EESC, EEGR, and CE 6364 – Fall 2024

Project Description – Online Clustering of Audio Signals via ART2

This project involves applying ART2 (Adaptive Resonance Theory 2) online clustering to features extracted from audio files. ART2 is particularly well-suited for online clustering or unsupervised learning as it does not require predefined labels or a predefined number of clusters. This flexibility makes it suitable for tasks where the number of clusters or classes is unknown to begin with and clusters evolve over time. Additionally, the ability of ART2 to operate in real-time makes it suitable for applications in streaming data processing, where features change in an on-the-fly manner.

Overview:

This project involves extracting frequency features from audio files, forming time-series, and applying ART2 clustering. Extensive experimentations are to be conducted for different frame sizes and audio features (frequency feature sets of MFCC and MFSC), combining features from three audio files to form ground truth time-series for clustering, and finding the right vigilance parameter to reach desirable clustering outcomes. Clustering outcomes are defined in terms of graphs showing predicted clusters vs. time compared with those of ground truth clusters vs. time.

Part 1: Feature Generation and Clustering

The first step in this part involves extracting and clustering features from the provided audio files. The initial step is to generate the features of MFCC (Mel-Frequency Cepstral Coefficients) and MFSC (Mel-Frequency Spectral Coefficients) by experimenting with the frame size and using the MFCC feature set alone, the MFSC feature set alone, and combining the MFCC with MFSC feature sets. Extensive experimentations are to be done to obtain the setting for the best clustering outcomes.

After extracting these features, the next step involves creating at least five distinct ground truth clustering time-series by combining features from the three audio files in a random manner generating combination time-series which have the same duration as the original audio files (about 2 mins). These combination time-series are to be used to act as the ground truth for the online operation of ART2 clustering as samples (features of a frame) are received one at a time.

Part 2: Feature Reduction and Clustering

In the second part, dimensionality reduction needs to be applied before ART2 clustering. This reduction is be done in a linear manner (PCA). At the end, one needs to be able to assess how well ART2 clustering performs on both the original features and dimensionality reduced features.

More details of each part are stated below.

Dataset

Three audio files are provided as the dataset for this project which is made available in eLearning under Project and can also be downloaded from utdallas.box. Link is given below.
https://utdallas.box.com/s/8p8h20g3yp2ba4v9yrkzufufc7jhwhyr

Evaluation Metrics

Generating Graphs:

1. **Predicted Clusters vs. Time**:
    - By using the timestamps of frame-by-frame processing, generate a graph or plot showing the predicted clusters obtained by ART2 over time. This graph gives a visual representation of how clustering evolves over time as an audio signal is captured, highlighting any transitions or shifts between different clusters or audio signals.

2. **Ground Truth Clusters vs. Time**:
    - Plot or graph the actual ground truth clusters over time alongside the predicted clusters. This comparison allows one to visually see how well ART2 clustering aligns with the actual clusters or classes and whether it correctly identifies changes in the audio signal in the combination time-series.

Quantitative Evaluation:

1. **Accuracy**:
    - To quantitatively evaluate the performance of ART2 clustering, compute the accuracy (percentage of correct classification) by comparing the predicted clusters to the ground truth clusters. This provides an insight into how accurately the clustering algorithm has identified the correct clusters or classes.

2. **Confusion Matrix**:
    - Generate a confusion matrix to further analyze the clustering outcome. The confusion matrix indicates which clusters are correctly predicted and where misclassifications occur.

Part 1 (Feature Generation and Clustering of Audio Signals)

Due Friday November 1, 12 pm

Audio signals are often dynamic and non-stationary. Thus, audio files are normally divided into segments called frames (duration in milliseconds) over which frequency content remains stationary. By processing audio signals in a frame-by-frame manner, one would be able to capture how frequency content evolves or changes over time.

Each frame is often multiplied by a window function (commonly Hamming or Hanning) to minimize spectral leakage. This occurs when a rectangular window is used. A smooth window is often used to avoid distortion due to such a leakage when extracting frequency content.

Frequency content is captured by applying Fourier transform to convert an audio signal from the time domain (where the signal is a waveform) to the frequency domain (where the signal is represented as a spectrum of frequencies). However, applying Fourier transform to the entire audio signal would result in the loss of time-related information. That is why an audio signal is broken up into overlapping frames (overlapping segments of the signal) to analyze how frequency content changes over time.

For the three provided audio files, perform frame-by-frame feature extraction. Experiment with different frame sizes (in milliseconds) and overlap to determine which size and overlap produces the best clustering outcome. The frame size has an impact on the generated features depending on the time-frequency resolution. Additionally, an overlap between consecutive frames needs to be used to ensure smooth transitions between frames and avoid missing important frequency content.

Steps to be taken:

1. Frame-by-Frame Audio Processing:

- Divide the audio signals into frames: Process the audio signals by dividing them into overlapping frames.
- Examine different frame sizes (in milliseconds) to identify which size produces the best clustering outcome.
- Frame overlap: Use 10% overlap between frames to ensure smooth transitions between frames and to avoid losing information between frames.

Important: While performing the above, collect timestamps for each frame. These timestamps are needed when generating graphs that compare the predicted clusters versus time with the ground truth clusters versus time for examining the clustering outcome.

2. Feature Extraction and Selection:

Extract the following two sets of features in the frequency domain:

- **MFSC (Mel-Frequency Spectral Coefficients)**: Representing the energy in each Mel-scaled frequency band or representing the power spectrum.

- **MFCC (Mel-Frequency Cepstral Coefficients)**: A compact, decorrelated representation of the Mel spectrum compressing the frequency content into a small set of coefficients. The lower-order coefficients capture broad spectral features, while higher-order coefficients capture finer details.

For computing MFSC (Mel-Frequency Spectral Coefficients) and MFCC (Mel-Frequency Cepstral Coefficients), use 40 coefficients for MFSC and 13 coefficients for MFCC. These values are commonly used in audio analysis to capture adequate frequency content while balancing computational efficiency. After generating the above sets of features, analyze which feature set or their combination generate the best clustering outcome. The ground truth time-series are to be used to compute the metrics of accuracy and confusion matrix. When extracting features from each audio file, assign a class_id to each frame to identify to which audio file or class a frame belongs. Additionally, it helps to save the extracted features in multiple formats for flexibility.

I. **CSV Format**: Save the features in a CSV file with appropriate headers for each column (timestamp, mfcc, mfsc, and class_id). This format allows one to easily inspect the features and verify that the features are generated correctly. A screenshot of what to turn in is provided below for reference.

class_id	timestamp	mfccs	mfsc
0	0	[-1131.370849609375, 0.0, 0.0, 0.0, 0.0, 0.0, 0	[-100.0, -100.0, -100.0, -100.0, -1
0	0.02	[-1131.370849609375, 0.0, 0.0, 0.0, 0.0, 0.0, 0	[-100.0, -100.0, -100.0, -100.0, -1
0	0.04	[-1131.370849609375, 0.0, 0.0, 0.0, 0.0, 0.0, 0	[-100.0, -100.0, -100.0, -100.0, -1
0	0.06	[-1131.370849609375, 0.0, 0.0, 0.0, 0.0, 0.0, 0	[-100.0, -100.0, -100.0, -100.0, -1
0	0.08	[-1131.370849609375, 0.0, 0.0, 0.0, 0.0, 0.0, 0	[-100.0, -100.0, -100.0, -100.0, -1
0	0.1	[-1131.370849609375, 0.0, 0.0, 0.0, 0.0, 0.0, 0	[-100.0, -100.0, -100.0, -100.0, -1
0	0.12	[-1131.370849609375, 0.0, 0.0, 0.0, 0.0, 0.0, 0	[-100.0, -100.0, -100.0, -100.0, -1
0	0.14	[-1131.370849609375, 0.0, 0.0, 0.0, 0.0, 0.0, 0	[-100.0, -100.0, -100.0, -100.0, -1
0	0.16	[-1131.370849609375, 0.0, 0.0, 0.0, 0.0, 0.0, 0	[-100.0, -100.0, -100.0, -100.0, -1
0	0.18	[-1131.370849609375, 0.0, 0.0, 0.0, 0.0, 0.0, 0	[-100.0, -100.0, -100.0, -100.0, -1
0	0.2	[-1131.370849609375, 0.0, 0.0, 0.0, 0.0, 0.0, 0	[-100.0, -100.0, -100.0, -100.0, -1

II. **Other Formats**: Optionally, the features could be saved in formats such as .pkl to gain efficiency for future use. This binary format allows for faster read/write operations, in particular, with large datasets.

3. Combining Features and Generating Time-Series:

- **Combine features from each of the three audio files**: Form at least five different cluster combination time-series by combining features from the three audio files having more or less the same total time duration (about 2 mins). Each combination time-series should include different time portions from the three audio files. Combination time-series can be formed by changing the time portions of the audio files.

Collect timestamps for each time-series to generate graphs of the predicted clusters vs. time as well as graphs of the ground truth clusters vs. time. Save each of the generated time-series in a separate CSV file with appropriate headers for each column (timestamp, mfcc, mfsc, and class_id). These files are to be turned in.

4. ART2 Clustering:

- Use the generated time-series as input for ART2 clustering. Experiment by comparing whether feeding the raw or original features directly produces better clustering outcomes or whether applying normalization to the features improves clustering outcomes.

- **Study vigilance parameter**: ART2 is sensitive to the vigilance parameter, which controls how far a new sample must be from the existing clusters in order to create a new cluster. Experiment with different vigilance parameter values for obtaining the best clustering outcomes for the five time-series.

Ensure that you implement ART2 clustering in a real-time manner, i.e. ART2 obtaining cluster labels one sample at a time allowing for online clustering as new samples are received on-the-fly.

Possible fluctuations in the clustering outcome can be smoothed out in a number of ways. For instance, one can average a certain number of frame**s** before inputting them into ART2 and/or by performing median filtering or majority voting after clustering.

Make sure that your results are saved in a CSV file with appropriate headers for each column (timestamp, prediction, ground truth).

Important: After performing clustering, generate the following graphs:

- **Predicted clusters vs. time**: Use the timestamps collected earlier to plot or graph the predicted clusters assigned by ART2 over time.

- **Ground truth clusters vs. time**: Compare the predicted clustering outcome with the ground truth and thus compute accuracy and confusion matrix.

What to turn in

eLearning: In a single zip file, include your codes and report. Name all the files (codes and report) and the zipped file Lastname-ProjectProgress. Upload the zipped file to eLearning. Make sure that the zipped file is properly loaded before logging out of eLearning.

Your report needs to include:

- All the steps required to run your codes.
- Generated graphs of predicted clusters vs. time and ground truth clusters vs. time for each of the 5 time-series.
- Accuracy and confusion matrix for each of the 5 time-series.
- Detailed and thorough analysis and discussion of ART2 performance using the time-series obtained from the raw or original feature sets, providing insight into

which combinations of frame size, feature sets, vigilance parameter, and smoothing produce the best clustering outcomes.

Box drive: Create a folder in your Box drive and name it 'Lastname-ProgressReport'. Place the CSV files generated from the feature extraction, time-series formation and ART2 clustering into this folder. Share your Box folder with (nxt230004@utdallas.edu) and allow permission to access.

Grading distribution for Part1 (40 points)

- Frame size selection, feature extraction and feature set selection according to the instructions (10 points)
- Generating combination time-series (5 points)
- ART2 clustering (10 points)
- Thoroughness of results and their analysis/discussion (15 points)

Part 2 (Feature Reduction Using PCA and Clustering with ART2)

Codes and report due Wed Nov 20, 12pm

(<u>This is a strict and non-negotiable deadline</u>)

Project presentation and code questioning conducted on Friday Nov 22

For this part, apply a dimensionality reduction technique to simplify the feature sets. The primary method to be used here is Principal Component Analysis (PCA), which reduces the dimensionality of features in a linear manner. You can experiment by applying PCA separately to MFSC and MFCC to reduce each feature set independently or apply PCA on the combined feature set (MFSC + MFCC) to reduce dimensionality across both sets together. For those who wish to do more or for extra points, Auto-Encoders (AE) can be used to achieve dimensionality reduction in a non-linear manner which is often more effective than PCA.

After applying PCA, you will create five new time-series by combining the reduced features from the audio files, similar to the first part. These five new time-series with the reduced feature sets are then fed into ART2 to explore how well the clustering is done when the dimensionality of the features is reduced. Examine whether feeding the reduced dimensionality features with and without normalization produce better clustering outcomes.

Ensure that you implement ART2 clustering in a real-time manner, where ART2 assigns cluster labels one sample at a time, allowing for online clustering as new samples are received on-the-fly.

Possible fluctuations in the clustering outcome can be smoothed out in a number of ways. For instance, one can average a certain number of frames before inputting them into ART2 and/or by performing median filtering or majority voting after clustering.

At the end, you should be able to discuss how well ART2 clustering performs on both the original and reduced dimensionality feature sets.

What to turn in

eLearning: In a single zip file, include your codes and report. Name all the files (codes and report) and the zipped file Lastname-FinalReport. Upload the zipped file to eLearning. Make sure that the zipped file is properly loaded before logging out of eLearning.

Your report needs to include:

- All the steps required to run your codes.
- Generated graphs of predicted clusters vs. time and ground truth clusters vs. time for each of the 5 time-series.
- Accuracy and confusion matrix for each of the 5 time-series.
- Detailed and thorough analysis and discussion of ART2 performance using the time-series obtained from both the original and reduced dimensionality feature sets,

providing insight into which combinations of frame size, feature sets and vigilance parameter, and smoothing produce the best clustering outcomes.

Box drive: Create a folder in your Box drive and name it 'Lastname-FinalReport'. Place the CSV files generated from the reduced feature extraction, time-series formation and ART2 clustering parts into this folder. Share your Box folder with (nxt230004@utdallas.edu) allowing permission to access.

Grading distribution for Part2 (35 points)

- Feature reduction (10 points)
- Thoroughness of results and their analysis/discussion (25 points)

Project total grade distribution (100 points)

- Project Part1– grading will be revised if further work is done for this part (40 points)
- Project Part2 (35 points)
- Project code questioning (15 points)
  - Conducted on November 22, 2024, from 9:00am to 12:00pm online (camera must be turned on) in eLearning's Blackboard Collaborate. Each student will be allocated 5 minutes. Sign up for your time slot using the following Google Sheet - https://docs.google.com/spreadsheets/d/1OYUQlVyqd4X_HHmQeprnr9VHxcehqSuNstAr5SdwpJY/edit?usp=sharing
  - You must join the online session ONLY at the time you sign up. You should not log in or be present at the code questioning eLearning session at other times.
  - Questions will be asked from your submitted implemented codes for both Part1 and Part2.
- Presentation and demo (10 points)
  - Email your presentation to (nafisazarrin.tasnim@utdallas.edu) by 10:00 am the latest on November 22, 2024. Rename is as 'Lastname.pptx'
  - Conducted on November 22, 2024, from 1:00pm to 3:45pm in the classroom.
  - You will get 5 minutes max to present your project work followed by questions. Your presentation must be focused on the results you obtained.

Resources for Implementation:

The following Python librosa functions are useful for implementing frame-by-frame audio processing and feature extraction:

- **librosa.load**:
  - This function is used to load an audio file at a particular sampling rate. It returns both the audio time-series (waveform) and the sampling rate.

- **librosa.stft**:
    - **STFT (Short-Time Fourier Transform)** divides an audio signal into overlapping frames, applies a window function (Hanning by default), and computes Fourier transform for each frame, returning frequency content for that time frame.
    - The result of librosa.stft is a complex number for each frequency bin of a frame, consisting of both magnitude and phase components. The interest here is in magnitude.
- **librosa.feature.melspectrogram**:
    - This function computes the Mel Spectrogram from an audio signal. STFT is used internally by librosa to compute the Mel Spectrogram, which is a representation of a signal power across the Mel-scaled frequency bands.
- **librosa.power_to_db**:
    - This function converts the power spectrogram (in linear scale) into decibels (dB), which is a logarithmic scale more appropriate for human hearing perception.
- **librosa.feature.mfcc**:
    - This function computes the MFCC features either directly from a time-domain audio signal (waveform) or from a precomputed Mel spectrogram depending on the input.

The following link can be referred to for more detailed information on how to use librosa functions:

https://librosa.org/doc/latest/core.html

Note: The above resources and functions are commonly used for audio processing, but there are many other libraries and tools available for similar tasks. Librosa is a well-known Python library for audio analysis, but depending on your use case, feel free to explore other libraries such as SciPy, PyDub, or TensorFlow's Audio module. Each library may offer unique functionality or optimizations that could better suit your specific requirements.

References:

1. Alamdari, N., and N. Kehtarnavaz, "A Real-Time Smartphone App for Unsupervised Noise Classification in Realistic Audio Environments," *Proceedings of IEEE International Conference on Consumer Electronics (ICCE)*, 2019.
2. McFee, *Framing - Digital Signals Theory*. https://brianmcfee.net/dstbook-site/content/ch09-stft/Framing.html.
3. Shabda, *Audio Basics*. https://shabda.readthedocs.io/en/latest/references/AudioBasics.html.
4. Nair, *Guide to MFCC*. https://medium.com/prathena/the-dummys-guide-to-mfcc-aceab2450fd.