Clean Energy and Smart Technology
Volume 04, Number 02, April 2026
DOI : https://doi.org/10.58641/
e-ISSN : 2964-2647

IMPLEMENTATION OF CONVOLUTIONAL NEURAL NETWORK (CNN) ALGORITHM IN MOBILE APPLICATION-BASED VOICE EMOTION CLASSIFICATION SYSTEM

Naufal Ammar Raihan[1)*], Muhamad Fuat Asnawi[2)], Iman Ahmad Ihsannuddin[3)], Nahar Mardiyantoro[4)], Muhammad Alif Muwafiq Baihaqy[5)]

[1)2)3)4)5)] Universitas Sains Al-Qur'an, Indonesia
[1)]naufalammar0909@gmail.com, [2)]fuatasnawi@unsiq.ac.id, [3)]ahmadihsan@unsiq.ac.id, [4)]nahar@unsiq.ac.id, [5)]aviqbaihaqy@gmail.com

*naufalammar0909@gmail.com

Submitted : 6 April 2026 | **Accepted** : 29 April 2026 | **Published** : 30 April 2026

Abstract: The ability of machines to recognize emotions from voice is known as Speech Emotion Recognition (SER). This study developed a voice emotion classification system using a Convolutional Neural Network (CNN) and implemented it in the form of an Android mobile application. The main problem raised is how to recognize human emotions through voice signals accurately, efficiently, and in real-time on mobile devices. The study was conducted with two training stages, namely pre-training using the RAVDESS dataset and fine-tuning with the IndoWaveSentiment dataset. Audio data was converted into a 128×128×1 Mel-spectrogram to be input to the CNN. The CNN model consists of three convolution and pooling blocks, as well as dense and softmax layers. After training, the model was converted to TensorFlow Lite format and integrated with the Android application through a client-server architecture using Flask. The test results showed that the system was able to recognize neutral, happy, disappointed, and surprised emotions with a high level of accuracy both on test data but not as good on live recorded voice. The system also features a SQLite-based history feature. Test results showed 96% accuracy on external test data and 55% on live recorded audio, with an average accuracy of 75.5%. This indicates the model performs very well in structured conditions, but still needs improvement for real-world input.

Keywords: CNN, voice emotion classification, Mel-spectrogram, TensorFlow Lite, Android

1. INTRODUCTION

The ability to recognize and respond to emotions is a crucial aspect of human communication. Emotions are conveyed not only through facial expressions and body language, but also through voice. The human voice conveys a variety of affective information through acoustic and prosodic characteristics such as pitch, rhythm, stress, and intonation. For example, a rising vocal frequency can indicate anger or joy, while a flat and low pitch is often associated with feelings of sadness or disappointment. (Wijaya et al., 2017). The ability of machines to recognize

Clean Energy and Smart Technology
Volume 04, Number 02, April 2026
DOI : https://doi.org/10.58641/
e-ISSN : 2964-2647

emotions from voice is known as Speech Emotion Recognition (SER). SER aims to identify emotions such as happiness, disappointment, or surprise from human voice signals. (Bansal & Kaur, 2024). This technology has great potential in various fields, including customer service, virtual assistants, education, and mental health. Its application allows interactions between humans and machines to become more natural and adaptive to the user's emotional state. Advances in deep learning, particularly Convolutional Neural Network (CNN) architectures, have driven improvements in the performance of SER systems. CNNs are capable of recognizing visual patterns in speech signal representations such as Mel spectrograms, which combine time and frequency information. A Mel spectrogram is a visual representation of a speech signal in the time-frequency domain using the Mel scale, a scale that mimics the way humans hear frequencies. (Sharan et al., n.d.) Previous research has shown that CNNs can extract emotional features from audio data with a good degree of accuracy. However, implementing CNNs directly on mobile devices still faces significant challenges, primarily related to limited computing power and model size. To address this, the TensorFlow Lite (TFLite) approach is used, a lightweight version of TensorFlow specifically designed to run efficiently on Android devices. (Aftab et al., 2021)

By combining transfer learning from large datasets like RAVDESS for pre-training and fine-tuning on local datasets like IndoWaveSentiment, the system achieves a balance between accuracy and efficiency. This system is designed to process speech data, extract Mel spectrograms, and perform emotion classification using a 2D CNN model converted to TFLite format. Furthermore, the system is integrated into an Android application using a client-server architecture, where the inference process is performed on the Flask server, while the Android application is only responsible for sending sound files and receiving prediction results. This approach allows the system to operate in real time and is lightweight on the client side. The application also features a SQLite-based history feature to store previous emotion analysis results locally. Several previous studies have addressed the topic of deep learning-based voice emotion recognition, specifically CNNs. For example, a study by Yusuf & Prasetio (2024) applied a denoising method using spectral gates to improve the quality of voice input before it was classified by a CNN in the context of stress detection. Although implemented on Android, this study focused only on one type of emotion (stress) with an accuracy of around 60%, without exploring other emotions.

Furthermore, Permana & Prasetio (2024) combined the Spectral Contrast method with CNNs to detect voice-based stress on Android. The results showed 87.5% accuracy, but they only focused on one type of emotion and did not compare the effectiveness of different feature extraction methods. Research by Rismanto & Handayani (2025) examined voice emotion classification using CNNs and recorded a training accuracy of 75.85%, but only 51.64% on test data, indicating potential overfitting and a lack of generalizability to real-world conditions. Based on the shortcomings of previous studies, this study takes a different approach: using Mel-spectrogram as a single feature for efficiency, training the model incrementally through transfer learning from a foreign dataset (RAVDESS) and fine-tuning on a local dataset (IndoWaveSentiment). This study also classifies various types of emotions (neutral, happy, disappointed, surprised) and implements the model in a lightweight and responsive client-server system for an Android application. Based on this background, this study aims to develop a Mel-spectrogram-based CNN model for detecting voice emotions, evaluate the accuracy of the CNN model in recognizing voice emotions in Indonesian, and integrate the CNN model into a Kotlin-based Android application with TensorFlow Lite and a Flask server.

2. METHOD

This research uses a Research and Development (R&D) approach with quantitative methods. The stages of this research process are as follows:

Clean Energy and Smart Technology
Volume 04, Number 02, April 2026
DOI : https://doi.org/10.58641/
e-ISSN : 2964-2647

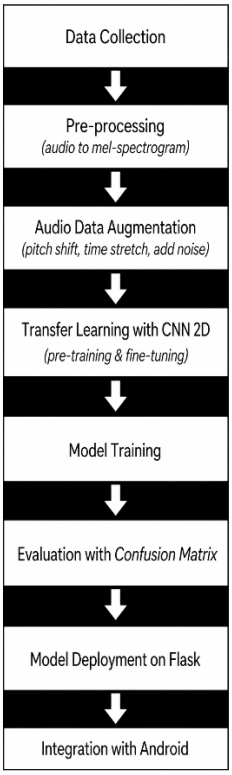

Fig. 1. Research Stages

1. Research Data

The datasets used were RAVDESS (Ryerson Audio-Visual Database of Emotional Speech and Song), consisting of 1,440 English sound files, covering various emotions such as calm, happy, sad, angry, fearful, surprise, and disgust, for pre-training. IndoWaveSentiment, consisting of 300 sounds, divided into five emotion labels: disgust, disappointment, neutral, happy, and surprised. Four categories were selected, with the disgust class removed for fine-tuning.

2. Pre-Processing

The main dataset in this study, IndoWaveSentiment, was divided into two parts: one for model training and one for model validation. The data was 80% training and 20% validation after data aggregation. The audio data was standardized by converting it to mono with a sampling rate of 16 kHz, then converted into a 128x128x1 Mel spectrogram using the Librosa library. The extracted log-mel images were normalized to the maximum value to ensure uniform scaling across files.

3. Data Augmentation

Facing the limited data size in the IndoWaveSentiment dataset, which only contains 48 files per class (60 – 12 for validation), a data augmentation process was performed to increase the variety and quantity of the data without changing the emotion labels. The augmentation techniques used included:

a. Pitch Shifting: Changing the pitch of the voice.
b. Time Stretching: Speeding up or slowing down the voice.
c. Noise Injection: Adding background noise.

Each original file generates three new versions, increasing the data to four versions per file (1 original file + 3 new files).

Clean Energy and Smart Technology
Volume 04, Number 02, April 2026
DOI : https://doi.org/10.58641/
e-ISSN : 2964-2647

Table 1. Dataset Distribution

Category	Training	Validation	Test
Disappointed	154	38	12
Neutral	154	38	12
Happy	154	38	12
Surprised	154	38	12
Total	**616**	**152**	**48**

The total data in the IndoWaveSentiment dataset is:

1. CNN Architecture Design

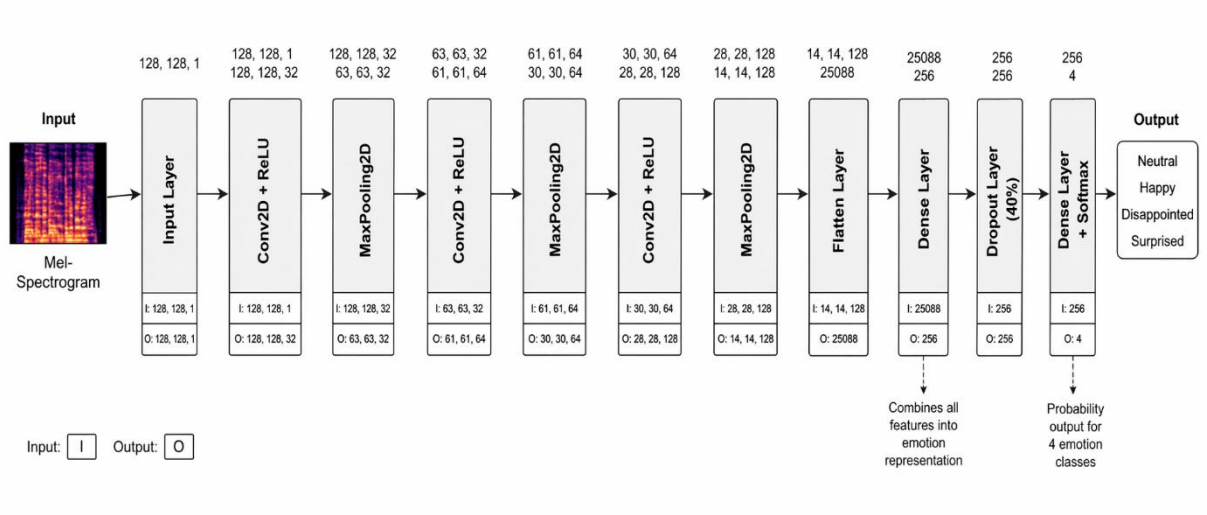

Fig. 2. CNN Layers

The 2D CNN architecture has several layers (Azmi & Defit, 2023).

1. 2D Convolutional Layer (3 layers)
2. 2D Max Pooling Layer (3 layers)
3. Flatten Layer
4. Dense Layer (2 layers)
5. Dropout Layer

The extracted sound data is fed into the Mel-spectrogram starting from the first convolutional layer. The first layer receives input in the form of a 128×128×1 Mel-spectrogram image. This layer applies 32 3×3 filters with a default padding of "valid" and ReLU activation. The convolutional output is then processed through a 2×2 Max Pooling layer to reduce the feature dimensionality and prevent overfitting. This process is repeated by adding a second and third convolutional layer with increasing numbers of filters, namely 64 and 128 filters, respectively. Each convolutional layer is always followed by a pooling layer, resulting in a total of three convolution-pooling blocks. After successfully extracting the important features, the CNN architecture enters the Fully Connected Layer (Dense) section, which consists of 256 neurons and one dropout layer with a ratio of 0.4 to prevent overfitting. Finally, there is an output layer that uses the Softmax activation function to generate the probability of each predicted emotion class. The number of neurons in this layer is adjusted to the number of emotion labels in the dataset, namely four classes: neutral, happy, disappointed, and surprised.

3. RESULT AND DISCUSSION

Data Collection

The data for this study comes from two primary datasets: the public RAVDESS dataset and IndoWaveSentiment, which are Indonesian voice data focused on four main emotional categories frequently encountered in everyday conversation: neutral, happy, disappointed, and surprised. The two datasets were divided into two: training and validation. In the RAVDESS dataset, the data was split 80% for training and 20% for

Clean Energy and Smart Technology
Volume 04, Number 02, April 2026
DOI : https://doi.org/10.58641/
e-ISSN : 2964-2647

validation during the pre-training stage. In the IndoWaveSentiment dataset, the data was split 80% for training and 20% for testing. This splitting was done evenly to prevent overfitting due to disproportionate data. The voice data from the two sources was then processed and organized into a folder structure based on emotion labels. This ensures structured, manageable, and efficient model training. To be used by the CNN in the training process, all voice files must be organized into a folder structure based on their labels.

struktur folder

```
def strukturkan_dataset_ravdess(src_folder, dst_folder):
    os.makedirs(dst_folder, exist_ok=True)
    for file in os.listdir(src_folder):
        if file.endswith('.wav'):
            parts = file.split('-')
            if len(parts) >= 3:
                emosi_kode = parts[2]
                label = ravdess_label_map.get(emosi_kode)
                if label:
                    label_folder = os.path.join(dst_folder, label)
                    os.makedirs(label_folder, exist_ok=True)
                    shutil.copy(os.path.join(src_folder, file), os.path.join(label_folder,
file))
```

Fig. 3. Source Code Dataset Structure

Pre-Processing

After the data is adjusted according to the label and made into one folder with the steps as shown in the picture, then the data is augmented to increase the diversity and quantity of data. Then, it is continued with the Mel-Spectrogram Feature Extraction process and label encoding. Mel Filterbank is calculating 128 mel filters using librosa.feature.melspectrogram. Then the Log Scale process to convert the sound to a log scale, and length normalization if the frame length is less than 128 frames (time steps), zero padding is added. The output of this function is a 2D array of size (128, 128), then a channel is added to make it (128, 128, 1) as the CNN input.

Ekstraksi Mel-Spectogram

```
def extract_mel_spectrogram(file_path, max_len=128):
    y, sr = librosa.load(file_path, sr=16000)
    mel = librosa.feature.melspectrogram(y=y, sr=sr, n_mels=128)
    log_mel = librosa.power_to_db(mel, ref=np.max)
    if log_mel.shape[1] < max_len:
        log_mel = np.pad(log_mel, ((0, 0), (0, max_len - log_mel.shape[1])),
mode='constant')
    else:
        log_mel = log_mel[:, :max_len]
    return log_mel
```

Fig. 4. Mel-Spectrogram Extraction

2D CNN Architecture

The CNN architecture consists of three Conv2D-MaxPooling blocks (with 32, 64, and 128 filters), followed by Flatten → Dense(256, ReLU) → Dropout(0.4) → Dense(4, Softmax) layers. The CNN model receives input in the form of a 128×128×1 Mel-spectrogram image. The first layer consists of 32 3×3 convolutional filters with ReLU activation and "valid" padding, followed by 2×2 Max Pooling for dimensionality reduction. This process is continued with two additional convolutional layers with 64 and 128 filters, respectively, which are also followed by pooling. After three convolution-pooling blocks, the obtained features are processed by a Dense layer with 256 neurons and a dropout of 0.4. The output layer uses Softmax activation with four neurons to map emotions: neutral, happy, disappointed, and surprised.

CNN 2D

```
def build_model(input_shape, num_classes):
    model = tf.keras.Sequential([
        tf.keras.layers.Conv2D(32, (3,3), activation='relu', input_shape=input_shape),
        tf.keras.layers.MaxPooling2D(2,2),
        tf.keras.layers.Conv2D(64, (3,3), activation='relu'),
        tf.keras.layers.MaxPooling2D(2,2),
        tf.keras.layers.Conv2D(128, (3,3), activation='relu'),
        tf.keras.layers.MaxPooling2D(2,2),
        tf.keras.layers.Flatten(),
        tf.keras.layers.Dense(256, activation='relu'),
        tf.keras.layers.Dropout(0.4),
        tf.keras.layers.Dense(num_classes, activation='softmax')
    ])
```

Fig. 5. 2D CNN Code

Clean Energy and Smart Technology
Volume 04, Number 02, April 2026
DOI : https://doi.org/10.58641/
e-ISSN : 2964-2647

Model Training

After successfully creating a 2D CNN model, the next step is training the model using the transfer learning method. Transfer learning is a technique that utilizes models previously trained on large datasets for new tasks with smaller or limited datasets (Rochman & Junaedi, 2020). In this study, the model was trained using two training stages utilizing two existing dataset sources. The first training phase involved pre-training the model using the RAVDESS dataset and fine-tuning using the IndoWaveSentiment dataset.

1. Pre-Training

The first stage is pre-training. The main goal of pre-training is to enable the model to understand general patterns, structures, and relationships in the data, thus gaining basic knowledge (Setiawan et al., 2023). This process is carried out using the RAVDESS dataset. This dataset is used as the initial source for forming the model's initial parameters because it contains a sufficient amount of high-quality data. Before training, the model first goes through a class weighting process to calculate the weights for each class to prevent bias toward the dominant class. The model will undergo 30 full cycles of training on the entire dataset. Each epoch is divided into smaller groups of 32 samples. This speeds up the process and helps stabilize the training. Class weighting is a technique in machine learning used to address the problem of class imbalance in classification data (Akbar & Sanjaya, 2023). The created class weights are used to address the imbalance in the amount of data between emotion classes by giving more weight to classes with less data. The pre-trained model results are stored in “cnn_ravdess_base.keras” for reuse in the fine-tuning stage.

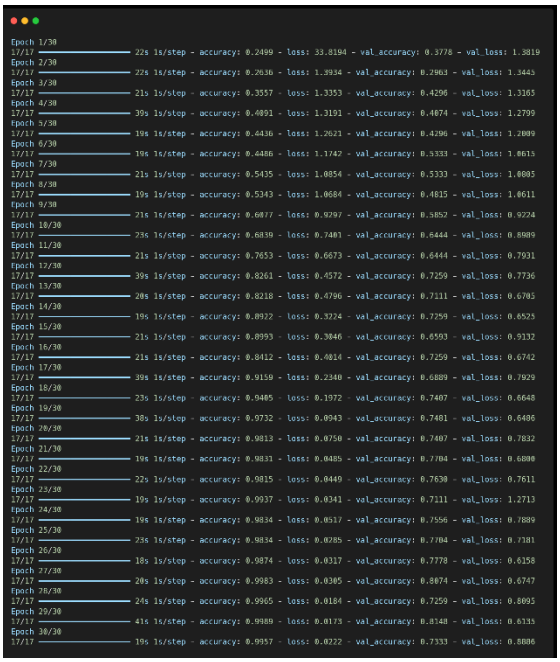

Fig. 6. Pre-Training Results

Based on the training results above, it shows that the model experienced a significant increase in accuracy from epoch to epoch, especially after the 10th epoch. The training accuracy increased to 99.57% at the 30th epoch. Although the validation accuracy was not as high as the training accuracy of only 73.33%, this value is quite stable and indicates that the model has learned to recognize basic emotional patterns quite well.

2. Fine-Tuning

Fine tuning is a model development process that allows for rapid and efficient adaptation to specific needs, leveraging existing knowledge in pre-trained models (Hakim et al., 2024). Fine-tuning in this study aims to adjust models that have learned from international data to be able to recognize the characteristics of Indonesian language voices more specifically. The pre-trained model on the RAVDESS dataset was reloaded from the

Clean Energy and Smart Technology
Volume 04, Number 02, April 2026
DOI : https://doi.org/10.58641/
e-ISSN : 2964-2647

cnn_ravdess_base.keras file. All layers in the model were made retrainable (unfreeze), so that all weights in the model could be adjusted to the characteristics of the IndoWaveSentiment data. The model was recompiled using the Adam Optimizer with a small learning rate of 1e-4 to prevent drastic weight changes during fine-tuning. The categorical crossentropy loss function was also used for multi-class classification. The model was trained using augmented voice data for 20 epochs with a batch size of 16, and 10% of the training data was used for internal validation.

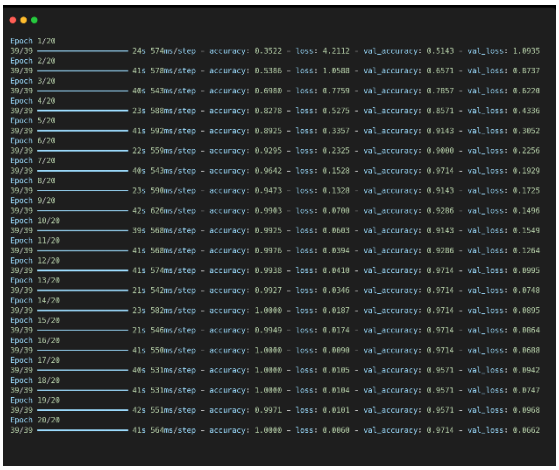

Fig. 7. Fine-Tuning Training Results

The training results show that in the initial epochs, the model was still in the learning stage with a training accuracy of 35% and a loss of 4.2. However, since the 5th epoch, the validation accuracy has increased significantly to 91% with a loss of 0.30. In epochs 10–20, the training accuracy reached 100%, while the validation accuracy remained stable in the range of 91%–97%, indicating good model generalization ability.

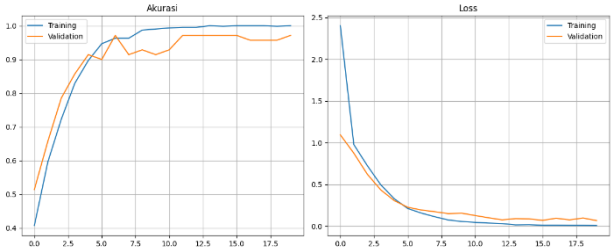

Fig. 8. Accuracy and Loss Graph

Model Evaluation

After the 2D CNN model training process was completed, an evaluation was conducted using test data from the IndoWaveSentiment dataset. The data used as test data constituted 20% of the Indonesian language data before augmentation and was not included in the model training. This test data consisted of 48 files, with 12 files in each class. One of the evaluation methods used was the Confusion Matrix, which is useful for observing the model's performance in classifying each emotion class individually. A confusion matrix is a table used to evaluate the performance of a classification model in machine learning by comparing the model's prediction results to the actual values. This table consists of four main components that represent the combination of predictions and actual conditions. (Bagas Prakosa & Radius Tanone, 2023)

Clean Energy and Smart Technology
Volume 04, Number 02, April 2026
DOI : https://doi.org/10.58641/
e-ISSN : 2964-2647

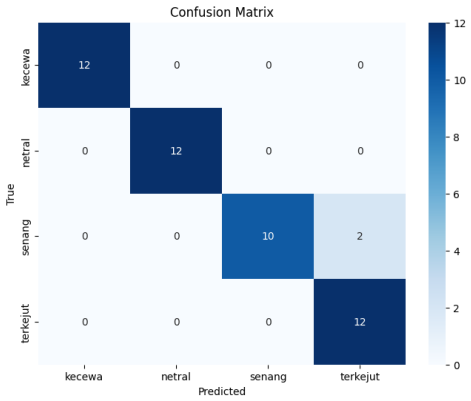

Fig. 9. Confusion Matrix

Based on the confusion matrix results, the model demonstrated an overall accuracy of 96% on the test data. This indicates that the model is able to recognize patterns from Mel-spectrogram features with good consistency on previously unseen data. Of the 48 test data sets, only two files were misclassified, while the other 46 were correctly recognized by the model. Precision, recall, and F1-score values for each class were calculated to evaluate classification performance per emotion category. The results show that the disappointed and neutral classes have very high precision and recall levels, indicating that the model rarely makes errors in detecting these two classes. However, there was a slight decrease in performance for the happy and surprised classes, particularly in recall, indicating that some instances of these two classes were still incorrectly classified as other classes.

Deployment Model

Before the emotion classification model can be integrated into an Android application, it must first be deployed to a Flask-based server. Flask is a web framework written in the Python programming language and is classified as a microframework. Flask is minimalist and does not provide the full range of built-in features found in full-stack frameworks (Santoso et al., 2023). This deployment was necessary because the emotion classification process used in this study requires several digital signal processing operations and complex audio feature extraction in the form of a Mel-Spectrogram. This relies heavily on Python libraries such as Librosa and Numpy (Farid et al., 2023). Therefore, the solution implemented in this study is a client-server approach. The client (Android) only functions to record the user's voice or select a .wav file, then send the file to the Flask server via HTTP request. The server (Flask) receives a .wav file, processes it using librosa to generate a 128×128 Mel-Spectrogram, then classifies the voice emotion using a 2D CNN model in .tflite format, and sends the results back to the Android application.

The Flask server deployment aims to provide an endpoint (API) for the application to send the voice file, perform inference using the .tflite-based 2D CNN model, and return the predicted emotion label and confidence score in JSON format. After the model is successfully deployed as an API using Flask, the next step is to integrate the Android application to perform online emotion prediction. In this integration, the application acts as a client, sending an audio file and receiving the prediction results from the server. Communication is done through the Retrofit library with an Ngrok address as the base URL. The .wav audio file is sent to the /predict endpoint using the POST method in multipart/form-data format. The Flask server then extracts features, runs the TFLite model, and returns the prediction results in JSON format for display in the application.

Application Development and Testing

The application was developed using Kotlin in Android Studio and serves as a client interface in a CNN-based voice emotion classification system. The application supports two input methods: direct voice recording via microphone and selecting a .wav file from device storage. Before being sent to the server, the sound can be replayed to ensure input quality. The file transfer process is carried out via the HTTP protocol using Retrofit, which is then forwarded to the Flask server for classification. The prediction results, in the form of emotion labels and confidence scores, are displayed to the user in a simple and informative interface. Each prediction result is also stored locally using SQLite, allowing users to review the history of previously analyzed emotions. With this client-server architecture, the application remains lightweight and can run in real-time on various Android devices.

Clean Energy and Smart Technology
Volume 04, Number 02, April 2026
DOI : https://doi.org/10.58641/
e-ISSN : 2964-2647

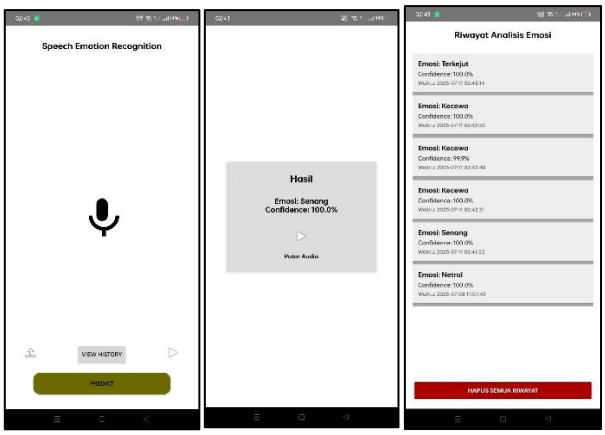

Fig. 10. Android Application View

The created application was tested using two methods: fetching files from storage and recording audio directly. The audio files tested using the fetch from storage feature were test data that were not included in the model training process. The application test results are as follows:

a. Audio from Storage

Table 2. Application Testing from Storage

No.	Audio File	Prediction	Actual	Confidence
1	09-01-01-01.wav	Neutral	Neutral	99.95%
2	10-01-01-01.wav	Neutral	Neutral	99.98%
3	09-02-01-01.wav	Happy	Happy	91.66%
4	10-02-02-02.wav	Happy	Happy	57.89%
5	09-03-01-01.wav	Surprised	Surprised	99.94%
6	10-03-02-01.wav	Surprised	Surprised	86.12%
7	09-05-01-01.wav	Disappointed	Disappointed	99.99%
8	10-05-02-01.wav	Disappointed	Disappointed	99.98%

b. b. Sound from live recording

Table 3. Application Testing from Recordings

No.	Audio File	Prediction	Actual	Confidence
1	Neutral Trial 1	Neutral	Neutral	89.4%
2	Neutral Trial 2	Disappointed	Neutral	88.3%
3	Neutral Trial 3	Neutral	Neutral	56.5%
4	Neutral Trial 4	Neutral	Neutral	72.6%
5	Neutral Trial 5	Happy	Neutral	65.5%
6	Happy Trial 1	Neutral	Happy	99.4%
7	Happy Trial 2	Disappointed	Happy	55.9%

Clean Energy and Smart Technology
Volume 04, Number 02, April 2026
DOI : https://doi.org/10.58641/
e-ISSN : 2964-2647

No.	Audio File	Prediction	Actual	Confidence
8	Happy Trial 3	Disappointed	Happy	70.7%
9	Happy Trial 4	Disappointed	Happy	50.5%
10	Happy Trial 5	Neutral	Happy	60.8%
11	Surprised Trial 1	Surprised	Surprised	99.88%
12	Surprised Trial 2	Surprised	Surprised	100%
13	Surprised Trial 3	Surprised	Surprised	99.6%
14	Surprised Trial 4	Surprised	Surprised	96.2%
15	Surprised Trial 5	Neutral	Surprised	55.1%
16	Disappointed Trial 1	Disappointed	Disappointed	99.9%
17	Disappointed Trial 2	Disappointed	Disappointed	99.8%
18	Disappointed Trial 3	Disappointed	Disappointed	97.6%
19	Disappointed Trial 4	Disappointed	Disappointed	99.8%
20	Disappointed Trial 5	Neutral	Disappointed	82.3%

Based on the results of model testing using two different input methods, the results showed a fairly high level of performance. The first test was conducted using audio files from test data that were not previously involved in the model training process. Of the 48 test samples, the model successfully classified 46 correctly, achieving an accuracy rate of 96%. Meanwhile, the second test was conducted using live voice recordings via an Android application. Of the 20 samples tested, only 11 were correctly classified, resulting in an accuracy rate of 55%. It can be concluded that the developed 2D CNN model successfully recognized voice emotions with an overall accuracy of 75.5%. The model was able to integrate its capabilities with the test data, but there was a decrease in performance when inputting live voice recordings.

Discussion

The accuracy of the model developed in this study, which was 96% on external test data and around 55% on live voice testing, is still considered quite good. This is consistent with the results of several previous studies, such as those conducted by Rismanto & Handayani (2025), where the CNN model they built experienced a significant decrease in accuracy on test data, from 75.85% during training to only 51.64% on test data. This indicates that the performance of the Speech Emotion Recognition (SER) model does tend to decline when tested using real or live voice recordings, compared to test data from structured datasets. (Rismanto & Handayani, 2025). The findings of this study indicate that the proposed CNN-based speech emotion recognition system can classify voice emotions effectively under structured testing conditions. The model achieved 96% accuracy on external test data, showing that the Mel-spectrogram representation was able to preserve relevant time–frequency information for emotion classification. This result supports the use of spectrogram-based input in deep learning audio classification, as Mel-spectrograms transform speech signals into two-dimensional representations that allow CNN models to learn spectral and temporal patterns more effectively (McLoughlin et al., 2026).

The high accuracy obtained from test data also demonstrates that the use of pre-training on RAVDESS followed by fine-tuning on IndoWaveSentiment contributed positively to model adaptation. The pre-training stage enabled the CNN model to learn general emotional speech characteristics from a larger dataset, while the fine-tuning stage adjusted the learned parameters to Indonesian voice characteristics. This finding is in line with recent developments in speech emotion recognition, where transfer learning and multi-source dataset adaptation are considered important strategies for improving model generalization across different speech conditions and emotional categories (Zhao et al., 2026). However, the decrease in accuracy from 96% on stored test data to 55% on live recorded voice indicates that the model still faces challenges in real-world deployment. Live recordings are generally affected by uncontrolled factors such as background noise, microphone quality, speaker accent,

Clean Energy and Smart Technology
Volume 04, Number 02, April 2026
DOI : https://doi.org/10.58641/
e-ISSN : 2964-2647

speaking style, distance from the device, and variations in emotional expression. This condition suggests that a model trained on structured datasets may not automatically perform equally well on spontaneous or natural speech. Rathnayake et al. (2026) also emphasized that speech emotion recognition in low-resource language contexts requires careful consideration of cultural, linguistic, and acoustic variations because emotion expression may differ across communities and languages.

The performance gap between dataset-based testing and live-recorded testing shows that data diversity is a crucial factor in developing a robust SER system. Although this study applied pitch shifting, time stretching, and noise injection as data augmentation techniques, the live-recorded results indicate that the augmentation process was not yet sufficient to represent the complexity of real-world Indonesian speech. Nagro (2026) reported that the use of hybrid dataset integration, augmentation techniques such as noise addition and pitch shifting, and loss functions designed for imbalanced data can improve the robustness of deep learning-based SER systems. Therefore, future development of this system should expand the Indonesian emotional voice dataset by involving more speakers, diverse accents, different recording devices, and natural environmental conditions. From the architecture perspective, the use of a 2D CNN with three convolution-pooling blocks proved effective for recognizing emotional patterns from Mel-spectrogram images. Nevertheless, the misclassification of several happy and neutral live recordings indicates that certain emotional classes may share similar acoustic characteristics, especially when expressed with low intensity. This suggests that relying only on Mel-spectrogram features may not fully capture all emotional cues in speech. Future studies may consider combining Mel-spectrogram with MFCC, chroma, pitch, energy, or prosodic features to improve class separability. Recent studies on SER have shown that feature fusion can enhance recognition performance because different acoustic features represent complementary aspects of speech emotion (Nagro, 2026; McLoughlin et al., 2026).

The implementation of the model into an Android-based application also provides practical value. The client-server architecture using Flask allows computationally intensive audio processing to be performed on the server, while the Android application remains lightweight and responsive. This design is suitable for early-stage mobile SER prototypes, especially when the mobile device has limited computational resources. However, because the inference process depends on server communication, the system still requires a stable internet connection. For future development, lighter architectures such as MobileNet, lightweight CNN, or model quantization can be explored to support on-device inference and reduce dependency on external servers. Overall, this study contributes to the development of Indonesian voice emotion recognition by combining Mel-spectrogram-based CNN, transfer learning, TensorFlow Lite conversion, and Android application integration. The results show that the model performs very well in structured testing but still requires improvement for real-world voice input. Therefore, the main contribution of this research lies not only in the achieved accuracy, but also in demonstrating the technical feasibility and limitations of deploying CNN-based SER into a mobile application. This finding is relevant for future applications in human–computer interaction, digital learning, virtual assistants, customer service, and mental health monitoring, where emotional awareness can improve the quality of interaction between users and intelligent systems.

4. CONCLUSION

Conclusion

This research successfully developed a Convolutional Neural Network (CNN)-based voice emotion classification system with Mel-spectrogram input, which was integrated into an Android application through a client-server architecture using Flask and TensorFlow Lite. The CNN model was trained incrementally using the RAVDESS and IndoWaveSentiment datasets. The test results showed 96% accuracy on external test data and 55% on live recorded voice, with an average accuracy of 75.5%. This indicates that the model performs very well in structured conditions, but still needs improvement for real-world input. The developed Android application successfully supports recording features, real-time emotion prediction, and SQLite-based history, making this system functional and ready to be used as an initial prototype for mobile-based SER. Several recommendations are recommended for further development of this voice emotion classification system. First, improving accuracy on real-world voice input could be achieved by increasing the amount and variety of training data, particularly from natural recordings with different accents and environmental conditions. Second, integrating advanced training techniques on real-world voices and using lighter model architectures such as MobileNet could improve the system's efficiency and accuracy on mobile devices. Furthermore, adding emotion classes such as anger or fear would broaden the classification scope. Cloud backend integration could also enable cross-device history

Clean Energy and Smart Technology
Volume 04, Number 02, April 2026
DOI : https://doi.org/10.58641/
e-ISSN : 2964-2647

synchronization and support long-term emotion tracking. Finally, in-app graphical visualization of emotion trends could be an additional feature that helps users reflect more deeply on their emotional states.

5. REFERENCES

Aftab, A., Morsali, A., Ghaemmaghami, S., & Champagne, B. (2021). *LIGHT-SERNET: A LIGHTWEIGHT FULLY CONVOLUTIONAL NEURAL NETWORK FOR SPEECH EMOTION RECOGNITION*.

Akbar, H., & Sanjaya, W. K. (2023). Kajian Performa Metode Class Weight Random Forest pada Klasifikasi Imbalance Data Kelas Curah Hujan. *Jurnal Sains, Nalar, dan Aplikasi Teknologi Informasi*, *3*(1). https://doi.org/10.20885/snati.v3i1.30

Azmi, K., & Defit, S. (2023). *Implementasi Convolutional Neural Network (CNN) Untuk Klasifikasi Batik Tanah Liat Sumatera Barat*. *16*(1), 2023.

Bagas Prakosa, A., & Radius Tanone, dan. (2023). IMPLEMENTASI MODEL DEEP LEARNING CONVOLUTIONAL NEURAL NETWORK (CNN) PADA CITRA PENYAKIT DAUN JAGUNG UNTUK KLASIFIKASI PENYAKIT TANAMAN. Dalam *Jurnal Pendidikan Teknologi Informasi (JUKANTI)* (Nomor 6). https://www.kaggle.com/datasets/n

Bansal, S., & Kaur, R. (2024). *THE SOUND OF EMOTION: CNN-BASED SPEECH EMOTION RECOGNITION FOR REAL-WORLD APPLICATIONS*. https://doi.org/10.56726/IRJMETS56879

Farid, M. N., Rahman, A. F., Wicaksono, H., & Kalimantan, I. T. (2023). *Jurnal Sistim Informasi dan Teknologi https://jsisfotek.org/index.php Analisis Pengaruh Kombinasi Fitur Spektral terhadap Tingkat Akurasi Speech Emotion Recognition*. *5*(2). https://doi.org/10.37034/jsisfotek.v5i1.234

Hakim, S. A., Ubaidillah, M., Ramadhan, A. R., Zulvia, R., Hawari, A., Rizky, A. B., Lutfi, R., Tsania, P., Hermanto, M., Yudistira, N., & Korespondensi, P. (2024). *KLASIFIKASI CITRA GENERASI ARTIFICIAL INTELLIGENCE MENGGUNAKAN METODE FINE TUNING PADA RESIDUAL NETWORK AI GENERATED IMAGE CLASSIFICATION USING FINE TUNING ON RESIDUAL NETWORK*. *11*(3), 655–666. https://doi.org/10.25126/jtiik.938118

McLoughlin, I., Pham, L., Song, Y., Miao, X. X., Phan, H., Cai, P., Gu, Q., Nan, J., Song, H., & Soh, D. (2026). Spectrogram features for audio and speech analysis. *Applied Sciences, 16*(2), 572. doi:10.3390/app16020572

Nagro, S. (2026). Optimization of speech emotion recognition using hybrid dataset integration and deep learning-based feature fusion with a novel balanced focal entropy loss. *Scientific Reports*. doi:10.1038/s41598-026-48975-5

Rathnayake, H., James, J., Leoni, G., Nicholas, A., Watson, C., & Keegan, P. (2026). A review on speech emotion recognition for low-resource and Indigenous languages. *Speech Communication, 176*, 103342. doi:10.1016/j.specom.2025.103342

Rismanto, M. E. P., & Handayani, I. (2025). Klasifikasi Emosi Berdasarkan Suara dengan Metode Convolutional Neural Network. *Jurnal Informatika Universitas Pamulang*, *9*(4), 163–171. https://doi.org/10.32493/informatika.v9i4.45236

Rochman, F., & Junaedi, H. (2020). *IMPLEMENTASI TRANSFER LEARNING UNTUK IDENTIFIKASI ORDO TUMBUHAN MELALUI DAUN*.

Santoso, B. B., Ocsa, P., & Saian, N. (2023). Implementasi Flask Framework pada Development Modul Reporting Aplikasi Sistem Informasi Helpdesk di PT.XYZ). *Jurnal Teknologi Informasi dan Komunikasi)*, *7*(2), 2023. https://doi.org/10.35870/jti

Setiawan, D., Ayu Dewi Karuniawati, E., Imelda Janty, S., & Bintan Cakrawala, P. (2023). *Peran Chat Gpt (Generative Pre-Training Transformer) Dalam Implementasi Ditinjau Dari Dataset*.

Sharan, R. V, Mascolo, C., & Schuller, B. W. (t.t.). *Emotion Recognition from Speech Signals by Mel-Spectrogram and a CNN-RNN*.

Wijaya, N., Soesanti, I., & Firmansyah, E. (2017). *Prosiding Seminar Nasional Teknologi dan Informatika, 2017 : Kudus, 25 Juli 2017*. Badan Penerbit Universitas Muria Kudus.

Wu, X., Lee, T., Lilhore, U. K., Simaiya, S., Alroobaea, R., Baqasah, A. M., Alsafyani, M., & Tekeste, L. G. (2026). A deep learning approach to emotionally intelligent AI for improved learning outcomes. *Scientific Reports, 16*, 7431. doi:10.1038/s41598-026-37750-1

Zhao, X., Liu, J., & Lin, L. (2026). Deep cross-modal affective memory networks with adaptive multi-source heterogeneous transfer learning in speech emotion recognition. *Scientific Reports*. doi:10.1038/s41598-026-47200-7