Emotion from music spectrograms

Matteo Cerutti
Politecnico di Torino
s265476@studenti.polito.it

Antonio Santoro
Politecnico di Torino
s264014@studenti.polito.it

Marco Testa
Politecnico di Torino
s265861@studenti.polito.it

Abstract

In this report the classification of songs' mood is exploited through spectrograms. The principal idea is to see if a neural network can be able to define song's mood starting from vocal audio per mood. Three networks are used: GoogLeNet, VGG and ResNet. The datasets used for training and testing part are different: RAVDESS emotional song audio for training, CAL500 for testing. The proposed technique is useful to extract the spectrograms from each dataset and treat the problem of the variable track duration. The results of the experiment are not so encouraging, but there are different improvements that can be implemented and that can increase the outcomes.

1. Introduction

Nowadays people need to have the possibility to select music and make playlists based on their mood. Many music platforms feature different music playlists made by hand that include popular and commercial songs aiming to maximise ratings. One of the most used feature on these platforms is to create playlists similar to other ones, the point is that all the songs that will be included are selected on the "similarity".

Our idea focuses on making classifications of music from a different point of view, in fact our aim is to exploit a visual representation of an audio file and try to catch features on that. After some researches, we found that our idea was applied to classify song genres, therefore starting from the article of Piotr Kozakowski and Bartosz Michalak [3], we adapted their work to our objective.

The interest is to train a neural network on different audio speeches that represent different human emotions, extract features and try to see whatever those peculiarities can be matched from music. Amiriparian *et al*. [1] showed that processing spectrograms into networks characterized by a different depth will change the result. This report presents results obtained from three networks: ResNet, VGG and GoogLeNet, trained on the RAVDESS Emotional song audio dataset [5] and tested on the CAL500 dataset [6].

This work is organized as follows. In Section 2 we give an explanation on how the training and testing dataset have been preprocessed to be adapted to our purposes, how the samples have been filtered and selected. Section 3 focuses on the networks training phase in which useful hyperparameters sets have been chosen in order to obtain significant validation and training scores. In Section 4 we present the testing algorithm that classifies each song. Finally, some conclusions and suggestions for future work are drawn in Section 5.

1.1. Classification pipeline

A trivial image classifier could have submitted poor performances, thus we figured out a model that is capable of slicing each song, treated as variable sized spectrograms as well, then choose a label after checking the rank of each slice that compose the whole track. Figure 1 shows the general model structure, in the final step the song will be classified by means of a voting algorithm.

2. Data preparation

2.1. Training dataset

The RAVDESS Emotional song audio consists of 1012 files of actors singing in a neutral North American accent. The portion used for this work includes calm, happy, sad, angry, and fearful emotions, each vocal is produced at two levels of emotional intensity, normal and strong.

Files are provided as .wav (16bit, 48kHz, mono, 4 seconds each) that need to be converted into raw spectrograms. For the purpose "SoX (Sound eXchange) sound processing utilities" has been used. This tool can process audio files and do things like trimming or filters frequencies. Spectrograms for the training dataset have been generated to fit the input size of the three networks, furthermore, to cope with the limited size of the dataset, augmentation has been applied like random grey colorizing, brightness, contrast and hue variations. Amiriparian *et al*. [1] showed that using different shades of colour could exhibit different outcomes. Unfortunately, since the hue transformation made by the PyTorch framework has a not negligible impact on the bright-

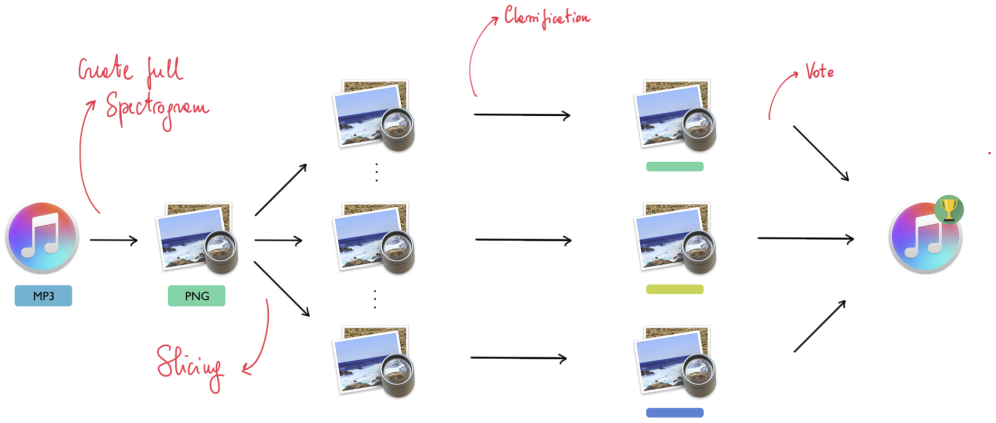

Figure 1. Classification pipeline.

ness of the image and the goal is to not corrupt the information brought by the spectrograms, the best choice was to stick with the original shades (Figure 2). Hence, playing with the contrast and with monochromes image could have a positive effect on capturing some features.

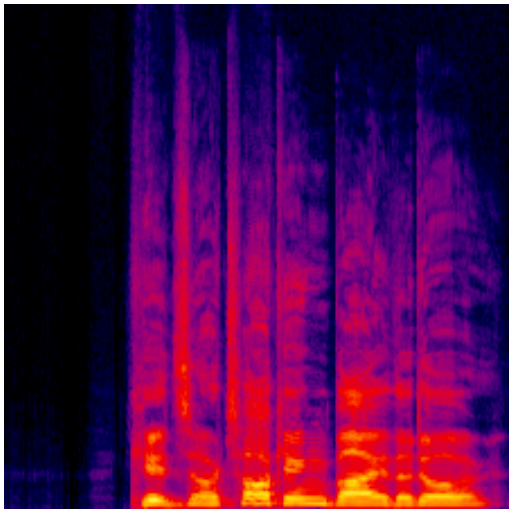

Figure 2. Training spectrogram sample.

2.2. Test dataset

The CAL500 dataset contains 500 songs performed by 500 unique artists, each song has been annotated by at least three people using a standard survey. Files are provided as .mp3 (32kbps, mono) along with one or more labels.

As Liu *et al*. [4] stated *preprocessing the spectrograms is a key point of successfully applying CNN on music spectrograms*, because the input to CNN requires to be a fixed size matrix and each track has a variable duration. The point is to let the network be able to recognize the emotion by taking a closer look at the song. The simplest approach would have been to extract spectrograms of the desired size after computing the number of overlaps. Yet, the chosen approach has been to not loose any data during the pre-processing step, thus the need is to define an algorithm that will generate spectrograms with variable length in order to match them to the duration of the tracks. For the purpose of making the dataset compatible with our testing environment, two actions have been performed.

2.2.1 Filtering

Since the training label set was a subset of the CAL500 labels, we selected only the songs which classes belong to the first set. Furthermore, an additional filtering step has been performed to remove all the redundant classes keeping only the relevant ones.

2.2.2 Slicing

The most challenging step was making the test dataset compatible with the training samples. To cope with the variable duration of each song, the extracted spectrograms have been sliced into squared images to fit to the network input size without losing any information. Each slice has been generated by sampling a proportional quantity of information equal to the training samples of four seconds. In fact, SoX allows to extract the spectrogram by setting the number of pixel per second and the input size of the image. Given that, a different testing approach has been implemented.

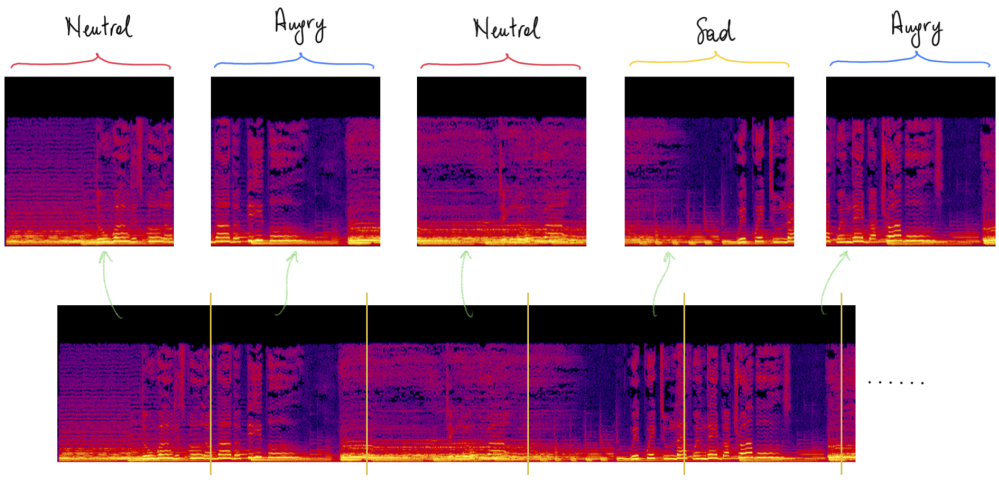

Figure 3. Voting system.

Network	LR	BS	WD	G
ResNet152	0.003	12	2e-05	0.6
ResNet50	0.0007	14	8e-04	0.7
VGG19	0.002	11	1e-06	0.1
VGG11	0.0009	11	7e-04	0.07
GoogLeNet	0.0001	8	5e-05	0.1

Table 1. Best values per hyperparameter and network.

3. Training phase

The research method to find the optimal set of hyperparameters has been the same for all networks[1] except for a slight difference related to GoogLeNet due to its three output branches. A lot of experiments have been done on different network variants of the same model to evaluate the impact of the networks' depth on the results.

The first step was to find a good starting hyperparameters set to make the network accomplish a full training. Due to Google Colab limitations, a random search has been used to evaluate 50 different hyperparameters sets, the best ones have been reported in Table 1.

Using the reported sets, all the networks have been trained for 100 epochs and evaluated using different split ratios between training set and validation set. The final values have been selected by doing some tuning by hand after evaluating the networks' performance during the epochs, moreover the values of each hyperparameter have been adjusted to address the problem of the high epochs number and to prevent the occurring overfit. Since the training dataset is very small, we dealt with the overfit problem by means of data augmentation, yet no significant improving has been noticed.

Table 2 contains the average best validation scores calculated on the validation set per network and variants. However we observed that choosing a validation set of a smaller size, the overall score of all networks will increase. Although the training dataset is small, applying a significative amount of regularization prevented the occurring of the overfitting phenomenon seen right after half of the epochs, but the limitations imposed by the platform kept us from increasing the number of epochs and evaluating more sets of hyperparameters.

Network	50/50	80/20	90/10
ResNet152	65%	63%	76%
ResNet50	66%	66%	66%
VGG19	72%	67%	80%
VGG11	74%	77%	84%
GoogLeNet	72%	73%	82%

Table 2. Average validation accuracy per network and split in 100 epochs.

4. Testing

The testing phase was characterized by two main parts: the network slices' prediction and the major voting. During the first part, the network assigned a prediction to each slice. The PyTorch *DataLoader* class, used in this part to collect

[1] ResNet50 & ResNet152, VGG11 & VGG19 and GoogLeNet (Inception v1)

a batch of images to be classified, was modified with the intention to have the image path as another return value, together with the image tensor and its labels. In this way it was possible to collect, for each song, a set of counters, one for each possible test class increased each time a prediction for a specific class was done.

In the second part all the data collected in the previous one were processed in order to decide which class was more predicted by the model, for each song, using a pure major voting strategy: the higher counter labeled the entire song. A preprocessing step for this part on the test dataset's labels was useful. In fact, for getting this values in an immediate way, a mapping on all songs was previously done, assigning for each song all the labels belonging to the subset obtained in Section 2.2.1.

Processing the data collected in network prediction's part, a dominant label was assigned to each song. A song is correctly classified if the predicted label was in the subset of the attributed labels. Finally, with the number of correct predictions and the total number of songs considered in the test dataset, it was possible to calculate the test accuracy.

Network	**50/50**	**80/20**	**90/10**
ResNet152	15%	15%	35%
ResNet50	19%	16%	16%
VGG19	17%	16%	15%
VGG11	31%	13%	33%
GoogLeNet	36%	34%	37%

Table 3. Test accuracy per network and split.

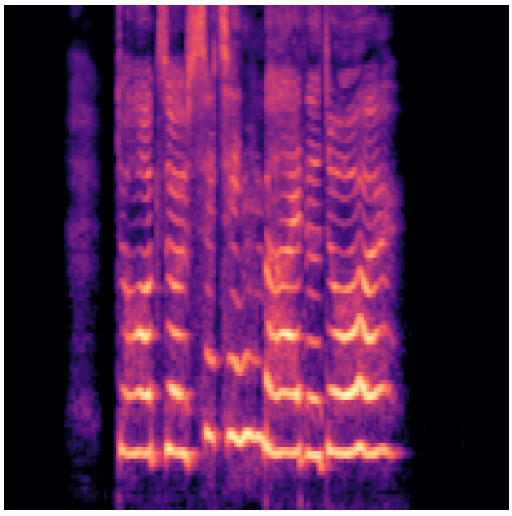

Figure 4. Mel spectrogram sample from the training set.

5. Conclusions

This report proposed an experimental method to classify songs with emotions captured from human vocal singing recordings.

Since the networks extracted features from spectrograms related to pure vocals, we had many doubts that the correct classification of spectrograms related to songs (vocal and instrumental) would be taken for granted. The accuracies' values suggest us that our doubts were real. Hence, the first immediate suggestion to achieve the right classification is to find emotions within a single domain. To stay in our domain (vocal), we should have extracted only the voice channel from each song, removing the instrumental part. Such process cannot be achieved within an acceptable time span. The biggest problem was the lack of availability of bigger and free datasets for this task.

This being cleared up, we can now suggest further improvements.

The use of mel spectograms (Figure 4) could have resulted in better performances as found out also by Eguino [2]. After a couple of validation runs, each network performed a lot better than training on raw spectrograms, actually, networks like VGG19 reached very high validation scores (94% using a 80/20 split). The preprocessing steps, unfortunately, required additional libraries like *librosa* that are not trivial to use for extracting variable spectrograms like those described in Section 2.2.

The testing phase can be improved in different ways. One possibility is to change the major voting algorithm. An example can be: insert a threshold to define the most predicted label, considering the percentage of the specific prediction on all slices. In this way the cases in which the most important label is not so dominant are excluded. Another possibility is to change how the predictions are considered as correct: having a definition of test class rank importance can determine different levels of prediction rightness. The consequence is to define a new method for calculate the accuracy on the test dataset.

Future works could be based on these suggested improvements. Full code is available at `https://github.com/anphetamina/AIML_project`.

References

[1] Shahin Amiriparian, Maurice Gerczuk, Sandra Ottl, Nicholas Cummins, Michael Freitag, Sergey Pugachevskiy, Alice Baird, and Bjorn Schuller. Snore sound classification using image-based deep spectrum features. 2017.

[2] Miguel Flores Ruiz de Eguino. Deep music genre.

[3] Piotr Kozakowski and Bartosz Michalak. Music genre recognition. 2016. `http://deepsound.io/music_genre_recognition.html`.

[4] Xin Liu, Qingcai Chen, Xiangping Wu, Yan Liu, and Yang Liu. Cnn based music emotion classification. 2017.

[5] Livingstone SR and Russo FA. The ryerson audio-visual database of emotional speech and song (ravdess): A dynamic, multimodal set of facial and vocal expressions in north american english. plos one 13(5): e0196391, 2018. https://doi.org/10.1371/journal.pone.0196391.

[6] Turnbull, Douglas, Barrington, Luke, Torres, David, Lanckriet, and Gert. Semantic annotation and retrieval of music and sound effects. *Audio, Speech and Language Processing, IEEE Transactions on*, 16(2):467–476, 2008.