Music Genre Recognition with Convolutional Neural Networks

Amerigo Aloisi[†], Jaswant Singh Bogan[†] and Anna Zorzetto[†]
Dipartimento di Ingegneria dell'Informazione, Università Degli Studi Di Padova
Email: [†]amerigo.aloisi@studenti.unipd.it, [†]jaswantsingh.bogan@studenti.unipd.it, [†]anna.zorzetto.1@studenti.unipd.it

***Abstract*—Music genre recognition is a challenging task in deep learning classification, and it has useful application in real life, such as the automatic music genre categorization for big datasets and music recommendation systems. In this work we propose four different deep learning architectures based on Convolutional Neural Networks (CNN). We used different representations for the audio signal to understand which one provides the best results. We developed a 1D CNN that takes as input the raw audio signals and two 2D CNNs that take as input the log Mel-Spectrogram representation and the Mel-Frequency Cepstral Coefficients respectevively. Finally, to combine the information provided by different features, we proposed a Late Fusion Recurrent CNN (RCNN) that incorporates a Long Short-Term Memory (LSTM) block to exploit the sequential characteristics of the signal and a simpler and computionally lighter Ensemble Classifier that averages the outputs of the other pre-trained models in two different modes. The best results were obtained from the Ensemble Classifier and from the 2D Mel-Spectrogram CNN classification with an overall 87% of accuracy and 48% of F1 score.**

***Index Terms*—Music Genre Classification, Convolutional Neural Networks, Late Fusion, Ensemble Classifier.**

I. Introduction

In recent years, with the widespread use of smartphones and music distribution applications, for the management and organization of large music datasets we are moving towards automation, reducing the reliance on manual human efforts. In this context, the interest in deep learning architectures for music classification based on different attributes has increased. To provide better music recommendations for the users, it's essential to have an algorithm that can automatically characterize the music. This process is called Musical Information Retrieval (MIR) and a specific example is music genre classification, which will be explored in this work. The advantage provided by deep leaning is the ability to learn audio features and use them for autonomous classification purposes. The blurred boundaries between genres, due to their subjective definition, introduce some difficulties in this task. Audio data can be represented in two main formats: 1D and 2D. The former corresponds to the raw time series of the audio signal, the latter corresponds to the frequency representation of the audio signal through the spectrogram-based images. Two commonly used frequency domain features for audio signal analysis are the spectrogram and the Mel-Spectrogram. These representations capture valuable information about the frequency content and temporal dynamics of the audio, enabling effective pattern recognition for tasks like music genre classification. The state-of-the-art architectures for audio signal classification are convolutional neural networks (CNNs) and recurrent neural networks (RNNs). In our work we tested four different methods to classify signals retrieved from the the small subset of the Free Music Archive (FMA) dataset [1]. The first is a CNN that processes raw audio signals, the second and the third are 2D CNNs that process the Mel-Spectrogram and the Mel-Frequency Cepstral Coefficients, respectively. The third architecture is a Late Fusion CNN that integrates two convolutional blocks: one for the raw audio signal, and one for the Mel-Frequency Cepstral Coefficients, the learned representations are then given in input to a LSTM layer. The last classifier is an Ensemble Classifier that combines the outputs of the pre-trained networks to provide a more robust prediction . The shift towards automated approaches driven by deep learning represents a significant step forward in the efficiency and scalability of music dataset management and organization. Section 2 of this paper presents related work, previous and alternative techniques for audio signal classification, in Section 3 the processing pipeline is described, Section 4 introduces the signals and features used for classification, Section 5 describes the four models' architectures and Section 6 reports the final results. Finally in section 7 we report the conclusions of our work and possible future works.

II. Related works

CNNs have been widely used for audio signal classification and music classification. Li et al. used CNNs to extract musical pattern features in audio signals [2]. This work demonstrated the ability of CNNs to extract features from the variations of musical patterns, although it showed poor performance in generalising well to unseen data. Zhang et al. developed a CNN architecture with k-max pooling layers with the aim of achieving more robust music representations[3]. Zhang et al. built a hierarchical architecture to extract invariant and discriminative audio representations [4]. Sander et al. automatically extracted features from raw audio signals by using CNNs and investigated their performance. They concluded that CNN-based methods did not outperform spectrogram-based approaches[5]. Zhang et al. proposed two ways to improve music genre classification using CNNs. In one case, they combined max- and average-pooling to provide more statistical information to higher level neural networks, reduce computation demands, and ensure invariance to small translations. In the second case, they used shortcut connections to skip one or more layers. The input to the CNN is simply the short-time Fourier transform of the audio signal [6]. Aytar et al. [7] presented a multimodal CNN for audio classification. They implemented a series of 1-D convolutional and max-pooling layers with large kernels and high stride values in the branch of their network that encodes the audio data. In addition to CNN, even RNN are also used for music classification, as they are able to exploit the repetitive pattern for the classification. In Kostrzew et al. work [8], different architectures of learning frames are tested: Convolutional Neural Network, Convolutional Recurrent Neural Networks with 1D and 2D convolutions, and Recurrent Neural Network with Long Short-Term Memory cells. In 2020 A.Elbir et al.[9] proposed MusicRecNet, a specific classifier for music genre classification. It's composed of a CNN of three layers with the addition of dropout layers. This system provided very good results: 81.8 % accuracy with soft-max classifier and 97.6 % accuracy with SVM.

III. Processing Pipeline

In this section, the technical description with a high-level introduction to our work is reported. Deep neural networks obviate the need for task-specific prior knowledge, as they autonomously tailor features for the given task. Each element in the dataset is a 30 second song. From the dataset were removed the audio files that are corrupted or not in compliance with the rest of the instances. We normalized every clip to the same sampling rate, after that we randomly selected a 6 seconds slide from the original audio as input data to deal with our limited computional resources. Some data augmentation techniques are performed on the audio signal to reduce the risk of overfitting. We then compared two different kinds of normalization procedure: standard z-score normalization and batch-normalization layer as first layer of each network. In the context of our work we introduce four different kind of neural networks for the music genre classification, they will be described in details in section 5. The 1D CNN receives as input raw audio signals. The two 2D CNNs receive two different features derived form the raw audio: the Mel-Spectrogram and the Mel-Frequency Cepstral Coefficients (MFCCs). The Late Fusion RCNN works with both raw audio and MFCCs features, including also a LSTM layer before the classification. The architecture of the network needs to be carefully designed as it significantly impacts system performance. CNNs enhance the local connectivity pattern between the input and the CNN neurons. The inputs of a neuron in layer m come from a subset of units in layer $m - 1$ that are timely contiguous to each other. Deeper neurons of a CNN have a larger receptive field, which improves global feature learning. In addition, since the kernel is shared across a feature map, useful features can be detected regardless of their position in the signal. This weight sharing mechanism also increases learning efficiency by greatly reducing the number of parameters to learn, resulting in a better generalisation of the model. Max and average-pooling operations are performed between each convolution to provide more statistical information to subsequent layers. We set the parameters of the architectures by looking at the reference article [6], more recent papers [8] [14] and by empirically trying different combinations and selecting the one which provided the best performances.

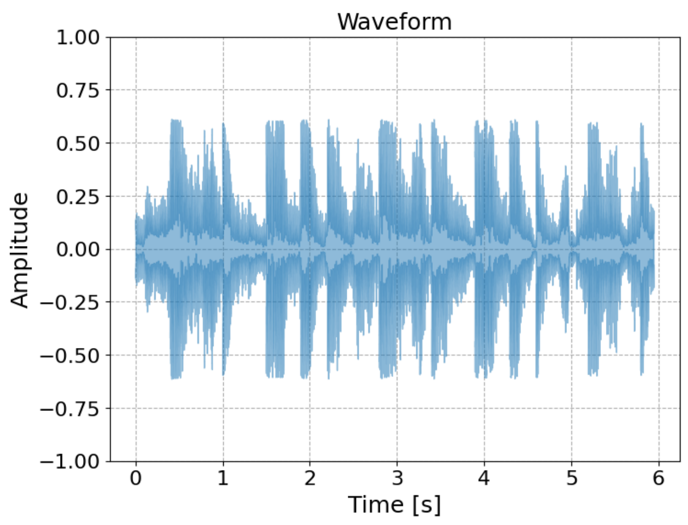

Fig. 1. Raw audio signal example

IV. Signals and Features

A. Dataset

The dataset we used is the Free Music Archive (FMA) [10], an open and easily accessible dataset suitable for evaluating several tasks, including music genre recognition. It provides full-length and high-quality audio, pre-computed features, along with track- and user-level metadata, tags, and free-form text such as biographies. Four different versions of the dataset of different sizes are provided: small, medium, large and full. Due to our limited computational resources, we used the small subset. It consists of 8,000 30s clips from the 8 most frequent genres: Electronic, Experimental, Folk, Hip-Hop, Instrumental, International, Pop and Rock. It's a balanced subset with 1,000 clips per genre, 1 root genre per clip. We used the 80/10/10 % split into training, validation and test sets, as suggested by the authors of the dataset. [6] We identified some audio samples that are corrupted and could not be loaded so we proceeded to remove them from the dataset, we are left with 7994 tracks and with the label "Electronic" having 994 samples left, which makes the dataset still balanced. The clips come with two different sampling rates (44.1KHz and 48KHz) so we proceeded to normalize every track to the same rate, the former. After this operations, a random set of 2^{18} samples, which correspond to approximately 6s, are extracted from each 30s clips. This is done as already said for computational purposes but also serves as a data augmentation technique, expanding significantly the virtual size of the dataset. Fig.1 shows the waveform representation of one audio clip after this first steps.

B. Features

The goodness of a learning model highly depends on the representation of the input data that is provided. Therefore, the selection of the appropriate features is crucial when employing DL-based methods. There are many different features that can be extracted from an audio signal and employed as input to a learning model: Chromagram, Mel Frequency Cepstral Coefficient (MFCC), Spectral Contrast, Swaragram, Tonnetz, Short Time Fourier Transform (STFT), Constant-Q Transform (CQT), Mel-Spectrogram, and Harmonic separated Mel-Spectrogram. [10] Among these, the ones we took into consideration are the Mel-Spectrogram and the Mel-Frequency Cepstral Coefficients.

a) Mel-Spectrogram: The human ear is not equally sensitive to different frequencies, and its resolution varies along the frequency axis. To analyze sound in a similar way to the human auditory system, it is better to use non-linear frequency scales. Mel scale is the frequency scale that corresponds to perceptual behaviour, i.e. it is related to the psychological perception of the pitch of a pure sound. Therefore, the Mel scale is designed so that pitch intervals are perceived as equally spaced by the human auditory system. A relationship between the Mel scale, $mel(f)$, and the frequency scale in Hertz, f, can be given as:

$$mel(f) = \frac{1000}{log2} * log(1 + \frac{f}{1000}) \qquad (1)$$

A spectrogram is a time-frequency representation of a signal obtained by performing a Short-Time-Fourier-Transform (STFT) on small overlapping segments of the signal. The STFTs are computed through the FFT algorithm. The segments are created with a window that slides through the signal with a certain hop size, we set the lenght of the FFT window to 1024 and the hop size to 512, this provides a good compromise between time and frequency resolution for the purposes of the task. The Mel-Spectrogram is obtained by applying a bank of 128 overlapping triangular filters to the spectrogram for calculating the energy of spectrum in each band. The shape of each Mel-Spectrogram is [128, 512]. In addition, the power amplitudes of the Mel-Spectrogram are converted to the decibel (dB) scale, which compresses the dynamic range of the signal, and makes it easier to see the relative strengths of the different frequency components in the spectrogram. Fig2 shows an example of the Log scale Mel-Spectrogram from an audio clip of our dataset.

b) Mel-Frequency Cepstral Coefficients: The Mel Frequency Cepstral Coefficients (MFCCs) are obtained by applying a Discrete Cosine Transform (DCT) on a Log scale Mel-Spectrogram. They represent the short-term power spectrum of a sound signal. MFCCs are a compact representation of the sound's spectrum, and they are less sensitive to noise than the raw spectrum. This makes them well-suited for use in speech recognition and music information retrieval. The Discrete Cosine transform (DCT) on the Mel filter bank [13]

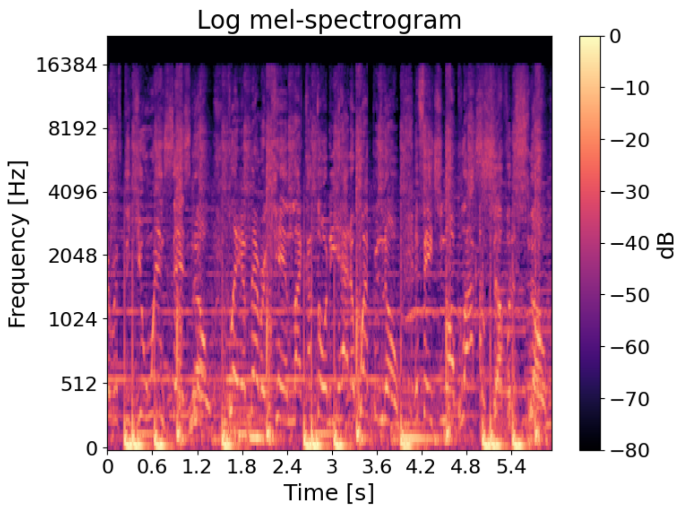

Fig. 2. Log Mel-Spectrogram example

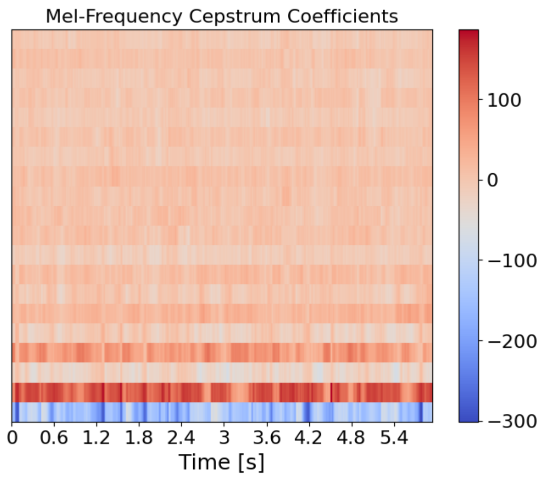

Fig. 3. Mel Frequency Cepstral Coefficients example

is computed by means of the following equation:

$$X(k) = \sum_{n=0}^{N-1} x_n * cos(\frac{2\pi jnk}{N}) \qquad (2)$$

where x_n is the discrete signal and N is the length of the signal. The shape of each MFCC representation is [20,512], each 2D representation consist of 20 coefficients for each of the 512 time frames of the audio signal. Fig3 shows the MFCCs computed on a clip of our dataset.

V. Learning Framework

We used the *librosa* library to load audio tracks and extract random clips of about 6 seconds. The usage of the entire 30 second signal as input to the learning model is useless and results in a waste of computational resources, in terms of cost, complexity, time and architecture's design. The random clips of 6 seconds have a beneficial effect, allowing the model to learn the genre of a song regardless of the exact starting point.

We also apply data augmentation transforms to the raw audio signals in order to increase the size of our training set and to introduce additional diversity, which can be beneficial in terms of generalization performance and to reduce overfitting. $audio\_transform$ is an audio transformation pipeline which uses the $audioamentations$ library. It consists of the introduction of:

- gaussian noise by means of the function $AddGaussianNoise$ to make the model more robust against random noise;
- time stretching to the audio signal, altering its duration by means of $TimeStretch$ function to make the learning model more robust against changes of velocity. This is done while maintaining the same total length of the transformed sample;
- Shifts the pitch of the audio signal by a certain number of semitones by means of $PitchShift$ function.

Each of these transforms have a probability of 0.5 of being applied. The *RawAudioDataset* class defines the input for the 1D CNN model. To reduce the huge dimensionality, each 6s audio clip has been furthermore downsampled to half the sampling rate (22.05KHz). We found that this choice sharply reduced the model complexity while providing the same performance of the not downsampled version. The *MelSpectrogramDataset* and *MFCCDataset* classes compute the respective inputs for the two 2D CNNs. Since the size of the images is fixed, the feature have been computed on the original 6s signal sampled at 44100 Hz to achieve a higher frequency resolution. Finally, we compared two ways of normalizing the different datasets. In one case the standard z-score normalization procedure is applied to the signal. In the case of the raw audio signal and the MFCCs, each sample is normalized by subtracting the mean and dividing by the standard deviation, both evaluated on the whole training set. In the case of the Mel-spectrogram, the normalization is performed to each Mel-band separately because value distributions differ significantly between frequency bands. In the second case, we instead placed a Batch-Normalization layer in input of each architecture. The difference is that the samples are normalized with respect to the batch, and not to the entire dataset. We empirically found that there is no difference in terms of model performance between these two choices, but the latter allows to save some heavy computation (loading all the audio samples, computing all the Mel-Spectrograms and MFCCs and their mean and variance) by just adding two more weights to the Neural Network for the Batch-Normalization layer.

A. Parameters settings

a) Optimizer: We used the Adaptive Moment Estimation (Adam) optimizer. It is extremely popular due to its good performance compared to other optimizers and typically it's the default choice.

b) Loss function: The loss function used is the cross-entropy for multimodal classification, whose formula is :

$$L_{CE} = -\sum_{c=1}^{M} y_{o,c} log(p_{o,c}) \tag{3}$$

where M is the number of classes, $y_{o,c}$ is the correct label for observation o, and $p_{o,c}$ is the predicted probability that observation o is of class c.

c) Activation function: Rectified Linear Units (ReLUs) are used as the activation functions in all convolutional and fully connected layers, except for the last layer of each network where the Softmax activation is used instead. The ReLU activation function is defined as:

$$f(x) = max(0, x) \tag{4}$$

The ReLU is useful for overcoming the gradient vanishing problem, it's simple since it's a linear function for positive values and it speeds up convergence.

d) Strategies to avoid overfitting: To avoid overfitting we used three strategies: data augmentation, early stopping, and dropout. Data augmentation consists in increasing the training set by applying some transformations to the instances and keeping the same labels. Early stopping consists in stopping the training phase when the validation loss stops decreasing according to a certain patience parameter, since it means that the model is approaching the overfitting thus the learning needs to be stopped. The idea of dropout consists of randomly remove some neurons at each epoch. By removing them, they are temporarily excluding them from the network along with their incoming and outgoing connections. The dropout has many beneficial effects: it reduces co-adaptation, each neuron is self-sufficient and performs well independently of the others. In this way, the learning model is more stable and robust.

e) Batch normalization: Batch normalization is a technique used in CNNs to stabilize and accelerate the training process. It works by normalizing the input layer by adjusting and scaling the activations. This process helps to mitigate the problem known as ”internal covariate shift,” where the distribution of each layer's inputs changes as the parameters of the previous layers change during training.

B. Baseline: 1D CNN for raw audio

The first CNN processes the audio waveform (input size is [1,131072]) to predict class labels. The structure of the 1D CNN is reported in figure 4. It consists of four convolutional blocks alternated by pooling layers and a classifier. The convolutional blocks are composed of three layers: a 1-dimensional convolutional layer, a batch-normalization layer and the ReLU activation function. After the last convolution block both a max and average pooling operations are performed along the whole time axis to summarize temporal information and provide statistical information to the following layers [6]. The outputs of these two pooling layers are then concatenated to be fed as input to the classifier. The classifier is composed by 3 fully connected blocks alternated by an activation function (ReLu

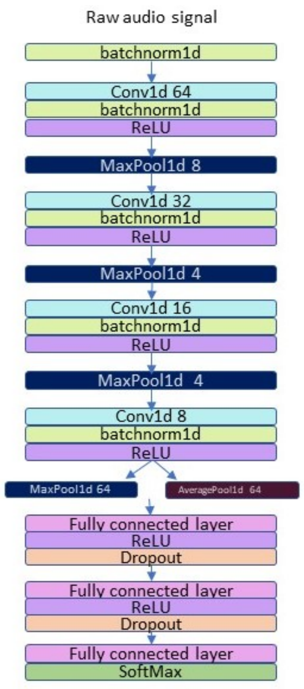

Fig. 4. 1D CNN

in the first two) and a dropout layer. Finally the SoftMax activation function is used for the final classification. The structure of this network is inspired by [6] and [14].

C. Baseline: 2D CNN for Mel-Spectrogram

The second baseline model is a 2D CNN that processes the Log Mel-Spectrogram for the classification. The structure of the network is shown in figure 5. There are four convolutional blocks (each composed of a convolutional layer, batch-normalization layer and the ReLU activation function) alternated by max-pooling operations. Again the output of the last convolutional block is fed as input to a max-pooling-layer and average-pooling layer and then these two outputs are concatenated. The classifier is the same of the previous architecture. The input image of dimension [128,512] is gradually reduced into a [2,2] square, in this way we retain both temporal and frequency information (grouped in 4 quadrants) that is summarised by the last 2D pooling operation.

D. 2D CNN for Mel-Cepstrum Coefficients

The structure of the 2D CNN for Mel-Cepstrum Coefficients classification is reported in figure 6. It's analogous to the structure of the previous presented network. In this case we decided to reduce the input of size [20,512] to a vector of size [1,64] to retain more temporal and sequential information about the signal.

E. Late Fusion Convolutional Recurrent Neural Network

Starting from the idea of integrating both time and spectral features in a single CNN model, we implemented an hybrid

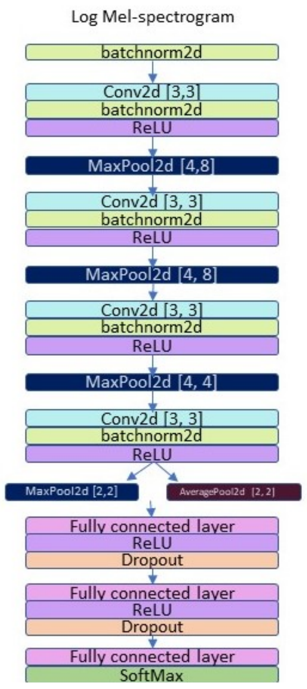

Fig. 5. 2D CNN for Mel-Spectrogram classification

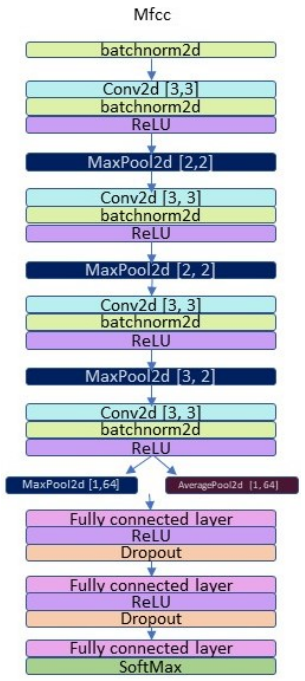

Fig. 6. 2D CNN for Mfcc classification

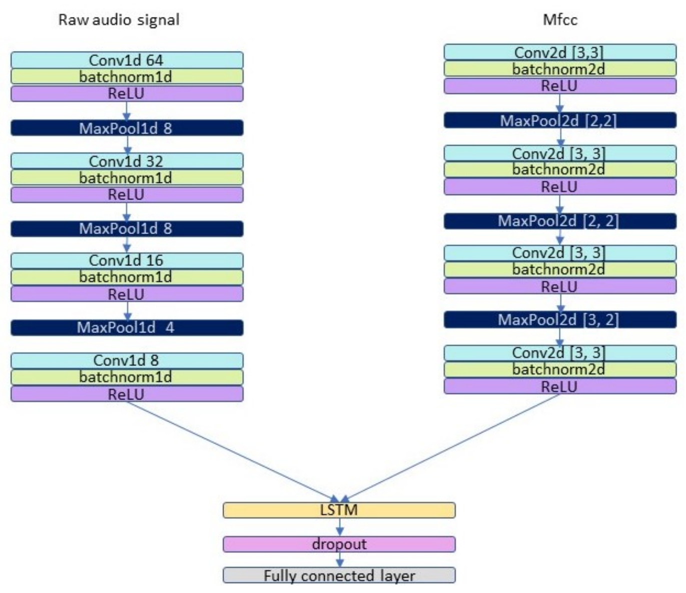

Fig. 7. multimodal classification using a LSTM layer

network that uses both raw audio and MFCCs for the classification. In particular, the late fusion approach suggests to merge the data one step before the classification. This is in contrast to the early fusion approach that integrates several features in only one data source at the first layer of the network, thus producing redundant information.[15] We reused the two different convolutional blocks previously described in the 1D CNN and the 2D MFCC CNN. Both these outputs condense temporal information in a [1,64] vector. To better exploit this sequential information, we decided to replace the simple feed-forward classifier with a single-layer Long Short Term Memory (LSTM) block. The advantage provided by the LSTM is that it retains information that can propagate in time without modifications even in long sequences. The structure of the layer is so that there's a straight path in the propagation of the gradient to allow backpropagation without passing multiple times through the $tanh$ activation function in order to overcome the vanishing gradient problem. The LSTM receives as input the sequences of 128-D feature vectors and it processes the sequential data in a way that allows it to remember and learn patterns over different time steps, by updating its hidden state and cell state. After the LSTM, a dropout layer is introduced to prevent the risk of overfitting, due to the increased complexity that the introduction of this layer brings in, and a fully connected layer with output size of 8 and Softmax activation function for the final prediction.

F. Ensemble Classifier

Finally we developed an ensemble classifier that combines multiple pre-trained models. This choice comes from the idea that each one of the different classifiers we built excels in recognizing certain patterns rather than others. By averaging their classification outputs we are also averaging the error of the individual classifiers, obtaining a more robust and

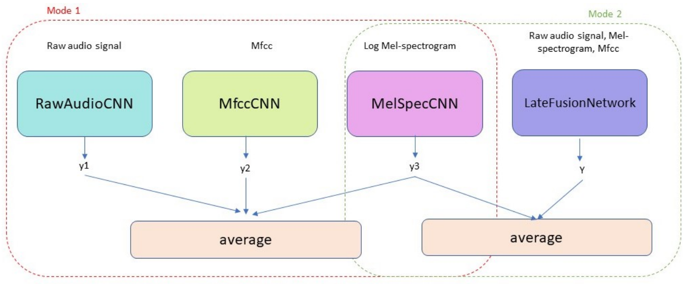

Fig. 8. Ensemble classifier schema

generalized performance. Other then that, the usage of pre-trained classifiers saves a significant amount of time and computational resources because we did not have to train a new model from scratch, but just use the combination of the already trained ones in the testing phase. In particular, we provided two different combination as shown in figure 8. The Ensemble Classifier in 'mode 1' averages the outputs from the 1D, 2D Mel-Spectrogram and 2D MFCC networks. The 'mode 2' uses instead the output of the Late Fusion (which has already combined together 1D and 2D MFCC) with the output of the Mel-Spectrogram 2D network.

VI. Results

We used four different metrics to evaluate the performance of our models. TPi, TNi, FPi, FNi and k correspond respectively to True positive, True negative, False positive, False negative and number of classes:

- accuracy calculated as:

$$\frac{1}{k}\sum_{i=1}^{k}\frac{TP_i+TN_i}{TP_i+TN_i+FP_i+FN_i} \tag{5}$$

  it corresponds to the fraction of correctly predicted labels over the total number of instances;
- precision

$$\frac{1}{k}\sum_{i=1}^{k}\frac{TP_i}{TP_i+FP_i} \tag{6}$$

  it's the ratio of true positive predictions over the total number of positive predictions made by the model;
- recall

$$\frac{1}{k}\sum_{i=1}^{k}\frac{TP_i}{TP_i+FN_i} \tag{7}$$

  it evaluates a model's ability to correctly identify all relevant instances of a certain class;
- F1-score

$$2*\frac{precision*recall}{precision+recall} \tag{8}$$

  it combines precision and recall. It ranges from 0 to 1, with higher values indicating better performance. A perfect classifier would have an F1 score of 1, while a completely ineffective classifier would have an F1 score of 0.

TABLE I
PERFORMANCE COMPARISON EVALUATED ON THE TEST SET

model	# of weights	accuracy	precision	recall	F1 score
1D-CNN	158106	83.66 %	37.35 %	34.62 %	35.94 %
2D-CNN Mel-Spectrogram	139306	86.84 %	47.90 %	47.38 %	47.64 %
2D-CNN MFCC	139306	85.47 %	41.37 %	41.88 %	41.62 %
Late Fusion	412728	84.53 %	38.63 %	38.12 %	38.38 %
Ensemble Classifier (mode1)	-	87.03 %	47.56 %	48.12 %	47.84 %
Ensemble Classifier (mode2)	-	84.25 %	35.35 %	37.00 %	36.15 %

Table 1 reports the performance comparison for the different architectures. Of the four proposed models, the 2D Mel-Spectrogram CNN outperforms the other architectures for each of the performance metrics. The 2D MFCC scores second with the same complexity of the first. The 1D CNN performs the worst. The Late Fusion network did not produce the expected improvement, as it ranks only third among the 4 models, while being about 3 times more complex. It is worth to mention that the Late Fusion model was trained for half the epochs of the others (10 versus 20), but it went early stopping, this suggests that the performance would not have increased anyways if trained for more epochs. This is in regard to models made from scratch. Regarding the Ensemble Classifier, we can give validity to our idea of combining strengths and weaknesses of each individual classifier to obtain a more robust model, as it achieves the best performance overall (best accuracy, recall and F1-score) when used in mode 1 (combination of raw audio, Mel-Spectrogram and MFCCs). It does not provide particularly good results when in mode 2 (Late Fusion and Mel-Spectrogram), probably because of the influence of the Late Fusion model which was already not so performing. The accuracy score is significantly higher than the precision and the recall for every model, this is the result of a large number of true negatives. In general the higher accuracy could be a misleading metric about the goodness of the model when the dataset is unbalanced and one class heavily outnumbers the others. However this is not the case at hand as the dataset has already be said to be balanced. Moreover, as it can be seen from the confusion matrix in fig 11, some classes are considerably more difficult to predict than others, with ”Electronic” and ”Instrumental” being the most difficult ones and ”Hip-Hop” being the easiest one. This clearly depends on the intrinsic characteristics of each music genre, with some of them being more easily recognizable than others due to some peculiarities (for example rap singing in Hip Hop music) and could also depend on the how the labeling was performed by the creators of the dataset.

VII. CONCLUDING REMARKS

In this study, we investigated the application of a deep learning framework to the challenge of music genre recognition while working under the constraints of limited computational resources. Our approach was driven by the need to reduce the complexity of our model, and led us to abandon extensive parameter optimization techniques. We compared the performances obtained by means of different architectures, such as 1D and 2D Convolutional Neural Networks, a Late

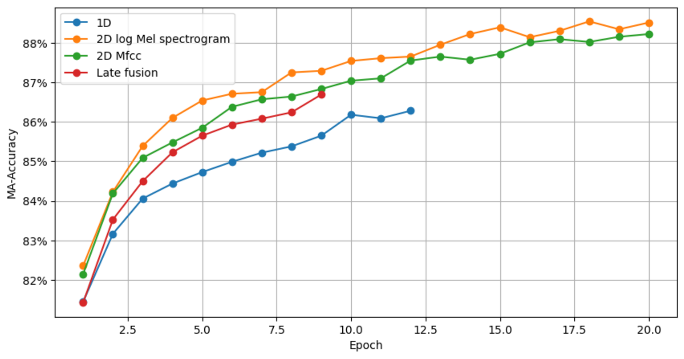

Fig. 9. comparison of the accuracy of the different models in the training phase

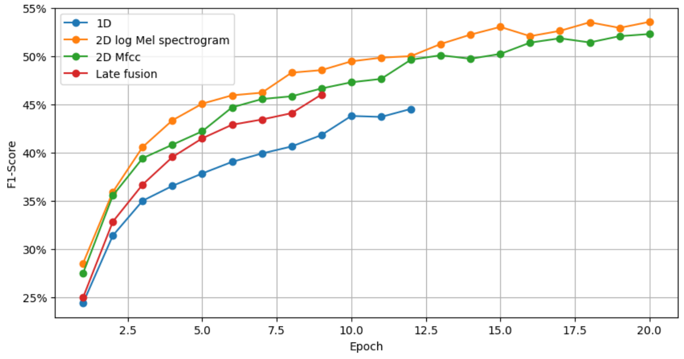

Fig. 10. comparison of the F1 score of the different models in the training phase

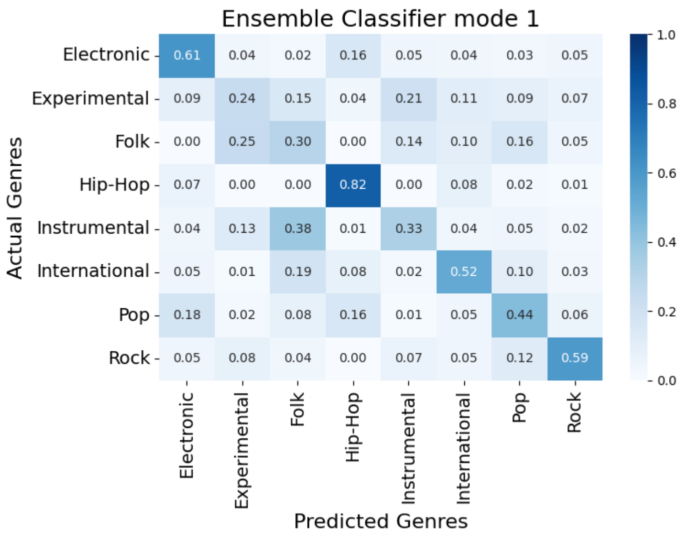

Fig. 11. confusion matrix of the Ensemble Classifier mode 1

Fusion Convolutional Recurrent Neural Network and an Ensemble Classifier. In the end, the best results were obtained when employing the Ensemble Classifier which averages the performance given by the Audio CNN, the Mel-Spectrogram CNN and Mel-Cepstrum Coefficients CNN, but the Mel-Spectrogram one alone achieved almost the same results as when combined with the other networks. We can therefore conclude that the most appropriate feature for such a task is the Mel-Spectrogram representation of the audio signal. The LSTM network did not perform as well as expected, but this could be a consequence of the limited size of the dataset, which makes the proposed model too complex for the dataset at hand. Future works could experiment with the availability of higher computational resources the usage of deeper models and different architectures with bigger datasets, to avoid being limited from the risk of overfitting. In addition one could try to increase the input length (more than 6s) to understand which one is the optimal length for the time segment that provides the best trade off between complexity and performance.

References

[1] M. Defferrard, K. Benzi, P. Vandergheynst, and X. Bresson, "FMA: A dataset for music analysis," in 18th International Society for Music Information Retrieval Conference (ISMIR), (Suzhou, China), Sept. 2017.

[2] T. L. Li, A. B. Chan, and A. Chun, "Automatic musical pattern feature extraction using convolutional neural network," in Proc. Int. Conf. Data Mining and applications, 2010.

[3] P. Zhang, X. Zheng, W. Zhang, S. Li, S. Qian, W. He, S. Zhang, and Z. Wang, "A deep neural network for modeling music," in Proceedings of the 5th ACM on international Conference on Multimedia Retrieval. ACM, 2015, pp. 379–386.

[4] C. Zhang, G. Evangelopoulos, S. Voinea, L. Rosasco, and T. Poggio, "A deep representation for invariance and music classification," in Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on. IEEE, 2014, pp. 6984–6988

[5] S. Dieleman and B. Schrauwen, "End-to-end learning for music audio," in Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on. IEEE, 2014, pp. 6964–6968.

[6] Zhang, Weibin et al. "Improved Music Genre Classification with Convolutional Neural Networks." Interspeech (2016).

[7] Y. Aytar, C. Vondrick, and A. Torralba, "SoundNet: Learning Sound Representations from Unlabeled Video," in Proceedings of the 30th International Conference on Neural Information Processing Systems, (Barcelona, Spain), pp. 892–900, Dec. 2016.

[8] Kostrzewa, D., Kaminski, P., Brzeski, R. (2021). Music Genre Classification: Looking for the Perfect Network. In: Paszynski, M., Kranzlmüller, D., Krzhizhanovskaya, V.V., Dongarra, J.J., Sloot, P.M.A. (eds) Computational Science – ICCS 2021. ICCS 2021. Lecture Notes in Computer Science(), vol 12742. Springer, Cham. pp. 55-67 https://doi.org/10.1007/978-3-030-77961-0_6

[9] Elbir, A. and Aydin, N. (2020), Music genre classification and music recommendation by using deep learning. Electron. Lett., 56: 627-629. https://doi.org/10.1049/el.2019.4202

[10] Yeshwant Singh, Anupam Biswas, Robustness of musical features on deep learning models for music genre classification, Expert Systems with Applications, Volume 199, 2022, 116879, ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2022.116879.

[11] D. C´ iric´, Z. Peric´, J. Nikolic´, and N. Vucˇic´, "Audio Signal Mapping into Spectrogram-Based Images for Deep Learning Applications," in 20th International Symposium INFOTEH-JAHORINA (INFOTEH), (San Francisco, CA, USA), pp. 1–6, Mar. 2021.

[12] Jacopo Pegoraro, Neural Network and deep learning, course notes, 2022/2023

[13] Z. K. Abdul and A. K. Al-Talabani, "Mel Frequency Cepstral Coefficient and its Applications: A Review," in IEEE Access, vol. 10, pp. 122136-122158, 2022, doi: 10.1109/ACCESS.2022.3223444.

[14] Sajjad Abdoli, Patrick Cardinal, Alessandro Lameiras Koerich, End-to-end environmental sound classification using a 1D convolutional neural network, Expert Systems with Applications, Volume 136, 2019, Pages 252-263, ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2019.06.040.

[15] Cheng, Yu-Huei Chang, Pang-Ching Kuo, Che-Nan. (2020). Convolutional Neural Networks Approach for Music Genre Classification. 399-403. 10.1109/IS3C50286.2020.00109.