1. **Approach**
   1. **Data collection**

A dataset of 1200 of popular trending songs labeled to four classes: sad, happy, energetic, and calm. These songs are uniformly almost distributed in these classes. Most of the previous works use western music/instrumentals as they are easier to classify into emotions and are less subjective than the top songs on the radio today. However I have chosen to work with these songs as they are used mostly listened to by the population.

The dataset comes with some Spotify extracted features using a Spotify API called Spotipy.

All of these songs are almost evenly distributed to the 4 classes. (see Figure 1)

A graph of blue bars

Description automatically generated with medium confidence

Figure

Due to the necessity of providing a CNN relevant inputs, it was necessary to somehow store the songs into a matrix. The methodology of downloading all the songs automatically was used through another python script. This script uses “youtube\_search” API to search the songs on YouTube by the name of the song and retrieve the corresponding URL. The URL is later passed to a function that would download the video (using PyTube API and requests) and extract only the audio using MoviePy API. The audio was then truncated to use only 30 seconds of the song starting from ¼ of the song’s duration and save it locally on the hard drive as an mp3 file.

Having the mp3 files for each song in the dataset, we would extract the following features [4]:

* *MFCC (Mel-frequency cepstral coefficients):* represent the spectral envelope of the audio signal and capture the power spectrum characteristics. They’re derived by applying a series of mathematical operations to the short-time power spectrum, widely used in speech and audio processing tasks
* *Spectral Centroid*: It indicates the "center of mass" of the spectrum, providing insights into the spectral brightness or the average frequency content of the signal. Higher values correspond to brighter or higher-pitched sounds.
* *Chroma Energy Normalized Statistics (CENS):* CENS computes a chromogram representation, summarizing the distribution of energy in different pitch classes. It's useful for analyzing tonal content in music and is more robust than raw chroma features against variations in tempo and dynamics
* *Mel Spectrogram***:** This representation is obtained by converting the linearly spaced frequency bins of a spectrogram into Mel-frequency bins using a Mel filter bank. It emphasizes frequencies relevant to human perception, facilitating tasks like audio classification or genre recognition

All of these features get concatenated and normalized using the Min-Max function. (see Figure 2)

A blue and green screen

Description automatically generated

Figure 2 (one sample of how a song is stored into memory)

* 1. **Feature extraction**

Spotipy is able to provide some unique values for features like *danceability, energy, key, loudness, speechiness, acousticness, instrumentalness, liveness, valence, tempo* and *time signature.* Unfortunately the methodology of how these features are extracted is not publicly available.

Having the songs stored into memory as in Figure 1, we then pass these songs to a CNN for relevant feature extraction.

* 1. **Experiments**

Training several different models with different features is the first step on a large understanding in the experiments I have conducted.

The models I have experimented with are:

* Spotify features:
  + Logistic Regressor with a Standard Scaler as feature standardizer
  + ANN with various hidden layer sizes and learning rate values
* CNN features:
  + ResNet18, finetuning parameters

As described in [6], CNN is a good way of extracting features from various inputs. I have decided to approach this problem as an “image classification” problem, where I try to convert a sequence of a song into an image. Just using a few of the features mentioned above in the Data collection section, features that were present in [6] aswell, I could easily assemble an image as a song.

Models’ performance is measured in accuracy. The training part uses a K-fold validation method with 5 folds, due to the fact that the dataset is small.

The parameters of the best registered accuracies on the conducted experiments are provided below:

* CNN (modified ResNet18 in Figure 3)
  + First convolutional layer is modified to use one channel instead of three; also the stride of 1, kernel size of 7 and padding of 2
  + Using a learning rate of 0.0009, a SGD optimizer with 0.9 momentum and a learning rate scheduler of step\_size=7 and gamma=0.1, batch size of 4
  + Added a fully connected layer at the end to predict into the 4 classes the data is labeled to
  + All layers are frozen except the last one, which is the one that is trained

A diagram of a graph

Description automatically generated

Figure 3 – Modified CNN, first Conv layer

After trying with various values for the learning rate, kernel size, stride and different optimizers, the above values gave the best results out of 10 epochs with 5 folds. The model outputs an accuracy of 80% and regular fully connected layers at the end for the classification problem. In [6], just using MFCCs as input, they obtained an accuracy of 88.70% on turkish songs with LSTM+DNN classifier.

A diagram of a confusion matrix

Description automatically generated with medium confidence

Figure 4 – Confusion matrix for CNN

* ANN
  + The model consists of 4 hidden layers with 128, 128, 64, 32 neurons in this order
  + Using a learning rate of 0.006, an Adamax optimizer with step\_size=50 and a gamma=0.1

After trying with different values for the learning rate, other optimizers and even adding more hidden layers with more neurons, the above values seem to be the best results out of 200 epochs. The model outputs an accuracy of 84% in average with Spotify’s features.

Interesting to note is the fact that adding more hidden layers to the network would not only increase the training time, but also would decrease the model’s performance dramatically.

* Logistic regressor
  + Used only a standard scaler to standardize the features by removing the mean and scaling to unit variance

This approach got an accuracy of 82% in average with Spotify’s features.

After these conducted experiments, we can safely say that the features extracted by Spotify are far greater than the “image” approach of a song classification. However, a CNN could have a far better performance if a better architecture using RNNs with LSTM/GRU gates for learning sequences better than only regular fully connected layers.

Compared to [5], where similar features were used, I managed to get to a fairly good accuracy almost close to an RNN (their 89.2% on instrumental music) on popular music.

1. **Previous works**

Although a lot of effort went into MER, this domain is something that can be very subjective. Compared to image classification where the ground truth is almost the same to every human being when given the prompt, music is way different. Some songs could easily be categorized in different classes of emotions by various people. A sad song could easily be categorized as a calm song for instance.

In [1], a regression approach is used to determine the emotion of the given song. To solve the dependency between arousal and valence and predict AV values, the paper formulates MER as a regression problem. Support Vector Regression outperforms other algorithms, reaching 58.3% for arousal and 28.1% for valence, plotted as a point on the Thayer’s arousal-valence emotion plane. This regression approach significantly improves emotion variation detection within music selections. Also the features are extracted with a professional computer program “PSYSOUND” [2].

In [3], a similar AV prediction is used, however the Thayer’s emotion plane is divided into more areas, big enough to store 11 emotions. They use SVMs to try to predict the emotions of the songs. Their accuracy of prediction reaches 94.55% on western pop songs.

In [5], on an instrumental music database, where they use RNNs and similar features found in this paper (MFCC, CENS, Spectral centroid, bandwidth, rollof, ZCR, Chroma), all combined and then fed into an RNN to classify the songs into Happy, Sad, Neutral and Fear for many instruments, reach a performance of 89.2%.

In [6], they use a CLSTM architecture to find the emotion given in Turkish songs. They use a CNN to extract relevant features from 30s sequences of songs and then feed the output as an input to LSTMs and reach an accuracy of 91.93% on the full feature set when using LSTM+DNN for classification and a 99.19% with CFS applied. After obtaining an 88.70% accuracy on MFCCs, they decided to increase the number of features used and apply CFS on them which boosted the model’s overall performance to 99.19%.

[1] Y. -H. Yang, Y. -C. Lin, Y. -F. Su and H. H. Chen, "A Regression Approach to Music Emotion Recognition," in IEEE Transactions on Audio, Speech, and Language Processing, vol. 16, no. 2, pp. 448-457, Feb. 2008, doi: 10.1109/TASL.2007.911513.

[2] D. Cabrera, “PSYSOUND: A computer program for psychoacoustical analysis,” in Proc. Australian Acoust. Soc. Conf., 1999, pp. 47–54 [Online]. Available: <http://www.psysound.org>

[3] B. Han, S. Rho, R. B. Dannenberg, and E. Hwang, “SMERS: Music emotion recognition using support vector regression,” in Proc. of the Intl. Society for Music Information Conf., Kobe, Japan, 2009.

[4] R. Panda, R. Malheiro and R. P. Paiva, "Novel Audio Features for Music Emotion Recognition," in IEEE Transactions on Affective Computing, vol. 11, no. 4, pp. 614-626, 1 Oct.-Dec. 2020, doi: 10.1109/TAFFC.2018.2820691.

[5] Rajesh, Sangeetha, and N. J. Nalini. "Musical instrument emotion recognition using deep recurrent neural network." Procedia Computer Science 167 (2020): 16-25, doi: 10.1016/J.PROCS.2020.03.178

[6] Hizlisoy, Serhat, Serdar Yildirim, and Zekeriya Tufekci. "Music emotion recognition using convolutional long short term memory deep neural networks." Engineering Science and Technology, an International Journal 24.3 (2021): 760-767, doi: 10.1016/J.JESTCH.2020.10.009