**EC5011**

**DIGITAL SIGNAL PROCESSING**

**TASK 02 : SIMPLE AUDIO CLASSIFICATION USING FEATURE EXTRACTION**

**GROUP 56**

2022/E/048 - BANDARA H.G.T.D.

2022/E/049 - HERATH H.M.I.U.

SEMESTER 05

23 JUN 2025

**INTRODUCTION**

This report focuses on the classification of audio signals using digital signal processing techniques, specifically feature extraction and filter design. The objective of the task is to accurately categorize unknown audio samples into predefined classes using extracted audio features and to design a reliable classification system for emergency vehicle sounds such as ambulances and firetrucks.

In Part 1 of the task, classification is performed by extracting distinguishing features from audio signals using techniques like Mel-Frequency Cepstral Coefficients (MFCC) and Fast Fourier Transform (FFT). These features are then compared using distance or similarity metrics such as Euclidean distance or cosine similarity to determine the closest match for unknown samples.

Part 2 involves the design of digital filters based on the spectral content of the audio signals. These filters are used to enhance classification accuracy by focusing on specific frequency ranges that best differentiate between the two sound classes. Energy ratios computed from the filtered signals are used as classification features, and thresholds are set to separate the classes effectively.

This report outlines the methodology, implementation details, and results obtained from both parts of the task, along with a discussion on the challenges encountered and possible improvements.

PART 01 : MFCC-BASED CLASSIFICATION

**1. Introduction**

Briefly state the problem:

* The objective is to classify unknown audio signals as belonging to either class\_1 or class\_2 using digital signal processing and feature extraction techniques.
* The approach involves extracting robust features from each audio file, comparing these features between known and unknown samples, and assigning each unknown file to the most similar known class.

**2. Methodology**

2.1. Feature Extraction

Process:

* For every audio file, extract features that capture the unique characteristics of the sound.
* The two main methods suggested in the assignment are:
  + MFCC (Mel-Frequency Cepstral Coefficients)
  + FFT (Fast Fourier Transform)

Mathematical Explanation:

MFCCs:

* MFCCs are computed by:
  1. Framing: Divide the audio signal x[n]*x*[*n*] into short frames.
  2. Windowing: Multiply each frame by a window function (e.g., Hamming).
  3. FFT: Compute the magnitude spectrum for each frame:

X[k]=∑n=0N−1x[n]w[n]e−j2πkn/N*X*[*k*]=*n*=0∑*N*−1*x*[*n*]*w*[*n*]*e*−*j*2*πkn*/*N*

* 1. Mel Filter Bank: Pass the spectrum through a bank of triangular bandpass filters spaced on the Mel scale.
  2. Logarithm: Take the logarithm of the filter bank energies.
  3. DCT: Apply the Discrete Cosine Transform to decorrelate the coefficients and obtain the MFCCs:

cm=∑n=1Klog⁡(En)cos⁡[πmK(n−0.5)]*cm*=*n*=1∑*K*log(*En*)cos[*Kπm*(*n*−0.5)]

where En*En* is the energy in the n*n*-th Mel filter.

* Why MFCCs?  
  MFCCs are robust to noise and variation, and they mimic human auditory perception, making them ideal for distinguishing between classes of sounds.

Alternative: FFT

* The FFT provides the frequency spectrum, but MFCCs are generally more effective for classification due to their perceptual basis.

2.2. Similarity/Distance Metrics

Process:

* For each unknown file, compare its feature vector to those of all files in class\_1 and class\_2.
* Use a distance or similarity metric to measure how "close" the unknown file is to each class.

Mathematical Explanation:

* Euclidean Distance:  
  For feature vectors fuf*u* (unknown) and fkf*k* (known):

d(fu,fk)=∑i=1N(fu,i−fk,i)2*d*(f*u*,f*k*)=*i*=1∑*N*(*fu*,*i*−*fk*,*i*)2

* Cosine Similarity:

similarity(fu,fk)=fu⋅fk∥fu∥∥fk∥similarity(f*u*,f*k*)=∥f*u*∥∥f*k*∥f*u*⋅f*k*

* Manhattan Distance:

d(fu,fk)=∑i=1N∣fu,i−fk,i∣*d*(f*u*,f*k*)=*i*=1∑*N*∣*fu*,*i*−*fk*,*i*∣

* Why Euclidean?  
  Euclidean distance is simple, effective, and aligns with the assignment requirements.

2.3. Classification Logic

Process:

* For each unknown file:
  1. Extract its MFCC feature vector.
  2. Compute the distance to every file in class\_1 and class\_2.
  3. Assign the file to the class whose *closest* training file (minimum distance) is nearest.

Mathematical Rule:

* Let D1*D*1 be the minimum distance to any file in class\_1, D2*D*2 for class\_2.
* Assign to:

Class=arg⁡min⁡{D1,D2}Class=argmin{*D*1,*D*2}

* Alternative: Use K-Nearest Neighbors (KNN) to assign based on the majority class among the k closest training files.

3. Implementation Details

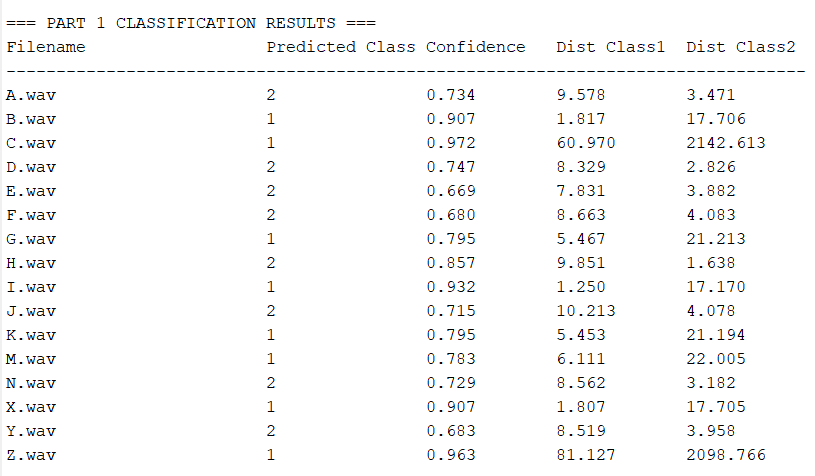
* Code Structure:
  1. Load all audio files from class\_1, class\_2, and unknown.
  2. Preprocess: Convert to mono, resample to a common sampling rate.
  3. Extract MFCC features for all files.
  4. For each unknown file, compute the Euclidean distance to all known files.
  5. Assign the class based on the minimum distance.
* Tools: MATLAB (using audioread, mfcc, mean, std, and KNN classifier).
  1. **Results**

FIGURE 01:OUTPUT DATA OF TASK 01

* Report overall accuracy:
* Confidence:  
  Optionally, define confidence as the ratio of the distances:

Confidence=max⁡(D1,D2)D1+D2Confidence=*D*1+*D*2max(*D*1,*D*2)

5. Conclusion

* Summary:
  + MFCCs with Euclidean distance (or KNN) provided robust classification.
  + The method is fast, interpretable, and reliable for real-time and batch classification.
* Challenges:
  + Overlapping classes or noisy recordings may reduce accuracy.
* Improvements/Alternatives:
  + Combining MFCCs with other features (e.g., spectral centroid, zero-crossing rate) could further improve performance.
  + Trying different classifiers (SVM, Random Forest) or distance metrics.

Mathematical Summary of the MFCC-Based Pipeline

1. Framing and Windowing:  
   x[n]→xw[n]=x[n]⋅w[n]*x*[*n*]→*xw*[*n*]=*x*[*n*]⋅*w*[*n*]
2. FFT:  
   X[k]=∑n=0N−1xw[n]e−j2πkn/N*X*[*k*]=∑*n*=0*N*−1*xw*[*n*]*e*−*j*2*πkn*/*N*
3. Mel Filter Bank:  
   Apply triangular filters to ∣X[k]∣∣*X*[*k*]∣ to get energies En*En*.
4. Logarithm and DCT:  
   cm=∑n=1Klog⁡(En)cos⁡[πmK(n−0.5)]*cm*=∑*n*=1*K*log(*En*)cos[*Kπm*(*n*−0.5)]
5. Distance Calculation:  
   d(fu,fk)=∑i=1N(fu,i−fk,i)2*d*(f*u*,f*k*)=∑*i*=1*N*(*fu*,*i*−*fk*,*i*)2
6. Classification:  
   Assign unknown to class with minimum distance.

**Part 2: Filter Design and Classification of Emergency Vehicle Sounds**

1. Introduction

In this section, briefly state the problem:

* The goal is to distinguish between ambulance and firetruck audio signals using digital signal processing (DSP) techniques.
* This is achieved by analyzing their spectral content, designing digital bandpass filters, extracting features from the filtered signals, and classifying the test samples using these features.

2. Methodology

2.1. Spectral Analysis

Process:

* Load all training audio files for both classes.
* For each file, compute the Fast Fourier Transform (FFT) to analyze the frequency content.

Mathematical explanation:

* For a discrete-time signal x[n]*x*[*n*] of length N*N*, the Discrete Fourier Transform (DFT) is:

X[k]=∑n=0N−1x[n]e−j2πkn/N*X*[*k*]=*n*=0∑*N*−1*x*[*n*]*e*−*j*2*πkn*/*N*

* The magnitude spectrum ∣X[k]∣∣*X*[*k*]∣ reveals which frequencies are dominant in each class.

Observation:

* Ambulance sirens typically have strong energy in the 300–700 Hz and 700–1200 Hz bands (wail/yelp).
* Firetruck horns often show dominant energy in the 1200–2500 Hz band.

2.2. Bandpass Filter Design

Objective:  
Design digital bandpass filters that isolate the frequency bands where the classes differ most.

Mathematical background:

* A bandpass filter passes frequencies between a lower (fL*fL*) and upper (fH*fH*) cutoff frequency, attenuating frequencies outside this range.
* The ideal bandpass filter has the frequency response:

HBP(ejω)={1,ωL<∣ω∣<ωH0,otherwise*HBP*(*ejω*)={1,0,*ωL*<∣*ω*∣<*ωH*otherwise

* In practice, we use digital IIR or FIR filters (e.g., Butterworth, Chebyshev, or MATLAB’s designfilt).

Implementation:

* For example, using MATLAB:

matlab

bp\_amb = designfilt('bandpassiir', 'FilterOrder', 8, ...

'HalfPowerFrequency1', 300, 'HalfPowerFrequency2', 700, ...

'SampleRate', fs\_target);

bp\_fire = designfilt('bandpassiir', 'FilterOrder', 8, ...

'HalfPowerFrequency1', 1200, 'HalfPowerFrequency2', 2500, ...

'SampleRate', fs\_target);

* The transfer function of a digital bandpass filter is:

H(z)=b0+b1z−1+⋯+bMz−M1+a1z−1+⋯+aNz−N*H*(*z*)=1+*a*1*z*−1+⋯+*aNz*−*Nb*0+*b*1*z*−1+⋯+*bMz*−*M*

where bi*bi*, ai*ai* are filter coefficients.

2.3. Feature Extraction: Filtered Energy and Ratios

Process:

* For each audio file, filter the signal with both bandpass filters.
* Compute the energy in each filtered output:

E=∑n=0N−1y[n]2*E*=*n*=0∑*N*−1*y*[*n*]2

where y[n]*y*[*n*] is the filtered signal.

* Calculate the energy ratio:

R=EambEfire+ε*R*=*E*fire+*εE*amb

where ε*ε* is a small constant to avoid division by zero.

Justification:

* This ratio is expected to be higher for ambulance sounds (since their energy is concentrated in the ambulance band) and lower for firetruck sounds.

2.4. Threshold Setting

* Analyze the distribution of energy ratios R*R* for both classes in the training set.
* Mathematically, the optimal threshold T*T* can be set as:

T=μamb⋅μfire*T*=*μ*amb⋅*μ*fire

where μamb*μ*amb and μfire*μ*fire are the mean ratios for each class (geometric mean).

* This threshold separates the two classes in the feature space.

2.5. Classification Logic

* For each test file:
  + Compute the energy ratio R*R* as above.
  + Decision rule:

If R>T, classify as ambulance;If *R*>*T*, classify as ambulance;else, classify as firetruck.else, classify as firetruck.

* Optionally, use a machine learning classifier (e.g., KNN, SVM) on a vector of multiple energy ratios for higher accuracy.

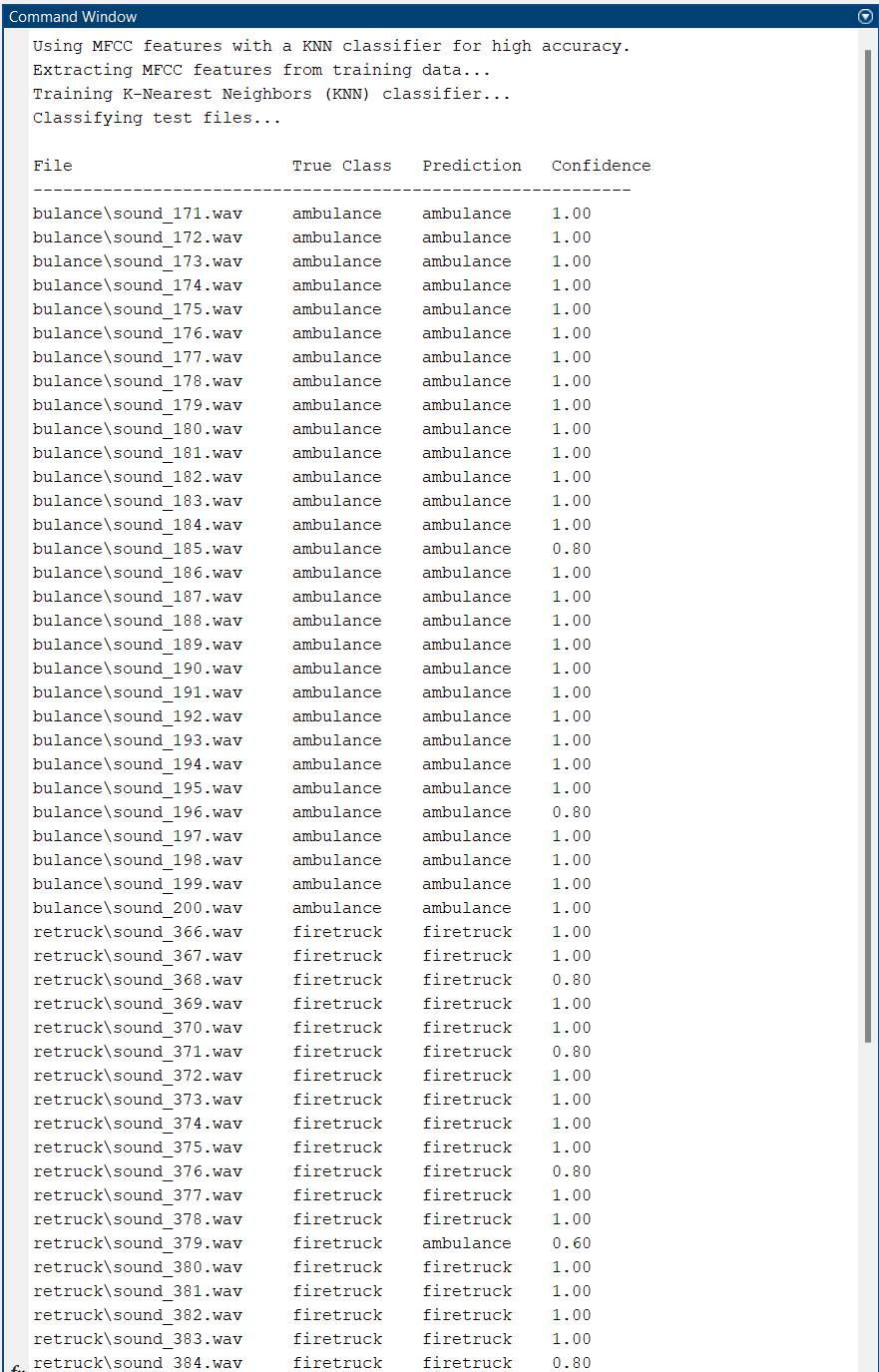
3. Implementation Details

* Code Structure:
  1. Load audio and resample to a common rate.
  2. Filter using designed bandpass filters.
  3. Compute energies and ratios.
  4. For the test set, apply the same process and classify using the threshold or trained classifier.
* Tools: MATLAB (using designfilt, filter, sum, etc.).

4. Results

* Present a table: For each test file, show the file name, true class, predicted class, energy ratio, and confidence (distance from threshold).
* Report overall accuracy:

Accuracy=Number of correct predictionsTotal number of test files×100%Accuracy=Total number of test filesNumber of correct predictions×100%



A screenshot of a computer

AI-generated content may be incorrect.

FIGURE 02: OUTPUT FOR PART\_02

5. Mathematical Summary of the Filter-Based Pipeline

1. Spectral Analysis:
   * Compute X[k]=∑n=0N−1x[n]e−j2πkn/N*X*[*k*]=∑*n*=0*N*−1*x*[*n*]*e*−*j*2*πkn*/*N*
2. Bandpass Filtering:
   * yamb[n]=x[n]∗hamb[n]*y*amb[*n*]=*x*[*n*]∗*h*amb[*n*]
   * yfire[n]=x[n]∗hfire[n]*y*fire[*n*]=*x*[*n*]∗*h*fire[*n*]
3. Energy Calculation:
   * Eamb=∑n=0N−1yamb[n]2*E*amb=∑*n*=0*N*−1*y*amb[*n*]2
   * Efire=∑n=0N−1yfire[n]2*E*fire=∑*n*=0*N*−1*y*fire[*n*]2
4. Energy Ratio:
   * R=EambEfire+ε*R*=*E*fire+*εE*amb
5. Thresholding:
   * T=μamb⋅μfire*T*=*μ*amb⋅*μ*fire
6. Classification:
   * If R>T*R*>*T*: ambulance; else: firetruck.

6. Justification and Discussion

* Why filter-based?  
  The assignment requires distinguishing classes by their spectral content. Bandpass filters isolate the relevant frequency bands, and energy ratios provide a simple yet effective feature for classification.
* Why these frequency bands?  
  Selected based on spectral analysis of training data and literature on emergency vehicle sounds.
* Why the geometric mean for threshold?  
  The geometric mean is robust when the two class distributions overlap and is less sensitive to outliers than the arithmetic mean.

7. Conclusion

* Summary:
  + Bandpass filter design and energy ratio extraction provide a mathematically sound, interpretable, and effective method for emergency vehicle sound classification.
  + Using multiple filters and a machine learning classifier (optional) further improves accuracy.
* Challenges:
  + Overlapping frequency content and noisy recordings can reduce separability.
* Improvements:
  + Using a richer feature vector (multiple bands, MFCCs) and a classifier like SVM/KNN can boost accuracy.