DEGREE PROJECT IN INFORMATION AND COMMUNICATION TECHNOLOGY,
SECOND CYCLE, 30 CREDITS
STOCKHOLM, *SWEDEN 2021*

Classification of Affective Emotion in Musical Themes

How to understand the emotional content of the soundtracks of the movies?

PAULA DIAZ BANET

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

Abstract

Music is created by composers to arouse different emotions and feelings in the listener, and in the case of soundtracks, to support the storytelling of scenes. The goal of this project is to seek the best method to evaluate the emotional content of soundtracks. This emotional content can be measured quantitatively thanks to Russell's model of valence, arousal and dominance which converts moods labels into numbers. To conduct the analysis, MFCCs and VGGish features were extracted from the soundtracks and used as inputs to a CNN and an LSTM model, in order to study which one achieved a better prediction. A database of 6757 number of soundtracks with their correspondent VAD values was created to perform the mentioned analysis.

As an ultimate purpose, the results of the experiments will contribute to the start-up Vionlabs to understand better the content of the movies and, therefore, make a more accurate recommendation on what users want to consume on Video on Demand platforms according to their emotions or moods.

Keywords

Music emotion recognition, Deep learning, Feature extraction, VGGish, Mel-frequency Cepstral Coefficients.

Abstract

Musik skapas av kompositörer för att väcka olika känslor och känslor hos lyssnaren, och när det gäller ljudspår, för att stödja berättandet av scener. Målet med detta projekt är att söka den bästa metoden för att utvärdera det emotionella innehållet i ljudspår. Detta känslomässiga innehåll kan mätas kvantitativt tack vare Russells modell av valens, upphetsning och dominans som omvandlar stämningsetiketter till siffror. För att genomföra analysen extraherades MFCC: er och VGGish-funktioner från ljudspåren och användes som ingångar till en CNN- och en LSTM-modell för att studera vilken som uppnådde en bättre förutsägelse. En databas med totalt 6757 ljudspår med deras korrespondent acrshort VAD -värden skapades för att utföra den nämnda analysen.

Som ett yttersta syfte kommer resultaten av experimenten att bidra till att starta upp Vionlabs för att bättre förstå innehållet i filmerna och därför ge mer exakta rekommendationer på Video on Demand-plattformar baserat på användarnas känslor eller stämningar.

Nyckelord

Music emotion recognition, Deep learning, Särdragsextraktion, VGGish, Mel-frequency Cepstral Coefficients

Contents

List of Figures

List of Tables

List of Acronyms and Abbreviations

CNN Convolutional Neural Network

DNN Dense Neural Network

LSTM Long Short Term Memory

MAE Mean Absolute Error

MER Music Emotion Recognition

MFCC Mel-frequency Cepstral Coefficients

MIR Music Information Retrieval

MSE Mean Square Error

RNN Recurrent Neural Network

SVM Support Vector Machine

VAD Valence, Arousal and Dominance

VoD Video on Demand

Chapter 1

Introduction

1.1 Background

Music has been in society since the beginning of time as it has the ability to stimulate emotions and feelings. Composers and creators employ numerous techniques and devices to arouse emotions and feelings in the listener, and in some contexts to support storytelling.

Nowadays, most film spectators use Video on Demand (VoD) platforms to watch movies. The competition in this sector is constantly increasing and therefore, companies have to reinvent themselves to keep their users loyal. What differentiates one service from another is the diverse content and good recommendation algorithms. In the end, if they are capable of suggesting correctly what the user wants to consume according to his current mood or personality. Therefore, understanding the overall emotional content of a movie contributes to a more accurate recommendation system for VoD platforms and a customized experience for users. Indeed they will spend less time searching for what content they prefer to consume [3]. This project aims to extract the emotions generated by movies soundtracks for this ultimate purpose of understanding better the content.

1.2 Problem

Music Emotion Recognition (MER) is the area of Music Information Retrieval (MIR)[1] that aims to identify perceived emotions in music by extracting and analyzing different audio features [5] or lyrics [6]. MER has multiple applications such as therapy or research, but above all, it is widely applied in automatic audio recommendation systems like Spotify [7].

Despite this, researchers find it challenging to develop algorithms with good performance due to three main reasons. Firstly, because of the subjectivity of emotions. Even though the composer intended to arouse a certain emotion in the listener, the perceived emotion can vary considerably from one user to another. The second reason is the lack of suitable datasets because labelling or gathering audio files is a demanding task. Lastly, developing robust algorithms which analyze sequential input data, such as audio files, is a challenging task because the input has a temporal relationship that should be taken into account, not just temporal relation with the past, but sometimes with the future as well [8][9].

Among the available music emotion datasets, there are two main approaches to classify emotions, the categorical and the dimensional [10]. The categorical one represents the emotion with discrete tags such as 'happy', 'sad', 'anger' or 'relaxed', i.e. measures them qualitatively [11]. With this approach, MER problem turns into a classification one. On the other hand, the dimensional model measures quantitatively the emotions, and so it becomes a regression problem [6][12]. A wider range of emotions can be represented using this model. The most popular dimensional model used across the literature is Russell's model [13]. It consists of a three-dimensional space defined by Valence, Arousal and Dominance, explained more in detail in Section 2.1.

1.3 Purpose

The purpose of this project is to analyze a dataset of soundtracks, extract their features and estimate the VAD values. Ultimately, the understanding of the emotional content of the soundtracks will contribute to analyse the overall emotional content of the movie.

[1] "Interdisciplinary science aimed to studying the processes, systems and knowledge representations required for retrieving information from music." Definition taken from [4].

1.4 Goals

The main goal of this project was the training a model to predict the VAD values of a given soundtrack. For this goal, a comparison between two different audio features explained in Section 2 was performed, in order to see which one carries more relevant information to predict the correct VAD labels. The result of this comparison was measured mainly in terms of loss, more detailed explained in Chapter 2.

1.5 Research Methodology

The Methodology followed was: do some research to find a suitable dataset and then, extract the audio features -MFCC and VGGish- explained in Chapter 2. Afterwards, two types of networks were used, a Long Short Term Memory (LSTM) and a Convolutional Neural Network (CNN) developed with TensorFlow [14], to perform the following experiments:

1. Compare the performance of VGGish versus MFCC on a Convolutional Neural Network (CNN).
2. Compare the performance of VGGish versus MFCC on a Long Short Term Memory (LSTM).

These experiments have been chosen in order to test the effectiveness of the audio features VGGish, i.e. evaluate if they carry enough information to predict VAD values. MFCCs are used for the comparison as they have widely been used across the literature with promising results.

Another method commonly used is extracting the spectrograms from the audio and used them to train a CNN. The problem with this approach is that the audio inputs are treated as images, losing their temporal relationship. It is explained more in detail in Chapter 2.

It was assumed that the best experiment would be training the VGGish features on a LSTM network, as these features carry more relevant audio information than the others since they have been trained on a larger dataset. Also, that the LSTM performs better with the temporal relationship relevant when dealing with audio.

1.6 Delimitations

To compare the emotional added value of the soundtrack to the overall of the movie is out of the scope of this project, since it would be needed a very complex dataset where the soundtracks are matched with their movies. The focus of this project is to evaluate which of the two features earlier mentioned performs better in predicting the VAD values. The performance of these features will be measured with two different models -LSTM and CNN-.

1.7 Outline

The project is structured as follows. In Chapter 2, the emotional model employed and why it was chosen is explained, as well as the theoretical background of the two network models and the two features used to perform the experiments. The chapter ends with a summary of the previous relevant related work. The description of the database acquisition and its validity is described in Chapter 3 and, in Chapter 4, the process of how the model was developed. To finalize, in Chapter 5 the results are exposed and discussed and, in Chapter 6, the conclusions and some of the future lines of research.

Chapter 2

Background

This chapter provides basic information and background of what has been previously done in the area of MER and, also with some concepts to fully understand the development of the project.

Machine Learning and Deep Learning approaches are the ones that perform the best when extracting emotions from music. Firstly, there is a brief introduction about the emotional model employed, followed by the explanation of the two network models utilized. Then, the audio features and the metrics utilized measure the results are described.

2.1 Emotional model

Traditionally, the most common way of defining emotions has been with mood labels. The first six emotions defined by Paul Ekman [15] were **sadness, happiness, disgust, anger, fear and surprise**, then they were enlarged across the years by different researches. This is defined as a categorical approach. Datasets such as the *4Q audio emotion dataset* [16] or the *MER500* [17] use this kind of labelling. The problem with this approach is the limited number of mood labels that can be employed.

Therefore, Russell in [13] proposed a dimensional model where emotions could be quantified in terms of valence and arousal. Later on, another dimension was added, the dominance. In Figure 2.1 some common emotions are represented using these labels. They are defined as:

- **Valence** expresses if an emotion is positive or negative, i.e. the pleasure the emotion transmits.
- **Arousal** refers to its affective activation, i.e. the intensity or activity.
- **Dominance** reflects the level of control of the emotion, i.e. the potency.

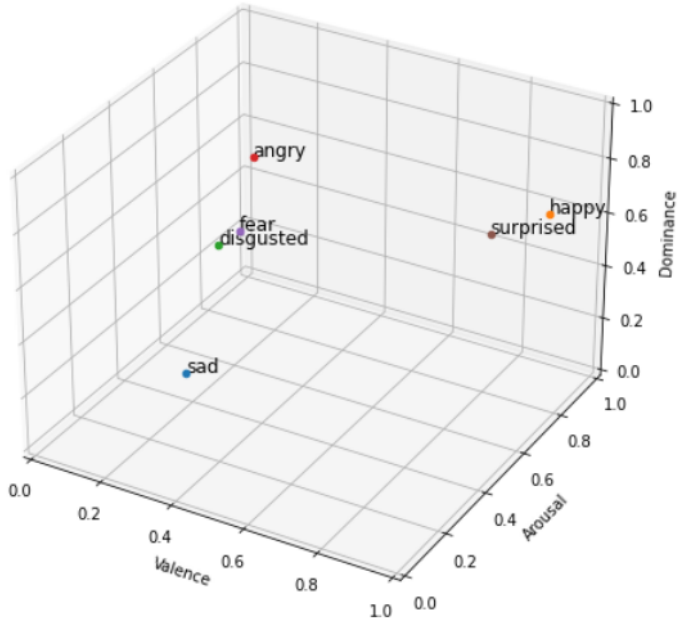


Figure 2.1: VAD values for some emotions taken from the NRC dataset.

Most of the databases of songs across the literature are labelled just with valence and arousal. The problem is that, without the dominance label, relevant emotional information is lost. For instance, strong crying or fury could not be differentiated without dominance. A couple of examples of these datasets are *PMEmo* [18] and *AMG1608* [19].

After some research, two suitable datasets with VAD values were found, one with 470 clips of soundtracks [20] and another one with 61902 general songs, Music Sentiment Dataset (MuSe) [21]. For the project, a customized version of the last one was created, explained more deeply in Chapter 3.

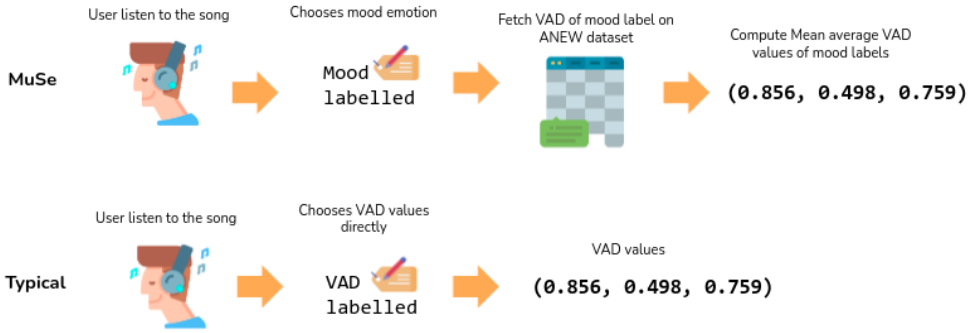


Figure 2.2: Comparison of how MuSe is labeled with respect to other more traditional datasets.

The hypothesis taken into account when using the customized MuSe dataset is that the VAD values are generated from the mood labels corresponding to each song, not actually from the users when listening to a specific song. The authors of this dataset utilised a another dataset that has VAD values for the majority of English words, the ANEW dataset [22], so in MuSe each song has the VAD values that come from the average of the VADs of all the mood labels that are associated with it. The process is illustrated in Figure 2.2 for its better understanding.

For this project, a more modern version from 2018 of the English words dataset is employed, the NRC dataset [2]. It has a total of 20,000 words with their corresponding VAD labels and it was generated in a more accurate way than its older version, ANEW.

In Table 2.1 there are some examples of mood labels with their VAD values retrieved from NRC dataset.

Table 2.1: Mood labels with their VAD values from NRC dataset [2].

Word	**V**	**A**	**D**
Sad	0.225	0.333	0.149
Happy	1	0.735	0.772
Disgusted	0.051	0.773	0.274
Angry	0.122	0.83	0.604
Fear	0.073	0.84	0.293
Surprised	0.784	0.855	0.539

2.2 Network models

In this section is explained the theoretical framework of the network models employed in the experiments, the CNN and the LSTM.

2.2.1 Convolutional Neural Network (CNN)

CNNs are the type of neural networks designed to outperform on image processing or classification [23]. They have the ability to deal with a bi-dimensional input, such as an image, instead of transforming it to a uni-dimensional vector thanks to the convolution operation with filters performed on each neuron. In addition, the spatial relation between pixels is understood when extracting different features, usually borders on the first layers and details on the deeper ones.

Another type of networks are Dense Neural Networks (DNNs) which are composed of several layers with a different number of neurons on each one. All neurons from one layer connect to all the neurons from the next layer, reason why they are also called fully connected. Data passes from the first input layer to the rest, the hidden layers, until it arrives to the output layer, procedure named *feed-forward*. On each neuron, the weights are updated and optimized, applying an activation function to introduce non-linearity.

In the beginning, only a DNN was going to be employed in the project, but after extracting the features, the number of data points to train was more complex than expected. Therefore a CNN was chosen becuase it reduces the number of parameters.

2.2.2 Long Short Term Memory (LSTM)

CNNs struggle to deal with the temporal relationship of the input data when performing the training. Recurrent Neural Networks (RNNs) [24] try to solve this problem by adding another input of each neuron, which is the output of the previous one. Basically, it provides the network with memory. The disadvantage is they do not handle correctly long-term dependency because the new information entering the network acquires more weight than the gathered memory. This problem is called *vanishing gradient problem* [25].

Hochreiter et al. [26] introduced LSTM to solve it. The main task of these networks is solving the long-term dependency problem. They also keep the information from previous neurons, similar to RNN, but with the difference that each neuron has three gates; the input gate, the forget gate, and the output gate. Thanks to them, the neuron can decide how much relevance it gives to the input information and how much previously gathered information it wants to remove [27].

2.3 Audio features

Extracting audio features, rather than introducing directly the audio raw signal to the networks, is a more convenient method in terms of simplicity and good performance according to previous works. MFCC are widely studied across the literature and, therefore, they are a secure approach to perform a comparison with VGGish features, novel in the area of MER.

2.3.1 Mel-frequency Cepstral Coefficients

These coefficients are widely used in audio processing, both in speech or music recognition. Briefly, MFCCs are an alternative representation of the spectrum of an audio signal.

Cepstrum represents the rate change in spectral bands. In normal spectrum analysis, a periodic signal in the time domain is represented with a peak in the frequency domain. Afterwards, when calculating the spectrum of the log of this original spectrum of the audio input, it can be observed a peak wherever there is a periodic element in the original time domain. They named this resulting spectrogram Cepstrum because it belongs to a completely different domain, neither frequency neither time, this new domain was named *quefrency*.

The next important concept to clarify is the *Mel scale*. It is a way of relating the frequency of the signal with how humans perceived the sound it generates. The *Mel scale* was defined in order to emphasize small changes at lower frequencies as they are better heard by humans.

The final reason for this process is that any human sound is determined by the shape of their vocal, and therefore knowing this shape any sound can be

accurately reproduced, which is the ultimate purpose of the MFCCs.

The process to extract these coefficients, Figure 2.3, is the following:

1. The signal is split into short frames according to the window size specified. In this project it has been used a window of 2048 frames, which sampled at a sample rate of 22050 Hz corresponds approximately to 93 ms per window. These frames are normally overlapped, defined by the parameter *hop length* which is the space between windows.
2. For each frame the Discrete Fourier Transform is calculated, generally with 512 FFT points.
3. The Mel-spaced filter bank is computed, a set of typically 26 filters that are applied to the DFT calculated in the previous step. There are more filters for lower frequencies to emphasise them as has been explained previously.
4. The log of the output of the computation of the bank of filters with the DFT from step 2 is computed.
5. Then, the DCT of the log filterbank energies, which are the output from the previous step.
6. Normally only the first 13 coefficients from the 26 generated with the filter banks are kept. The rest are values close to zero due to the DFT from step 2, which concentrates the information in the first coefficients.

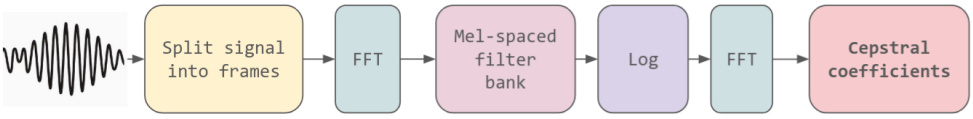


Figure 2.3: Chart explaining how MFCC features are obtained.

2.3.2 VGGish

VGGish is a variant of the VGG model [28] to enable it to receive long Mel spectrograms audio inputs. The Mel spectrograms are representations in the frequency domain of the audio signal converted into the Mel Scale. This scale,

as mentioned in the explanation of MFCCs, is a non-linear transformation of the frequency scale to adapt it to how humans perceive sounds. The idea is that the distance in pitch -how frequency is named in speech and music processing- sounds equal to the listener, despite the frequency. As humans are more sensitive to lower frequencies, the changes are less abrupt on lower than on higher frequencies, see Figure 2.4.

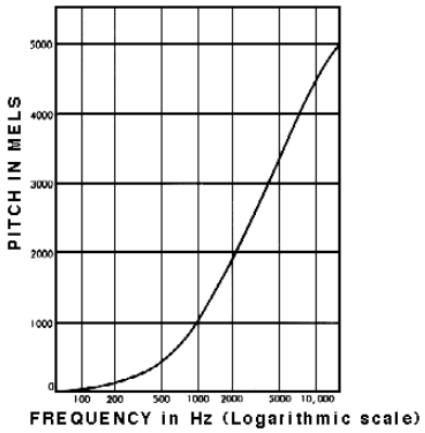


Figure 2.4: Mel Scale taken from [1].

The VGGish model has been trained on the YouTube-8M dataset [29] and generates 128-D dimensional embeddings for approximately 1 second of each audio input. There are two ways of using this model, first as part of a larger model, normally employed for training larger or more complicated audio datasets, and secondly, as a feature extractor. The audio input is framed in windows of 0.96 seconds and for each one, a high-level 128-D feature is extracted resulting in a 2D tensor of size [N, 128], with N the approximate number of seconds of the audio input. These features seem promising as they have more semantic information than raw audio features. The model is available in TensorFlow Hub. The limitation is that it has been trained on millions of YouTube videos, and therefore, if the audio inputs, the soundtracks, are very different from those, the result would not be as promising as expected.

2.4 Metrics

A regression algorithm has to be evaluated with specific measures for it, explained in detail in this section. Mean Square Error (MSE) and Mean Absolute Error (MAE) correspond to the negative measure that evaluates the loss, the lower the better.

2.4.1 Mean Square Error

The MSE is the standard deviation of the prediction errors. It measures how far are predictions from the true measure, i.e. the Euclidean distance between prediction and ground truth. The formula is defined as follows:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}\left(y_i - \hat{y}_i\right)^2$$

N corresponds to the number of data points, y_i to the true value and $\hat{y}_i$ to the predicted value in i. This metric is useful when big errors need to be punished. It focuses more on them thanks the squared difference.

2.4.2 Mean Absolute Error

MAE is defined as the sum of the absolute of the difference between the real value and the predicted one divided by the total number of predictions N. This metric is widely used in regression models. In comparison with MSE, it has a linear behaviour. The formula is defined as follows:

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i|$$

2.5 Related work area

In the past years, many researchers have developed different algorithms to solve the problem of extracting emotions from music. These works can be classified according to different characteristics such as the audio features used as inputs, the classification algorithm employed or the output labels. In this project, the literature is divided according to the classification algorithm applied.

Support Vector Machine (SVM) is employed as a classifier in several works, for instance in [30], in order to determine to which emotion a particular song belongs within a given dataset. They used Russell's model to then convert the VAD values into four emotional categories. This project achieves a high accuracy but the labels are not detailed enough to consider it a good result. Panda et al. [16] also applied SVM to classify emotions into four quadrants after a high preprocessing of the audio features, similarly done in [31]. Chapeneri et al. [32] applied a deep Gaussian regression model on a dataset labelled with valence and arousal as well and they used Bayesian acoustic features as inputs. Delbouys et al. [6], instead of analyzing the audio features to extract emotions, focused on the lyrics, embedding words to be the input of the classifier. The best performance is achieved when merging audio and lyrics features, employing SVM as well as other classical model for classification.

Machine learning algorithms are widely used in this area, as well as in many others. For instance, in [33] a non-invasive brain-machine technique, electroencephalography, is used to evaluate the brain reactions when listening to music. Then, they classify the emotions extracted applying a DNN. DNN are a basic neural network that has been widely applied in works such as [34] were they also use the valence and arousal model. They compare the behaviour of SVM and DNN, obtaining better results with this last one. In [35] several types of DNN are compared, arriving to the conclusion that ResNet50 [36] performs outstandingly compared to the rest of DNN models.

Good results are achieved with this approach but with the disadvantage of high preprocessing data. Therefore, a large number of studies have focused on evaluating CNN, due to its effective performance on images. The most popular approach is to extract the spectrograms from the audio, and use them as the input for the CNN [5][37]. The disadvantage of these networks is they do not exploit the temporal relation of the audio. They just focus on the spatial relationship between the input pixels, reason why they perform so well on

images.

To deal with this problem Begio et al. [24] introduced the RNN. They achieved a better performance on audio files, specially the LSTM networks [26], a type of RNN. Different audio features can be used as inputs, from raw audio [10], passing through low-level acoustic descriptors -loudness, sharpness, harmonicity, energy, etc.- to MFCC [27].

More recently, comparisons between deep audio embedding methods have been performed in order to analyze which one carries more emotional semantic information. In [38] they compare L^3-Net embeddings with VGGish, with the result that the first one contains more useful information for the purpose of predicting the MER.

Chapter 3

Methodology

This chapter describes how the database was created, its reliability and validity and why it was chosen this approach.

3.1 Database creation

The database created for this project followed the same procedure as the authors of [21] did for the creation of the MuSe (Music Sentiment) database, but retrieving only soundtracks. The procedure has been the following:

1. **Collection of soundtracks**: All songs from Last.fm with the tag *Soundtrack* were retrieved from the API[1], in total 9550 soundtracks.
2. **VAD for English words dataset**: In the MuSe database the authors used the ANEW database [22] which contains 13,915 lemmas with their correspondent VAD value to calculate the VAD labels. In this project a newer version of this database with more than 20,000 words was employed, the NRC-VAD dataset [2]. The score of the VAD labels ranges from 0 to 1 and they were acquired in a more reliable way. More details about it can be found on the provided link [2].
3. **Retrieval of all tags from the soundtracks**: Again with the use of

[1] https://www.last.fm/api/show/tag.getTopTracks[2] https://saifmohammad.com/WebPages/nrc-vad.html

Last.fm API[1], all the tags with their correspondent weights for each soundtrack were retrieved.

4. **Filter the genre tags**: From the retrieved tags, the ones corresponding to genres did not contribute to the emotional information. Therefore, they were filtered before calculating the final VAD values of the soundtracks. Some examples of these tags were: soundtrack, movie, electronic, indie, rock, pop, country, film, composer, intro, musical, cover, etc.

5. **Calculate VAD for each soundtrack**: As it has been done for the MuSe database, for each mood tag its VAD values were taken from the ANEW dataset. In this case from NRC database, and later the mean average was calculated by multiplying with their correspondent weight. Some tags were not in the NRC dataset resulting in 7413 songs with VAD annotations. An example of how it was calculated the mean average is shown in Figure 3.1.

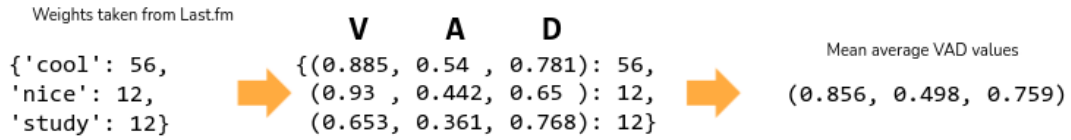


Figure 3.1: Mean average VAD values from *Simple as This* by Jake Bugg.

6. **Get their Spotify ID**: With Spotify API[2] the Spotify ID for each soundtrack was added to the dataset, together with the duration of the song. Some of the songs were not available on Spotify and therefore the number was reduced to 6826 soundtracks, but still, a very significant number to train correctly a network. Table 3.1 presents a summary of the final output of the dataset.

[1] https://www.last.fm/api/show/track.getTopTags[2] https://developer.spotify.com/console/get-search-item/

Table 3.1: Final structure of the dataset created, the columns are: *id, track, artist, duration, weights, spotify_id, valence, arousal and dominance.*

track	artist	...	V	A	D
Only hope	Mandy Moore	...	0.833	0.462	0.551
The End Of The World	Skeeter Davis	...	0.555	0.446	0.423
Lovefool	The Cardigans	...	0.644	0.410	0.522
Nobody Does It Better	Carly Simon	...	0.818	0.461	0.552
Blow Me Away	Breaking Benjamin	...	0.673	0.610	0.658

7. **Get the audio using Savify**[1]: With the Spotify ID and the use of Savify, the audio was downloaded to later extract the features, MFCC and VGGish.

In Figure 3.2 all the VAD labels from the final database are plotted in a 3D space to visualize their distribution. Some parts of the space have no value because there are some VAD combinations that do not correspond to any real emotion.

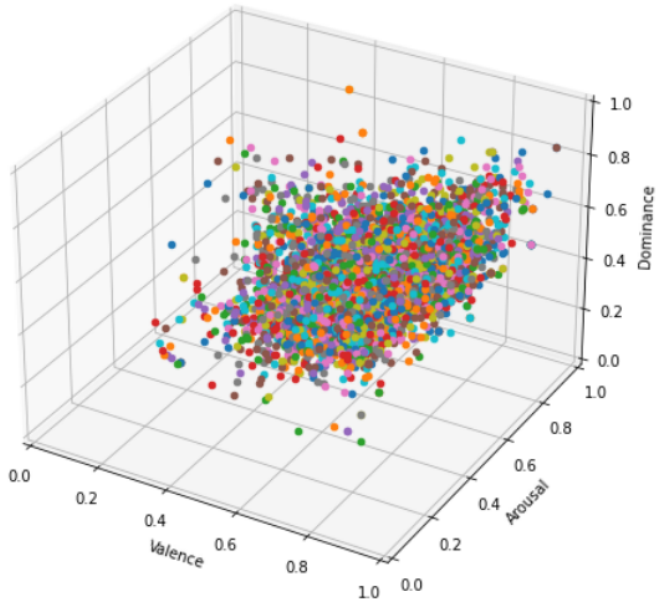


Figure 3.2: VAD labels from the whole database.

[1] https://github.com/LaurenceRawlings/savify

The number of available soundtracks was reduced from 6826 to 6757 for a couple of reasons. Some of their names started with punctuation marks so they could not be searched in Spotify and songs larger than 28MB had to be removed as they were very heavy for the program to be able to extract the features correctly.

In Figure 3.3 all the available fields of song number 11, *Simple as This* by Jake Bugg, can be observed.

```
id                                                    11
track                                        Simple as This
artist                                           Jake Bugg
duration                                            199453
weighted_mood        {'cool': 56, 'nice': 12, 'study': 12}
spotify_id                         59xYAblNy1Xr4U1IsJ3gZb
valence_tags                                      0.85695
arousal_tags                                      0.49845
dominance_tags                                     0.7594
Name: 11, dtype: object
```

Figure 3.3: *Simple as This* by Jake Bugg saved in dataframe format.

3.1.1 Hardware/Software used

The hardware used in the project was the laptop of the student, a Dell New Inspiron 13 5391 with 8GB of RAM and 512GB of memory and a Linux operating system, Ubuntu 20.04. Both the acquisition of the dataset and the extraction of the features were done locally, but the training of the models was done in Google Cloud Platform (GCP) with credentials provided by the company. From the resources available in GCP, a Machine type named *n1-standard-8* was selected, with 8 CPUs and 30GB of memory, in addition to a GPU, *NVIDIA_TESLA_T4*, when the models were trained with the whole dataset.

About the software, all the code was written in Python, the generation of the dataset, underlining the use of the library *pandas* to treat with data frames, and the architecture of the models using Tensorflow 2.4 [14]. The Integrated Development Environment (IDE) employed for developing the project was Visual Studio Code for Ubuntu.

3.2 Reliability and validity

The goal of this section is to justify the validity and reliability of the database created in first place, and of the training procedure in second place.

3.2.1 Database created

During the creation of the database, there were two steps where the labelling relied on users opinions and feelings: the weighted moods retrieved from Last.fm and the VAD labels assigned to each word.

Firstly, the songs were retrieved from Last.fm with their respective social-based tags. Those public tags correspond to the most voted ones by the users of the platform. They labelled them with the emotions felt after listening to the whole song. The drawback encountered here is they did not only tag the songs with the emotions felt but also with genres and other words that can describe music such as *alternative*, *blue*, *retro* or *contemporary*. For that reason, these tags had to be filtered before calculating the average VAD labels.

Afterwards, when translating the mood labels into VAD labels with the NRC database, the question that arised here was how these VAD labels were assigned by the users. According to the creators of this database, there are mainly four problems with rating scales: the fixed granularity, the difficulty for annotators to be consistent with themselves and with the rest, and the scale region influence. To deal with these problems they made use of the best-worst scaling [39] which leads to more reliable annotations according to several sources. Moreover, half of the annotations were compared to the other half to prove their mutual consistency, a process called 'Average split-half reliability (SHR)' [40]. It resulted in a higher similarity between both halves than the one achieved with the ANEW dataset [2].

In Figure 3.4 is displayed the final distribution of the calculated VAD labels.

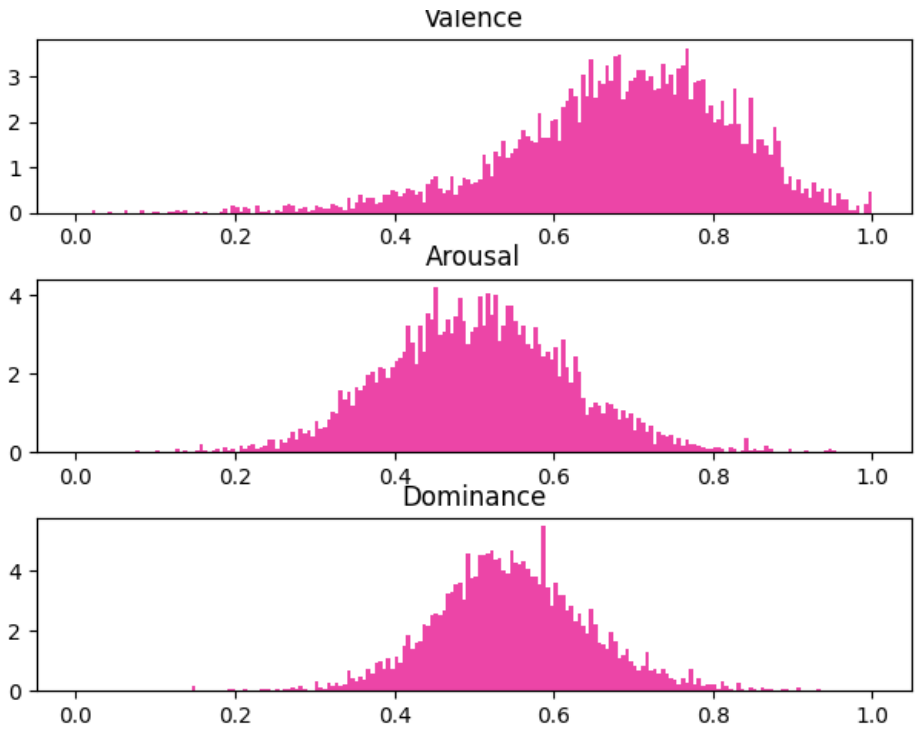


Figure 3.4: Histogram of Valence, Arousal and Dominance values from the database created.

It can be observed on the plots that the valence is slightly biased to the positive values,which it will impact on the results of the training.

3.2.2 Training configuration

The three VAD labels were trained separately even though they are not independent. According to [2], valence and dominance have a significant correlation (0.488), but for the remaining combinations, arousal-dominance (0.302) and valence-arousal (0.268) are quite low. They normally are assumed to be independent because is easier for the networks to converge.

As it is explained in the next chapter, the songs were split into frames of 15 and 30 seconds as input to the networks. All the frames belonging to the same song have the same VAD labels even though the beginning of the song could transmit different emotions compared to the middle part. The hypothesis

posed in this sense is that songs with an average relax feeling, for example, will contribute with a larger number of frames with the appropriate VAD values, compensating the intro frame of a heavy song that could be more relaxing and has VAD labels for a heavier and harder feeling. Therefore, when performing the test, even though all the frames from each song should be predicted with the same ground truth, each predicted VAD for each frame will approximate more to measure the emotion felt in those seconds, but in average they will tend to the ground truth.

3.2.3 Metrics

Both metrics, MSE and MAE were calculated for all the experiments. As mentioned previously, MSE punished bigger errors, unlike MAE that works better on a database with outliers. In this case, the outliers are important as they will be the songs with the most defined emotions, and therefore the MAE metric is more accurate when measuring the loss.

Chapter 4

Developing the model

This chapter describes how the model was developed to achieve the results that are presented in Chapter 5. Firstly, how the features were extracted is explained, which is followed by the description of how the architectures of the models were tuned.

4.1 Features extraction

The theoretical explanation of the features employed in the project has already been described in Section 2.3. This section presents how the MFCCs and the VGGish were concretely extracted for this project.

4.1.1 MFCCs

A popular Python package used in the state-of-the-art for music and audio analysis is *Librosa* [41]. It is very convenient because the MFCCs can be extracted just by applying the following function.

```
mfcc = librosa.feature.mfcc(x, sr=sr)
```

The song of Figure 4.1 is utilised as an example to explain the output dimension from the MFCCs, important for later understand the input dimension

to both models.

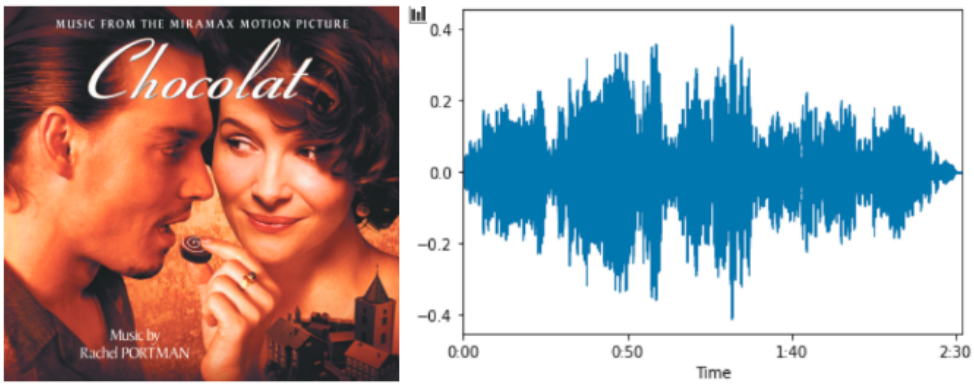


Figure 4.1: *Passage of Time* by Rachel Portman, song number 8857 from the dataset with a duration of 153 ms.

Firstly, when loading the audio with Librosa, it samples it at the default sample rate of 22050 Hz. Therefore, the number of frames in this song is:

$$\#Frames = duration[seconds] * sr \left[\frac{frames}{seconds}\right] = 152,137 * 22050 = 3354620,85$$

Afterwards when applying the MFCCs, the sampled audio is divided into windows, with a window size of 2048 and a hop length of 1024. It represents the space between windows so they are overlapped, graphically explained in Figure 4.2.

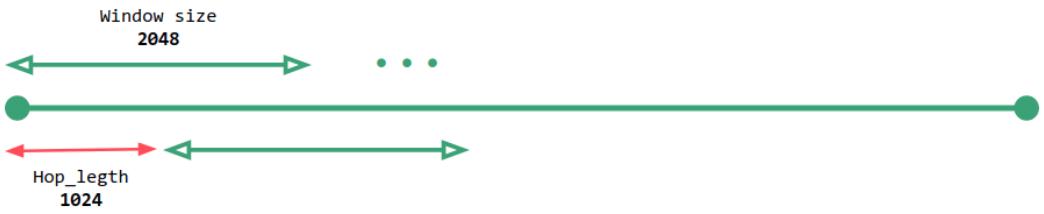


Figure 4.2: Representation of how the MFCCs windows are divided.

Therefore, by dividing the total number of frames of the song by the hop length, the result is the number of windows generated.

$$\#Frames\ MFCCs = \frac{duration[sec] * sr\ \left[\frac{frames}{seconds}\right]}{hop\ length[frames]} = \frac{152,137 * 22050}{1024} = 3276$$

From each of these windows, 13 MFCCs are generated following the steps defined in the theoretical part in Section 2.3. As a result, the final output size is for the song treated in this case is **(13, 3276)**. Figure 4.3 shows the resulting spectrogram. The horizontal axe refers to the time length which is half of the total audio length due to the hop length selected of 1024, and the vertical the MFCCs coefficients.

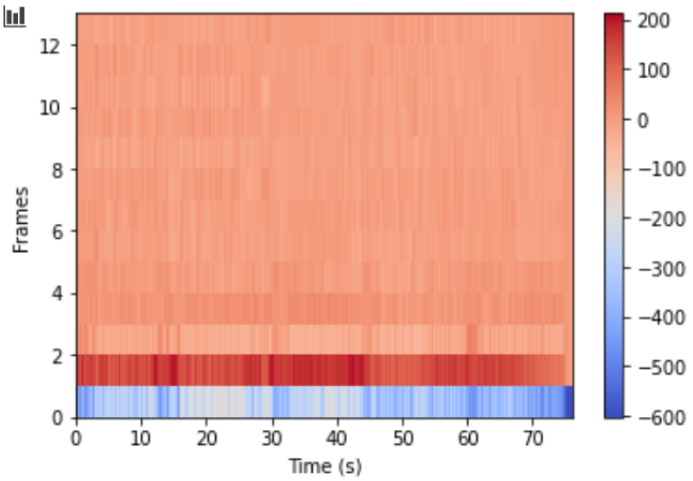


Figure 4.3: MFCCs coefficients of the song 8857.

Treating different audio length as input for the networks is out of the scope of this project, reason why the experiments were defined to employ audio inputs of 15 and 30 seconds. The MFCCs output was split in **(13, 323)** for 15 seconds and **(13, 646)** for 30 seconds. These numbers were computed following the same formula as before but changing the audio duration:

$$\#Frames\ 15s = \frac{duration[sec] * sr\ \left[\frac{frames}{seconds}\right]}{hop\ length[frames]} = \frac{15 * 22050}{1024} = 323$$

$$\#Frames\ 30s = \frac{duration[sec] * sr\ \left[\frac{frames}{seconds}\right]}{hop\ length[frames]} = \frac{30 * 22050}{1024} = 646$$

After processing the MFCCs for all the songs, they were converted into *tfrecords* for an efficient training in Google Cloud Platform.

4.1.2 VGGish

The approach followed to extract these features was simpler compared to the MFCCs as the model is saved in Tensorflow Hub and it only had to be loaded and applied to all the songs from the database. However, as pointed out in the documentation of the model, the audio was trained with a sample rate of 16k Hz, and therefore, when loading the audio files with Librosa to extract the VGGish, it had to be specified this new sample rate so it did not employ the default one, 22050 Hz.

For the song of Figure 4.1, the output size of the VGGish is **(158, 128)**. In Figure 4.4 is represented the output. These features had to be sliced as well to take 15 and 30 seconds. In this case, with a direct computation as the first dimension of the output of VGGish is approximately one second (0.96 seconds). Hence the slices chosen were **(15, 128)** for 15 and **(30, 128)** for 30 seconds.

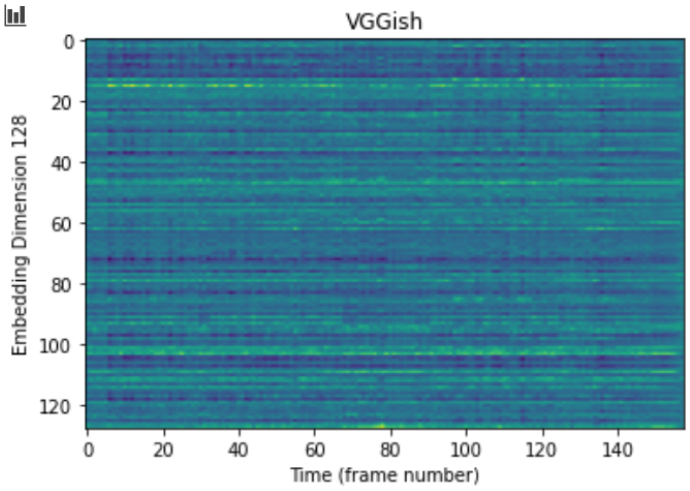


Figure 4.4: VGGish from song 8857.

4.2 Architecture experimentation

This section describes how the final architecture of both models, CNN and LSTM, was developed. Several configurations were analysed applying the MFCCs features, with the final goal of finding the architecture that achieves the best result, for later test the VGGish on that same one. The hypothesis, stated before, is that the VGGish will perform better than the MFCCs.

4.2.1 CNN

Initially, the approach was to use a DNN instead of a CNN. The reason behind it was that the features have already extracted the relevant information from the audio and therefore, the model architecture was not required to be very complex for it to converge. This approach was soon discarded after finishing with the features extraction and understanding their shapes. Because they generated a high number of parameters in comparison to the available data inputs. Replacing the first layers with convolutional ones reduced the number of trainable parameters and, consequently, the relation between them and the number of data points was more reasonable and augured better results. More details about the number of parameters and the data input points are given in Section 4.2.3.

The final CNN architecture is the result of several experiments changing the number of convolutional, dense layers and their correspondent neurons. Firstly, the model had three convolutional layers followed by four dense, but it collapsed to one single value even though the loss was decreasing. The reason was the high complexity of the model for the concrete scenario.

Consequently, the following approach was to reduce the number of layers to 2 convolutional and 2 or 3 dense layers with few neurons, between 32, 64, or 128. When the model was too simple, with only 2 dense layers, it was not able to learn the VAD labels. Some trials with Dropout and Batch normalization were executed as well but with no improvement in the results.

The trade-off between collapsing and not learning enough was found in a model with 2 convolutional layers and three dense with 128, 128 and 64 neurons respectively. The final architecture is shown in Figure 4.5.

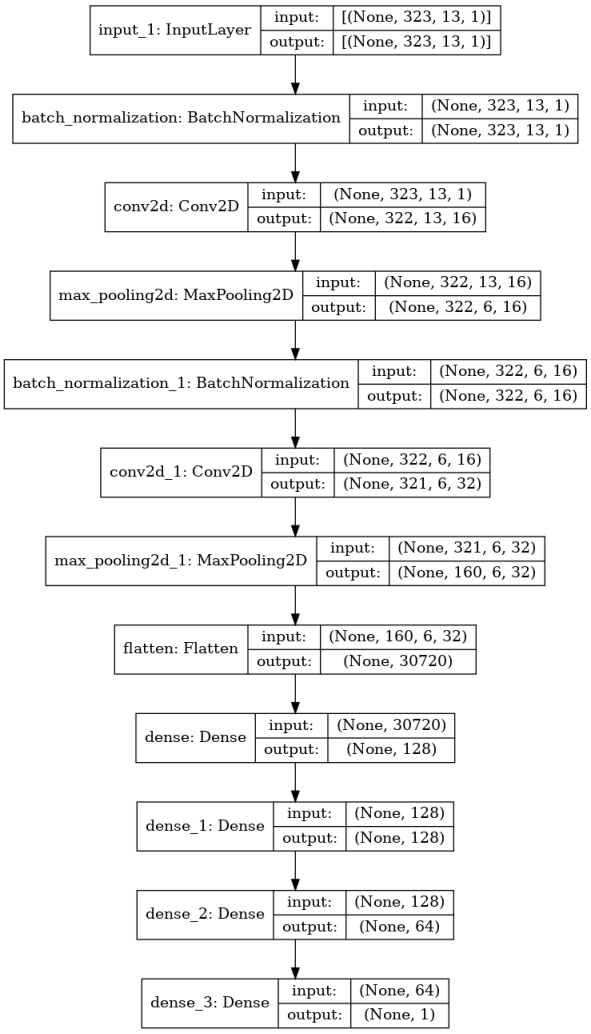


Figure 4.5: Final architecture of the CNN model.

4.2.2 LSTM

For the LSTM model, the experiments were made changing the number of LSTM neurons and followed by the same three layers used in the CNN, with 128, 128 and 64 neurons respectively. First, high numbers of LSTM neurons, such as 100, were tested but with no success as the number of trainable parameters was very high compared to the amount of data inputs. Instead, with a low number such as 10, the model could not converge. The trade-off was found in 512 neurons. In Figure 4.6 the final architecture is represented.

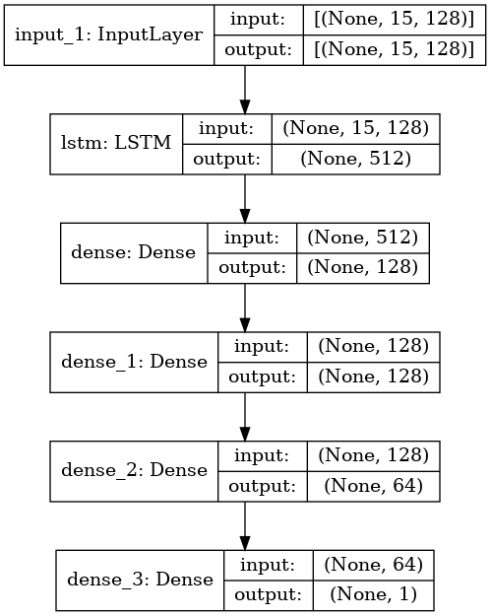


Figure 4.6: Final architecture of the LSTM model.

4.2.3 Number of parameters clarification

The purpose of this explanation is to prove that the number of input data points is higher than the number of parameters but keeping a reasonable relation. A rule of thumb is that the ratio between both should not be larger than 100. If the number of parameters is higher than the data inputs the model will overfit.

To calculate approximately the number of data inputs, first, the mean duration of the songs from the training set -4000 songs- was computed, resulting in 225 seconds. Dividing it by frames of 15 seconds resulted in 15 frames on average for each song. As follows is explained how they were calculated for each type of feature.

- For the MFCCs: As the input size for each frame is (323, 13) with 15 frames per song and 4000 songs for the training, the data inputs resulted in

$$4000\ songs * 15\ frames * 323 * 13 = 251940000\ data\ inputs$$

  - CNN: # parameters 3,958,344. Relation with data inputs ≈ 63.

  - LSTM: # parameters 1,167,745. Relation with data inputs ≈ 215.

- For the VGGish: The input for each frame is (15, 128), also with 15 frames per song and 4000 songs for the training. Therefore,

$$4000\ songs * 15\ frames * 128 * 15 = 115200000\ data\ inputs$$

  - CNN: # parameters 1,599,048. Relation with data inputs ≈ 72.
  - LSTM: # parameters 1,403,265. Relation with data inputs ≈ 82.

Chapter 5

Results and Analysis

This chapter contains the results of the experiments conducted with the purpose of finding out which of both features presented and which model is able to predict the VAD labels with lower loss. It is divided into two sections according to both features.

As mentioned previously, even though they are correlated, the three labels are trained separately since it is easier for the network to converge.

5.1 Experiments with MFCCs

After reaching the final architecture of both networks, the following task was to find the most appropriate way to input the audio features. For this purpose two parameters were chosen:

1. **Frame length**: It was chosen according to three facts. First, it had to be long enough for the emotion to be recognizable by the listeners. Secondly, the log Mel spectrum should be computed on a long frame, if not relevant information could be lost. And lastly, as the dataset VAD labels are labelled for the whole songs, the longer the frame the more the predicted value resembled the ground truth. The trade-off values found satisfying these requirements were 15 and 30 seconds.
2. **Remove offset**: When calculating the DCT of the MFCCs, an offset

with no information is generated, so according to previous literature, removing it will result in a more accurate prediction.

Four experiments were executed according to these two parameters on the LSTM model, to discover which parameter combination outputs the lowest loss. Afterwards, the same configuration was employed on the CNN. The results are shown in Table 5.1. The lowest loss is achieved with a frame of 15 seconds and without removing the offset. The rest of the experiments were carried out under this configuration.

Table 5.1: Results of MFCC trained on the LSTM.

Frame	**Offset**	**Label**	**MSE**	**MAE**
15s	True	val	**0.0153486**	**0.0965546**
30s	True	val	0.0154745	0.0970154
15s	False	val	0.0154395	0.0965895
30s	False	val	0.0156131	0.0974047

The following graphs show the Tensorboard for valence, arousal and dominance. First trained on the CNN, Figure 5.1, and then on the LSTM, Figure 5.2. The plot underneath the *epoc_loss* graphs compares the predicted value -vertical axis-, with the true value -horizontal axis-, when running it on the validation set. For a perfect prediction, the values should be distributed across the plot's diagonal.

On Figure 5.1 by observing the bottom graphs it can be deduced that the CNN does not learn properly the data as it tends to predict the average value.

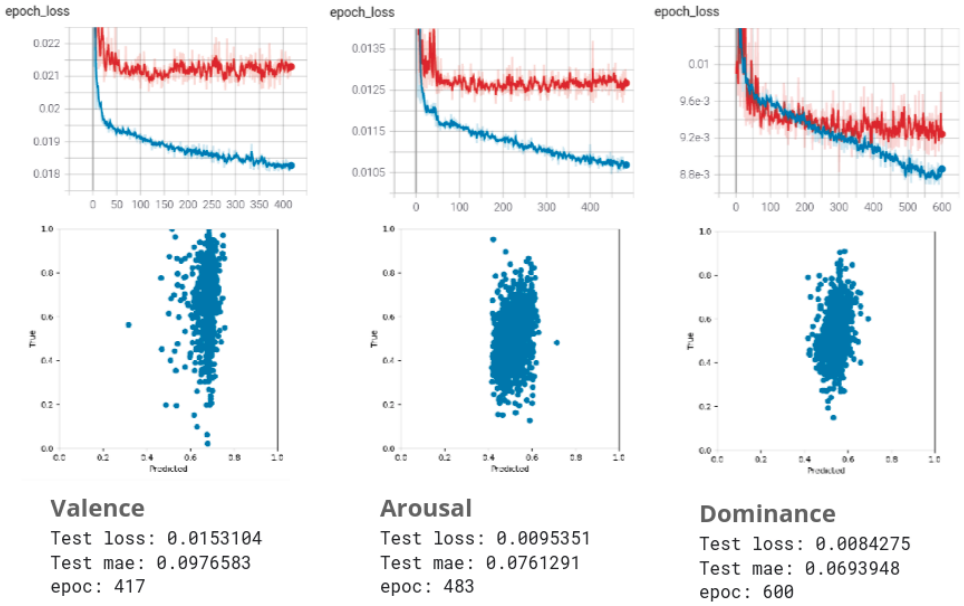


Figure 5.1: VAD results on CNN with MFCCs, with frame of 15 seconds and without removing the offset.

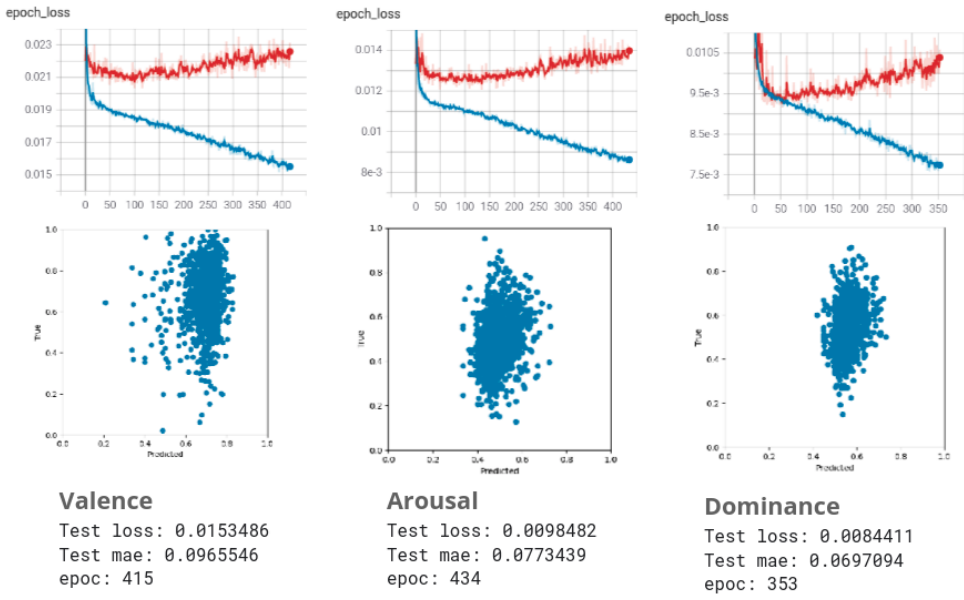


Figure 5.2: VAD results on LSTM with MFCCs, with frame of 15 seconds and without removing the offset.

The results on the LSTM were not as expected since they achieved a higher loss than on the CNN when training the three labels. However, it can be observed in the bottom graphs of Figure 5.2 that the data is more spread out compared to the plots of Figure 5.1 of the CNN, but still far from approaching the diagonal.

The emotion that achieves the best result is the dominance, both on the LSTM and the CNN. One reason for this is that it has the narrowest distribution, as can be observed in Figure 3.4. The predicted distribution for the three labels displayed on the bottom plots resembles the range of each emotion label, for more details about these ranges refer as well to Figure 3.4.

5.2 Experiments with VGGish

The VGGish features were introduced in the networks directly with frame of 15 seconds to perform a proper comparison with the MFCCs. The results for valence, arousal and dominance executed on the CNN with this feature configuration are displayed on Figure 5.3, and the ones ran on the LSTM in Figure 5.4.

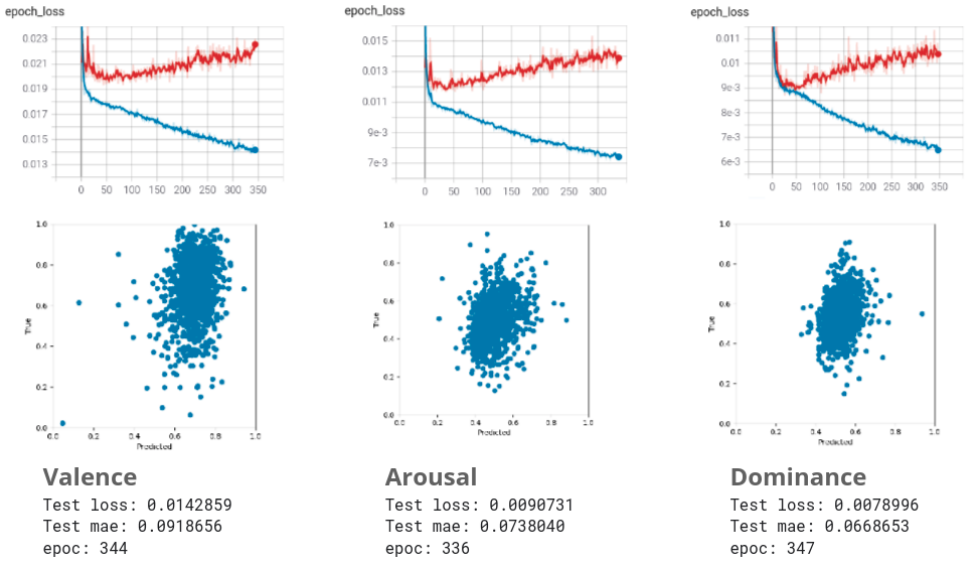


Figure 5.3: VAD results on CNN with VGGish.

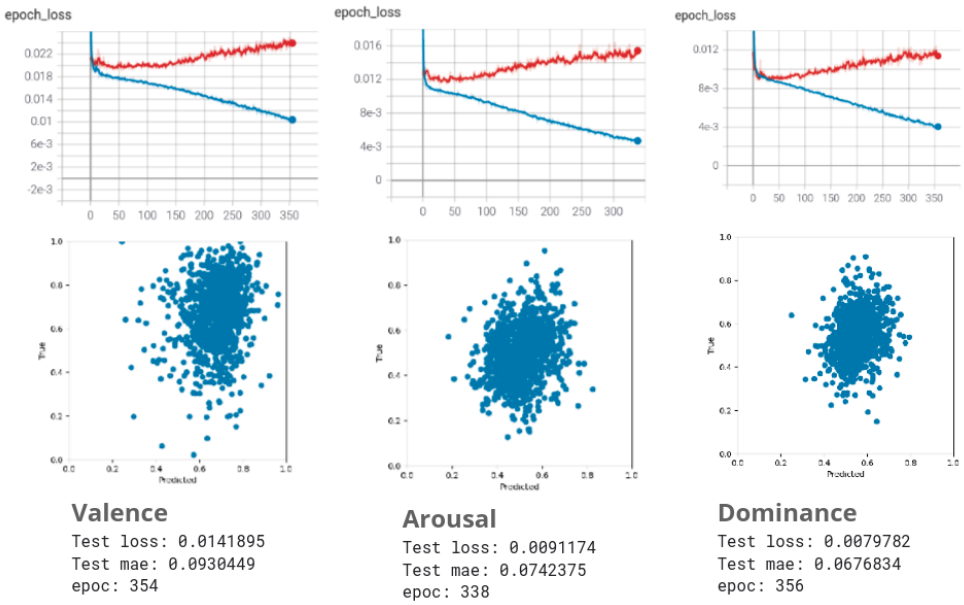


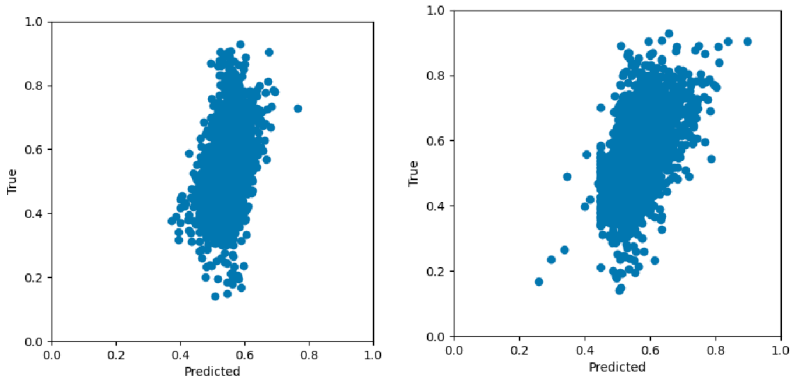
Figure 5.4: VAD results on LSTM with VGGish.

In this scenario, the LSTM achieves a lower loss than the CNN as it was posed initially. Furthermore, empirically observing the bottom plots that compare the predicted with the true value, it can deduced that more values are placed across the desired diagonal. Additionally, they present a higher variance and values further away from the average value as it occurred with the CNN.

Equal to the MFCCs features, the one that achieves the lowest loss of the three labels is the dominance which supports the previous interpretation of this happening because it is the one with the narrowest value distribution.

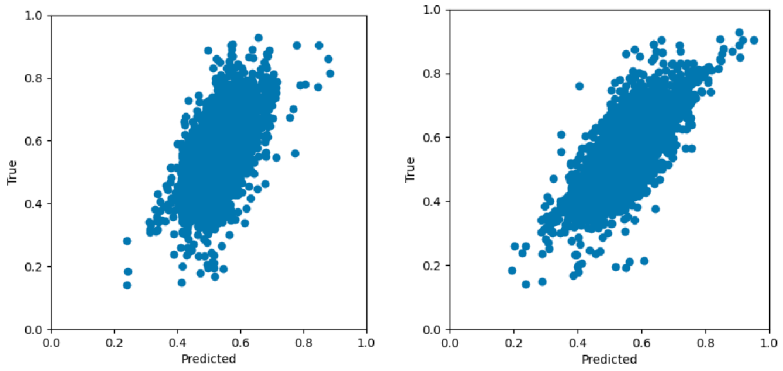
To conclude this section, the following plots show the performance of the training of the two models with the VGGish and MFCCs with the dominance, chosen for its lowest loss. The training plots anticipate the previous results. About the MFCCs, Figure 5.5, the CNN plot tends to the average value of 0.5 and the LSTM appears to collapse to one value close to 0.4, another reason why it is achieving a worse result.

On the other hand, Figure 5.6 shows the behaviour of the VGGish training where the LSTM is capable of learning the values across the diagonal considerable better than the CNN.



(a) CNN (b) LSTM

Figure 5.5: Training plots of the MFCC with dominance.



(a) CNN (b) LSTM

Figure 5.6: Training plots of the VGGish with dominance.

5.3 Summary of Results and Discussion

This summary aims to present the results for each emotion label comparing both networks and both features in a more synthesized way, together with a discussion about them.

Table 5.2: Results on Valence.

Network	Feature	Emotion	MSE	MAE
CNN	MFCC	val	0.0153104	0.0976583
LSTM	MFCC	val	0.0153486	0.0965546
CNN	VGGish	val	0.0142859	**0.0918656**
LSTM	VGGish	val	**0.0141277**	0.0919397

Table 5.3: Results on Arousal.

Network	Feature	Emotion	MSE	MAE
CNN	MFCC	aro	0.0095351	0.0761291
LSTM	MFCC	aro	0.0098482	0.0773439
CNN	VGGish	aro	0.0090731	0.0738040
LSTM	VGGish	aro	**0.0089451**	**0.0733317**

Table 5.4: Results on Dominance.

Network	Feature	Emotion	MSE	MAE
CNN	MFCC	dom	0.0084275	0.0693948
LSTM	MFCC	dom	0.0084411	0.0773439
CNN	VGGish	dom	0.0078996	0.0668653
LSTM	VGGish	dom	**0.0076934**	**0.0662203**

Regarding the results on the MFCCs, the MSE is lower when employing the CNN with the three emotion labels. The interpretation given is that, since the MFCCs demand a high pre-processing, the initial temporal information that the audio posses can be lost, provoking a poor performance on the LSTM. These features are evenly spaced to respect perception which resemble more the image concept and therefore contributes to the CNN to achieve a lower loss.

On the other hand, the pre-processing executed on the VGGish is not as complex as the MFCCs one, but as they belong to a pre-trained model on a large dataset of audio -the Youtube-8M Dataset-, their temporal relationship was not lost but rather enhanced, resulting in a better performance on LSTM.

The previous analysis was carried out with the MSE results. However, when concerning the MAE, it measures more accurately the valence as its distribution occupies a larger range in the space, providing the data with more outliers. As explained previously, with this type of data distribution the MAE performs a more accurate measure, and with more dense distributions, like the arousal and dominance one, the MSE performs better because the square difference punishes the outliers.

Based on these results and interpretations, the selected configuration to perform the final test is the VGGish features and the LSTM network.

The plots of Figure 5.7 show the results performed on the test set, 1600 songs. As mentioned before, the closer the points are to the red diagonal, the more accurate the prediction is. On the test set, they tend to the average value, which could be solved by increasing the number of data inputs or refining the architecture model.

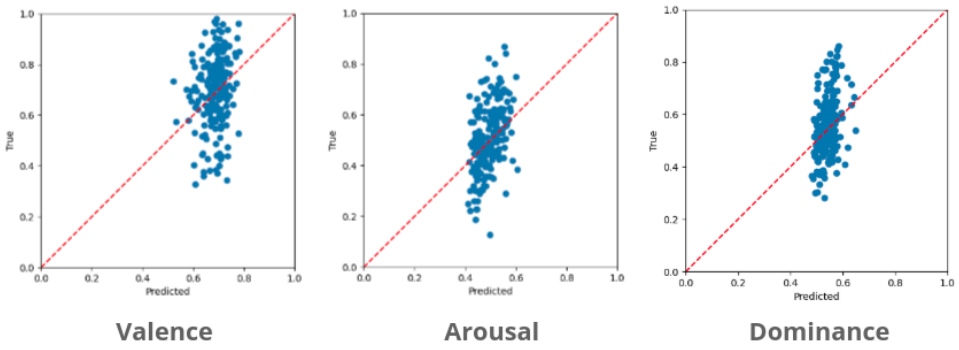
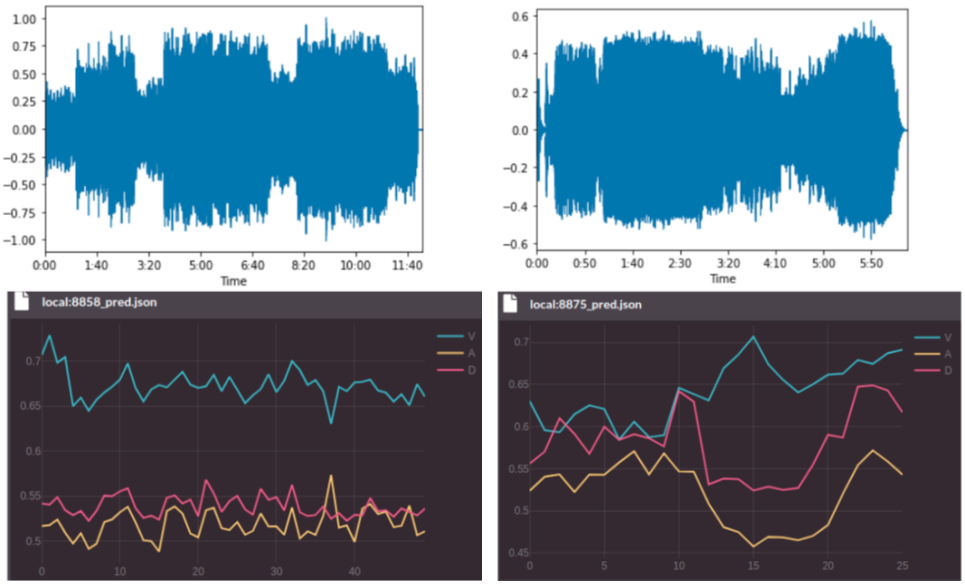


Figure 5.7: Plots ran on the test set, 1600 songs, with the configuration with the lowest loss, 15 seconds of VGGish frames on LSTM.

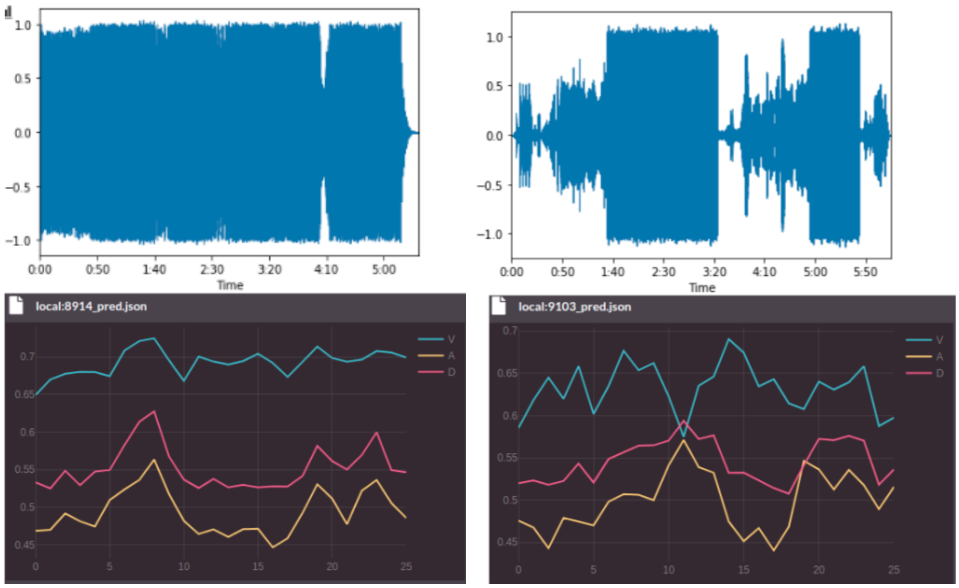
Below, some examples of the predicted VAD are represented in time series with their respective wave plot of the audio. A link to the song on YouTube is provided when clicking on the title of each song, together with the ground truth of the VAD labels.

It can be observed that abrupt changes in the wave plot, which correspond to changes in the listening experience, are predicted as peaks in the VAD space for each song. Additionally, it can appreciate the valence bias of the dataset towards positive values on these test songs.



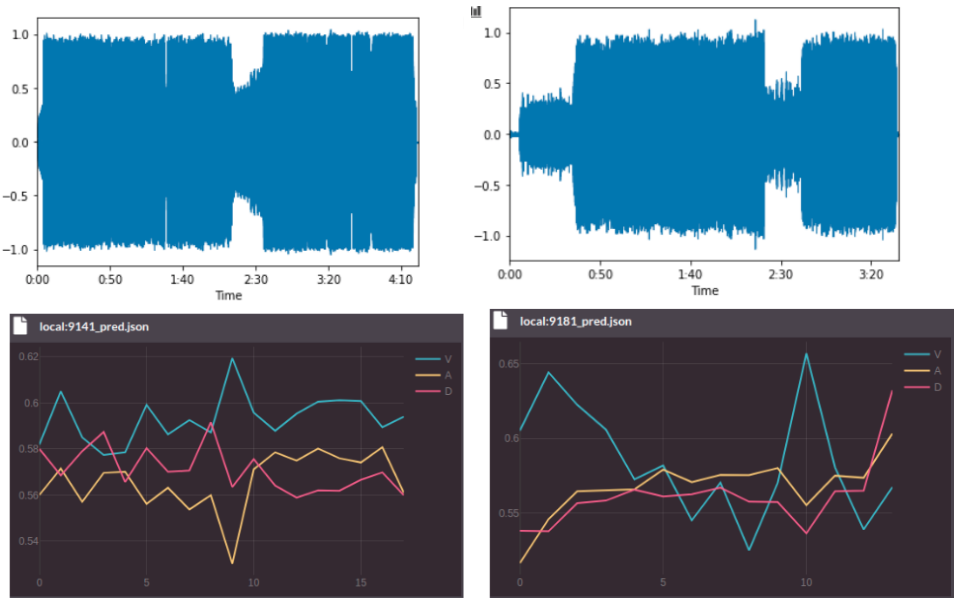
(a) VAD real values: 0.637, 0.559, 0.587 (b) VAD real values: 0.841, 0.731, 0.784

Figure 5.8: (a) Memorial - Michael Nyman, (b) The Tower That Ate People - Peter Gabriel.



(a) VAD real values: 0.762, 0.309, 0.464 (b) VAD real values: 0.659, 0.429, 0.494

Figure 5.9: (a) Poisoned Chalice - Hans Zimmer, (b) Taxi (Ave Maria) - John Murphy.



(a) VAD real values: 0.531, 0.583, 0.519 (b) VAD real values: 0.560, 0.502, 0.532

Figure 5.10: (a) Just Another Victim - Helmet House Of Pain, (b) Wunschkind - Oomph!

In Appendix A, the same results are plotted with a fixed range of [-1, 1] to better appreciate the bias of the valence towards the positive values, and to compare the six examples with fixed axes values.

Chapter 6

Conclusion and Future work

6.1 Conclusion

The goal of this project was to address the stated research question of understanding the amount of emotional information the soundtracks of the movies carry. For that purpose a database of 6757 soundtracks with their correspondent valence, arousal and dominance was created in order to train a model which could learn to predict these labels.

For that aim, two different models, CNN and LSTM, were tested with two different features, MFCCs and VGGish, to understand which configuration achieved a lower loss when predicting the VAD labels.

After performing several experiments, the conclusion obtained was that the VGGish features trained on an LSTM outperformed compared to the other four combinations, proving the initial hypothesis.

The gained insights are that time dependency matters when analyzing audio. The relation between the number of trainable parameters of the model and the number of data inputs is crucial to achieve the lowest possible loss. And features perform better when they are trained on more data -VGGish- rather than subjecting them to a high preprocessing -MFCCs-.

6.2 Future work

In this section, some of the possible lines that could be addressed in the future are presented.

Firstly, according to the dataset created, a good approach will be to filter the mood weights retrieved from Last.fm to keep the ones that transmit more precise emotions. It could be done by filtering them with the WordNet-Affect dataset, a set of words that refer just to emotions.

Additionally, finding a way of matching each soundtrack with its movie would be a very interesting future line. A database of those characteristics will contribute to understanding the emotional overall impact of the soundtrack in the whole movie, which is the ultimate purpose of this project.

Regarding the network model, it will be interesting to experiment with other architecture such as BiLSTM or Transformers, after proving that the LSTM performs way better than the CNN, and therefore, that the time dependency is relevant.

Lastly, different audio features could be tested as inputs. For example, another pre-trained model, *openl3* [42], from which audio embeddings can also be extracted and compared with the VGGish to analyze which one extracts more relevant information to later predict the VAD labels.

References

[1] R. Perera and O. Luening, *The development and practice of electronic music*. Prentice Hall, 1975.

[2] S. M. Mohammad, "Obtaining reliable human ratings of valence, arousal, and dominance for 20,000 english words," in *Proceedings of The Annual Conference of the Association for Computational Linguistics (ACL)*, Melbourne, Australia, 2018.

[3] A. Tobin, "Making AI and machine learning pay," Apr 2020. [Online]. Available: https://www.digitaltveurope.com/longread/making-ai-and-machine-learning-pay/

[4] C. Laurier and P. Herrera, "Automatic detection of emotion in music: Interaction with emotionally sensitive machines," in *Machine learning: Concepts, methodologies, tools and applications*. IGI Global, 2012, pp. 1330–1354.

[5] Y. Dong, X. Yang, X. Zhao, and J. Li, "Bidirectional convolutional recurrent sparse network (bcrsn): An efficient model for music emotion recognition," *IEEE Transactions on Multimedia*, vol. 21, no. 12, pp. 3150–3163, 2019. doi: 10.1109/TMM.2019.2918739

[6] R. Delbouys, R. Hennequin, F. Piccoli, J. Royo-Letelier, and M. Moussallam, "Music mood detection based on audio and lyrics with deep neural net," *arXiv preprint arXiv:1809.07276*, 2018.

[7] A. Huq, J. P. Bello, and R. Rowe, "Automated music emotion recognition: A systematic evaluation," *Journal of New Music Research*, vol. 39, no. 3, pp. 227–244, 2010. doi: 10.1080/09298215.2010.513733

[8] M. B. Er and Aydilek, "Music emotion recognition by using chroma spectrogram and deep visual features," *International Journal*

of Computational Intelligence Systems, vol. 12, 12 2019. doi: 10.2991/ijcis.d.191216.001

[9] X. Yang, Y. Dong, and J. Li, "Review of data features-based music emotion recognition methods," *Multimedia systems*, vol. 24, no. 4, pp. 365–389, 2018.

[10] N. HE and S. Ferguson, "Multi-view neural networks for raw audio-based music emotion recognition," in *2020 IEEE International Symposium on Multimedia (ISM)*, 2020. doi: 10.1109/ISM.2020.00037 pp. 168–172.

[11] J. Bai, K. Luo, J. Peng, J. Shi, Y. Wu, L. Feng, J. Li, and Y. Wang, "Music emotions recognition by cognitive classification methodologies," in *2017 IEEE 16th International Conference on Cognitive Informatics Cognitive Computing (ICCI*CC)*, 2017. doi: 10.1109/ICCI-CC.2017.8109740 pp. 121–129.

[12] J. Brotzer, E. Mosqueda, and K. Gorro, "Predicting emotion in music through audio pattern analysis," in *IOP Conference Series: Materials Science and Engineering*, vol. 482, no. 1. IOP Publishing, 2019, p. 012021.

[13] J. A. Russell, "A circumplex model of affect." *Journal of personality and social psychology*, vol. 39, no. 6, p. 1161, 1980.

[14] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: A system for large-scale machine learning," in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, 2016, pp. 265–283.

[15] P. Ekman, E. R. Sorenson, and W. V. Friesen, "Pan-cultural elements in facial displays of emotion," *Science*, vol. 164, no. 3875, pp. 86–88, 1969. doi: 10.1126/science.164.3875.86

[16] R. Panda, R. Malheiro, and R. P. Paiva, "Novel audio features for music emotion recognition," *IEEE Transactions on Affective Computing*, vol. 11, no. 4, pp. 614–626, 2018.

[17] M. Velankar, "MER500 — music emotion recognition," Kaggle, Jun 2020, accessed: 2021-03-10.

[18] K. Zhang, H. Zhang, S. Li, C. Yang, and L. Sun, "The pmemo dataset for music emotion recognition," in *Proceedings of the 2018 ACM on International Conference on Multimedia Retrieval*, ser. ICMR '18. New

York, NY, USA: ACM, 2018. doi: 10.1145/3206025.3206037. ISBN 978-1-4503-5046-4 pp. 135–142.

[19] Y.-A. Chen, Y.-H. Yang, J.-C. Wang, and H. Chen, "The amg1608 dataset for music emotion recognition," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2015, pp. 693–697.

[20] T. Eerola and J. K. Vuoskoski, "A comparison of the discrete and dimensional models of emotion in music," *Psychology of Music*, vol. 39, no. 1, pp. 18–49, 2011.

[21] C. Akiki and M. Burghardt, "Toward a Musical Sentiment (MuSe) Dataset for Affective Distant Hearing," *CHR 2020: Workshop on Computational Humanities Research*, November 2020.

[22] A. B. Warriner, V. Kuperman, and M. Brysbaert, "Norms of valence, arousal, and dominance for 13,915 english lemmas," *Behavior research methods*, vol. 45, no. 4, pp. 1191–1207, 2013.

[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, p. 84–90, May 2017. doi: 10.1145/3065386

[24] J. Guignot and P. Gallinari, "Recurrent neural networks with delays," in *ICANN '94*, M. Marinaro and P. G. Morasso, Eds. London: Springer London, 1994. ISBN 978-1-4471-2097-1 pp. 389–392.

[25] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE transactions on neural networks*, vol. 5, no. 2, pp. 157–166, 1994.

[26] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[27] D. de Benito-Gorron, A. Lozano-Diez, D. T. Toledano, and J. Gonzalez-Rodriguez, "Exploring convolutional, recurrent, and hybrid deep neural networks for speech and music detection in a large audio dataset," *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2019, no. 1, pp. 1–18, 2019.

[28] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[29] S. Abu-El-Haija, N. Kothari, J. Lee, P. Natsev, G. Toderici, B. Varadarajan, and S. Vijayanarasimhan, "Youtube-8m: A large-scale video classification benchmark," *arXiv preprint arXiv:1609.08675*, 2016.

[30] Y.-S. Seo and J.-H. Huh, "Automatic emotion-based music classification for supporting intelligent iot applications," *Electronics*, vol. 8, no. 2, 2019. doi: 10.3390/electronics8020164

[31] R. Panda, R. Malheiro, and R. P. Paiva, "Musical texture and expressivity features for music emotion recognition," in *19th International Society for Music Information Retrieval Conference (ISMIR 2018*, 2018, pp. 383–391.

[32] S. Chapaneri and D. Jayaswal, "Deep gaussian processes for estimating music mood," in *2018 15th IEEE India Council International Conference (INDICON)*, 2018. doi: 10.1109/INDICON45594.2018.8987036 pp. 1–5.

[33] J.-L. Hsu, Y.-L. Zhen, T.-C. Lin, and Y.-S. Chiu, "Affective content analysis of music emotion through eeg," *Multimedia Systems*, vol. 24, no. 2, pp. 195–210, 2018.

[34] J. H. Juthi, A. Gomes, T. Bhuiyan, and I. Mahmud, "Music emotion recognition with the extraction of audio features using machine learning approaches," in *Proceedings of ICETIT 2019*, P. K. Singh, B. K. Panigrahi, N. K. Suryadevara, S. K. Sharma, and A. P. Singh, Eds. Cham: Springer International Publishing, 2020, pp. 318–329.

[35] J. Li, L. Han, X. Li, J. Zhu, B. Yuan, and Z. Gou, "An evaluation of deep neural network models for music classification using spectrograms," *Multimedia Tools and Applications*, pp. 1–27, 2021.

[36] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[37] R. Sarkar, S. Choudhury, S. Dutta, A. Roy, and S. K. Saha, "Recognition of emotion in music based on deep convolutional neural network," *Multimedia Tools and Applications*, vol. 79, no. 1, pp. 765–783, 2020.

[38] E. Koh and S. Dubnov, "Comparison and analysis of deep audio embeddings for music emotion recognition," *arXiv preprint arXiv:2104.06517*, 2021.

[39] J. J. Louviere, T. N. Flynn, and A. A. J. Marley, *Best-worst scaling: Theory, methods and applications*. Cambridge University Press, 2015.

[40] T. Pronk, D. Molenaar, R. W. Wiers, and J. Murre, "Methods to split cognitive task data for estimating split-half reliability: A comprehensive review and systematic assessment," *Psychonomic Bulletin & Review*, pp. 1–11, 2021.

[41] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in *Proceedings of the 14th python in science conference*, vol. 8. Citeseer, 2015, pp. 18–25. [Online]. Available: https://github.com/librosa/librosa/tree/0.8.1rc2

[42] J. Cramer, H.-H. Wu, J. Salamon, and J. P. Bello, "Look, listen, and learn more: Design choices for deep audio embeddings," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 3852–3856.

Appendix A

Results in more detail

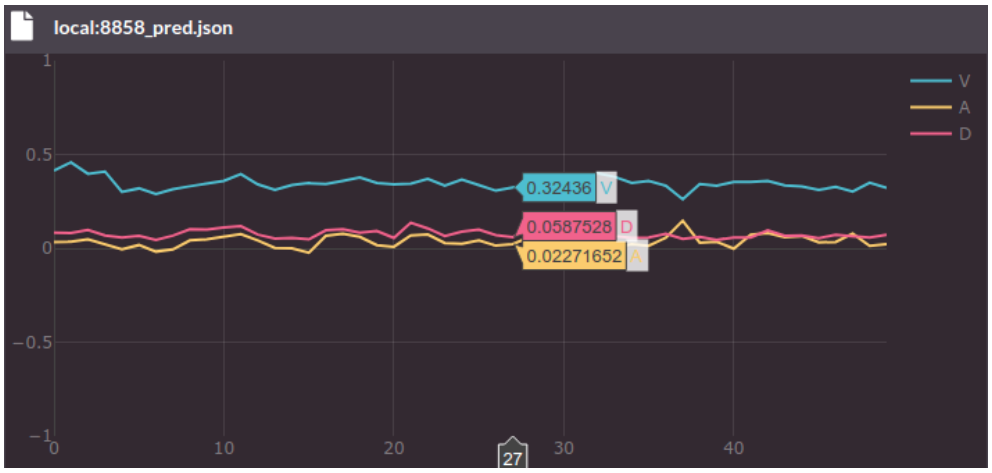


Figure A.1: VAD true labels: [0.274, 0.117, 0.175].

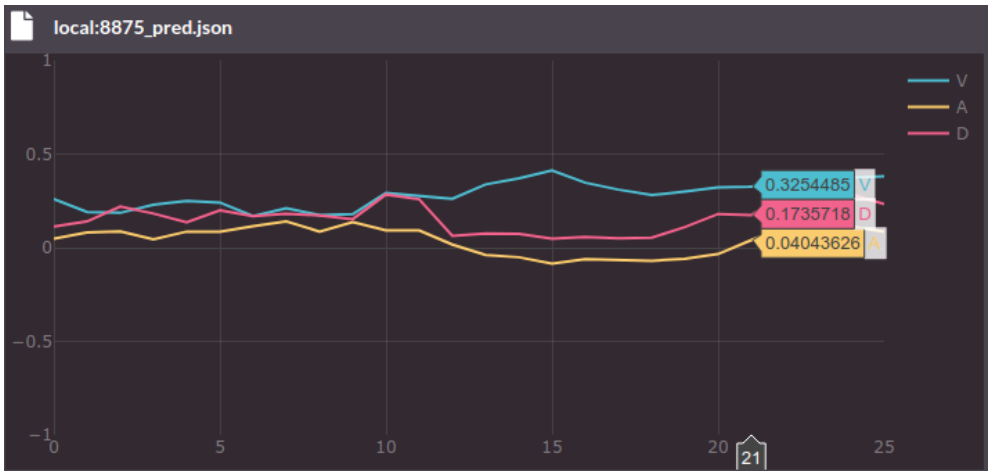


Figure A.2: VAD true labels: [0.682, 0.461, 0.567].

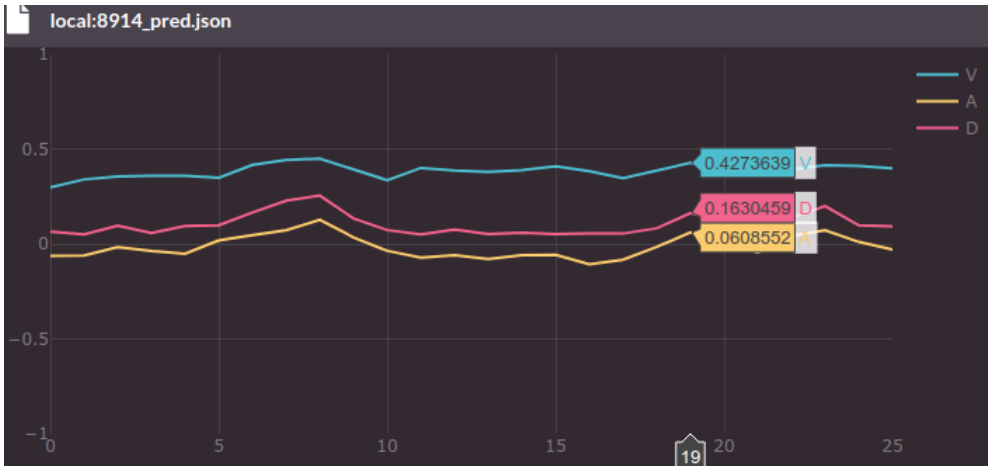


Figure A.3: VAD true labels: [0.524, -0.381, -0.072].

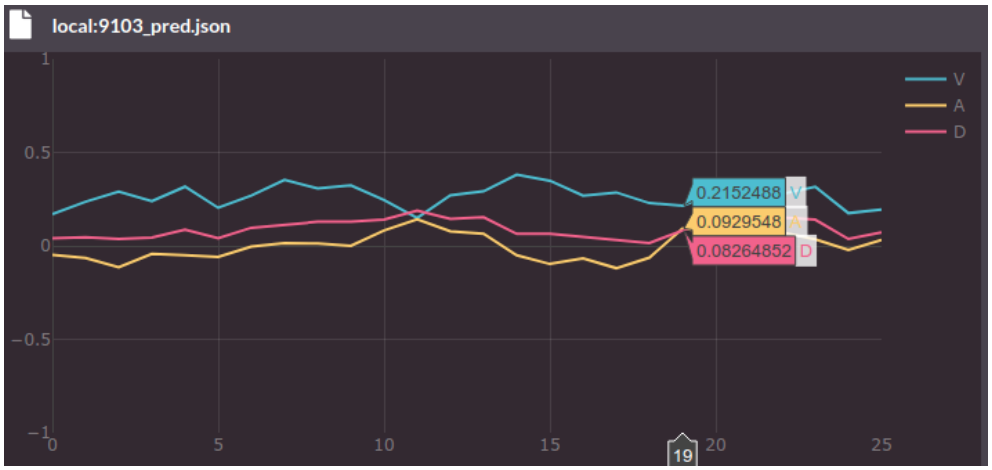


Figure A.4: VAD true labels: [0.318, -0.142, -0.011].

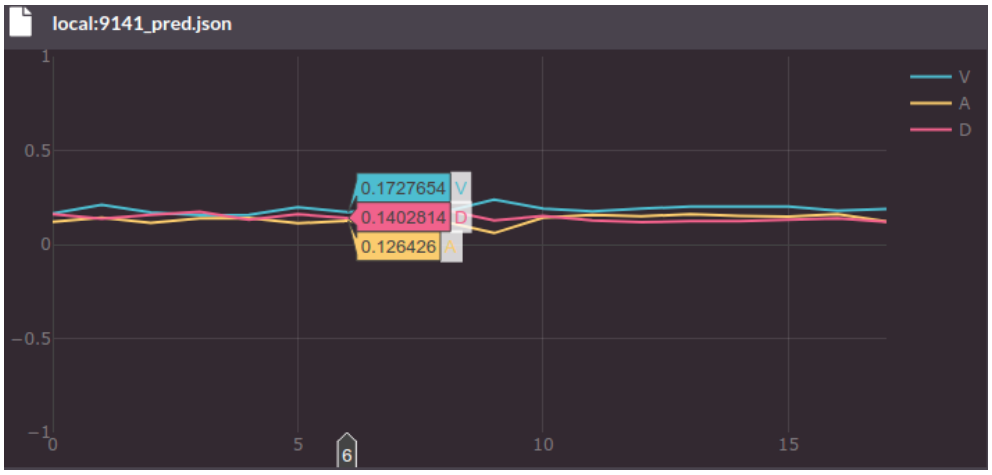


Figure A.5: VAD true labels: [0.063, 0.165, 0.037].

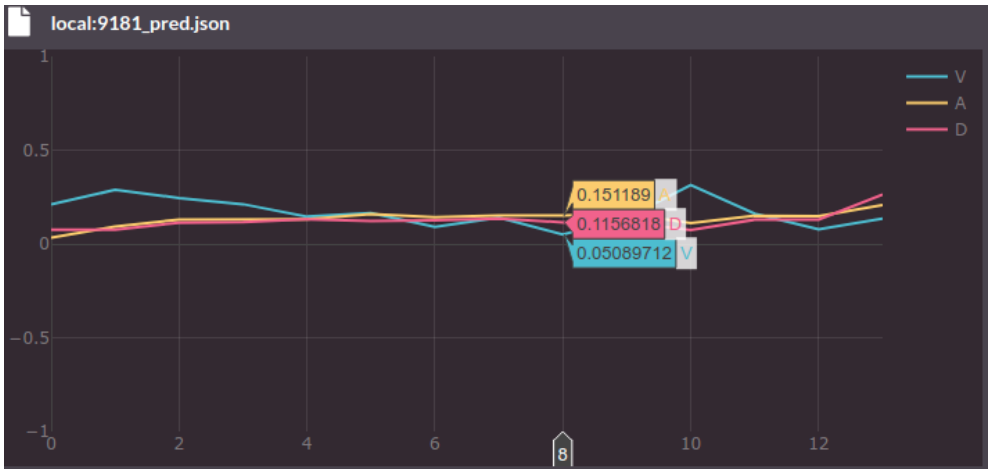


Figure A.6: VAD true labels: [0.120, 0.005, 0.065].

TRITA-EECS-EX-2021:784