Speech Emotion Recognition Project Report

1 Introduction

Speech Emotion Recognition (SER) is a process of recognizing human emotions and affective states from speech. This project involves extracting features from audio files, processing these features, and using a neural network to classify the emotions.

2 Data Collection and Preprocessing

2.1 Dataset Description

The dataset consists of two main directories containing audio files: `Audio_Speech_Actors_01-24` and `Audio_Song_Actors_01-2`
Each directory contains subfolders representing different actors, and each subfolder contains `.wav` files of speech or song.

2.2 Feature Extraction

We extracted three main types of features from each audio file:

- **MFCC (Mel-Frequency Cepstral Coefficients)**: Captures the power spectrum of sound.
- **Chroma Features**: Represents the 12 different pitch classes.
- **Mel Spectrogram**: Represents the short-time power spectrum of sound.

The `extract_features` function reads each audio file, computes these features, and stores them in a list.

Listing 1: Feature Extraction Function

```
def extract_features(main_dir, list_of_features):
    features_list = []
    labels = []

    for subfolder in os.listdir(main_dir):
        subfolder_path = os.path.join(main_dir, subfolder)
        if os.path.isdir(subfolder_path):
            for filename in os.listdir(subfolder_path):
                if filename.endswith('.wav'):
                    file_path = os.path.join(subfolder_path, filename)

                    y, sr = librosa.load(file_path)

                    features = []
                    if 'mfcc' in list_of_features:
                        mfcc = librosa.feature.mfcc(y=y, sr=sr, n_mfcc=13)
                        features.extend(mfcc.mean(axis=1))

                    if 'chroma' in list_of_features:
```

```
                stft = np.abs(librosa.stft(y))
                chroma = librosa.feature.chroma_stft(S=stft, sr=sr)
                features.extend(chroma.mean(axis=1))

            if 'melspectrogram' in list_of_features:
                mel_spect = librosa.feature.melspectrogram(y=y, sr=sr)
                mel_spect_db = librosa.power_to_db(mel_spect, ref=np.max)
                features.extend(mel_spect_db.mean(axis=1))

            features_list.append(features)
            labels.append(filename[7])

    return features_list, labels
```

2.3 Data Scaling and Encoding

We used `StandardScaler` to standardize the features and `LabelBinarizer` to one-hot encode the labels.

Listing 2: Data Scaling and Encoding

```
scaler = StandardScaler()
features_array2 = scaler.fit_transform(features_array2)

lb = LabelBinarizer()
labels_one_hot = lb.fit_transform(labels_array2)
```

2.4 Data Splitting

The dataset was split into training and test sets with an 80-20 split.

Listing 3: Data Splitting

```
X_train, X_test, y_train, y_test = train_test_split(features_array2, labels_one_hot, test_s
```

3 Model Development

3.1 Model Architecture

We built a simple neural network with one hidden layer. The input layer has neurons equal to the number of features, and the output layer uses a softmax activation function to classify the emotions.

Listing 4: Model Architecture

```
model = tf.keras.Sequential([
    Dense(300, activation='relu', input_shape=(X_train.shape[1],)),
    Dense(y_train.shape[1], activation='softmax')
])
```

3.2 Model Compilation and Training

The model was compiled using the Adam optimizer and categorical cross-entropy loss. We trained the model for 200 epochs with a batch size of 256.

Listing 5: Model Compilation and Training

```
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
model.fit(X_train, y_train, epochs=200, batch_size=256, validation_split=0.1)
```

3.3 Model Evaluation

The model was evaluated on the test set, achieving an accuracy of approximately 75.36%.

Listing 6: Model Evaluation

```
test_loss, test_accuracy = model.evaluate(X_test, y_test, verbose=2)
print(f'Test Accuracy: {test_accuracy*100:.4f}')
```

4 Results and Discussion

4.1 Classification Report

The classification report shows precision, recall, and F1-score for each emotion class. The overall accuracy of the model is 75.36%.

Listing 7: Classification Report

```
y_pred = model.predict(X_test)
y_pred_labels = np.argmax(y_pred, axis=1)
y_test_labels = np.argmax(y_test, axis=1)
print(classification_report(y_test_labels, y_pred_labels))
```

```
              precision    recall  f1-score   support

           0       0.88      0.74      0.80        38
           1       0.75      0.84      0.79        81
           2       0.76      0.68      0.72        73
           3       0.72      0.80      0.76        71
           4       0.77      0.80      0.79        69
           5       0.78      0.72      0.75        80
           6       0.65      0.58      0.61        45
           7       0.74      0.82      0.78        34

    accuracy                           0.75       491
   macro avg       0.76      0.75      0.75       491
weighted avg       0.76      0.75      0.75       491
```

4.2 Analysis

- The model performed well on most emotion classes with precision and recall above 70%.
- Some classes like 0 and 6 had slightly lower recall, indicating that these emotions were sometimes misclassified.
- The overall accuracy suggests that the model is effective but has room for improvement.

5 Conclusion

This project demonstrates a successful implementation of a Speech Emotion Recognition system using feature extraction and neural networks. The model achieved a test accuracy of 75.36%, showing good performance in classifying different emotions from audio data. Future work could focus on improving the feature extraction process and exploring more complex neural network architectures to further enhance performance.