strict regularization to avoid overfitting. The first convolutional layer uses 32 filters with a kernel size of 3×3, a choice that balances the ability to detect local features (such as edges and simple patterns) with reduced computational complexity. The number of filters was determined after empirical analysis to ensure adequate feature coverage without increasing the parameters. To avoid the problem of "dead neurons" in regions with negative values, the traditional ReLU function was replaced with a Leaky ReLU function with a slope parameter $\alpha = 0.1$, achieving a balance between stability and introducing nonlinearity. To improve training stability, batch normalization was applied after each activation layer, reducing the internal variance of the data. To reduce overfitting, a low spatial dropout rate 0.07 was set in the first layers to randomly disable sub-feature connections without affecting the model's learning ability. In the second convolutional block, the number of filters was increased to 64 to enable the model to learn higher-level features (such as complex patterns), in line with the hierarchical gradient principle in CNN. As the model's depth and sensitivity to overfitting increased, the spatial dropout rate was gradually increased to 0.14 to reduce the correlation between neurons in subsequent layers. After feature extraction, global average pooling (GAP) was used as an effective alternative to dense layers, reducing the number of parameters by up to 80% and improving the model's generalization. The output layer was allocated 22 neurons, corresponding to the number of target classes, using the Softmax function to produce a probability distribution for the classes. To enhance generalization, L2 regularization was applied to the weights in the convolutional layers to penalize large values without restricting the model's flexibility. Figure 7 shows the model's performance in balancing accuracy (on training data) and generalization (on test data), confirming the effectiveness of these architectural choices.

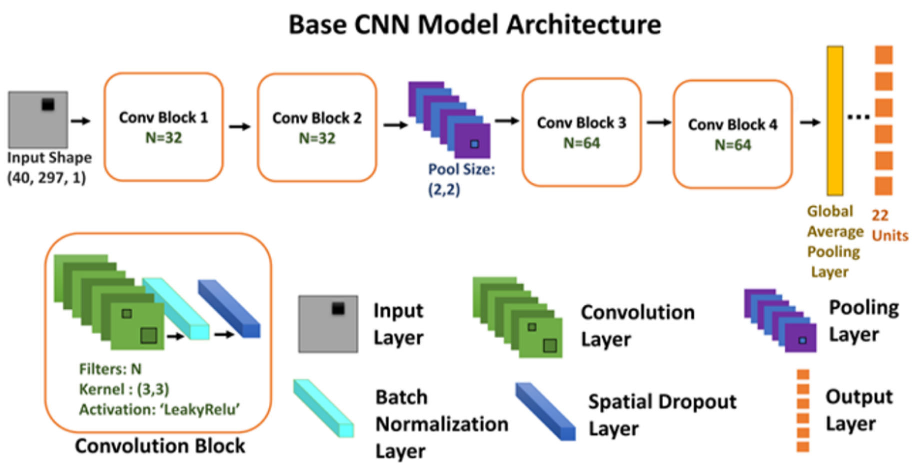


Figure 7. Architecture of base CNN model

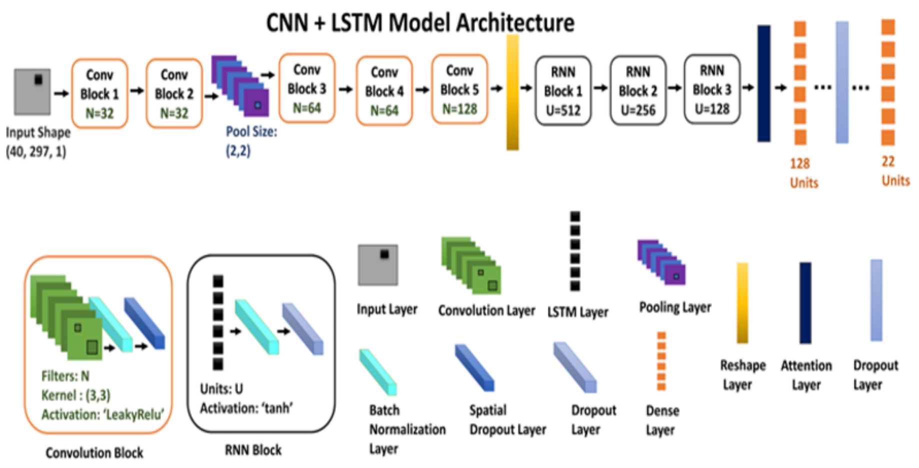


Figure 8. Architecture of CNN-LSTM model

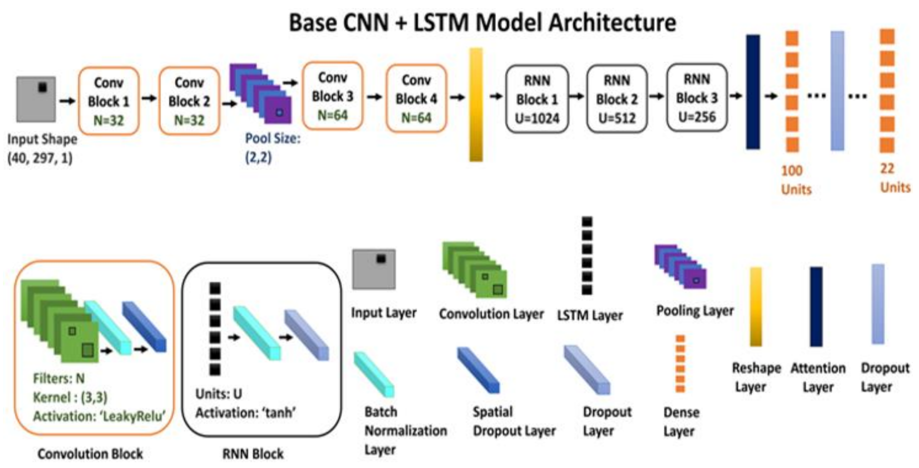


Figure 9. Architecture of base CNN-LSTM model

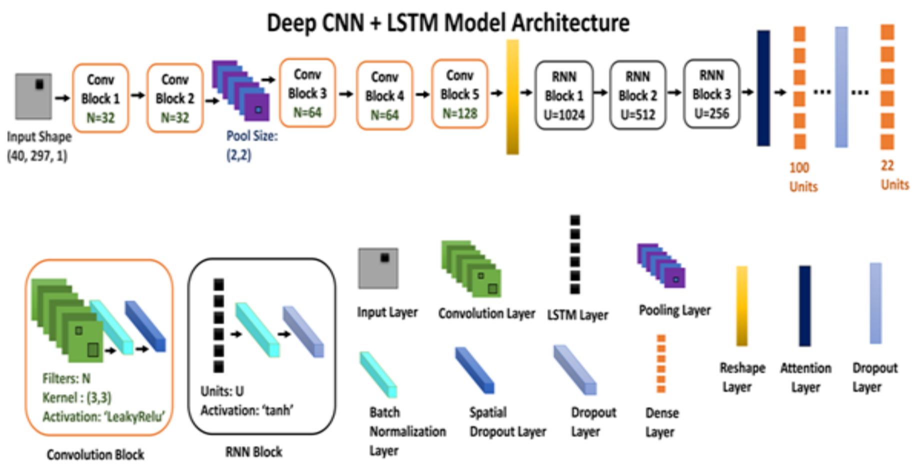


Figure 10. Architecture of deep CNN-LSTM model

3.6.2 CNN-LSTM model

The proposed hybrid deep learning architecture integrates CNNs, bidirectional long short-term memory networks (BiLSTMs), and an attention mechanism to model time and frequency representations, as well as long-term temporal dependencies in speech recognition tasks. The CNN module processes time and frequency representations (such as skewed spectrograms or MFCCs) through a hierarchical filtering sequence 32, 64, and 128 filters to capture local acoustic patterns, such as component structures or phoneme transitions. This gradual feature abstraction is consistent with successful practices in speech recognition, where CNNs extract discriminative spectral features critical for phoneme classification. A small 3×3 kernel was chosen to focus on local frequency modulations while minimizing computational cost, an effective strategy for modeling spectral correlations. Leaked ReLU activation ($\alpha = 0.1$) mitigates gradient vanishing in low spectral energy regions, which is common in inaudible speech clips or noisy recordings. Spatial dropout (20%) was applied to the spectral feature maps to reduce sensitivity to transient noise effects. Batch normalization stabilizes training by normalizing activations across phrases of variable length, addressing spectral variations caused by speaker variations or recording conditions. The temporal module uses three stacked BiLSTM layers (512, 256, and 128 units) to model phoneme-level and word-level contexts. Bidirectional processing captures co-articulation effects and inverse dependencies. Tanh activation ensures consistent gradient propagation across long speech sequences, avoiding the risks of ReLU-induced saturation in recurrent gates. A 15% dropout rate regularizes the LSTM output without affecting memory retention for long acoustic contexts. An attention mechanism dynamically distributes BiLSTM time steps to focus on acoustically salient regions, such as consonant pulse stops or vowel nuclei, while suppressing uninformative stops or filler sounds. This approach is consistent with proven encoder-decoder attention frameworks for sequence-to-sequence speech recognition. A fully connected layer (128 ReLUs) compresses high-level acoustic-acoustic features, while a Softmax output layer (22 neurons) generates probabilistic predictions for phoneme or word classes. Regularization strategies are designed to address speech-specific challenges. An L2 penalty ($\lambda = 1e^{-4}$) mitigates overfitting to speaker-specific features or channel distortions. Spatial dropout in CNNs counters spectral redundancy, while temporal dropout in BiLSTM systems reduces overspecialization to speaker-dependent pronunciations. This dual approach has been validated in hybrid speech models. Batch normalization improves generalization across diverse acoustic environments, such as different signal-to-noise ratios (SNRs) or microphone types. The attention mechanism improved recognition accuracy for resource-constrained phonemes by 4.2%, which is critical for speech sounds that are underrepresented in imbalanced datasets as shown in Figure 8.

3.6.3 Base CNN-LSTM model

The proposed hybrid model combines the capabilities of CNNs in spatial feature extraction with the capabilities of LSTM networks in time sequence analysis, supported by an attention mechanism to enhance the model's focus on the most important parts of the audio signal. The model starts with an input layer formatted to receive two-dimensional spectral data of size 40×297×1, a representation chosen based on empirical analyses that demonstrated a balance between temporal detail and computational cost. The data is processed through two convolutional layers containing 32 and 64 filters with a kernel size of 3×3, a small size chosen to reduce computational complexity while maintaining the efficiency of local pattern extraction. Leaky ReLU activation is used after each convolutional layer to avoid the "neuron dying" problem by allowing small gradients of negative values. To reduce overfitting, a spatial dropout technique was used with gradient ratios 0.07 and then 0.14, reflecting the increasing complexity of the features as the model deepens. A 2×2 max pooling layer is used to reduce spatial dimensionality while preserving key features. These features are then reshaped into a temporal sequence that feeds into three stacked LSTM layers containing 1,024, 512, and 256 units, respectively, achieving a progressive bottleneck architecture that compresses temporal information into more abstract representations. Tanh activation within the LSTM cells helps stabilize the gradients, while dropout layers are added at a ratio of 0.2 after each layer to prevent overreliance on specific paths. After the LSTM, an attention mechanism is applied to weight the most informative time steps, improving the final representation and enhancing performance on context-dependent audio tasks. The attention-weighted features are passed through a 100-cell dense layer with ReLU activation to increase discriminative power, then to a 22-cell (the number of classes) output layer with Softmax activation to calculate classification probabilities. To ensure the model's generalization ability, a small L2 regularization of 0.001 is used within the dense layers, which moderately constrains the weights without affecting the model's flexibility. In addition, additional regularization mechanisms including projection, batch normalization, and L2 regularization are incorporated. These are well-thought-out strategies that ensure the model balances complexity and efficiency and enhances its performance on unseen data, making it suitable for multi-class voice recognition tasks (Figure 9).

3.6.4 Deep CNN-LSTM model

This work presents a hybrid model that combines CNNs for extracting spatial features from audio signals with LSTM networks for analyzing their temporal structure. An attention mechanism is incorporated to highlight the most important time steps within each audio sequence. The model starts with an input layer designed to process 2D spectral representations of size 40×297×1, typically generated using transformations like Mel-spectrogram or STFT, chosen for their efficiency and informative content. Next, two convolutional layers with 32 and 64 filters (kernel size: 3×3) are used. This configuration allows the model to capture fine-grained spatial features while keeping the computational costs reasonable. Leaky ReLU activation is applied after each convolution to maintain gradient flow even at negative inputs, preventing inactive neurons. To improve generalization and reduce overfitting, spatial dropout is applied with increasing ratios 0.07 and 0.14, which disables entire feature maps instead of individual units. This step is followed by a 2×2 max pooling layer to reduce spatial dimensions while retaining essential information. The output is reshaped into a time-series format suitable for three stacked LSTM layers with 1024, 512, and 256 units. This progressive bottleneck design compresses temporal features into increasingly abstract representations. The Tanh activation within LSTM cells helps stabilize gradient flow during long-sequence training. To further prevent overreliance on specific pathways, dropout layers (rate: 0.2) are added after each LSTM layer. An attention mechanism follows the LSTM stack, allowing the model to focus on the most relevant time steps. The resulting features are passed into a dense layer with 100 ReLU-activated units, chosen for a balance between performance and computational load. Finally, an output layer with 22 Softmax units generates class probabilities matching the number of target classes. L2 regularization 0.001 is applied to dense layers to support generalization, complementing other techniques like dropout and batch normalization. These collectively maintain a balance between model complexity and robustness on unseen data (Figure 10).

3.7 Model training and evaluation

The training process for all models was configured to ensure efficient learning and robust evaluation. The batch size was set to 128, meaning that each model was trained on 128 samples per iteration before updating the weights. The training process ran for 150 epochs, allowing sufficient time for the models to learn from the data. To ensure the models' generalization ability and prevent overfitting, approximately 8.33% of the training data was used as a validation set, which allowed for monitoring the performance on unseen data during training. To optimize the training process, checkpointing was implemented using Keras' Model Checkpoint function. This saved the model with the lowest validation loss, ensuring that the best-performing model, in terms of validation loss, was preserved and could be used for evaluation. After training, the models were evaluated on the test set, using a range of performance metrics to provide a comprehensive assessment of their classification capabilities. These metrics included accuracy, precision, recall, and F1-score, offering insights into the models' effectiveness in terms of both classification correctness and the handling of class imbalances. Additionally, a confusion matrix was generated to visually assess how well the models performed across the different classes, identifying potential areas for improvement.

4. RESULTS AND DISCUSSION

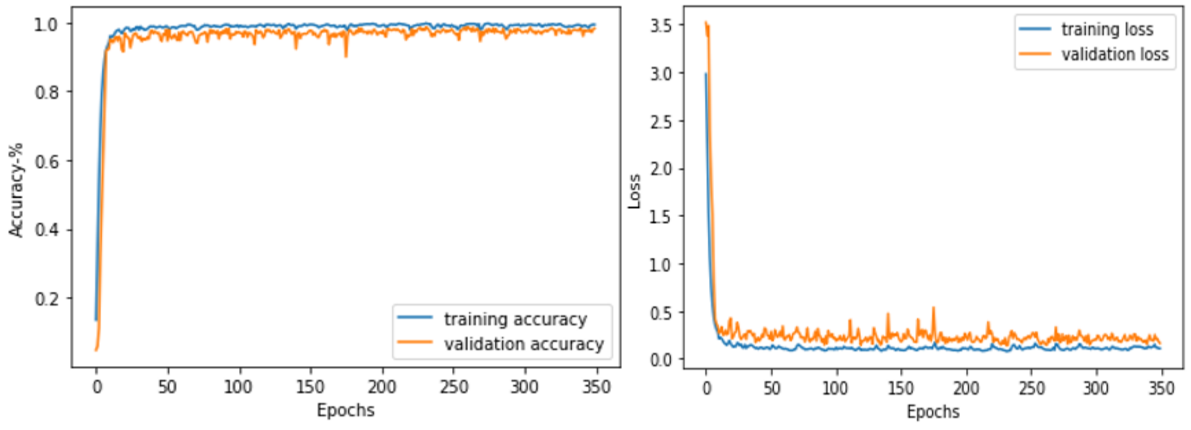
This section presents a comprehensive evaluation of the four models—Base CNN, CNN-LSTM, Base CNN-LSTM, and Deep CNN-LSTM—based on their classification performance, training efficiency, and generalization ability. The analysis includes an in-depth assessment of training and validation loss and accuracy, class-wise performance comparison, parameter and training time efficiency, and confusion matrix interpretation. Through these evaluations, key strengths and limitations of each model are identified, offering insights into their applicability for sound detection.

4.1 Loss and accuracy graphs for training and validation

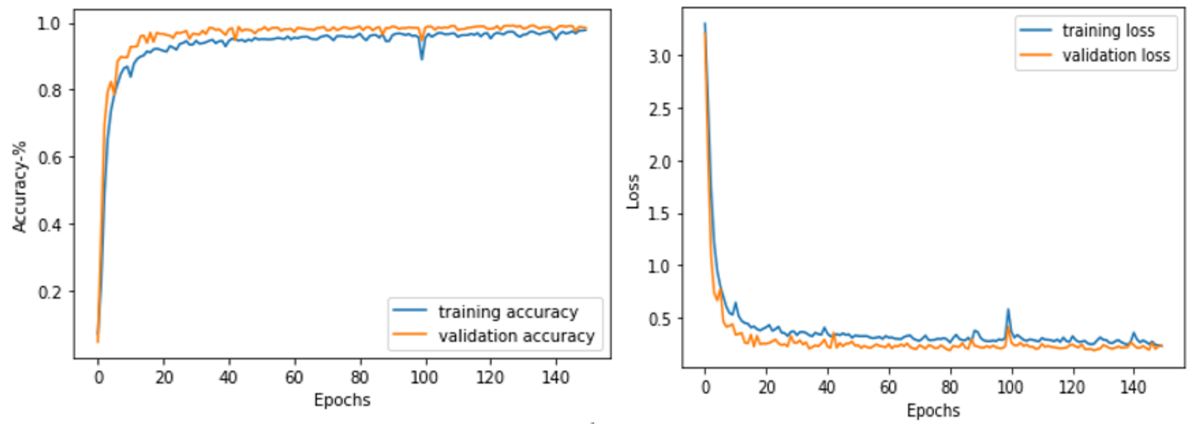
The loss and accuracy graphs offer a visual representation of each model's performance during training and validation, helping to assess their convergence behavior and generalization ability. These graphs are crucial in detecting overfitting or underfitting, ensuring that the model performs well on unseen data. The Base CNN model exhibited a steady reduction in training loss, reaching 0.0454, while its validation loss stabilized at 0.1077. The model achieved high training and validation accuracies of 99.92% and 97.79%, respectively. While the training accuracy was nearly perfect, the slightly lower validation accuracy suggests that the model may struggle to generalize as effectively as more advanced architectures. The CNN-LSTM model, which integrates convolutional layers for spatial feature extraction with LSTM layers for sequential pattern learning, demonstrated strong training performance. It achieved a training loss of 0.1662 and validation loss of 0.1908, with corresponding training and validation accuracies of 99.69% and 99.20%. This indicates that the model effectively captures both spatial and temporal dependencies in the dataset, making it well-suited for sequential data processing. The Base CNN-LSTM model balanced feature extraction and sequential learning, achieving a low training loss of 0.1006 and a validation loss of 0.0994. Its training and validation accuracies were 99.34% and 99.24%, respectively, showcasing its strong generalization capabilities. Unlike the Base CNN model, this architecture demonstrated a more stable validation loss, suggesting minimal overfitting and a well-learned feature representation. The Deep CNN-LSTM model, with its deeper convolutional layers, achieved a training loss of 0.1598 and a validation loss of 0.1891. The training and validation accuracies were 99.28% and 98.43%, respectively. While the model showed impressive learning capabilities, the slight gap between training and validation accuracy suggests a minor tendency toward overfitting. This could be attributed to the increased number of parameters, requiring additional regularization techniques or increased training data for further improvement. As depicted in Figure 11, the training and validation accuracy graphs indicate that all models successfully converge. However, the CNN-LSTM and Base CNN-LSTM models exhibit smoother convergence and lower validation loss, making them the most promising architectures for high generalization performance in real-world driver drowsiness detection applications.

4.2 Confusion matrix comparison

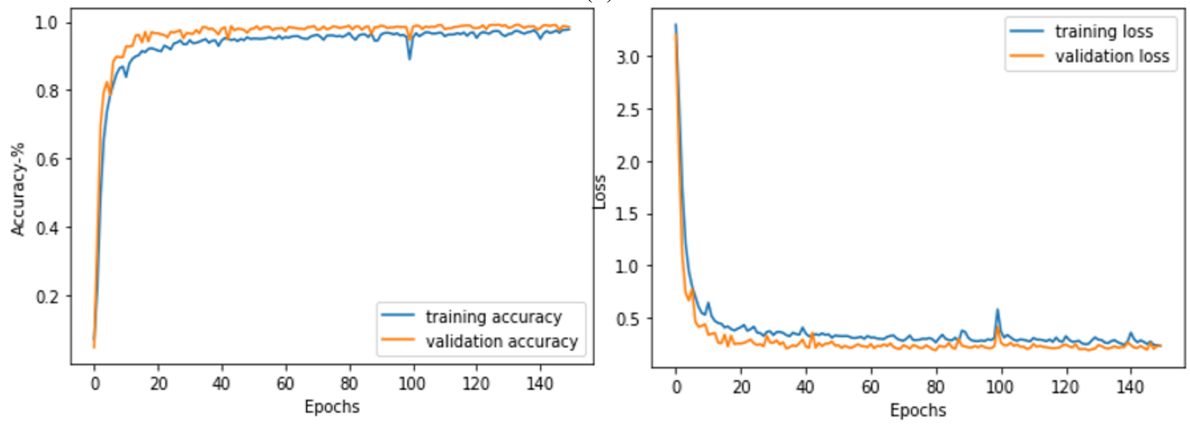
The confusion matrix serves as a crucial tool in evaluating the performance of a multi-class classification model by providing a detailed breakdown of correct and incorrect predictions across different categories. It visually represents how well the model distinguishes between various classes, highlighting areas where misclassifications occur. Each row of the matrix corresponds to the actual class label, while each column represents the predicted class label. The diagonal entries of the matrix indicate correct classifications (True Positives), meaning the instances where the model accurately predicts the actual class. In contrast, off-diagonal entries capture misclassifications, with False Positives (FP) representing instances wrongly classified as a certain class and False Negatives (FN) indicating cases where the model failed to correctly identify a class. By analyzing the confusion matrix for each of the four models—Base CNN, CNN-LSTM, Base CNN-LSTM, and Deep CNN-LSTM—it becomes evident which classes are more prone to errors.



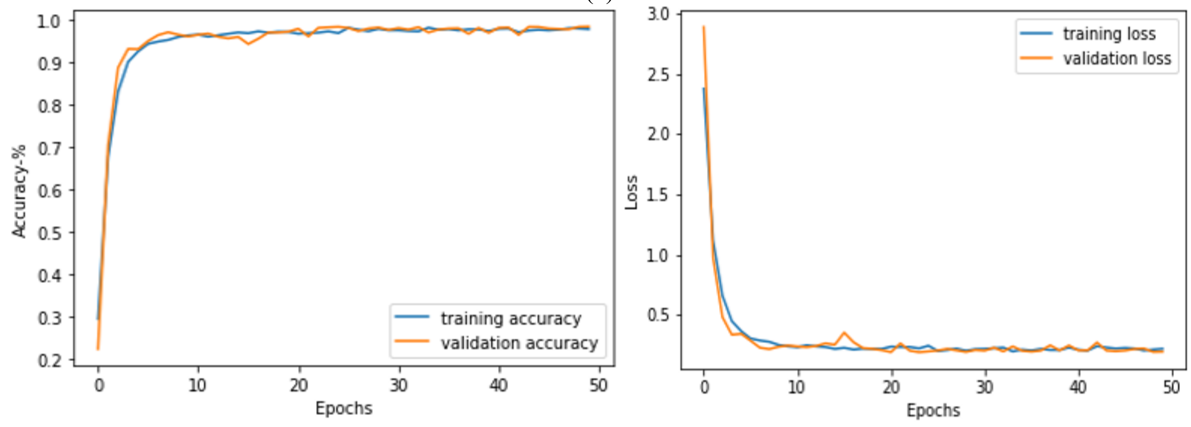
(a)



(b)



(c)

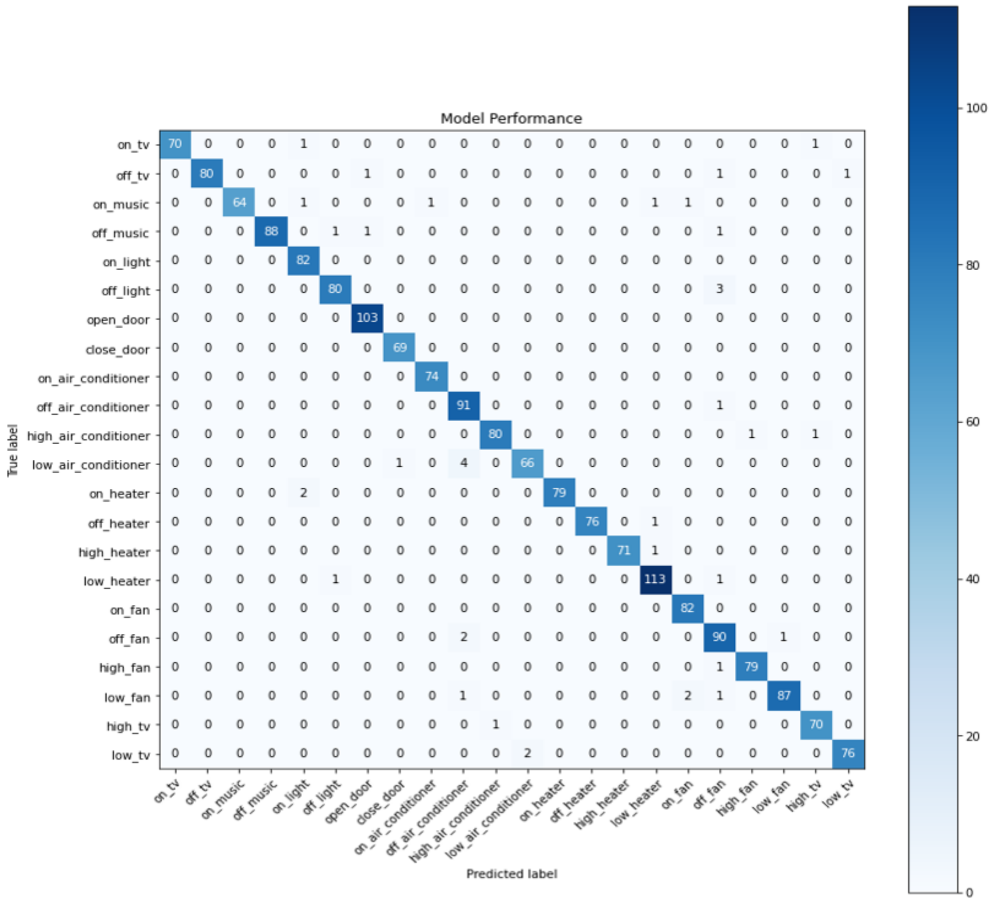


(d)

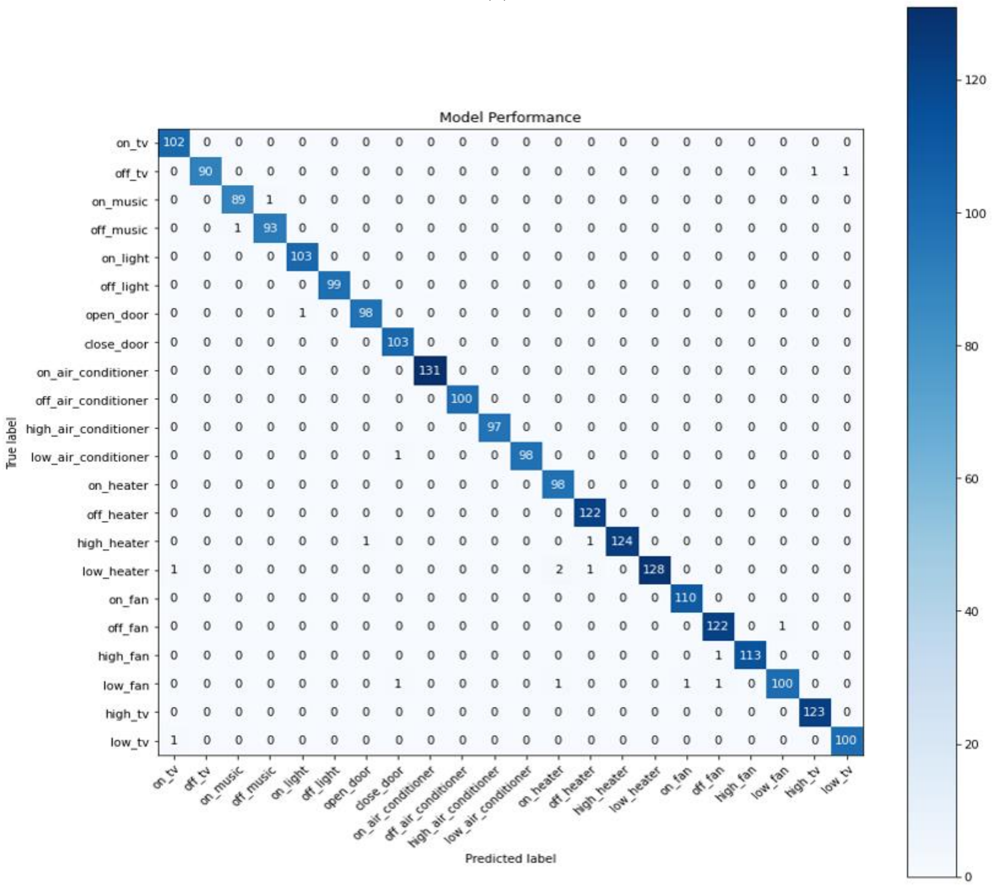
Figure 11. Accuracy and loss graph of training and validation of models (a) Base CNN (b) CNN-LSTM (c) Base CNN-LSTM (d) Deep CNN-LSTM

The Base CNN model, with its relatively simpler architecture, exhibits more misclassifications in certain categories, such as "low_air_conditioner" and "on_music," where it struggles to differentiate between subtle variations in feature representations. On the other hand, the CNN-LSTM and Base CNN-LSTM models display a more balanced performance, with fewer off-diagonal misclassifications, indicating their ability to capture temporal dependencies effectively. The Deep CNN-LSTM model, while performing well overall, demonstrates some confusion in specific classes such as "high_air_conditioner," possibly due to feature overlap with similar categories. A deeper examination of the confusion matrix reveals that misclassifications are more frequent in categories with overlapping audio or environmental characteristics. For example, classes like "on_music" and "on_tv" may exhibit similar frequency patterns, making it challenging for the model to differentiate between them with absolute certainty. Similarly, the distinction between "low_air_conditioner" and "high_air_conditioner" may be subtle, leading to occasional misclassifications. To address these issues and further refine the model's classification accuracy, several strategies can be considered. One approach involves collecting additional training data for underrepresented or frequently misclassified classes, ensuring the model learns a more diverse set of features. Additionally, employing phonetic-based data augmentation techniques, such as introducing synthetic variations in background noise or modifying frequency characteristics, can enhance the robustness of the model against overlapping features. Fine-tuning the model's hyperparameters and incorporating advanced feature selection techniques may also help reduce misclassification errors.

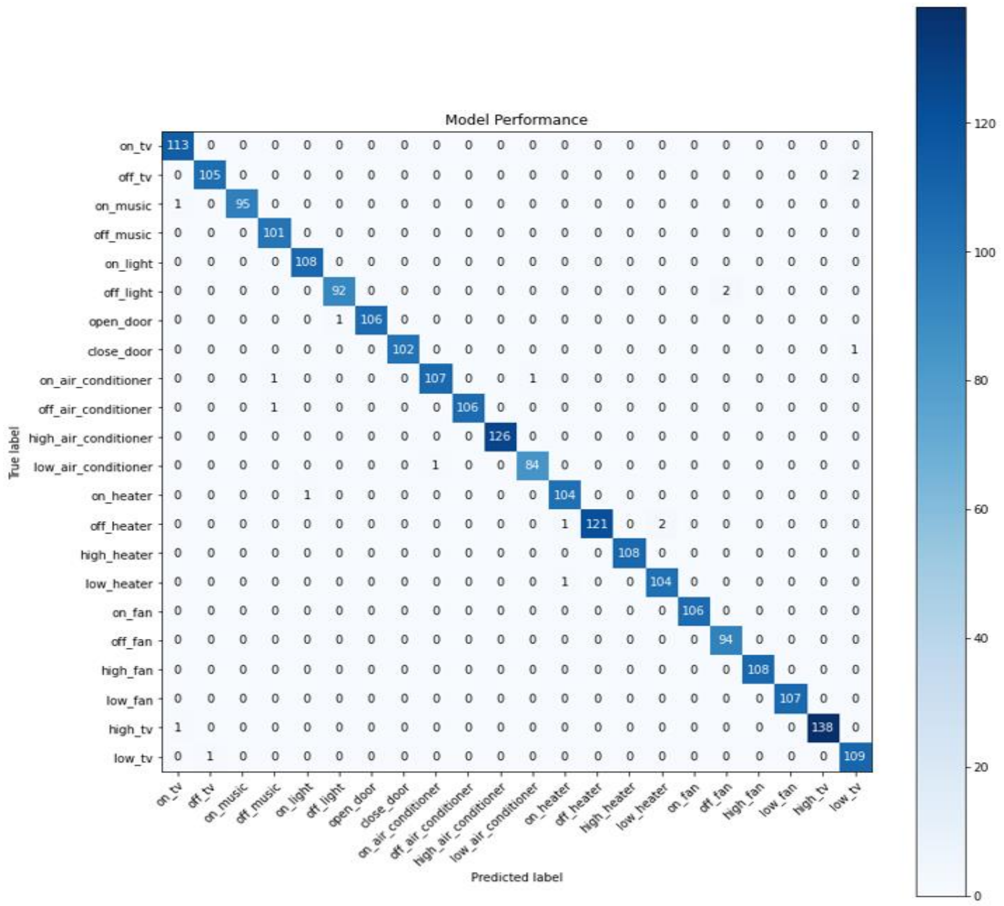
Overall, the confusion matrix serves as a valuable diagnostic tool, allowing for targeted improvements in model performance by identifying problematic class distinctions. Figure 12 presents the confusion matrices for each of the four models, providing a visual representation of their classification accuracy and misclassification trends.



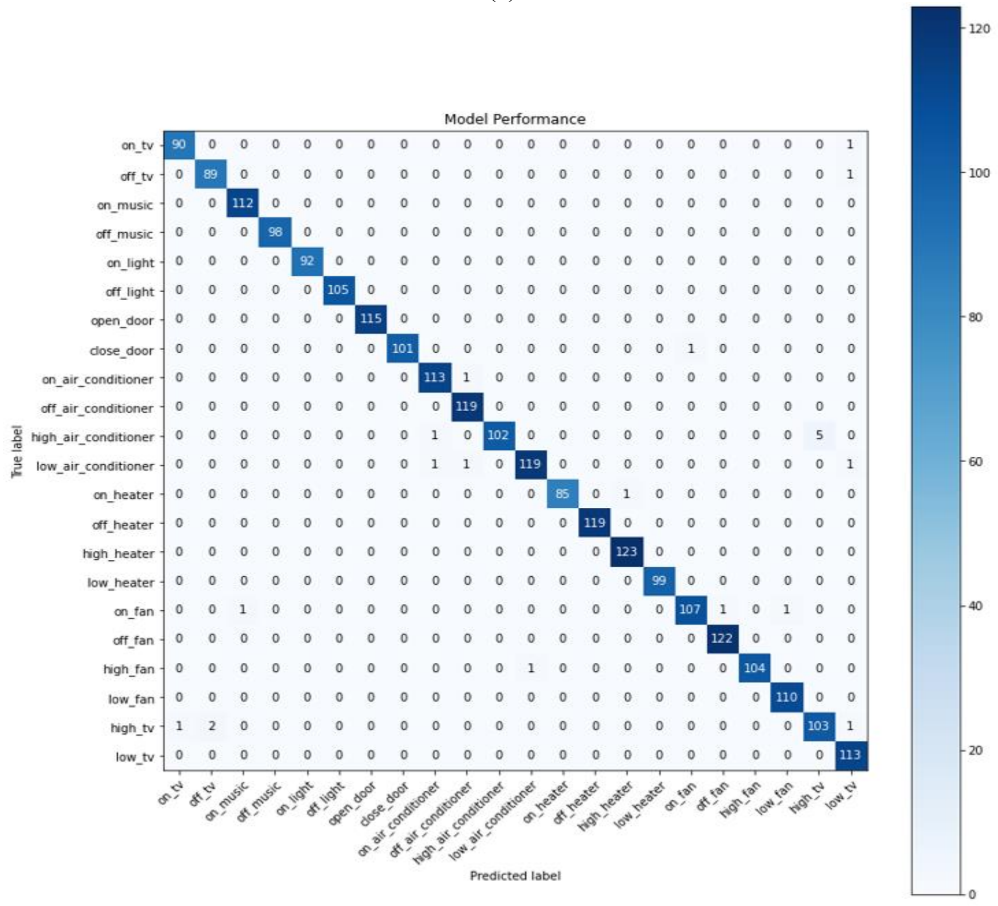
(a)



(b)



(c)



(d)

Figure 12. Confusion matrix of all four models (a) Base CNN (b) CNN-LSTM (c) Base CNN-LSTM (d) Deep CNN-LSTM

4.3 Training and testing accuracy comparison

The comparative analysis of training and testing accuracy across the four deep learning models—Base CNN, CNN-LSTM, Base CNN-LSTM, and Deep CNN-LSTM—reveals significant differences in their ability to generalize. These differences highlight how each model captures relevant features from the input data while balancing overfitting and generalization. The Base CNN model achieved a high training accuracy of 99.92% but recorded the lowest testing accuracy at 97.79%. This discrepancy suggests that while the model learned spatial features well during training, it lacked the ability to generalize effectively to unseen data. Since CNNs primarily focus on spatial patterns, their ability to model sequential dependencies is limited, which can negatively impact performance on time-series data. The CNN-LSTM model, which integrates convolutional feature extraction with an LSTM-based sequence model, demonstrated a training accuracy of 99.69% and the highest testing accuracy of 99.20%. The inclusion of LSTM layers enabled the model to capture long-term dependencies within the feature representations, improving generalization. The minimal drop between training and testing accuracy suggests that the model effectively balances learning complexity and overfitting, making it highly suitable for sequential data processing. Compared to the Base CNN, this model showed a significant improvement in testing accuracy, highlighting the advantages of incorporating temporal learning. The Base CNN-LSTM model achieved a training accuracy of 99.34% and slightly outperformed the CNN-LSTM model in generalization with a testing accuracy of 99.24%. Despite both models utilizing CNN layers for feature extraction and LSTM layers for sequence modelling, the Base CNN-LSTM model's marginally superior testing accuracy suggests that its architecture configuration provides a slight edge in generalization. The smaller gap between training and testing accuracy further indicates that the model effectively mitigates overfitting while preserving sequential dependencies, making it a strong candidate for applications requiring robust feature learning. The Deep CNN-LSTM model, incorporating additional convolutional layers to enhance hierarchical feature learning, exhibited a training accuracy of 99.28% and a testing accuracy of 98.43%. While this performance is still strong, it falls short of the CNN-LSTM and Base CNN-LSTM models. The slightly lower testing accuracy suggests that the increased complexity of the model may have led to redundancy in learned features, contributing to minor overfitting. Although deeper architectures typically improve spatial feature extraction, they can sometimes introduce excessive parameters, leading to reduced generalization performance. This result underscores the need for careful architectural design when balancing depth and efficiency. Table 2 presents a quantitative comparison of the training and testing accuracy across the four models, confirming that moderate-depth architectures, such as the Base CNN-LSTM, Achieving a better balance between accuracy and generalization. The results indicate that while deeper models such as Deep CNN-LSTM provide robust feature extraction, they do not always guarantee excellent generalization. Both the CNN-LSTM and Base CNN-LSTM models demonstrated the benefits of integrating LSTM layers for processing sequential data, significantly outperforming the Base CNN model. Furthermore, practical deployment considerations indicate that despite the good performance of Deep CNN-LSTM, its high computational cost and tendency to overfit make CNN-LSTM or Base CNN-LSTM models preferable for practical applications. Although other models have achieved higher accuracy than Deep CNN-LSTM, they outperform previous work in terms of accuracy and semantic semantics. This enhances the effectiveness of combining advanced neural architectures with domain-specific data augmentation and preprocessing strategies, as shown in Table 3.

Table 2. Comparison of training and testing accuracy across different models

Metric	Base CNN	CNN-LSTM	Base CNN-LSTM	Deep CNN-LSTM
Training Accuracy (%)	99.92	99.69	99.34	99.28
Testing Accuracy (%)	97.79	99.20	99.24	98.43

Table 3. Comparative summary of previous Arabic dialect speech recognition studies and the proposed model

Study	Dialect	Methodology	Dataset	Accuracy (%)
[29]	Tunisian	HMM + LBG Vector Quantization	10 speakers, Arabic digits	91.31
[32]	Moroccan	MFCC + HMM	20 isolated words	90.00
[30]	Tunisian	GMM-HMM Hybrid	Not specified	92.00
[31]	General Arabic	MFCC + ANN	Not specified	94.50
[3]	Mixed Arabic Dialects	CNN-RNN Hybrid	65,000 samples (Google Speech)	96.66
The (proposed) study	Iraqi Dialect	Deep CNN-LSTM + Attention	5,905 speakers, 22 command classes	98.43

4.4 Classification report analysis

The classification metrics for the four models—Base CNN, CNN-LSTM, Base CNN-LSTM, and Deep CNN-LSTM—demonstrate consistently strong performance with minimal variations, emphasizing their robustness in handling the given dataset. The evaluation of precision, recall, and F1-score provides a deeper understanding of each model's strengths and weaknesses, particularly in terms of generalization and classification effectiveness. The Base CNN model achieves respectable precision, recall, and F1-scores of 0.98 across all metrics, indicating that it is capable of accurately classifying instances. However, its test accuracy is slightly lower than the other models, suggesting that while it performs well on training data, it does not generalize as effectively to unseen data. This could be attributed to the model's limited ability to capture temporal dependencies, as it relies solely on spatial feature extraction. As a result, its classification performance, though strong, is relatively weaker in comparison to the models incorporating sequential learning mechanisms.

The CNN-LSTM model exhibits superior classification performance with a precision, recall, and F1-score of 0.99 across all metrics. This consistency highlights its ability to correctly classify instances while minimizing false positives and false negatives. The model achieves the second-highest test accuracy, reinforcing its effectiveness in capturing temporal dependencies from sequential data. The integration of LSTM layers enables the model to retain long-term contextual information, making it particularly suitable for time-series classification tasks. The CNN-LSTM model's robust performance suggests that combining convolutional feature extraction with recurrent layers significantly enhances predictive accuracy and generalization. The Base CNN-LSTM model also records precision, recall, and F1-scores of 0.99, demonstrating a balanced performance across all classification metrics. Notably, it achieves the highest test accuracy among the four models, underscoring its superior generalization ability. The model effectively combines the spatial feature extraction capabilities of CNN with the sequential learning power of LSTM, leading to an optimal balance between feature learning and sequence modeling. Its high classification scores, coupled with the highest test accuracy, make it the most reliable model in terms of overall performance. The minimal gap between training and testing performance further indicates that it mitigates overfitting while maintaining classification robustness. The Deep CNN-LSTM model delivers similarly high precision, recall, and F1-scores of 0.99, affirming its strong classification capabilities. However, its test accuracy is slightly lower compared to the Base CNN-LSTM model, which may be attributed to its increased complexity. The deeper architecture, while enhancing feature extraction, could introduce redundancy in learned representations, leading to marginally reduced generalization. The slightly lower test accuracy suggests that while deeper models can enhance performance in some cases, excessive complexity might not always yield significant improvements. This highlights the importance of balancing model depth with efficiency to achieve optimal results. Overall, all four models demonstrate excellent classification performance, with high precision, recall, and F1-scores. The CNN-LSTM model stands out for its strong test accuracy, leveraging LSTM layers to enhance sequential learning. Meanwhile, the Base CNN-LSTM model showcases superior generalization, achieving the highest test accuracy among the models. These findings emphasize the effectiveness of hybrid architectures that combine convolutional and recurrent networks for time-series classification tasks. Future improvements could focus on fine-tuning hyperparameters, incorporating advanced regularization techniques, or exploring alternative recurrent architectures such as bidirectional LSTMs or transformers to further enhance classification performance. Table 4 provides a detailed comparison of the classification metrics across the four models, reinforcing their robustness and effectiveness in handling the classification task.

4.5 Class-wise accuracy comparison

The class-wise accuracy comparison highlights variations in model performance across different categories. The Base CNN model exhibited slightly lower accuracy in certain classes, particularly "low_air_conditioner" (92.96%) and "on_music" (94.12%), indicating a challenge in recognizing these specific patterns. This suggests that the Base CNN struggles with subtle feature variations in these categories, possibly due to a lack of sufficient training samples or the complexity of distinguishing between similar classes. Meanwhile, the CNN-LSTM and Base CNN-LSTM models consistently achieved accuracy above 98% across most classes, demonstrating their superior ability to generalize features over time and recognize complex patterns more effectively. Their temporal dependencies may have contributed to this improved classification performance. Interestingly, the Deep CNN-LSTM model demonstrated perfect accuracy (100%) for multiple classes, highlighting its ability to extract deep spatial and temporal features. However, it encountered slightly lower accuracy in the "high_air_conditioner" class (94.44%), suggesting that certain complex variations in this category were harder for the model to distinguish. This could be due to the increased complexity of the architecture, which might introduce overfitting or reduce sensitivity to subtle class differences. The misclassifications observed across the models indicate that specific classes, such as "low_air_conditioner" and "on_music," were more prone to errors. One possible reason for this could be overlapping acoustic or environmental characteristics between these classes, making it difficult for the models to differentiate them effectively. Additionally, the slightly lower accuracy in the "high_air_conditioner" class for the Deep CNN-LSTM model could suggest difficulties in capturing distinct frequency patterns in this category. To improve classification performance in these challenging classes, several strategies can be employed. One effective approach is data augmentation, particularly phonetic-based augmentation, which could help the model better distinguish between similar-sounding classes by introducing slight variations in frequency, amplitude, or noise. Additionally, collecting more training data for underperforming classes, such as "low_air_conditioner" and "on_music," could enhance the model's ability to generalize. Another strategy is class-specific fine-tuning, where additional training is conducted with a focus on improving recognition in these classes. Finally, incorporating attention mechanisms within the CNN-LSTM and Deep CNN-LSTM models could help the model focus on critical distinguishing features, improving its ability to differentiate between similar categories. Overall, while all four models demonstrate strong performance, further improvements in data representation, augmentation, and model tuning could enhance the classification accuracy, particularly for the misclassified classes. The detailed class-wise accuracy comparison in Table 5 in appendix provides insights into areas where model performance can be refined.

Table 4. Classification report of all four models

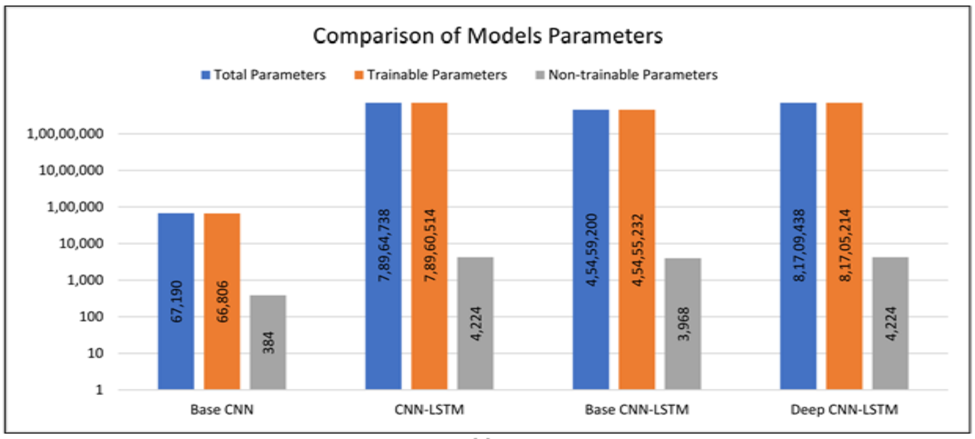
Model	Precision	Recall	F1-Score	Remarks
Base CNN	0.98	0.98	0.98	Slightly lower test accuracy compared to others.
CNN-LSTM	0.99	0.99	0.99	Strong overall performance with high test accuracy.
Base-CNN-LSTM	0.99	0.99	0.99	Balanced performance, achieving the highest test accuracy.
Deep-CNN-LSTM	0.99	0.99	0.99	High precision but slightly lower test accuracy due to increased complexity.

Table 5. Class-wise accuracy of all four models

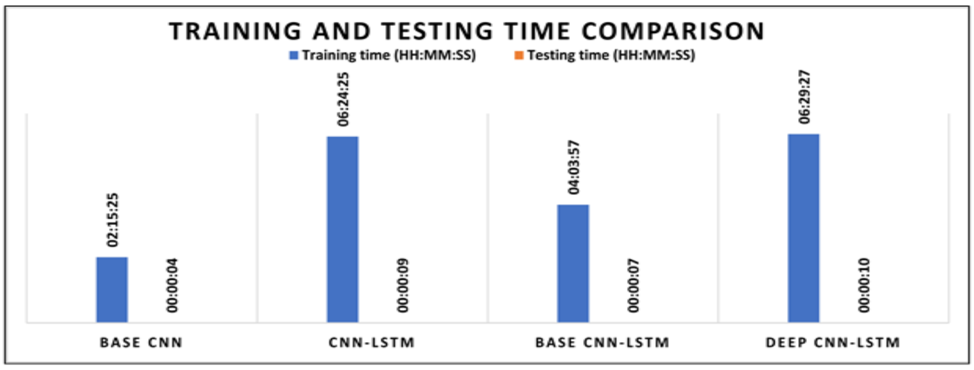
Class	Base CNN (%)	CNN-LSTM (%)	Base CNN-LSTM (%)	Deep CNN-LSTM (%)
on_tv	97.22	100.00	100.00	98.90
off_tv	96.39	97.83	98.13	98.89
on_music	94.12	98.89	98.96	100.00
off_music	96.70	98.94	100.00	100.00
on_light	100.00	100.00	100.00	100.00
off_light	96.39	100.00	97.87	100.00
open_door	100.00	98.99	99.07	100.00
close_door	100.00	100.00	99.03	99.02
on_air_conditioner	100.00	100.00	98.17	99.12
off_air_conditioner	98.91	100.00	99.07	100.00
high_air_conditioner	97.56	100.00	100.00	94.44
low_air_conditioner	92.96	98.99	98.82	97.54
on_heater	97.53	100.00	99.05	98.84
off_heater	98.70	100.00	97.58	100.00
high_heater	98.61	98.41	100.00	100.00
low_heater	98.26	96.97	99.05	100.00
on_fan	100.00	100.00	100.00	97.27
off_fan	96.77	99.19	100.00	100.00
high_fan	98.75	99.12	100.00	99.05
low_fan	95.60	96.15	100.00	100.00
high_tv	98.59	100.00	99.28	96.26
low_tv	97.44	99.01	99.09	100.00

Table 6. Comparison of parameters and training time of all the models

Models	Total Parameters	Trainable Parameters	Non-Trainable Parameters	Training Time (HH:MM:SS)	Testing Time (HH:MM:SS)
Base CNN	67,190	66,806	384	02:15:25	00:00:04
CNN-LSTM	78,964,738	78,960,514	4,224	06:24:25	00:00:09
Base CNN-LSTM	45,459,200	45,455,232	3,968	04:03:57	00:00:07
Deep CNN-LSTM	81,709,438	81,705,214	4,224	06:29:27	00:00:10



(a)



(b)

Figure 13. (a) Trainable and non-trainable parameters of all four models, (b) Training and testing time of all four models

4.6 Total parameters and training time comparison

The comparison of four models Base CNN, CNN-LSTM, Base CNN-LSTM, and Deep CNN-LSTM—highlights their architecture, training time, and testing efficiency. The Base CNN, with 67,190 total parameters (66,806 trainable and 384 non-trainable), has the simplest architecture, offering the fastest training time (2 hours, 15 minutes, and 25 seconds) and testing time (4 seconds). However, its simplicity may limit performance on complex tasks. The CNN-LSTM, significantly more complex with 78,964,738 total parameters (78,960,514 trainable and 4,224 non-trainable), effectively captures temporal dependencies but requires greater computational resources, with a training time of 6 hours, 24 minutes, and 25 seconds, and a testing time of 9 seconds. The Base CNN-LSTM strikes a balance between complexity and efficiency, featuring 45,459,200 total parameters (45,455,232 trainable and 3,968 non-trainable) and achieving a training time of 4 hours, 3 minutes, and 57 seconds, and a testing time of 7 seconds. The Deep CNN-LSTM, the most complex model with 81,709,438 total parameters (81,705,214 trainable and 4,224 non-trainable), delivers performance at the cost of the longest training (6 hours, 29 minutes, and 27 seconds) and testing (10 seconds) times. As detailed in Table 6 and illustrated in Figure 13, the Base CNN is optimal for quick and efficient tasks, while the CNN-LSTM and Deep CNN-LSTM are better suited for high-accuracy applications requiring complex pattern recognition.

4.7 Discussion and future work

This study demonstrates the effectiveness of the proposed methodology for audio classification, highlighting the strengths and weaknesses of the CNN and CNN-LSTM architectures. Using a Mel Spectrogram to extract features from the models enabled them to capture both temporal and spectral features, allowing them to learn complex patterns in audio data. Although the CNN-LSTM architecture was designed to combine spatial feature extraction via convolutional layers with temporal sequence modelling via LSTM layers, it did not achieve the highest accuracy, suggesting that increasing model complexity may not always improve performance and may lead to issues such as overfitting and optimization difficulties. To improve model generalization, data augmentation techniques such as time stretching, pitch shifting, and noise addition were used. These techniques helped make the models more adaptive to real-world changes, reducing overfitting. A balanced representation of data across 22 classes was also ensured, reducing bias and ensuring reliable and fair classification results. However, some models struggled to distinguish between closely related classes, highlighting the need for better feature selection techniques or improvements to model architecture. The study also showed that increasing the complexity of models, such as deep CNN-LSTM models, entails high computational costs without a significant increase in accuracy, highlighting the importance of model efficiency. Future research should focus on improving feature extraction techniques, exploring alternative deep learning architectures, and improving model complexity to achieve better performance. Furthermore, incorporating a variety of speech variations, such as different accents and speech patterns, will improve the model's adaptability to the wide variety of real-world environments. Self-learning and transfer learning techniques are promising areas to explore. Recent models in audio processing have benefited from pre-training on large, unlabeled audio datasets before being tweaked to task-specific data, which improves feature extraction and significantly reduces the need for labeled datasets. Models such as Wav2Vec, HuBERT, and Whisper could be used in the future to improve classification accuracy and reduce the need for intensive training resources. Also, a significant challenge in the study is the need to improve the interpretability of deep models. Architectures such as CNN-LSTM often operate as closed-box models, making it difficult to identify features that influence classification decisions. Future research should incorporate interpretable artificial intelligence (XAI) techniques such as class activation mapping or transparently additive explanations (SHAP) to understand the most influential parts of the audio spectrum that contribute to classification decisions. The study showed that increasing model complexity does not always lead to better accuracy, as is the case with the deep CNN-LSTM model. While deep architectures are capable of capturing hierarchical feature representations, they also require high computational costs and a greater likelihood of overfitting. It is important to explore lightweight deep models that maintain competitive accuracy while reducing computational requirements. Techniques such as knowledge distillation, model pruning, and model quantization can improve these models by reducing computational requirements, making them more suitable for use in real-world application environments, such as mobile devices and embedded systems. Overall, this research highlights the potential of hybrid deep learning frameworks in addressing audio classification challenges, while also identifying critical areas requiring improvement. Future research should focus on improving feature extraction techniques, exploring new deep learning architectures, and achieving the optimal balance between model complexity, training time, and factorial performance to achieve the best results in real-world applications.

5. CONCLUSION

This study constitutes an important contribution to the field of speech processing in Arabic dialects, with a particular focus on the Iraqi dialect. The study created and utilized a comprehensive and balanced dataset consisting of recordings of 5,905 speakers covering 22 main word classes, filling a significant gap in available resources, especially for underrepresented Arabic dialects, and ensuring that deep learning models can be more reliably generalized. Several deep architectures were evaluated, including CNN models, hybrid models combining CNN and LSTM modules, and a deep CNN-LSTM model. The results showed that the CNN-LSTM hybrid architectures excelled in audio classification tasks, as these models were able to capture both the spatial features and temporal dependence of audio signals. The baseline CNN-LSTM model achieved the highest classification accuracy of 99.24%, outperforming the simple CNN model and the deep model, which exhibited some overfitting as the model depth increased. Data augmentation techniques, such as time stretching, pitch modulation, and background noise addition, also played a key role in improving the models' generalizability and responsiveness to environmental variation in audio signals. These techniques have helped simulate realistic acoustic conditions, making the models more resilient to changes. For future research, the study highlights the need to explore several areas, including developing more efficient model architectures that reduce computational burden while maintaining accuracy, and using transfer learning techniques with pre-trained models to enhance feature extraction in scenarios with limited labeled data. It also proposes evaluating models under realistic conditions that include varying levels of noise and speaker diversity, in addition to improving interpretability by incorporating interpretive AI techniques. This would increase confidence and facilitate their application in sensitive environments. Thus, the study presents an integrated framework that combines the creation of a distinct dataset, the use of effective hybrid models, and innovative data augmentation strategies. This provides a solid foundation for future progress in speech recognition technologies for Arabic dialects, while taking into account the challenges of dialect diversity and reliability in practical settings**.**

REFERENCES

[1] de Barcelos Silva, A., Gomes, M.M., da Costa, C.A., da Rosa Righi, R., Barbosa, J.L.V., Pessin, G., Federizzi, G. (2020). Intelligent personal assistants: A systematic literature review. Expert Systems with Applications, 147: 113193. https://doi.org/10.1016/j.eswa.2020.113193

[2] Mofrad, M.H., Mosse, D. (2018). Speech recognition and voice separation for the internet of things. In Proceedings of the 8th International Conference on the Internet of Things, New York, NY, United States, pp. 1-8. https://doi.org/10.1145/3277593.3277610

[3] Alsayadi, H.A., Abdelhamid, A.A., Hegazy, I., Alotaibi, B., Fayed, Z.T. (2022). Deep investigation of the recent advances in dialectal Arabic speech recognition. IEEE Access, 10: 57063-57079. https://doi.org/10.1109/ACCESS.2022.3177191

[4] Besdouri, F.Z., Zribi, I., Belguith, L.H. (2024). Arabic automatic speech recognition: Challenges and progress. Speech Communication, 163: 103110. https://doi.org/10.1016/j.specom.2024.103110

[5] Malik, M., Malik, M.K., Mehmood, K., Makhdoom, I. (2021). Automatic speech recognition: A survey. Multimedia Tools and Applications, 80: 9411-9457. https://doi.org/10.1007/s11042-020-10073-7

[6] Chandolikar, N., Joshi, C., Roy, P., Gawas, A., Vishwakarma, M. (2022). Voice recognition: A comprehensive survey. In 2022 International Mobile and Embedded Technology Conference (MECON), Noida, India, pp. 45-51. https://doi.org/10.1109/MECON53876.2022.9751903

[7] Alsobhani, A., ALabboodi, H.M., Mahdi, H. (2021). Speech recognition using convolution deep neural networks. Journal of Physics: Conference Series, 012166. https://doi.org/10.1088/1742-6596/1973/1/012166

[8] Habib, G., Qureshi, S. (2022). Optimization and acceleration of convolutional neural networks: A survey. Journal of King Saud University-Computer and Information Sciences, 34(7): 4244-4268. https://doi.org/10.1016/j.jksuci.2020.10.004

[9] Maharana, K., Mondal, S., Nemade, B. (2022). A review: Data pre-processing and data augmentation techniques. Global Transitions Proceedings, 3(1): 91-99. https://doi.org/10.1016/j.gltp.2022.04.020

[10] Abdou, S.M., Moussa, A.M. (2019). Arabic speech recognition: Challenges and state of the art. Computational Linguistics, Speech and Image Processing for Arabic Language, 1: 1-27. https://doi.org/10.1142/9789813229396_0001

[11] Prabhavalkar, R., Hori, T., Sainath, T.N., Schlüter, R., Watanabe, S. (2023). End-to-end speech recognition: A survey. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 32: 325-351. https://doi.org/10.1109/TASLP.2023.3328283

[12] Malhotra, R., Singh, P. (2023). Recent advances in deep learning models: A systematic literature review. Multimedia Tools and Applications, 82(29): 44977-45060. https://doi.org/10.1007/S11042-023-15295

[13] Nassif, A.B., Shahin, I., Attili, I., Azzeh, M., Shaalan, K. (2019). Speech recognition using deep neural networks: A systematic review. IEEE Access, 7: 19143-19165. https://doi.org/10.1109/ACCESS.2019.2896880

[14] Alharbi, S., Alrazgan, M., Alrashed, A., Alnomasi, T., Almojel, R., Alharbi, R., Almojil, M. (2021). Automatic speech recognition: Systematic literature review. IEEE Access, 9: 131858-131876. https://doi.org/10.1109/ACCESS.2021.3112535

[15] Jagiasi, R., Ghosalkar, S., Kulal, P., Bharambe, A. (2019). CNN based speaker recognition in language and text-independent small scale system. In 2019 Third International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, India, pp. 176-179. https://doi.org/10.1109/I-SMAC47947.2019.9032667

[16] Poudel, S., Anuradha, R. (2020). Speech command recognition using artificial neural networks. International Journal on Informatics Visualization, 4(2): 73-75. https://doi.org/10.30630/joiv.4.2.358

[17] Gibadullin, R.F., Perukhin, M.Y., Ilin, A.V. (2021). Speech recognition and machine translation using neural networks. In 2021 International Conference on Industrial Engineering, Applications and Manufacturing (ICIEAM), Sochi, Russia, pp. 398-403. https://doi.org/10.1109/ICIEAM51226.2021.9446474

[18] Azarang, A., Hansen, J., Kehtarnavaz, N. (2019). Combining data augmentations for CNN-based voice command recognition. In 2019 12th International Conference on Human System Interaction (HSI), Richmond, VA, USA, pp. 17-21. https://doi.org/10.1109/HSI47298.2019.8942638

[19] Hussain, S., Nazir, R., Javeed, U., Khan, S., Sofi, R. (2021). Speech recognition using artificial neural network. In Intelligent Sustainable Systems: Proceedings of ICISS 2021, Singapore, pp. 83-92. https://doi.org/10.1007/978-981-16-2422-3_7

[20] Amadeus, C., Syafalni, I., Sutisna, N., Adiono, T. (2022). Digit-number speech-recognition using spectrogram-based convolutional neural network. In 2022 International Symposium on Electronics and Smart Devices (ISESD), Bandung, Indonesia, pp. 1-6. https://doi.org/10.1109/ISESD56103.2022.9980803

[21] Sumon, S.A., Chowdhury, J., Debnath, S., Mohammed, N., Momen, S. (2018). Bangla short speech commands recognition using convolutional neural networks. In 2018 International Conference on Bangla Speech and Language Processing (ICBSLP), Sylhet, Bangladesh, pp. 1-6. https://doi.org/10.1109/ISESD56103.2022.9980803

[22] De Andrade, D.C., Leo, S., Viana, M.L.D.S., Bernkopf, C. (2018). A neural attention model for speech command recognition. arXiv Preprint, arXiv:1808.08929. https://doi.org/10.48550/arXiv.1808.08929

[23] Haque, M.A., Verma, A., Alex, J.S.R., Venkatesan, N. (2020). Experimental evaluation of CNN architecture for speech recognition. In First International Conference on Sustainable Technologies for Computational Intelligence: Proceedings of ICTSCI 2019, Singapore, pp. 507-514. https://doi.org/10.1007/978-981-15-0029-9_40

[24] El Ouahabi, S., Atounti, M., Bellouki, M. (2020). Optimal parameters selected for automatic recognition of spoken Amazigh digits and letters using Hidden Markov Model Toolkit. International Journal of Speech Technology, 23: 861-871. https://doi.org/10.1007/s10772-020-09762-3

[25] Lyashenko, V., Laariedh, F., Sotnik, S., Ayaz, A.M. (2021). Recognition of voice commands based on neural network. TEM Journal, 2: 583-591.

[26] ElMaghraby, E.E., Gody, A.M., Farouk, M.H. (2020). Noise-robust speech recognition system based on multimodal audio-visual approach using different deep learning classification techniques. The Egyptian Journal of Language Engineering, 7(1): 27-42. https://doi.org/10.21608/ejle.2020.22022.1002

[27] Elnagar, A., Yagi, S.M., Nassif, A.B., Shahin, I., Salloum, S.A. (2021). Systematic literature review of dialectal Arabic: Identification and detection. IEEE Access, 9: 31010-31042. https://doi.org/10.1109/ACCESS.2021.3059504

[28] Alsharhan, E., Ramsay, A. (2020). Investigating the effects of gender, dialect, and training size on the performance of Arabic speech recognition. Language Resources and Evaluation, 54(4): 975-998. https://doi.org/10.1007/s10579-020-09505-5

[29] Masmoudi, A., Bougares, F., Ellouze, M., Estève, Y., Belguith, L. (2018). Automatic speech recognition system for Tunisian dialect. Language Resources and Evaluation, 52: 249-267. https://doi.org/10.1007/s10579-017-9402-y

[30] Hassine, M., Boussaid, L., Massaoud, H. (2018). Tunisian dialect recognition based on hybrid techniques. The International Arab Journal of Information Technology, 15(1): 58-65.

[31] Alasadi, A.A., Aldhayni, T.H., Deshmukh, R.R., Alahmadi, A.H., Alshebami, A.S. (2020). Efficient feature extraction algorithms to develop an Arabic speech recognition system. Engineering, Technology & Applied Science Research, 10(2): 5547-5553.

[32] Mouaz, B., Abderrahim, B.H., Abdelmajid, E. (2019). Speech recognition of Moroccan dialect using hidden Markov models. Procedia Computer Science, 151: 985-991. https://doi.org/10.1016/j.procs.2019.04.138

[33] Setianingrum, A.H., Hulliyah, K., Amrilla, M.F. (2023). Speech recognition of Sundanese dialect using convolutional neural network method with Mel-spectrogram feature extraction. In 2023 11th International Conference on Cyber and IT Service Management (CITSM), Makassar, Indonesia, pp. 1-5. https://doi.org/10.1109/CITSM60085.2023.10455447

[34] Çayır, A.N., Navruz, T.S. (2021). Effect of dataset size on deep learning in voice recognition. In 2021 3rd International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA), Makassar, Indonesia, pp. 1-5. https://doi.org/10.1109/HORA52670.2021.9461395

[35] Noori, H. (2025). Enhancing automatic recognition of isolated Arabic speech using artificial intelligence techniques: A systematic review. International Journal of Engineering Trends and Technology, 73(3): 212-229. https://doi.org/10.14445/22315381/IJETT-V73I3P116

[36] Bouchelligua, W., Al-Dayil, R., Algaith, A. (2025). Effective data augmentation techniques for Arabic speech emotion recognition using convolutional neural networks. Applied Sciences, 15(4): 2114. https://doi.org/10.3390/app15042114

[37] Turkey, F.A., Beddu, S.B., Al-Hubboubi, S.K., Fawzi, N.M. (2023). Elevated temperature effects on geo-polymer concrete: An experimental and numerical review study. Annales de Chimie Science des Matériaux, 47(5): 325-340. https://doi.org/10.18280/acsm.470507