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

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". Since the intention was to stay inside the computer vision domain, we have to treat audio files as images so the first idea was to exploit spectrograms. 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 [2], 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 the result will change. This report presents results obtained from three networks, ResNet, VGG and GoogLeNet, trained on the RAVDESS Emotional song audio dataset [3] and tested on the CAL500 dataset [4].

1.1. Classification pipeline

Figure 1 shows the model structure that classifies each slice and then after collecting all the predicted labels, 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 four seconds 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) that need to be converted into a raw spectrogram. 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 shade of colour could exhibit different outcomes but, since the hue transformation made by the PyTorch framework has a not negligible impact on the brightness of the image and the goal is to not alter any information on the spectrogram, the best choice was to stick with the original shades (Figure 2), hence playing with the contrast and with a monochrome image could have a positive effect on capturing some features.

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.

In order to make the dataset compatible with our testing environment, two actions have been performed:

- **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.

- **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

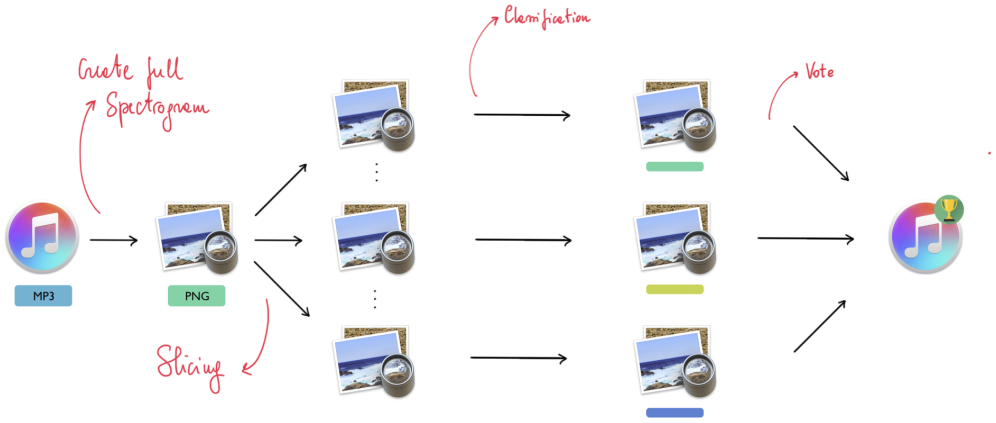

Figure 1. Classification pipeline.

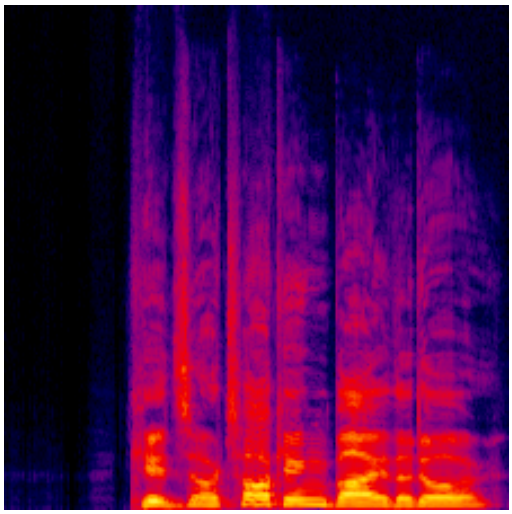

Figure 2. Training spectrogram sample.

proportional quantity of information equal to the training samples of four seconds. Given that, a different testing approach has been implemented.

3. Training phase

The research method used to find the optimal set of hyperparameters was 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 70 different hyperparameters sets, the best ones have been reported in Table 1.

Network	LR	BS	WD	G
ResNet	toadd	toadd	toadd	toadd
VGG	0.0008	8	2e-05	0.05
GoogLeNet	0.0001	8	5e-05	0.1

Table 1. Best values per hyperparameter.

Network	Validation accuracy
ResNet	toadd
VGG	70%
GoogLeNet	82%

Table 2. Average validation accuracy per network.

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' perfomance 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 overfit. Since the training dataset is very small, we dealt with the overfit problem trying to solve it by means of data augmentation which led our training dataset to increase its size.

Table 2 contains validation accuracies calculated on the validation set per network. The results referred to the first two networks are the averages between the single result of each network variant.

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

4. Testing

5. Conclusions

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

[3] 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`.

[4] 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.

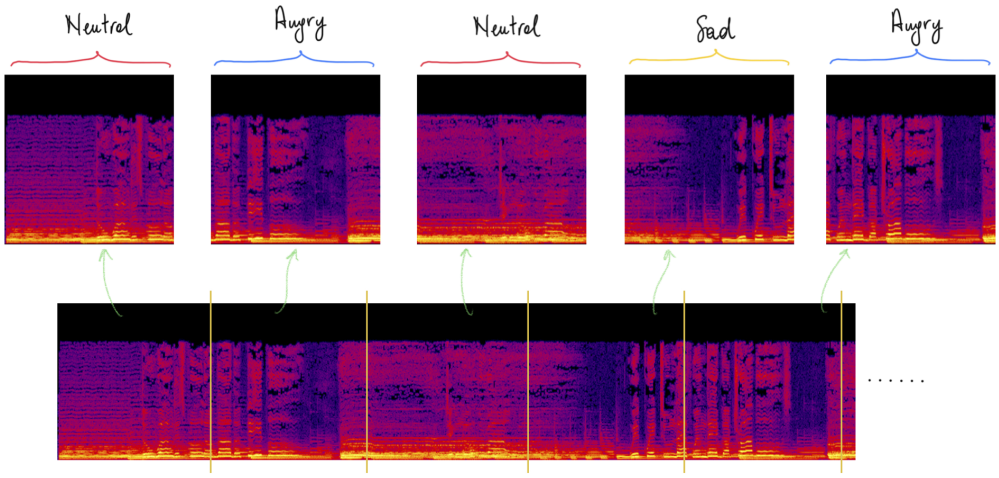

Figure 3. Voting system.