Music Genre Classification with Convolutional Recurrent Neural Networks

Luca Campana
Politecnico di Torino
Student ID: s290085
s290085@polito.studenti.it

Gianvito Liturri
Politecnico di Torino
Student ID: s290464
s290464@polito.studenti.it

Dario Padovano
Politecnico di Torino
Student ID: s291475
s291475@polito.studenti.it

***Abstract*—From the birth of AI algorithms, there were few - successful or less - attempts to project a full-working ML model for music classification. In particular, one of the most challenging tasks has been genre recognition. Among the best results obtained in this field, it is worth mentioning the work made by Tzanetakis et al. [1]: first of all, it provided the GTZAN dataset, which has become the main and most complete source for the above-mentioned assignment; moreover, it granted a baseline, based on simple ML structures, for future developements. Here, the models exploited three different set of frequency-based features that were directly extracted from audio signals.**

This paper firstly reproduces the described approach, and then takes the task to the state of the art by exploiting the power of CRNNs, an established deep framework that succeeds in understanding temporal structure in audio spectrograms, following the approach described by Nasrullah et al. [2]

The best performing model achieved an average F1 score of 0.8920 from three independent trials, with a micro-precision equal to 1.0 for 5 out of 10 genres it has been trained with. The results obtained strongly outperform both the results obtained with SVM and previous CRNN based attempts.

***Index Terms*—audio classification, music genre classification, audio augmentation, music, convolutional recurrent neural network, deep learning, information retrieval**

I. Introduction

Defining a specific musical genre to be assigned to a certain song has always been a fairly simple task for humans, as it is hardly connected with the feelings transmitted by it. On the other side, describing this process with a series of logical and arithmetical constraints is nearly impossible: this makes it difficult to design a proper application that is able to achieve good performances on this task, without regarding AI. Moreover, the fact that often there aren't strict boundaries between similar genres further hardens the job.

However, it is clear that, together with the emotional sphere, songs embody a series of more rational features that follow recurrent patterns among the same genre. Those are the ones that machines can work with, understanding their meanings and exploiting them in order to obtain good results.

Nowadays, the need of a proper automatization for the task of genre recognition is increasing: music industry owes most of its incomes from mechanisms like user profilation, music recommendation and market investigation. These processes could heavily benefit from innovations in this direction, in terms of both efficiency and incomes. In view of all this, it clearly appears how important it is to focus on the development of successful algorithms for this job.

The late spread of AI application over music is due to the lack of available computing power: in the early stages, musical records were summarized through low-dimensional feature representations, which failed in depicting organically the songs they were originated from. As a consequence, it was really difficult to achieve remarkable results in this domain. In addition, only a little part of the existing music was recorded, so the feature dimensionality was also influenced by the datasets' cardinality, and it had to remain low in order to avoid curse of dimensionality [3].

Nowadays, thanks to the technological progress, there are much more resources to be exploited by researchers, both at hardware level and data availability. This means that audio signals can be depicted by the means of spectrograms, high-dimensional feature representations that can be interpreted and analyzed in a profitably way by the new deep neural models, such as CRNNs.

By a practical point of view, spectrograms, contrary to previous representations that mainly focus on frequency ranges, preserve the temporal structure of the song. In this way, they can enclose more information: as it is logical, a more accurate representation leads to a more accurate and performing model. As it follows, it is obvious the choice of a CRNN to address music genre classification: in fact, the early convolutional layers are useful to learn the global frequency structure, while the latter recurrent ones can find and interpret time patterns in the spectrogram.

Following the approach of Nasrullah et al. [2], we decided to use a properly sized examplar of CRNN, that we trained with an augmented version of GTZAN dataset [1]. Songs from the dataset have been splitted into frames of equal length, in order to obtain a more robust algorithm and to gain data cardinality. As it is logical, different performances have followed from different split lengths, so it has been mandatory to carry out a qualitative analysis.

With the best obtained model, we achieved good performances over validation data in terms of F1 score and micro-precision. This proves that our architecture is able to recognize its domain of application, similarly to the one obtained by Nasrullah et al. As a further proof, we tried to submit to the model real scenarios, such as audio extracts from YouTube music videoclips. The fact that also in this case our algorithm worked properly is a clear sign of its robustness, and what it is capable of: we are strongly confident that our model could at least compete with the state of the art.

II. Related works

A. Genre Classification Baselines

During the last years, the task of recognising the musical genre of a song has been widely addressed, through several different approaches. One of the most valuable works was done in 2002 by Tzanetakis et al. [1], and it consisted in a 10 genres classification task. In this project, they tested a wide variety of classical architectures - such as Support Vector Machines, Gaussian Mixture Models and K-Nearest Neighbors - by means of three low-dimensional sets of features. In particular, the first two sets (regarding pitch and rhythmic aspects) are obtained directly from the whole audio signal, while the last one, concerning song timbral textures, is calculated as the running average of the single quantities evaluated over song splits, required to apply the Short-Time Fourier Transform (STFT). This whole process results in the creation of a single feature vector for each different song. As a consequence, cardinality of the provided data is rather low, and, above all, they do not manage to express song temporal structures in an adequate way.

The described approach leads to the creation of an imperfect architecture, that is not capable of retrieving the correct amount of information from the data. This causes low overall accuracy scores with any of the architecture tested, with a best result equal to 0.61 obtained for the 10 genres classification with a GMM.

This work laid a foundation to a prosperous research in the field of music genre classification, and it can be seen as an important baseline for all following projects, including the one described in this paper.

B. Audio Representations and Spectrograms

Previous approaches heavily rely on features obtained by the means of Mel-frequency cepstral coefficients (MFCC), that provide a compact representation of the song spectral envelope in a way that approximates the human auditory system's responses. However, this representation can not describe properly temporal patterns, so researchers started to look towards a more innovative approach, that would better perform information retrieval from audio files.

One of the most valuable techniques that have recently spread in MIR field consists in generating spectrograms of audio splits. In fact, spectrograms are representations of frequency content over time, and they can be practically calculated by taking the squared magnitude of the discrete STFT of the signal. The mathematical form of STFT reminds of a discrete convolution, and it is described by Eq. (1): here, $x[n]$ is related to the equation of the quantized input signal, while $w[n]$ regards the window function of the transformation.

$$\mathbf{STFT}\{x[n]\}(m,\omega) \equiv X(m,\omega) = \sum_{n=-\infty}^{\infty} x[n]w[n-m]e^{-j\omega n} \tag{1}$$

As one can see, both STFT and spectrograms are functions of both frequency (ω) and sampling times (m), so they can be seen as a sort of 'images', or more formally two-dimensional matrices, that show defined temporal and frequency structures, and are particularly suited for the usage of a Convolutional Recurrent Neural Network, as we will describe organically in section II-D. As it is shown in Figure 1, obtained spectrograms are quite difficult to understand from a human point of view, but on the other side we can clearly perceive that there are significant differences among different genres: therefore, they can be usefully exploited by deep algorithms in order to find implicit patterns, both in time and frequency domain, and then perform classification.

C. Convolutional and Recurrent Neural Networks

The proposed approach in this paper was strongly influenced by recent developments in ML fields: in particular, it is worth mentioning the spread of two deep architectures, respectively called CNNs and RNNs.

Convolutional Neural Networks [4] have been true game changers to handle high-dimensional data, such as images and spectrograms, for several reasons. Firstly, the convolution filter strategy allows the network to share parameters through all same-level input features, resulting in a much more agile architecture, that can be trained in reasonable times. Moreover, the stack of different convolutional layers makes the model able to learn features, at different levels of abstraction, in a hierarchical structure. By this way, CNNs maintain multiple overviews of input features, both in local and global standpoint. One last advantage granted by this approach consists in the fact that the network, during training steps, autonomously understands the most fruitful ways to extract features, in order to perform the given task: by this way, researchers can avoid to specify any knowledge by hand, so their work is utterly simplified.

Recurrent Neural Networks, on the other way, were born with the specific aim to analyze time series, in which the information is enclosed not only in the single input feature, but also in their reciprocal positions. This is obviously the case of audio tracks, both speech and music: here, the intrinsic sequentiality of the input can be properly interpreted and evaluated during the computation.

D. Previous approaches of CRNNs to music related tasks

Since CNNs and RNNs are tailored to interpret complementary aspects of the spectrograms, soon researchers have tried to project a miscellaneous model that would be able to merge the two peculiar powers, in order to enhance the ability to manage spectrograms themselves.

Among the most valuable works that have followed this approach, an honorable mention is given to the one by

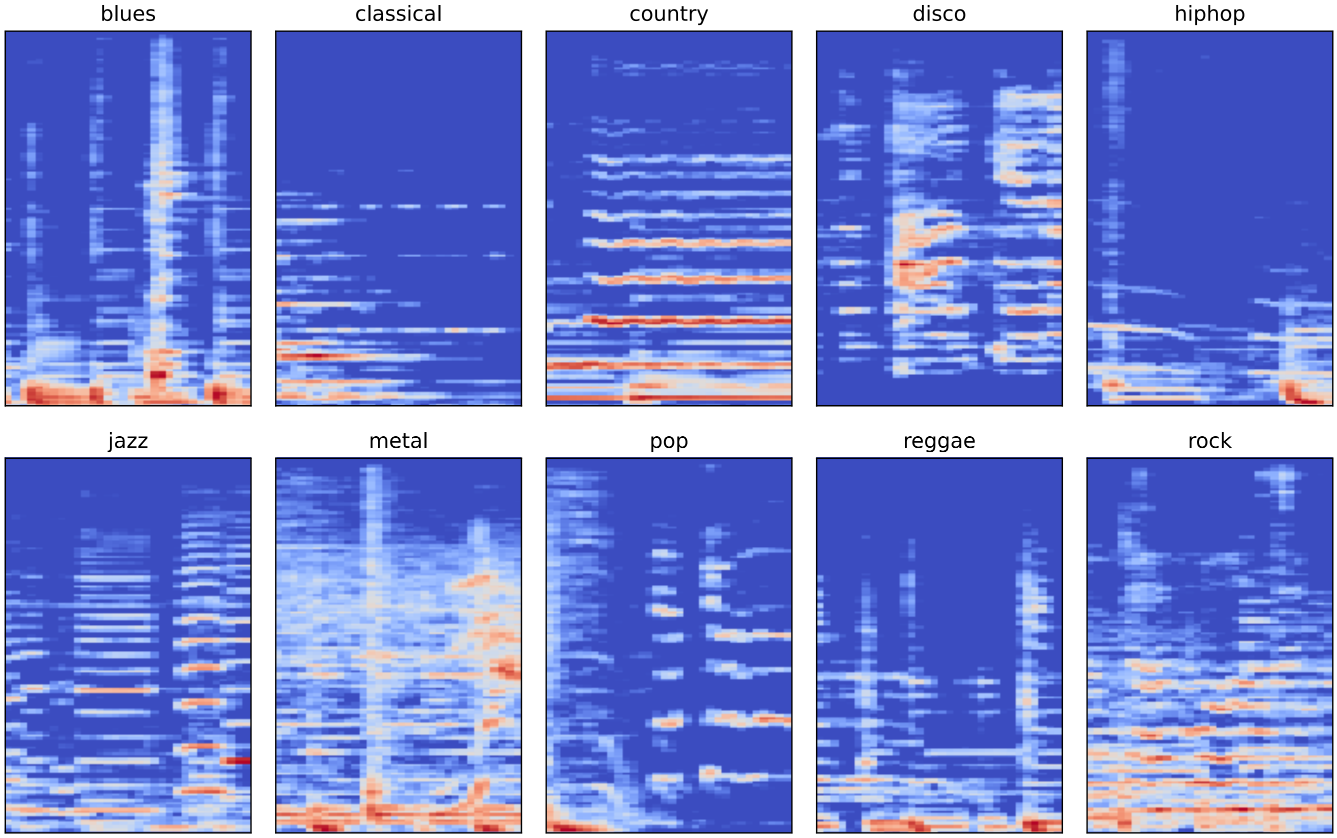


Fig. 1: Spectrograms examples for all genres in the GTZAN dataset (songs randomly chosen; duration one second)

Nasrullah et al. [2]: in this case, a Convolutional Recurrent Neural Network, hereinafter called CRNN, has been applied to audio splits turned into spectrograms in order to perform music artist classification. This work set a new baseline for this particular task, and demonstrated that the usage of a similar architecture improves overall performances.

Another major contribution made by this paper consisted in the introduction of the system of 'majority vote' to obtain a coherent prediction for all splits related to the same song. This has further improved model quality and accuracy standards.

III. Methodology

A. Dataset

The dataset we dealt with is the well known GTZAN set, that is the most complete and widely used resource for the music genre classification. It is a balanced set of 1000 song samples stored as *.wav* files, each long 30 seconds, labeled into 10 different classes: classical, country, disco, hiphop, jazz, rock, blues, reggae, pop, metal.

The files in GTZAN were collected in 2000-2001 from a variety of sources, including personal CDs, radio, microphone recordings, in order to represent a variety of recording conditions. The sampling rate is constant for all the records, and it is equal to 22 050 Hz; the bit-rate is equal to 16 kbps, while the channel is unique (Mono).

B. Data preprocessing

1) Data Augmentation: The first important issue to be solved regarded the fact that data cardinality was too limited to carry out a proper model training. So, we decided to boost both cardinality and overall robustness through a well-known regularization technique, called data augmentation [5]. For every training sample, we produced two augmented versions: one in which we shifted the pitch (*pitch factor* = 1), and the other in which we added some form of additive gaussian white noise (*noise factor* = 1). The result obtained is a training set that has three times the cardinality of the initial one.

2) Mel-spectrogram conversion: After the augmentation, the next step consisted of the conversion from raw audio files into mel-spectrograms, that could be submitted in the CRNN. By a practical point of view, raw files are used to compute a common spectrogram, that is mapped first into Mel-scale (with a fixed number of Mel bins, equal to 128) and then into decibels, according to the following equations.

$$m = 2595 \log_{10}(1 + f/700) \tag{2}$$

$$d = 20 \log_{10}(m) \tag{3}$$

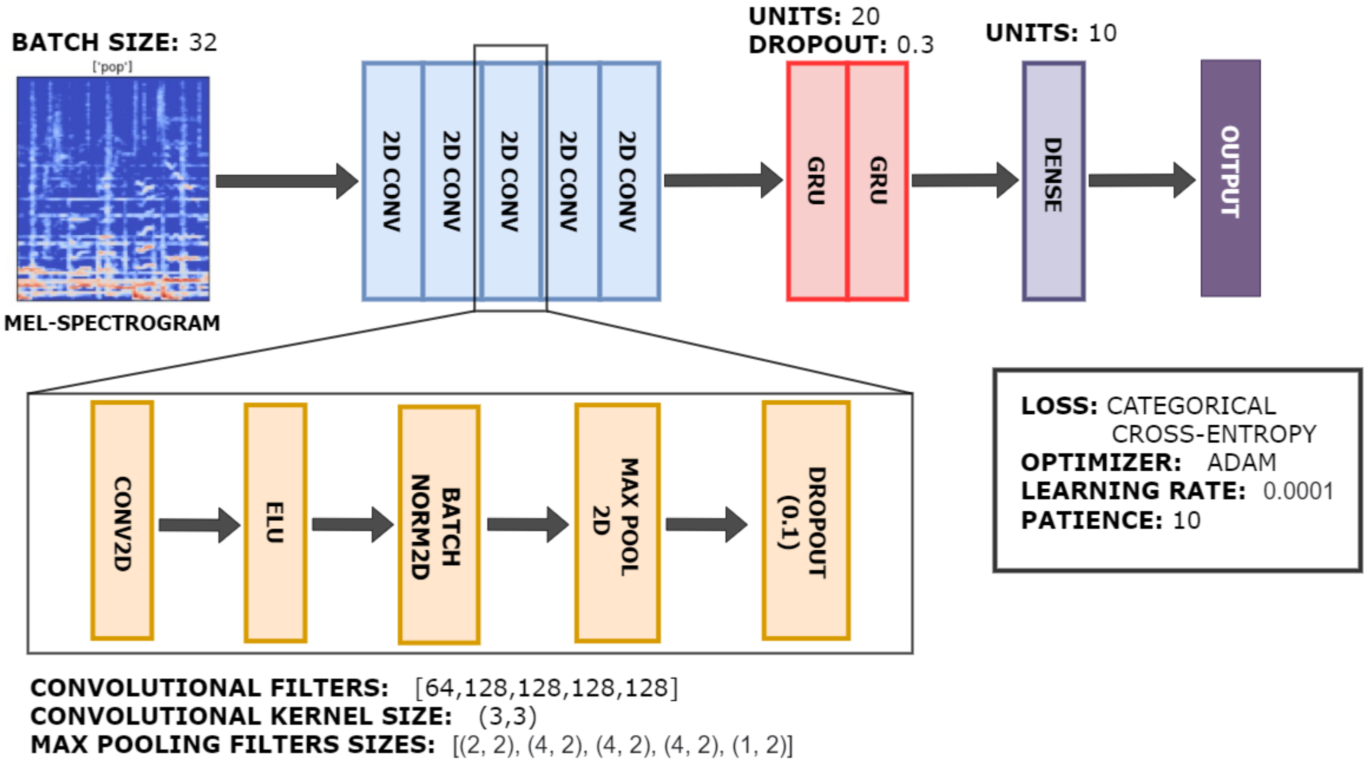


Fig. 2: CRNN Architecture for genre classification

Here, f represents the frequencies obtained in the spectrogram, while m and d are the related transformation outputs. These operations are considered standard practices for audio processing and have been shown in prior work [2] to improve performance in audio classification tasks.

3) Partition into train/test/validation and split into slices: Once we obtained the logarithmic Mel-spectrograms, the next step involved the partitioning of samples into the three usual bins of train, test and validation. Here, the only precautions to take are to ensure that augmented data are put only into the train set, and that those augmentations would not be originated from songs belonging to the two other sets.

Then, we decided to split each Mel-spectrogram into equal time intervals of length t, which varied throughout the research within a range of six fixed lengths. Since slicing is done after the split into train, test and validation sets, slices of the same song can not be found into different bins.

The benefit of this approach is that it allows for experimentation with song level predictions, as it will be described in section III-D, and, above all, it yields a greater number of training samples.

As we can see from Table I, the longer is the slice length, the bigger are input dimensions and the smaller is the train dataset cardinality. This is one of the causes of 'curse of dimensionality': to work well, an algorithm that analyzes n features per instance should be trained with at least $10n$ training instances. This is verified only for the very last slice length, but we choose to test all lengths because longer clips may contain more temporal structure within each training sample.

Slice length	Input dimensions	Number of train istances
30s	1*128*911 = 116608	2430
15s	1*128*465 = 59520	4860
10s	1*128*312 = 39936	7290
5s	1*128*157 = 20096	14580
3s	1*128*94 = 12032	21870
1s	1*128*32 = 4096	70470

TABLE I: Comparison between input dimensionality and cardinality, at each split level

C. Model Architecture

Following the approach defined by Nasrullah, we implemented a CRNN structure (Figure 2): as it can be clearly seen, it is composed of three main constitutive blocks.

The first one, i.e. the convolutional one, is composed of five convolutional 2D layers. The kernel size, the number of filters and the pooling sizes are adapted from the previous work and are shown in Figure 2. Here we used the Exponential Linear Unit (ELU) as activation, which is a better alternative to usual ones. Normalization and regularization techniques used here are 2D batch normalization and drop-out.

In order to be sumbitted to the next block, data are reshaped to 1-dimensional vectors. The recurrent component, consisting of two GRU units, is then used to perform temporal summarization.

After applying a final drop-out, data go through the last block, that is a simple fully-connected layer, composed of 10 units. It produces the final 10 scores, one for each distinct label. The final prediction is then computed as the $argmax$ of this resulting vector and backpropagation is then performed through the calculus of Categorical Cross-Entropy loss, with a constant learning rate and the Adam optimizer.

To reduce overfitting possibilities, we choose to apply also Early Stopping [5], with a patience equal to 10.

D. Frame Level vs. Song Level Evaluation

Since songs in every dataset are sliced into time windows, in each of them there will be a certain number of samples that belong to the same song. This allows to carry on two different types of evaluation.

The first one is a traditional evaluation, for which the classification performances are assessed by considering every frame as independent; the second one, instead, adopts a slightly different approach, that allows to get coherent predictions for all the slices of the same song. In fact, after we got all single predictions over a whole song, we can align them by replacing all with the most predicted value, and then evaluate the classifier performances at the 'song level', by the means of these modified predictions. This process can be seen as a sort of ensembling technique, and, if made with a proper number of slices, can enhance the model robustness and improve the accuracy of predictions.

Lastly, both approaches are evaluated through the F1 score, that allows to correctly consider all types of misclassifications, both false negative and false positive.

IV. Results

	30s	15s	10s	5s	3s	1s
mean	0.638241	0.627125	0.724114	0.754750	**0.782234**	0.768398
max	0.686493	0.686182	0.748980	0.763427	**0.786740**	0.773742
std	0.050954	0.049294	0.026256	0.009836	0.005353	**0.004942**

TABLE II: Test F1 scores for **frame level** Audio Features (three independent runs)

	30s	15s	10s	5s	3s	1s
mean	0.638241	0.629275	0.735132	0.811512	0.846969	**0.891998**
max	0.686493	0.701818	0.772676	0.831062	0.864464	**0.909619**
std	0.050954	0.068231	0.041238	0.022230	**0.016162**	0.016712

TABLE III: Test F1 scores for **song level** Audio Features (three independent runs)

The model has been tested on six different slice lengths {30s, 15s, 10s, 5s, 3s, 1s}, every time with three independent runs, and then the prediction have been evaluated both at frame and song level. Results, in terms of F1 scores, are summarized in Table II and Table III.

Regarding frame level evaluation, small slice lengths increase the accuracy and reduce the standard deviation across the three runs, as it can be seen from Table II. In particular, the best scores are obtained for 3s time slices, both in average (0.782234) and max value (0.786740), while the best standard deviation (0.004942) corresponds to 1s slices. This could be due to the fact that, having a wider training set (as shown in section III-B3), the model can better retrieve information from data, in a way that remains the same across different runs.

For song level evaluation, the situation is slightly different. As shown in Table III, applying majority voting to same song predictions results in an overall performance boost, that is much more noticeable as slice lengths become narrower. This time, best results are obtained with 1s slices (average F1: 0.891998, best F1: 0.909619), while the best standard deviation (0.016162) is achieved for 3s slices. With respect to the previous evaluations, the majority vote also causes a little overall variance increase, since most predicted genres for a certain song could change from one run to another, resulting in a major fluctuation among F1 scores.

To better understand the model quality, we choose to implement as a baseline a simple SVM structure, based on frequency feature sets, as mentioned in section II-A. In this case, timbral textures are calculated basing on 3s long windows. The model achieved a 0.7126 F1 score for frame level evaluation, that rises up to 0.7911 for song level evaluation. As one can notice, CRNN strongly outperforms this baseline for each of the three shortest slice lengths: this proves that the quality boost coming from the enlargement of datasets' cardinality overcomes the loss of global temporal structure deriving from the slicing procedure.

One last run with the best performing slice length has been carried out, and it has been used for some final visualizations.

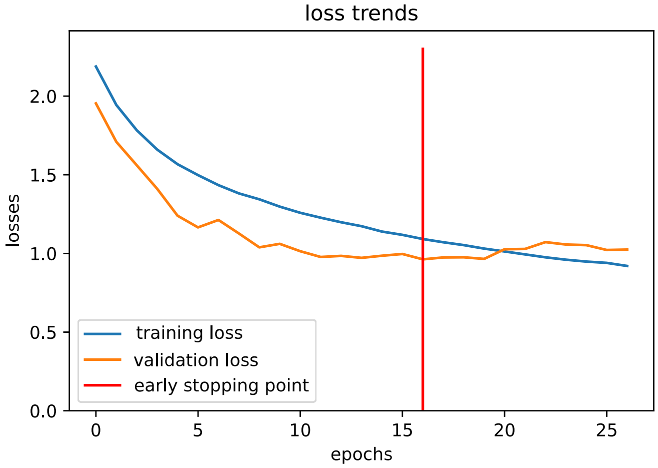
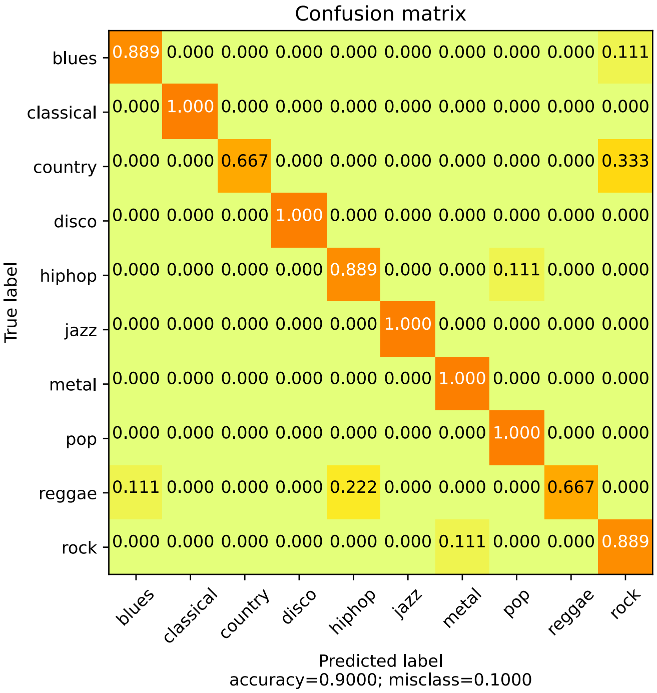


Fig. 3: Trends of training and validation losses across the epochs, plus early stopping threshold

The first one (Figure 3) regards the losses' evolution across the epochs: at the beginning both losses decrease quickly, while after a few iterations the validation one reaches a plateau. Thanks to early stopping, we are able to intercept the epoch for which we get the minimum value for validation loss, save it as the final result and therefore avoid the overfitting zone, for which the validation loss starts to diverge.



	blues	classical	country	disco	hiphop	jazz	metal	pop	reggae	rock
blues	0.889	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.111
classical	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
country	0.000	0.000	0.667	0.000	0.000	0.000	0.000	0.000	0.000	0.333
disco	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000	0.000
hiphop	0.000	0.000	0.000	0.000	0.889	0.000	0.000	0.111	0.000	0.000
jazz	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000
metal	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	0.000
pop	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000
reggae	0.111	0.000	0.000	0.000	0.222	0.000	0.000	0.000	0.667	0.000
rock	0.000	0.000	0.000	0.000	0.000	0.000	0.111	0.000	0.000	0.889

Fig. 4: Genre confusion matrix obtained from the best run

After the training phase, the model was applied to the test set, and outputted predictions have been used, together with ground truths, to compute a final confusion matrix (Figure 4). The results obtained are absolutely satisfying: the model managed to achieve the maximum micro-precision score for 5 classes out of 10, and also the remaining ones are generally good. The only genres that show relevant misclassifications are *country*, that in 33.3% of the cases was confused with *rock*, and *reggae*, that sometimes got mistaken for *blues* or *hiphop*. Since those mistakes refer to semantically similar genres, we can assume that those genres share common patterns also at spectrogram level, and that the model is able to find and use them in the prediction process.

A further extension regarding the song level evaluation consists in the definition of a *top-K* function, that outputs, for each song, the top K most predicted genres, and checks if the ground truth is among them. Its application to the described predictions shows that the correct class is among the two most predicted ones in 95.62% of total cases, and among the three most predicted ones in 98.88% of cases.

As last step, we tested our algorithm in a real-case scenario: we defined a simple function that submits to the model audio tracks collected from YouTube videoclips, after having preprocessed them according to section III-B. The application of this function to two songs of distinct genres has completed successfully, and the model has managed to predict correct labels, showing a good level of generalization.

V. Conclusion and Future Directions

In view of all the described approach and results, we can consider our work organic and satisfactory, and we hope it could act as a source of inspiration for future researches in this particular field.

For the sake of completeness, we describe a few possible forthcoming developements, that could help to furtherly improve genre classification quality and obtain better working structures.

- *Augmentation through masking frequencies*: one other way to augment data could consist of taking training spectrograms and masking a certain range of frequencies, chosen in a random fashion.
- *Overlapping time slices*: slicing spectrogram in a ’sliding window’ fashion, for which generated slices overlap each other, could help the model quality in two ways: firstly, it would bring a greater cardinality, acting as a form of augmentation; lastly, it could preserve some forms of temporal structures that may be lost with the not-overlapping approach.
- *Extension of already implemented augmentations*: data cardinality may be enhanced again by increasing the number of provided values for parameters (*noise factor* and *pitch factor*) on which the performed augmentation was done. To better understand this mechanism, refer to section III-B1.

References

[1] G. Tzanetakis and P. Cook, “Musical genre classification of audio signals,” *IEEE Transactions on speech and audio processing*, vol. 10, no. 5, pp. 293–302, 2002.

[2] Z. Nasrullah and Y. Zhao, “Music artist classification with convolutional recurrent neural networks,” in *2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, IEEE, 2019.

[3] D. L. Donoho *et al.*, “High-dimensional data analysis: The curses and blessings of dimensionality,” *AMS math challenges lecture*, vol. 1, no. 2000, p. 32, 2000.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.

[5] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.