AML Milestone 1

Darren Midkiff, Cheng-Wei Hu

November 2020

Overview

This project aims to identify the genre of music in an audio sample. Features will be extracted from the analog data, and a genre will be predicted using a KNN model, trained on labelled audio samples from the freely available FMA dataset.

Training Data

The FMA Dataset (https://github.com/mdeff/fma) is comprised of over 100,000 tracks from 161 genres. In order to make the problem more manageable, we will use the small version of the dataset, which includes 8,000 tracks from 8 top-level genres. The dataset also includes dozens of features – year released, location of artist, number of listens, etc. Because this project aims to identify genre using only audio signal, all of these features are irrelevant and will be dropped.

```python
import pandas as pd

#read full metadata file
metadata = pd.read_csv("./fma_metadata/tracks.csv", skiprows=[0,2], low_memory=False)

# drop all tracks that are not in fma_small dataset
metadata = metadata[metadata["subset"].eq("small")]
# add name to track_id column (missing because of stupid CSV formatting)
metadata = metadata.rename(columns={"Unnamed: 0": "track_id"})
# drop all columns that don't relate to genre
# we will not have this metadata from the audio file
metadata.drop(metadata.columns.difference(["track_id","genre_top"]),1,inplace=True)
# reset indices accounting for dropped rows
metadata = metadata.reset_index(drop=True)

# #write only relevant metadata to file for use in training
metadata.to_csv("fma_small_genres.csv")
```

Observing the trimmed data reveals the 8 top-level genres to be Hip-Hop, Pop, Folk, Experimental, Rock, International, Electronic, and Instrumental.

Feature Extraction

In order to apply machine learning techniques to audio samples, useful features must be extracted from the signals. We will extract four features: Zero Crossing Rate, Spectral Centroid, Spectral Rolloff, and Mel-Frequency Cepstral Coefficients.

In our project, we will use librosa (https://librosa.org/doc/latest/index.html), a Python package for music and audio analysis, to extract features from raw audio. Much of the code is excluded here, as it simply takes up too much space.

```
# Extract the features from audio files
# Some of the audio files are damaged. So we skipped those files.
# The damaged files are: './fma_small/099/099134.mp3', './fma_small/108/108925.mp3', './fma_

train_audio_features_all = []
fail_file_names_all = []
fail_file_idx_all = []
fail_file_names_dict_all = {}

for idx, file in enumerate(file_names):
    print(idx, file)
    try:
        single_audio_features = extract_feature_from_audio(file)
        row = []
        row.append(file.split('/')[-1])
        for f in single_audio_features:
            row.append(f)
        train_audio_features_all.append(row)
    except:
        print("Failed: ", idx, file)
        fail_file_names_all.append(file)
        fail_file_idx_all.append(file)
        fail_file_names_dict_all[file] = True
```

Pre-Processing

In order to make the model training easier, we have to preprocess the data and combine the 'features_all.csv' files with 'fma_small_genres.csv', storing all calculated features with their respective track IDs. This will be extremely helpful in model training.

```
# Read features_all.csv and process the file_name columns
import pandas as pd
```

```python
raw_data = pd.read_csv("./features_all.csv", low_memory=False)
raw_data = raw_data.rename(columns={"file_name": "track_id"})
raw_data['track_id'] = raw_data['track_id'].str[:-4].astype(int)

# Read features_all.csv and make it a dictionary
 and process the file_name columns
metadata = pd.read_csv("./fma_small_genres.csv", low_memory=False)
metadata_mapping = dict([(i,g) for i, g in zip(metadata.track_id, metadata.genre_top)])

# Insert a column genre_top into raw_data with each track's corresponding genre_top
raw_data["genre_top"] = raw_data["track_id"].map(metadata_mapping)
raw_data.to_csv("fma_small_train_data.csv")
```

Model

Now that we have labeled training data and corresponding features, the only remaining step is to train a model. We are strongly considering a nearest-neighbors model, but the size of the training dataset may prove prohibitive to this strategy. We will analyze the effectiveness of a variety of models before making a final decision.