POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

NAML Report
Music genre recognition

Author(s): **PAESANO Emanuele**
PIZZUTI Gianluca

Academic Year: 2022-2023

Contents

Contents **1**

Abstract **2**

1 Introduction **3**

2 Related works **4**

3 Theoretical background **5**

3.1 Melspectrogram 5

3.2 Convolutional neural networks - CNN 6

3.3 Recurrent neural networks - RNN 6

3.3.1 Gated Recurrent Unit (GRU) 7

4 Dataset **9**

4.1 Data description 9

4.2 Preprocessing 9

5 Methodology **11**

5.1 Model structure 11

5.2 Motivations of GRU 13

5.3 CNN and GRU: A good combination 14

6 Results **15**

7 Conclusions **17**

Bibliography **18**

Abstract

The aim of this project is to present a method to automatically identify the genre of musical clips. The genre is an abstract feature, but still, it is considered to be an important characteristics of music. Existing algorithms for genre recognition predominantly rely on feature extraction techniques. These extracted characteristics are then utilized to build a classifier on the genre label. However, genre is a fuzzy concept which is also subject to personal interpretation, and there may be overlapping of genres in the same song. For these reasons, genre recognition is an inherently difficult task, and the accuracy of a classifier based on the extracted features can be very sensitive to the datasets on which it is trained and tested. In our project, we utilize a pipeline for extracting mel-spectrogram features from audio clips coming from the popular GTZAN dataset, and subsequently build a classifier for the genre. The classifier model is based on a convolutional recurrent neural network architecture, and is trained on the dataset of the extracted mel-spectrograms, after a train-validation-test split of 80%, 10% and 10% respectively. Our model reaches an accuracy of around 73% in 25 epochs in the test set.

1 | Introduction

Music genre recognition (MGR) represents a widely explored area within music information retrieval, captivating the attention of researchers globally. Although this is a widely navigated area, it has been noted that there is no general, objective definition that describes the term genre in this context. This is due to the fact that there is significant overlapping between different classes of genres. The genre of music is subject to interpretation by a listener, and the definition seems to be fuzzy. This ambiguity makes the identification of the genre an inherently difficult task. While humans can effortlessly assign an abstract label to a music clip, it is difficult to project an automated system for genre classification.

The exploit on using powerful supervised deep learning architectures like convolutional neural network (CNN) and recurrent neural network (RNN) displaced almost all previously used methods based on hand-crafted features, and/or support vector machines (SVM). This because CNNs can capture spatial dependencies of data in the feature space, whereas RNNs can capture the temporal dependency among the input sequences). In the case of MGR, the spatio-temporal information of the music provides meaningful insight, which may improve the performance of the classifiers. The convolutional layers are responsible for extracting features from the melspectrogram of the music clips, while the recurrent layer is used for temporal aggregation of these features. However, traditional RNN layers encounter challenges such as training latency and exploding gradient problem. Trying to avoid as much as possible this issues, this work implements a variant method where is added a recurrent layer called "gated recurrent unit - GRU" to the CNN architectures to reach better classification. This study also suggests that the short audio clips of 3-s duration can also be used to identify the music genre.

2 | Related works

When considering related works, we are particularly interested in MGR tasks which involve a deep learning model as a classifier. Deep learning has been explored extensively as a possible way to solve MGR tasks, and the GTZAN dataset is commonly used as a benchmark to test different classifiers. Many of the proposed architectures rely on neural networks with a convolutional part and either a GRU layer or a LSTM (long short-term memory) layer for the recurrent part. For instance [2] uses a GRU architecture, while [3] uses LSTM. In our work we mainly test a model based on a convolutional recurrent neural network (CRNN) based on a GRU layer, such as the ones featured in [1] and [4]. The latter uses an architecture where the recurrent and convolutional layers are applied to the data independently, and the output is subsequently combined in a final softmax layer. As opposed to [4] (and similarly to [1]), we will use a sequential architecture, where a convolutional and max-pooling layers are followed by a GRU layer and a final softmax layer.

3 | Theoretical background

3.1. Melspectrogram

To make the data classifiable we need a suitable feature space to represent the initially unstructured audio files. In general, the frequency domain of the audio signal captures most of its distinguishing characteristics, so it is particularly useful for content-based audio classification tasks. As such, the power spectrum of the audio hits our target. As a second step, we need to consider that genre mainly depends on how human ear perceives the sound, and it is subject to human psychology. Thus, we need to somehow incorporate perception phenomena along with the production phenomena in the data representation.

Studies have shown that humans do not perceive frequencies on a linear scale. Indeed, we are better at detecting differences in lower frequencies than higher frequencies. In 1937, Stevens, Volkmann, and Newmann invented the mel scale, a psychoacoustic scale of pitch such that equal distances in pitch sound equally distant to the listener. The concept of a mel-spectrogram combines the characteristics of both the power spectrum and the mel scale. A mel-spectrogram provides an acoustic representation of audio in the time-frequency domain. It is calculated based on the power spectral density, denoted as $\mathrm{P}(f, t)$, which is sampled at specific time instants (t_i) and frequencies (f_j). The frequency is expressed on a mel scale.

The mel frequency scale is defined as follows:

$$mel = 2,595 * \log_{10}(1 + \frac{Hertz}{700})$$

And its inverse is:

$$Hertz = 700 * (10.0^{\frac{mel}{2,595.0}} - 1)$$

3.2. Convolutional neural networks - CNN

CNN is a variant of multilayer perceptrons and has been extensively used in DNNs. Figure 1 shows the block diagram of a typical convolution network. In the context of CNN, convolution refers to the mathematical operation of element-wise multiplication of input source with a kernel function. The kernel (or filter) function slides throughout the domain of the source function. Typically, the convolution layer is followed by a max-pooling layer. Max-pooling layer downsamples the output produced by the convolution layer. It also partitions the output into a set of nonoverlapping regions, and for each such region, the maximum value is taken as the output.

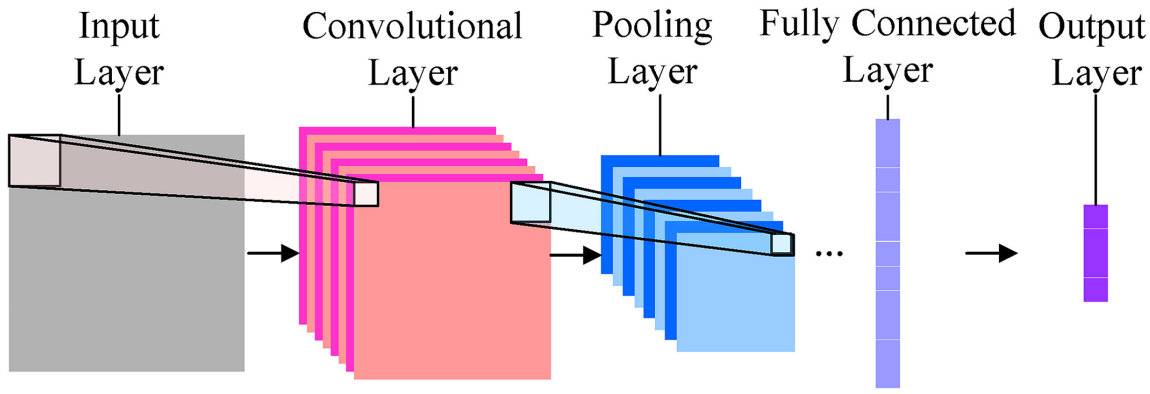

Figure 3.1: CNN structure

In convolution step, because the kernel is shared across the layer, it behaves like a pattern detector that outputs a high value when a specific pattern is found in the input and vice versa. The transformation such as melspectrogram exhibits unique variations in its pattern for different classes of audio genres. CNN can extract the musical pattern descriptors that can be used for classifying genres.

3.3. Recurrent neural networks - RNN

An RNN (Recurrent Neural Network) is a type of Deep Neural Network that is specifically designed to model sequence data. In an RNN, the connections between artificial neurons form a directed cycle. Unlike traditional neural networks where inputs and outputs are assumed to be independent of each other, RNNs take into account the dependence of outputs on previous inputs and outputs. The fundamental concept behind RNNs is the utilization of sequential information. However, the basic implementation of RNNs suffers

from a significant issue known as the vanishing gradient problem. During backpropagation, the gradients become extremely small, preventing effective training of the initial layers of the RNN. To address this problem, in this study we use the GRU architecture: a gating mechanism in RNNs that enables the learning of long-term dependencies.

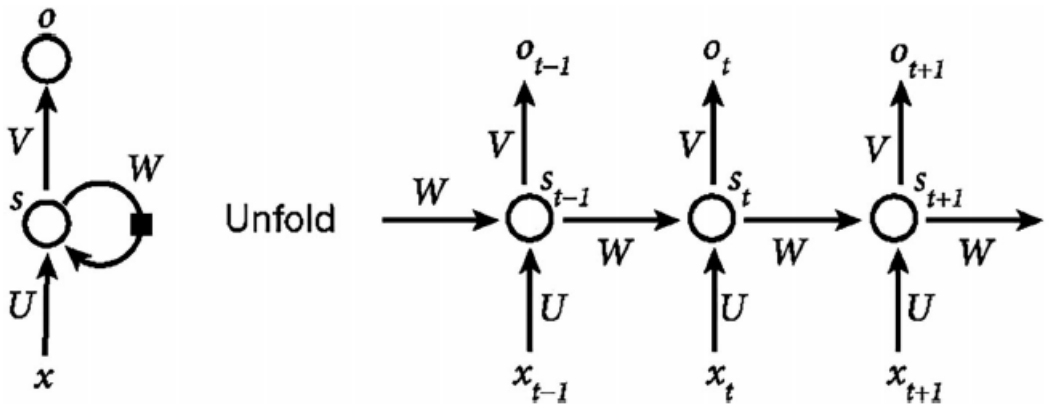

Figure 3.2: RNN structure

3.3.1. Gated Recurrent Unit (GRU)

The Gated Recurrent Unit is, together with the LSTM (long-short term memory), one of the main variants of the RNN. The GRU solves the vanishing gradient problem in RNN by using update and reset gates, which are two vectors that decide what information should be passed to the output. These gates can be trained to keep information from long ago without changing it through time or remove information which is irrelevant to the prediction. In particular, the reset gate is used to select which part of the previous information will be used, while the update gate is used to determine how to update the memory. Below we report the structure of a GRU.

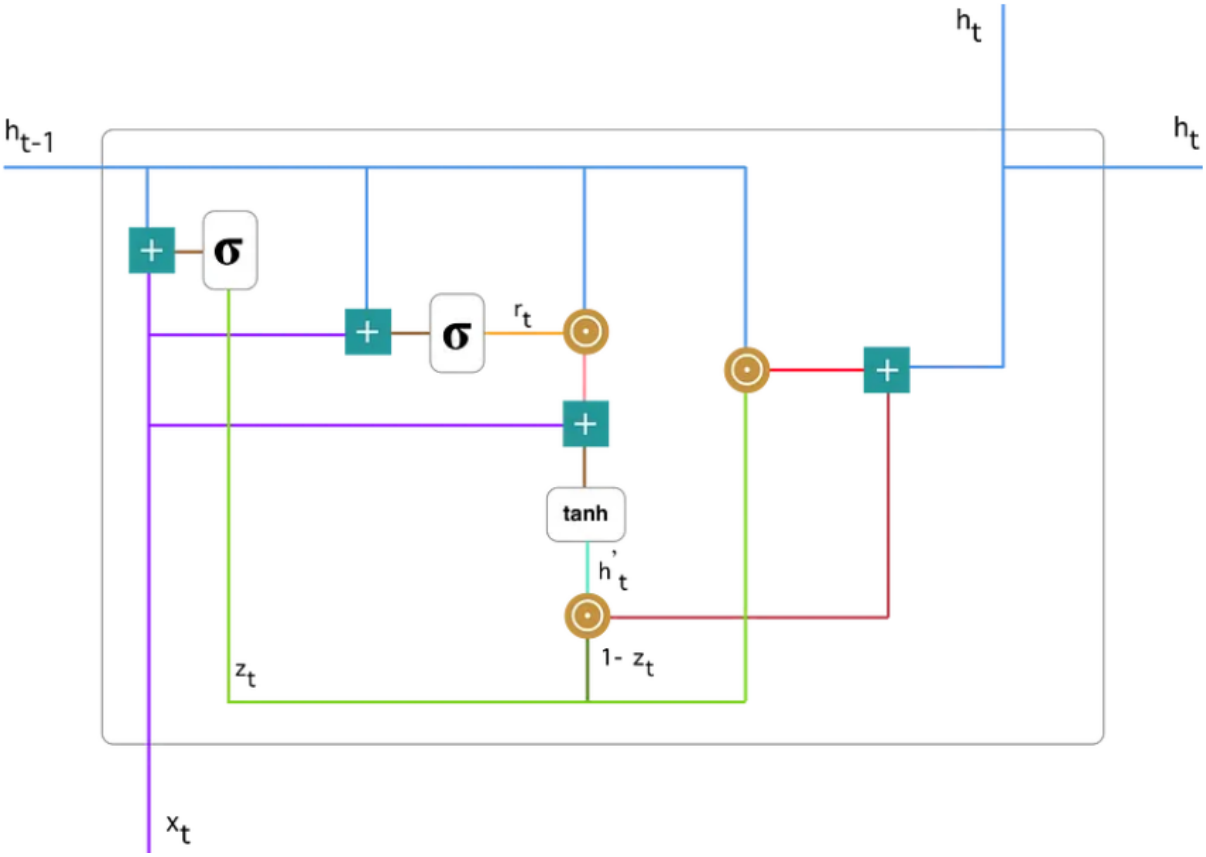

Figure 3.3: A Gated Recurrent Unit

In Fig. 3.3, h_{t-1} refers to the memory accumulated so far (in the previous $t-1$ units), while x_t is the current input. As shown in figure, the update gate z_t is learned with two weight matrices $W^Z_{x_t}$ and $W^Z_{h_{t-1}}$, and it is used to select how much of the old information h_{t-1} and of the new information h'_t to keep. In theory, a GRU could just pass the whole h_{t-1} to the next step ($z_t = 1$), bypassing the current unit and eliminating the risk of vanishing gradients.

The reset gate is learned with two different matrices $W^R_{x_t}$ and $W^R_{h_{t-1}}$, and it is used to generate h'_t, effectively selecting which parts of the vector h_{t-1} to use in the computation. This allows the model to keep only the relevant information from the past.

4 | Dataset

4.1. Data description

The dataset used in this work is the benchmark GTZAN dataset (Tzanetakis & Cook, 2002), which is widely used for genre recognition tasks. The dataset consists of 100 30-second audio clips for each of 10 different genre labels (except for jazz which has 99), which are:

`blues, classical, country, disco, hiphop, jazz, metal, pop, reggae, rock.`

Hence we have a total of 999 songs and 10 labels. Deep learning models rely on a significant amount of data for effective training, so that a CRNN architecture trained on the mel-spectrograms of the raw dataset may be prone to overfitting. For this reason, as part of our pre-processing pipeline, we split each song into 10 3-second snippets, leading to a total of 10000 snippets. In fact, it has been observed that even music clips of very short duration (such as 3 seconds) can provide sufficient information to assign a genre to them.

4.2. Preprocessing

The preprocessing of our dataset mainly consists in 2 parts:

1. divide the songs in 3-second snippets
2. extract the mel-spectrogram of each snippet

As MGR is such a popular topic, there is a wide supply of available software to achieve the desired tasks. In particular, the SLAPP (single-label audio processing pipeline) software is tailored for the GTZAN dataset and automates the entire data processing workflow for single-label audio classification task. It offers a number of options for building a suitable dataset for classification starting from raw audio clips. Documentation is available at

`github.com/MaxHilsdorf/single_label_audio_processing_pipeline`

We use it to divide each 30-second track into 3-second clips, and seubsequently to ex-

tract the mel-spectrogram of each clip. This way, the final dataset is composed of 9990 mel-spectrograms, which are shuffled and then split into training set (80%), validation set (10%) and test set (10%). Each spectrogram is an array of shape (130,100). We report below two samples of mel-spectrograms from a "blues" and a "rock" song.

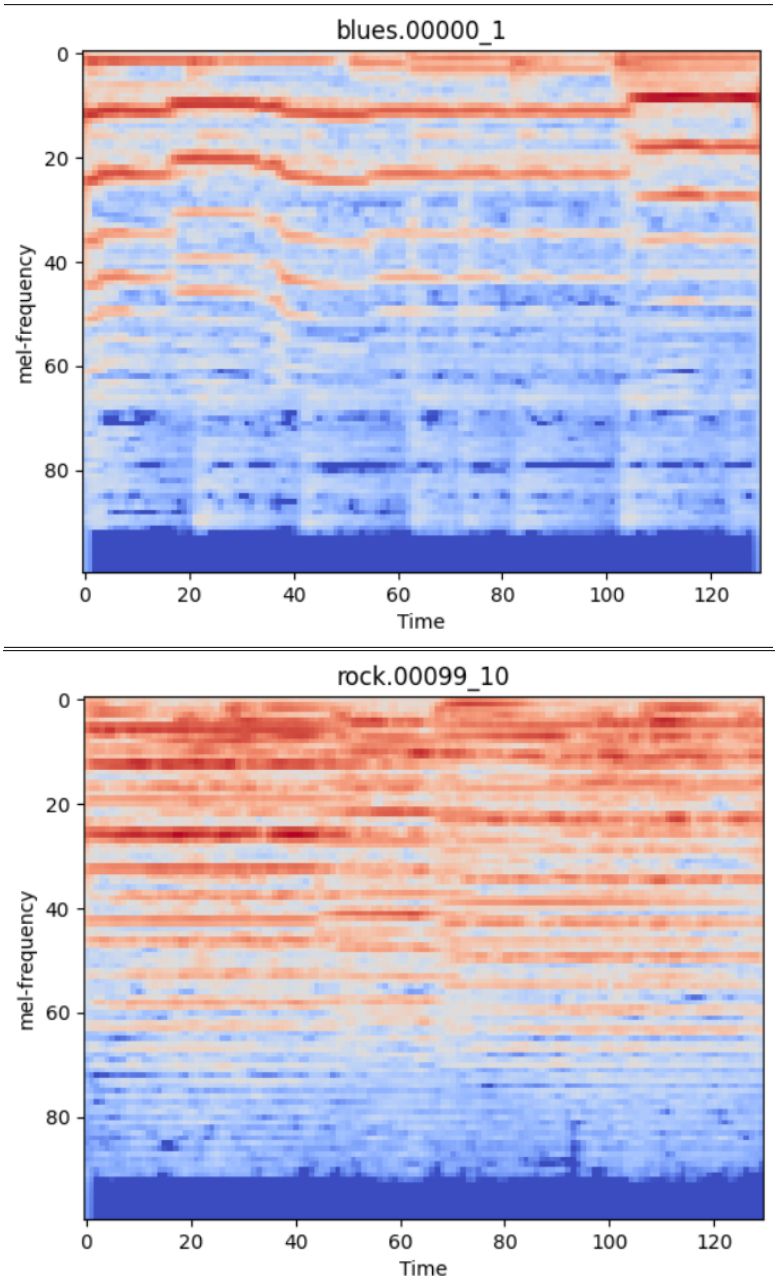

Figure 4.1: spectrograms from a blues and a rock song

5 | Methodology

5.1. Model structure

After preprocessing the data, our work has focused on the construction of the neural network to be used for classification. The model describing the process is a combination of a CNN with an RNN layer (GRU): this way, we obtain a Convolutional Recurrent Neural Network (CRNN). The model is composed of two convolutional blocks, both consisting of a 2D convolutional layer followed by a 2D max-pooling layer. This enables CNNs to look at non-overlapping regions of the audio signal and output the maximum value. By eliminating non-maximal values, it reduces computation for upper layers. A Batch Normalization layer is added after the max-pooling operation to suppress overfitting and accelerate the convergence of the training.
The aim of these convolutional blocks is to extract useful features to be given as input to the GRU layer for genre classification. Before the GRU layer, there is a reshape layer that effectively squeezes the frequency dimension.
In the end, the dense layer precedes the final output layer, which is used as a classifier to automatically classify the input audio into different genres. Fig.5.1 contains a depiction of the model architecture.

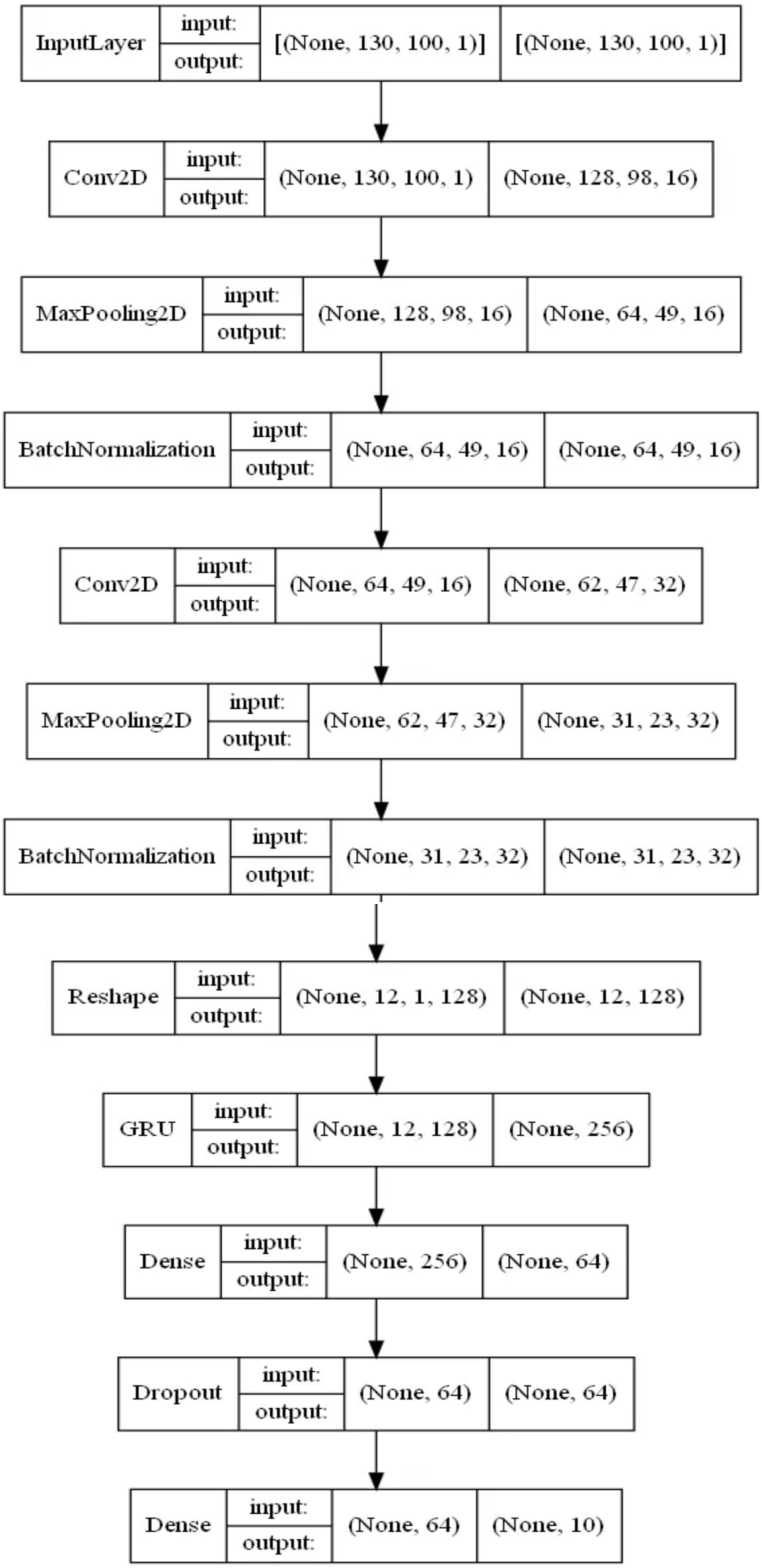

Figure 5.1: CRNN Architecture

As for the optimization algorithm, we use the Adam optimizer. Adam is an extension of the Adagrad algorithm, which is a Stochastic Gradient Descent variant that features an adaptive learning rate, which is scaled for each component of the parameter vector. However, Adagrad tends to lead to a very fast decrease of the learning rates which can prevent learning. Adam realizes the benefits of both the RMSProp and Adagrad algorithms, by combining the first and the second moments of the gradients, and has empirically shown good performance in general. We choose a batch size of 64 for our algorithm.

As regards the loss function, we deploy the "categorical cross entropy" loss. This loss function computes the cross-entropy between the true label and the predicted label, and is appropriate for classification problems. Cross-entropy is computed as:

$$J(w) = -\frac{1}{N}\sum_{i=1}^{N}[y_i log(\hat{y}_i) + (1 - y_i) log(1 - \hat{y}_i)]$$

Where w refers to the current parameters of the model, $\hat{y}_i$ are the predicted labels and y_i are the true labels.
We then proceed to training our model on the labeled spectrograms. We stop our training after 25 epochs, as we notice that the training accuracy does not improve significantly with further training.

5.2. Motivations of GRU

In the music genre recognition field there are a lot of adopted strategies to implement the classification model, most of them differ by the type of RNN's layer they decide to use. Typically, the most used technologies refer to the GRU and to the long short-term memory networks (LSTM).
In this project we have chosen to adopt a gated recurrent unit for many reasons.
First of all, the GRU's ability to capture temporal dependencies: music is a sequential and temporal data type and GRU, like other RNN variants, is well-suited for modeling sequential data and capturing long-term dependencies. It can learn to recognize patterns and structures in music by considering the context of previously observed data points. Another important aspect that leads us to our choice it was the efficient handling of vanishing gradients performed by the GRU. The gating mechanism allows the network to selectively update and forget information, mitigating the vanishing gradient problem. This helps the network effectively capture dependencies over longer sequences, which is crucial for music genre recognition.

Important GRU feature regards its simpler architecture compared to other RNN variants, such as LSTM. It has fewer parameters and computations, which can result in faster training and convergence and it also offers good trade-off between model complexity and performance. In this case, GRU performs comparably to LSTM while being computationally more efficient. Hence, GRU is relatively easier to interpret compared to LSTM due to its simpler architecture. The gating mechanism in GRU is more straightforward and can provide insights into the model's decision-making process. Additionally, the simplicity of GRU makes it more accessible and easier to implement in practical applications. This is beneficial when working with music datasets, as it reduces training time and computational requirements.

5.3. CNN and GRU: A good combination

The main reason behind the choice of implementing the model structure combining CNN and GRU lies in the ability of this structure of incorporate the temporal aspect of music and capture long-term dependencies. Motives for which GRU is a popular choice when combined with CNNs regard in particular the shape of the music data, the mitigation of the vanishing gradient problem and feature fusion.
In fact, we know that Music is a sequential data format where the order of notes, chords, and rhythms matters and GRUs are recurrent neural networks that excel at modeling sequential data. By incorporating a GRU layer after the CNN, the model can capture the temporal dependencies between different frames or segments of music.
As said before GRUs are designed to mitigate the vanishing gradient problem that often occurs in traditional recurrent neural networks, which makes them effective at capturing long-term dependencies. In music genre recognition, these long-term dependencies are crucial for identifying genre-specific structures that develop over time, such as the arrangement of sections, chord progressions, or melodic patterns. CNNs can extract local features from the input music signal but these features alone may not capture the global characteristics of a music piece that are important for genre recognition. By combining the output of the CNN with a GRU layer, the model can fuse local features from the CNN with temporal information from the GRU, creating a more comprehensive representation of the music data. In summary, The CNN is proficient at capturing local patterns and features within the music spectrogram, extracting information related to frequency and time-localized structures. On the other hand, the GRU, as a type of recurrent layer, excels at modeling sequential and temporal dependencies in the data. This fusion allows the model to create a more comprehensive representation of the music data by incorporating both local features and temporal information.

6 | Results

The model reaches an accuracy of 0.9889 in the training set after 25 epochs. However, we notice that the accuracy on the validation set struggles to keep up during the training, reaching a plateau at around 0.7 . This is a clear hint that the model is over-fitting the training set.
Unfortunately, the data which is available to us is limited, and more data would make the model able to generalize better. In any case, the performance on the test set is satisfactory, reaching an accuracy of 0.7139. Below we report the confusion matrix of the assigned labels versus the real labels in the test set.

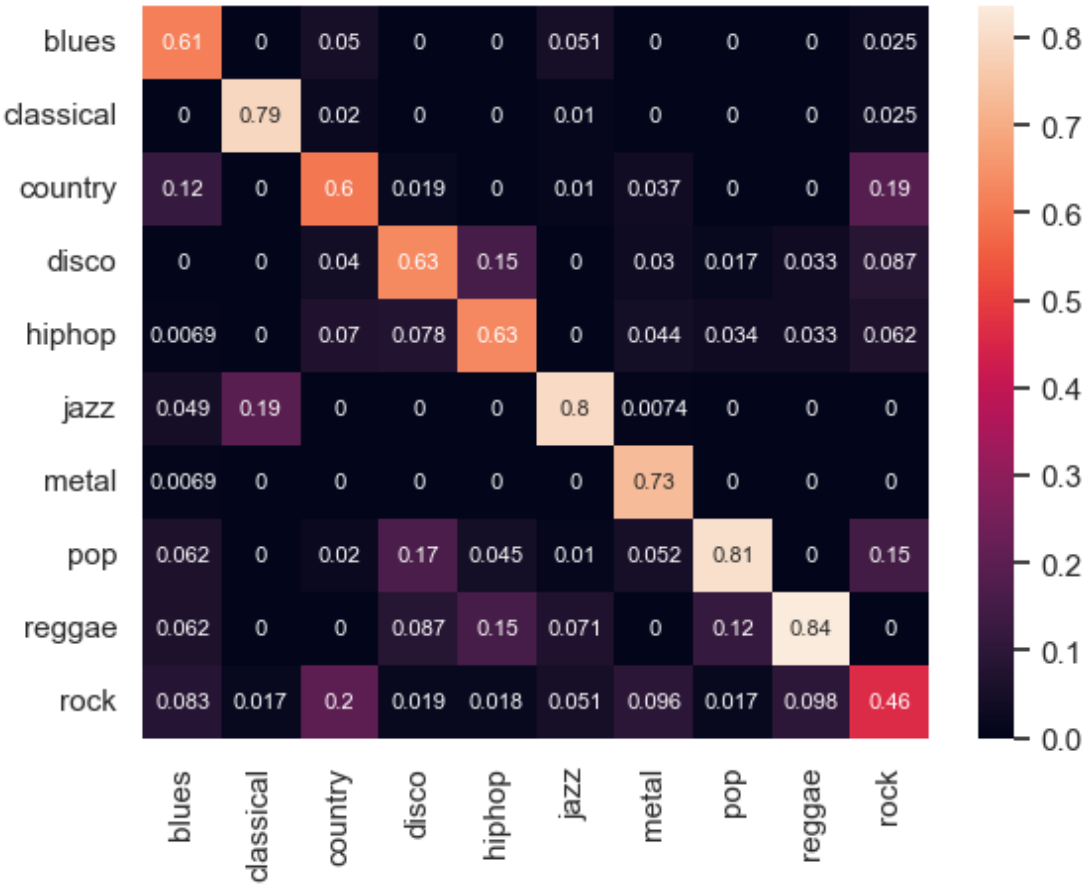

Figure 6.1: Confusion matrix of model

The model is generally able to predict the correct label, with the main source of confusion being the distinction between rock and country.
Possible causes of the mediocre accuracy of the model are attributable to the low dimensionality of the data. A sample size of 7983 spectrograms for the training set seems not to be sufficient to prevent over-fitting.
A possible strategy to circumvent this issue might be resorting to data augmenting techniques such as adding noise or shifting the pitch to the audio files. However, it is not clear how distorting the audio signal might affect the genre of the clip, as genre can be a very subtle and fuzzy feature. Additionally, it may be possible to experiment with different architectures to look for a better result.

7 | Conclusions

We provided a possible deep learning model to determine the genre of a song starting from an mp3 file. The model provides a sufficient accuracy in the test set, however it suffers from over-fitting problems, that might be solved with the availability of a larger dataset.

In order to look for a performance improvement, we have also experimented with substituting the GRU layer of the model with a LSTM (long short-term memory) layer. LSTM is the main alternative to GRU when it comes to RNN models. LSTM are a slightly more complex model, with a forget gate, an input gate and an output gate. However, the accuracy and performance of the overall model does not improve. Indeed, as the LSTM is a more complex model, it is at least as prone to overfitting as the GRU.

To conlude, possible future work might comprehend extracting a larger dataset of pre-labeled songs, and possibly generalizing the model to accommodate for a larger set of genres and subgenres, understanding more of the nuances of music.

Bibliography

[1] R. H. L. Dipjyoti Bisharad. Music genre recognition using convolutional recurrent neural network architecture.

[2] https://medium.com/@maxhilsdorf. Music genre classification using a divide conquer crnn.

[3] https://medium.com/@premtibadiya. Music genre classification using rnn-lstm.

[4] H. W. J. Y. S. L. RUI YANG, LIN FENG. Parallel recurrent convolutional neural networks based music genre classification method for mobile devices.