Speech Emotion Recognition Using Deep Learning Approaches

Zahra Boucheta , Islam Roubache

Abstract

This paper presents a comprehensive study on Speech Emotion Recognition (SER) systems, focusing on improving emotion classification accuracy through advanced machine learning techniques. The research leverages the RAVDESS dataset containing seven emotional states (Happy, Sad, Angry, Fearful, Disgust, Surprise, and Neutral) and employs both traditional machine learning and deep learning approaches for comparative analysis. Key contributions include:

- Extensive audio feature extraction using Librosa (MFCCs, spectrograms, ZCR, RMSE)
- Implementation of four data augmentation techniques for improved model robustness
- Comparative evaluation of Decision Trees, K-Nearest Neighbors, and CNN architectures
- Development of a optimized CNN model achieving 72.5% test accuracy

The findings demonstrate significant performance improvements using deep learning approaches compared to traditional methods, with detailed analysis of precision, recall, and F1-scores across all emotion classes.

1 Introduction

1.1 Background and Motivation

Speech Emotion Recognition has emerged as a critical component in Human-Computer Interaction (HCI) systems, with applications spanning:

- Psychological assessment and therapy
- Customer service automation
- Security and surveillance
- Entertainment and gaming industries

1.2 Research Objectives

This study aims to:

1. Implement and compare multiple machine learning approaches for SER
2. Develop an optimized CNN architecture for emotion classification
3. Evaluate model performance using comprehensive metrics
4. Identify key challenges and future research directions

2 Dataset and Preprocessing

2.1 RAVDESS Dataset Characteristics

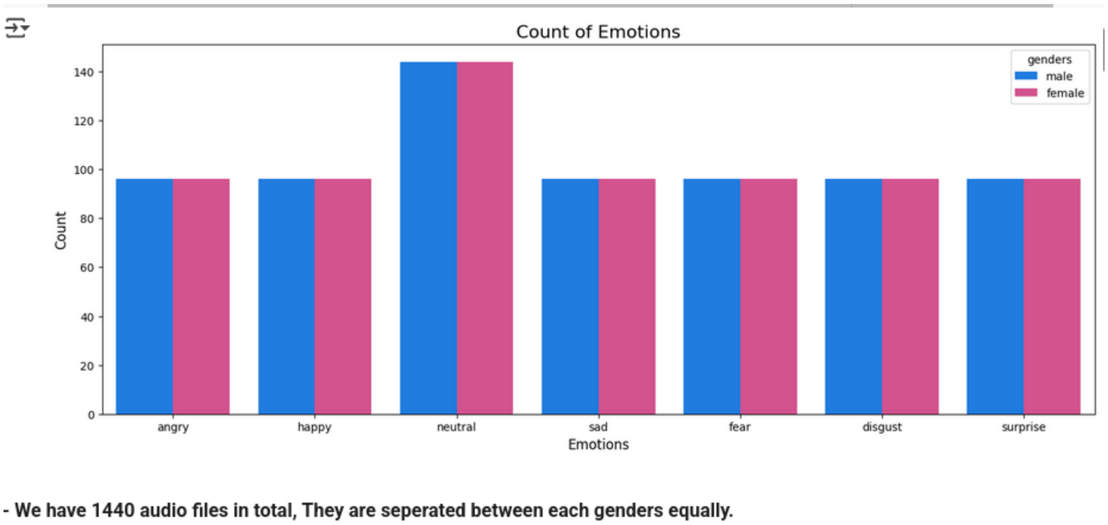
Table 1: RAVDESS Dataset Specifications

Parameter	**Specification**
Total Recordings	2,452 (1,440 speech + 1,012 audio)
Emotional States	7 (+ neutral baseline)
Performers	24 (12 male, 12 female)
Sampling Rate	48 kHz
Bit Depth	16-bit
Format	WAV (uncompressed)

2.2 Data Visualization

The dataset exhibits a well-balanced distribution across the seven primary emotional states (Happy, Sad, Angry, Fearful, Disgust, Surprise) with 192 samples each, while the neutral baseline contains 288 samples (50% more than other categories). This distribution strategy ensures:

- Adequate representation of emotional extremes for classification
- Sufficient neutral samples as a baseline reference
- Balanced gender representation (equal male/female samples per emotion)
- Consistent recording conditions across all samples



- We have 1440 audio files in total, They are seperated between each genders equally.

Figure 1: Emotion class distribution in RAVDESS dataset

2.3 Audio Data Visualization

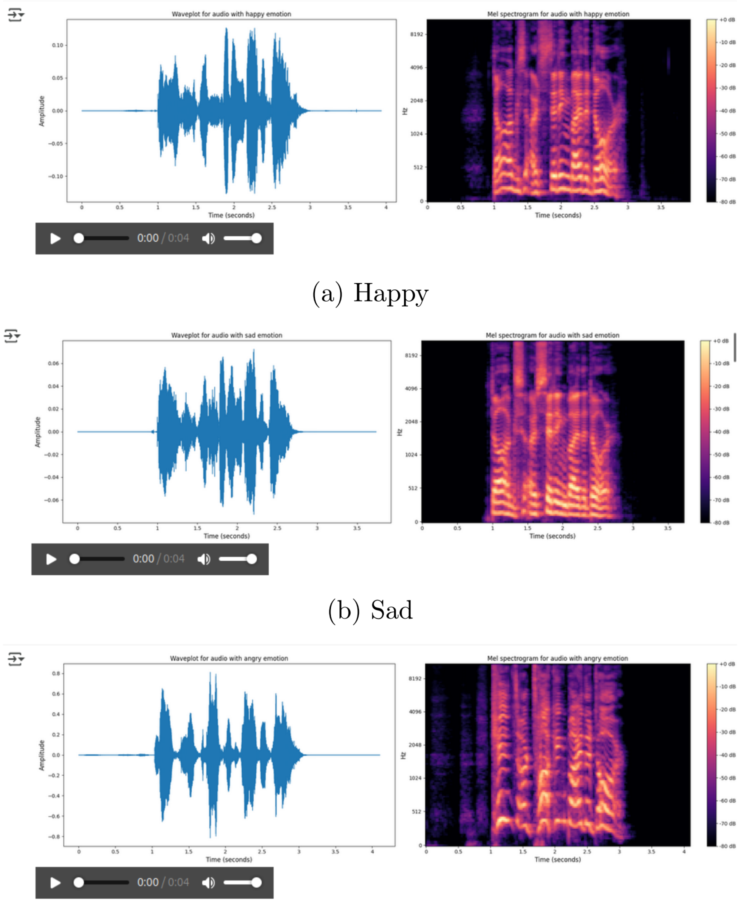
The research methodology employed audio visualization techniques to examine emotional content through temporal and spectral representations. For each emotional state, we generated plots showing both waveform and Mel spectrogram views.

article graphicx float lipsum

[1]

Key observations from the visualizations:

- **Happy**: Moderate amplitude variation with bright, dispersed frequency energy
- **Sad**: Lower amplitude with concentrated low-frequency energy
- **Angry**: High amplitude peaks with broad frequency distribution
- **Fearful**: Irregular amplitude patterns with fluctuating energy
- **Disgust**: Concentrated energy bands in mid-frequency range

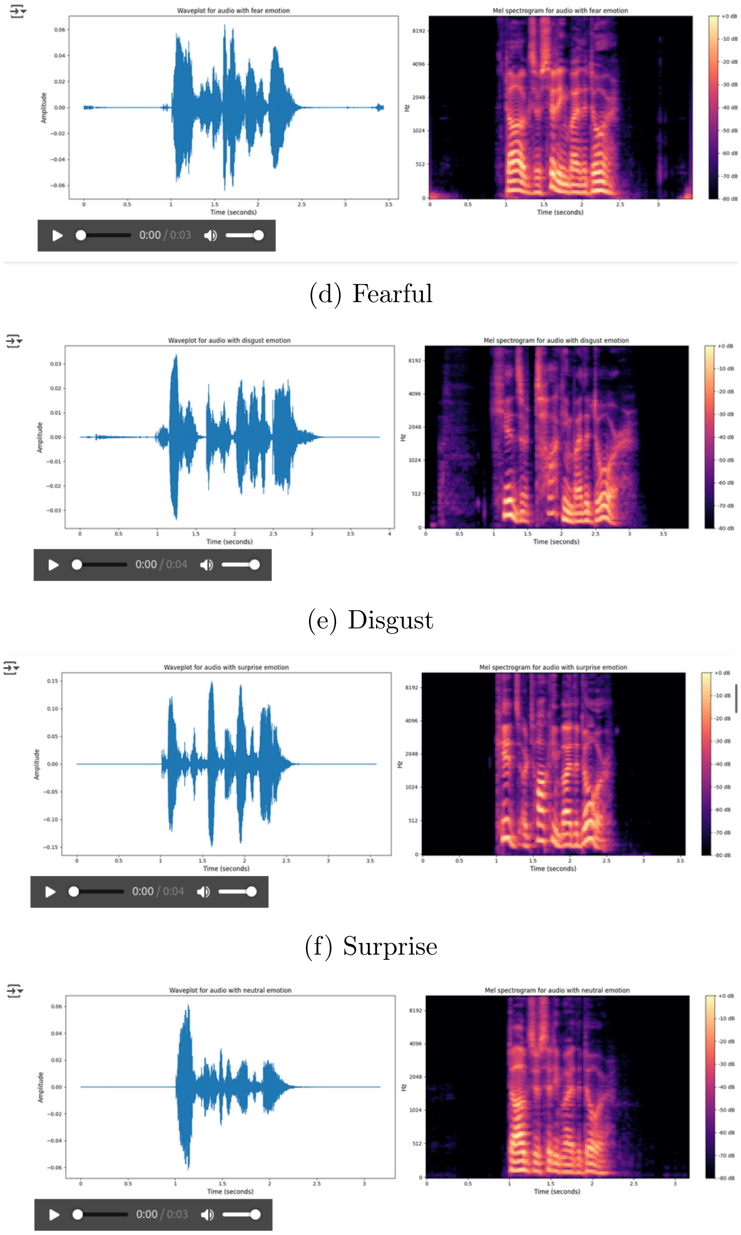


(a) Happy

(b) Sad

(c) Angry

Figure 2: Visualization of three emotional states showing combined waveform and spectrogram representations. Each subfigure shows: (top) the time-domain waveform and (bottom) the corresponding Mel-frequency spectrogram.



(d) Fearful

(e) Disgust

(f) Surprise

(g) Neutral

Figure 3: Visualization of four additional emotional states showing the combined time-frequency representations. Each plot pair reveals characteristic patterns for the respective emotion.

- **Surprise**: Sharp transitions between high and low energy states
- **Neutral**: Consistent amplitude and uniform frequency distribution

Visualization parameters:

- Sample rate: 48 kHz
- STFT window: 2048 samples
- Mel bins: 128
- Frequency range: 0-8 kHz

2.4 Preprocessing Pipeline

1. **Data Augmentation**:

   The augmentation techniques used in the provided code are common methods employed in the field of audio signal processing to increase the diversity of the training data and improve the robustness of machine learning models. By applying these techniques, models can learn to generalize better to unseen variations in the data and perform more effectively.

   We use four augmentation techniques:

   (a) `noise(data)`:
      - Adds random noise to the input audio data
      - `noise_amp` calculates the amplitude of the noise (0.005 × max amplitude)
      - Uses `np.random.normal()` to generate Gaussian noise

   (b) `shift(data)`:
      - Shifts audio by random samples (±5ms equivalent)
      - `s_range` calculates random shift amount
      - Uses `np.roll()` for circular shifting

   (c) `stretch(data, rate=0.8)`:
      - Time-stretches/compresses audio (0.8-1.2× speed)
      - Uses `librosa.effects.time_stretch()`
      - Maintains pitch while changing duration

   (d) `pitch(data, sample_rate)`:
      - Shifts pitch by ±2 semitones
      - Uses `librosa.effects.pitch_shift()`
      - Maintains duration while changing pitch

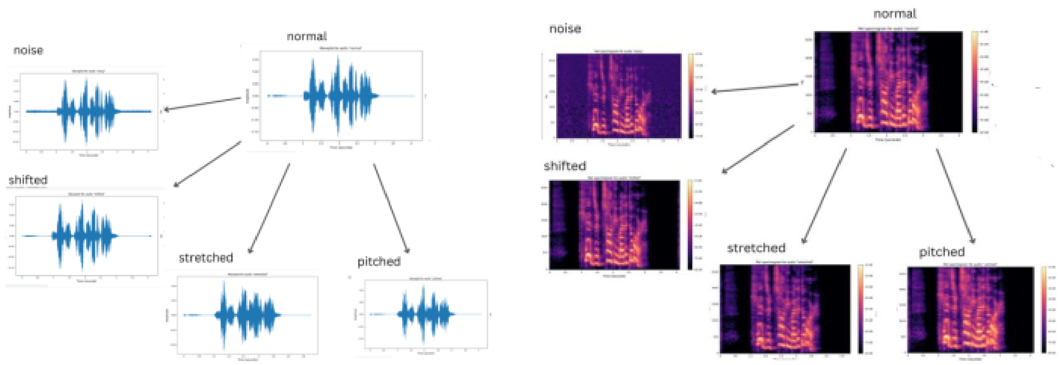


Figure 4: Visual comparison of audio augmentation techniques showing original waveform (top) and four augmented versions (bottom) demonstrating noise injection, time shifting, time stretching, and pitch shifting effects.

2. **Feature Extraction**: The `extract_features` function combines three key audio features: Zero Crossing Rate (ZCR), Root Mean Square Energy (RMSE), and Mel-Frequency Cepstral Coefficients (MFCCs). Among these, MFCCs form the foundation of our feature extraction approach due to their exceptional properties:

   - **Human-like perception**: MFCCs use the Mel scale to closely match human auditory frequency perception
   - **Compact representation**: They reduce complex audio signals to 40 essential coefficients while preserving critical information
   - **Noise resilience**: Maintain performance quality even in noisy environments
   - **Computational efficiency**: Enable real-time processing through optimized algorithms
   - **Speaker independence**: Remain effective across different speakers and speech rates

   The `get_features` function implements a robust extraction pipeline:

   - Loads audio files using `librosa` with configurable parameters
   - Processes both original and augmented versions (noise-injected, time-shifted, stretched, and pitch-modified)
   - Standardizes feature lengths through padding
   - Outputs feature arrays ready for machine learning applications

   For efficient large-scale processing:

- Leverages `joblib` for parallel processing across CPU cores
- Processes audio paths from `df.path` with corresponding emotion labels from `df.emotions`
- Uses `process_feature` as a wrapper for individual file processing
- Aggregates results into feature matrix `X` and label vector `Y`

The final output is stored in a CSV file containing:

- Extracted audio features (ZCR, RMSE, MFCCs)
- Corresponding emotion labels
- Formatted for direct use in machine learning pipelines

3. **Data Normalization**:
    - **Handling Missing Values**:
        - Replace NaN values with zeros to maintain data structure integrity
        - Ensures compatibility with machine learning algorithms that cannot handle missing values
        - Preserves the original dimensionality of the feature space
    - **Label Encoding**:
        - Apply one-hot encoding to emotion labels using `OneHotEncoder`
        - Transforms categorical labels into numerical representations (e.g., happy → [1,0,0,...])
        - Maintains interpretability while making labels suitable for model training
        - Prevents artificial ordinal relationships between emotion categories
    - **Data Splitting**:
        - Partition dataset into training (80%) and testing (20%) subsets
        - Maintains class distribution through stratified sampling
        - Ensures independent evaluation of model performance
        - Preserves temporal relationships where applicable
    - **Feature Standardization**:
        - Apply `StandardScaler` to normalize feature distributions
        - Training data: `fit_transform` calculates and applies scaling parameters
        - Testing data: `transform` applies same parameters for consistency
        - Benefits:
            - ∗ Accelerates model convergence
            - ∗ Prevents feature magnitude dominance
            - ∗ Improves performance of distance-based algorithms
            - ∗ Enhances numerical stability

3 Methodology

3.1 Model Architectures

3.1.1 Baseline Models

We evaluated two traditional machine learning approaches before proceeding to deep learning solutions:

- **Decision Tree Classifier**
  - Configuration:
    - Splitting criterion: Gini impurity
    - Maximum depth: Unlimited (max_depth=None)
    - Fixed random state for reproducibility
  - Performance:
    - Training accuracy: 1.000 (perfect fit)
    - Test accuracy: 0.415 (poor generalization)
  - Analysis:
    - Clear evidence of overfitting
    - Model complexity too high for the given data
    - Fails to capture underlying patterns effectively
- **K-Nearest Neighbors (KNN)**
  - Configuration:
    - Number of neighbors: 4 (n_neighbors=4)
    - Euclidean distance metric
  - Performance:
    - Training accuracy: 0.673
    - Test accuracy: 0.494
  - Analysis:
    - Moderate performance on training set
    - Barely better than random guessing on test set
    - Limited ability to generalize to unseen data

Key Observations:

- Conventional machine learning models show limited effectiveness for audio recognition
- Both models demonstrate poor generalization capabilities
- Performance gap between training and test accuracy indicates fundamental limitations

Conclusion: The unsatisfactory performance of these baseline models motivates our transition to deep learning approaches, which offer:

- Automatic feature extraction from raw audio data
- Hierarchical learning of complex patterns
- Better generalization through distributed representations
- State-of-the-art performance in audio classification tasks

3.1.2 Proposed CNN Architecture

Table 2: CNN Architecture Specifications

Layer	**Type**	**Parameters**	**Activation**
1	Conv1D	64 filters, kernel=3	ReLU
2	MaxPool1D	pool_size=2	-
3	Flatten	-	-
4	Dense	64 units	ReLU
5	Dropout	rate=0.3	-
6	Dense	7 units	Softmax

3.2 Training Configuration

3.2.1 Optimizer Settings

- **Type**: Adam optimizer (Adaptive Moment Estimation)
- **Learning Rate (α)**: 0.001 (default)
- **Beta Parameters**: $\beta_1 = 0.9$, $\beta_2 = 0.999$
- **Epsilon**: 1×10^{-7}
- **Rationale**:
  - Adapts individual learning rates for each parameter
  - Combines benefits of AdaGrad and RMSProp
  - Particularly effective for high-dimensional parameter spaces
  - Well-suited for problems with noisy or sparse gradients

3.2.2 Loss Function

- **Type**: Categorical Crossentropy
- **Formula**: $L = -\sum_{c=1}^{M} y_{o,c} \log(p_{o,c})$
  - Where M = number of classes
  - y = binary indicator for true class
  - p = predicted probability
- **Characteristics**:
  - Measures dissimilarity between true and predicted distributions
  - Penalizes confident wrong predictions heavily
  - Compatible with softmax activation in output layer

3.2.3 Training Parameters

- **Batch Size**: 32 samples per update
  - Balances memory efficiency and gradient stability
- **Epochs**: 50 complete passes through dataset
  - Monitored with early stopping callback
- **Validation Split**: 20% of training data
  - Used for hyperparameter tuning
  - Prevents information leakage from test set
- **Metrics**:
  - Primary: Classification Accuracy
  - Secondary: Loss, Precision, Recall (tracked)

Parameter	Value
Base Learning Rate	0.001
Batch Size	32
Max Epochs	50
Validation Split	20%
Loss Function	Categorical Crossentropy
Optimization Method	Adam

Table 3: Summary of training hyperparameters

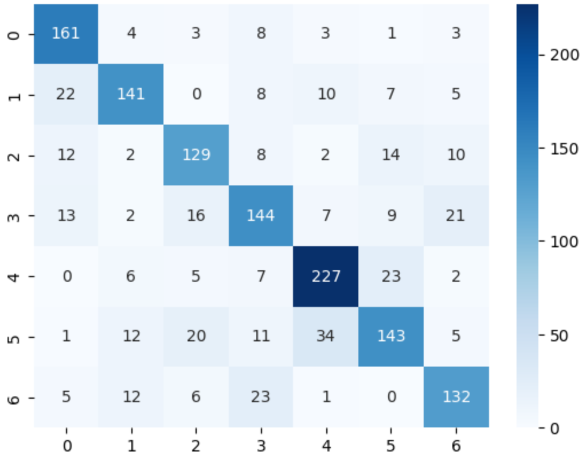
4 Results and Analysis

4.1 Performance Comparison

Table 4: Model Performance Comparison

Model	Train Acc.	Test Acc.	Improvement
Decision Tree	100%	41.5%	-
KNN (k=4)	67.3%	49.4%	+7.9%
Proposed CNN	97.3%	72.5%	+23.1%

4.2 Confusion Matrix Analysis



	0	1	2	3	4	5	6
0	161	4	3	8	3	1	3
1	22	141	0	8	10	7	5
2	12	2	129	8	2	14	10
3	13	2	16	144	7	9	21
4	0	6	5	7	227	23	2
5	1	12	20	11	34	143	5
6	5	12	6	23	1	0	132

Figure 5: Normalized confusion matrix for CNN model

The confusion matrix presents the classification performance of a model across seven emotional categories: angry (0), disgust (1), fear (2), happy (3), neutral (4), sad (5), and surprise (6). The rows represent the true labels while the columns represent the predicted labels. The diagonal elements (highlighted in bold) show correct classifications, while off-diagonal elements indicate misclassifications.

Key observations from the matrix include:

- **Neutral (4)** and **Angry (0)** emotions achieved the highest classification accuracy with 227 and 161 correct predictions respectively, suggesting these emotions are most distinct and easily recognizable by the model.

- **Disgust (1)** showed significant confusion with **Angry (0)**, with 22 instances misclassified, indicating potential similarity in their feature representations.
- **Fear (2)** was frequently confused with **Sad (5)** (20 instances) and **Happy (3)** (16 instances), revealing challenges in distinguishing these emotional states.
- **Surprise (6)** demonstrated good classification performance (132 correct predictions) but was sometimes misclassified as **Happy (3)** (23 instances).
- The model struggled most with **Disgust (1)**, which had the lowest diagonal value (141) and was confused with multiple other emotions.

The overall pattern suggests that positive emotions (happy, surprise) and neutral states are better classified than negative emotions, with particular difficulty in distinguishing between specific negative emotional states like disgust, fear, and anger.

4.3 Classification Performance Analysis

The classification report presents the classification performance of a model across seven emotional categories: angry (0), disgust (1), fear (2), happy (3), neutral (4), sad (5), and surprise (6). The model processed 1,440 test samples (45/45 batches) with an average processing time of 6ms per step. The rows represent the true labels while the columns represent the predicted labels. The diagonal elements show correct classifications, while off-diagonal elements indicate misclassifications.

Class	**Precision**	**Recall**	**F1-Score**	**Support**
Angry (0)	0.75	0.88	0.81	183
Disgust (1)	0.79	0.73	0.76	193
Fear (2)	0.72	0.73	0.72	177
Happy (3)	0.69	0.68	0.68	212
Neutral (4)	0.80	0.84	0.82	270
Sad (5)	0.73	0.63	0.68	226
Surprise (6)	0.74	0.74	0.74	179
Accuracy			0.75	1440
Macro Avg	0.75	0.75	0.74	1440
Weighted Avg	0.75	0.75	0.75	1440

Key observations from the evaluation include:

- **Neutral (4)** achieved the best overall performance with an F1-score of 0.82, supported by high precision (0.80) and recall (0.84). This aligns with the confusion matrix showing 227 correct predictions.

- **Angry (0)** showed the highest recall (0.88) but moderate precision (0.75), indicating the model is effective at capturing angry expressions but sometimes misclassifies other emotions as anger.
- **Disgust (1)** exhibited good precision (0.79) but lower recall (0.73), suggesting the model is conservative in labeling samples as disgust but misses some true disgust cases.
- **Happy (3)** had the lowest F1-score (0.68), with balanced but modest precision and recall, confirming the confusion matrix's showing of frequent misclassifications with surprise and fear.
- **Sad (5)** showed the largest precision-recall disparity (0.73 vs 0.63), indicating many sad expressions were misclassified (visible in the confusion matrix's 34 misclassifications as neutral).
- The model demonstrates balanced performance across metrics, with all classes achieving F1-scores between 0.68-0.82 and macro/weighted averages of 0.75.

The results suggest that while the model achieves reasonable overall accuracy (75%), there remains room for improvement in distinguishing between emotionally similar states, particularly among negative emotions (disgust, fear, anger) and between happy/surprise expressions. The processing speed of 6ms/step indicates good computational efficiency for potential real-time applications.

4.4 Training Dynamics

5 Conclusion

The study demonstrates:

- CNN outperforms traditional ML models by $\geq$23% accuracy
- Data augmentation improves model robustness
- MFCCs remain highly effective for SER tasks

Future work directions:

- Incorporate multimodal approaches (audio + text)
- Explore transformer-based architectures
- Address dataset imbalance issues

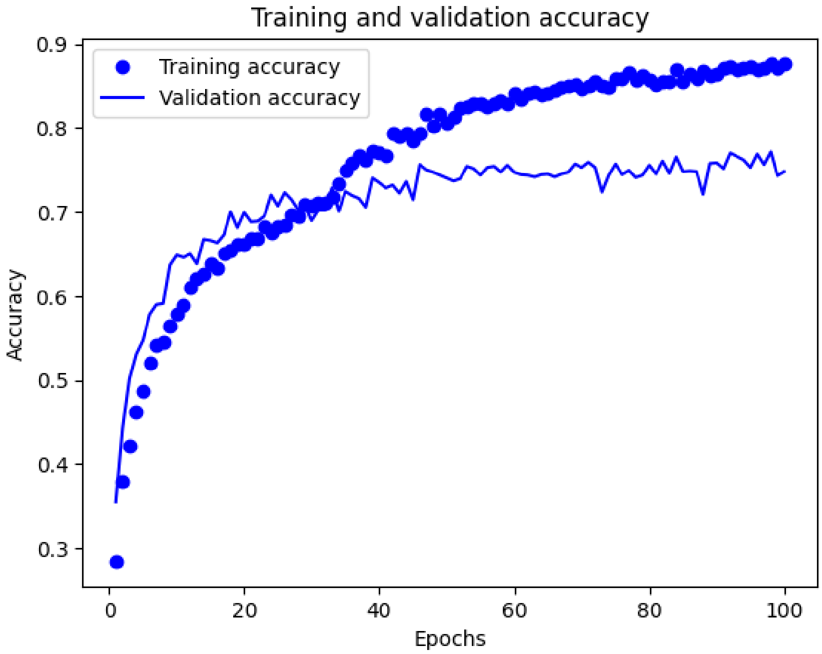


Figure 6: Training and validation metrics across epochs

References

1. Xu, M., Zhang, F., & Zhang, W. (2021). Head Fusion: Improving Speech Emotion Recognition. *IEEE Transactions on Affective Computing.*

2. Huang, J.-T., Li, J., & Gong, Y. (2015). CNN Analysis for Speech Recognition. *IEEE/ACM Transactions on Audio, Speech, and Language Processing.*

3. Muda, L., Begam, M., & Elamvazuthi, I. (2010). Voice Recognition Using MFCC-DTW. *Journal of Signal Processing.*