Improved Music Genre Classification with Temporal Convolutional Neural Networks

Filippo Festa[†], Filippo Orlando[†]

***Abstract*—In recent years, research has demonstrated the efficacy of residual deep neural networks in several classification tasks, including music genre classification (MGC). A rare occurent example is the Temporal Convolutional Neural Network (TCNN). In this paper we employ temporal convolutions and residual learning to tackle the challenges of MGC, often characterized by intricate temporal patterns and dependencies. Our study proposes three methods to enhance MGC: 1) combine TCNNs with residual learning; 2) test TCNNs with different temporal data types (waveform, Mel spectrogram, images); 3) design a TCNN with bottleneck residual configuration to enhance the depth of the network. The obtained results are partial and not entirely satisfactory. However, the ones related to the latter method are encouraging and could lead to improved performance in various fields where TCNNs have already been implemented, such as speech recognition, video analysis, sequence modelling and forecasting.**

***Index Terms*—FMA Dataset, Residual Learning, Convolutional Neural Networks, Temporal Convolutional Neural Network, Music genre classification, Bottleneck configuration.**

I. Introduction

In recent years, the proliferation of personal multimedia devices has led to a significant increase in the availability of music across various application platforms. However, the volume of this music has made it increasingly challenging for individuals to effectively structure and organise it.

Automated music genre classification is essential for efficiently organizing vast music collections. It consists of two key steps: feature extraction, in which various attributes are derived from the audio signal, and machine learning-based classification that assigns genres to music tracks using these features. Its applications range from streamlined music archive management to profiling users of audio streaming and media services providers, that are the foundation of modern music recommendation systems.

Deep learning has emerged as a valuable method for categorising music content. A relevant method for constructing such networks consists of sequentially connecting convolutional layers, resulting in a plain network. However, as networks become deeper, residual learning is frequently utilised, since it allows the incorporation of an ample quantity of layers without substantially increasing the number of parameters, boosting the network's ability to learn intricate features [1].

Temporal convolutional layers are a crucial subset of convolutional layers when dealing with data that has a noticeable temporal dependency. These layers are skilled at identifying time-dependent features, which makes them indispensable for numerous applications dealing with time-series data. While their primary use is usually associated with time series forecasting and modeling, it should be noted that in some less frequent instances, researchers have investigated their suitability for classification tasks [2], although such application is not as common within the music domain. The versatility of temporal convolutional layers for modelling temporal patterns denotes their adaptability across various domains, which is beneficial for a wide range of machine learning and deep learning applications.

The temporal nature of music data makes it a suitable candidate for networks that include temporal convolutions. Our strategy consists of merging the abilities of residual learning and temporal convolutions in order to improve the accuracy of classic convolutional networks. This study holds the potential to advance the current state-of-the-art in music genre classification, and it emphasises the significance of tailoring neural network structures to the distinct characteristics of the available data. We propose three strategies to improve music genre classification accuracy: 1) implement a temporal residual neural network (TCNN); 2) discover which types of temporal data (waveform, Mel spectrogram, images) fit better with TCNNs; 3) construct a TCNN with bottleneck residual configuration to enhance the depth of the network.

This paper is organised as follows. In Sect. II are introduced the related work and the latest advance of deep learning for the TCNN design. We then describe the details of our methodologies in Sec. III, followed by the signal and features, learning framework and results (Respectively, Sec. IV, V, VI). Finally, we draw a conclusion and describe potential future work in Sec. VII.

II. Related Work

The classification of musical genres using Residual Networks has indeed been a focal point in various research studies. These investigations, such as the one outlined in [3] and [1], emphasize the advantages of employing Residual Networks in this context. RNN have proven to be essential in the development of exceptionally deep neural architectures while consistently delivering outstanding results in terms of both computational efficiency and classification accuracy when compared to traditional plain networks [4].

In the field of convolutional neural networks for processing sequential data, a crucial focus is on the study of TCNNs. Several studies have contributed to the development and

[†]Both authors contributed equally to this paper
email: `filippo.festa@studenti.unipd.it`
email: `filippo.orlando.1@studenti.unipd.it`

improvement of this design [5], [2]. TCNNs draw inspiration from the accomplishments of Convolutional Neural Networks (CNN), but have been tailored to analyse time series. One of the initial breakthrough studies was conducted in the Oord et al. paper [2], who proposed TCNN as a robust substitute for recurrent networks in a the sequence generation task. It showed how dilated convolutions allow the receptive field to grow longer in a much cheaper way than using LSTM units. Later, Bai et al. [5] carried on this idea and demonstrated that a simple TCNN outperforms LSTMs in sequence modeling, while demonstrating longer effective memory. Other researchers have explored the use of TCNNs in classification domain. van den Oord et al. [2] claimed that although WaveNet was designed as a generative model, it can straightforwardly be adapted to discriminative audio tasks such as speech recognition. However, this aspect was only addressed as a side topic and not explored further. Pelletier et al. focused on the classification of satellite image time-series (SITS) [6], aiming at obtaining up-to-date and accurate land cover maps of the Earth's surfaces. The experimental results showed that TCNNs are more accurate than random forest and recurrent neural networks, which are the current state of the art for SITS classification. Yet, rare work has focused on classification tasks in the domain of MGC, thus this paper aims to exploit the effectiveness of temporal convolution joined with residual learning, in order to untangle the temporal structure of music tracks.

Historically, the bottleneck residual network was introduced as an optimisation technique to enhance the computational efficiency of deep residual networks. Relevant literature on the subject encompasses the results obtained made by He et al. [4]. In this study, the authors suggest the use of "bottleneck" configuration to decrease the dimensionality of feature maps in residual blocks, resulting in lower computational costs without significant performance drawbacks. This concept significantly impacted the design of deep neural architectures, allowing for deeper and more intricate networks with fewer parameters. Given the absence of prior research on the concurrent use of this structure alongside temporal convolutions, this paper seeks to introduce this novel approach.

Furthermore, we employ the mixed pooling technique for aggregating features. Research on mixed pooling includes studies like those of Zhang et al. [1] and Gu et al. [7], which demostrate the effectiveness of this technique. Mixed pooling is notable for its capacity to gather contextual information from various spatial scales by blending different pooling methods, including max pooling and average pooling.

Regarding the data, the FMA dataset has inspired various research on music analysis and machine learning [8]. Many studies have examined the use of this dataset, which is rich in music and is useful for several purposes. For instance, the research conducted in [8] showcased the dataset and emphasised its musical variety and effectiveness for tasks, including genre recognition, music content analysis, and personalised recommendations. Regarding MGC on the FMA dataset, studies by Luo [9], Kostrzewa et al. [10] used plain CNNs that struggled to achieve satisfactory results, better results are instead accomplished using different architectures and data preprocessing techniques. Zhang et al. [11] trained a convolutional recurrent neural network using raw audio to provide real-time classification and obtained an accuracy of 65.23%. Bian et al. [12] conducted a comprehensive study on music audio classification using time overlapping and pitch shifting data augmentation approaches. They employed improved CNNs, including a Densely Connected Convolutional Network, achieving an accuracy of 68.9%. Finally, El Achkar et al. [13] achieved an accuracy of 74.39% employing an upgraded version of a CNN-based music classifier named Bottom-up Broadcast Neural Network.

III. Processing Pipeline

A. *Residual Block*

Training very deep networks can be challenging due to the vanishing gradient problem, where gradients diminish quickly during backpropagation, slowing down the training process. RNNs were introduced to tackle this issue by using shortcut connections, this is the biggest difference respect to plain CNNs (Fig. 1).

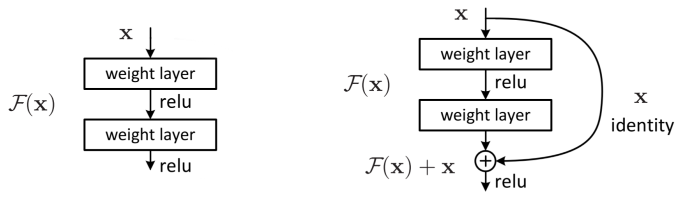

x
weight layer
relu
weight layer
relu
$\mathcal{F}(\mathbf{x})$
x
weight layer
relu
weight layer
$\mathcal{F}(\mathbf{x})$
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Fig. 1: **Left**: the plain block, **Right**: the residual block (Source)

RNNs draw inspiration from He et al.'s concept of residual learning [4]. Let us consider $H(x)$ as the fundamental mapping to be modeled by a series of stacked layers, with x denoting the inputs to these layers. Since multiple nonlinear layers can asymptotically approximate complex functions, they can also model residual functions, represented as $H(x) - x$, provided the input and output share the same dimensions. So we explicitly instruct these layers to model the residual function $F(x) := H(x) - x$ instead of $H(x)$, and thus express the original function as $F(x) + x$. The degradation problem indicates that solvers may struggle to approximate identity mappings using multiple nonlinear layers [4]. The concept of residual learning aims to address this issue and enables the construction of deeper networks while preserving more stable learning.

RNNs feature two main types of building blocks (Fig. 2):

- **"Identity block (IB)"**: is the standard RNN block used to maintain the input and output dimensions. It is characterized by less complexity than its convolutional counterpart in terms of parameters.
- **"Convolutional block (CB)"**: is specifically designed for cases where the input and output dimensions do not align.

It incorporates a convolutional layer within the shortcut path. This convolutional layer serves the crucial purpose of resizing the input to a different dimension, ensuring compatibility for the final addition operation that merges the shortcut value with the main path.

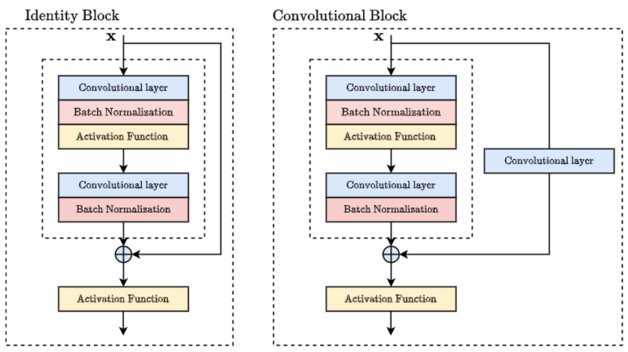


Fig. 2: **Left**: the identity block, **Right**: the convolutional block.

B. *Temporal Convolutional Neural Network (TCNN)*

Temporal Convolutional Neural Network (TCNN) is an architecture which employs causal convolutions and dilations, and it is particularly suitable for sequential data. The characteristics of TCNNs are [5]:

- the capability to process time sequences of varying lengths, ensuring that the network produces an output of the same length as the input time sequence;
- the ability to prevent the information loss from the future to the past, forcing the model not to violate data causality.

To achieve the first point, TCNN employs a 1D fully-convolutional network architecture, ensuring through padding that each hidden layer matches the input's length. To address the second point, causal convolutions are employed (Fig. 3), such that each output at time t depends solely on elements at time t and earlier in the previous layer.

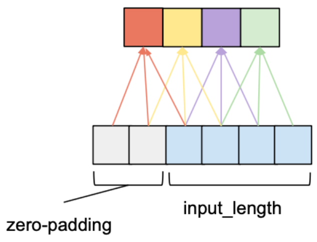


Fig. 3: Casual Convolution with kernel size 3 acting on a 1D input of length 4.

A desirable quality of a model dealing with time series, in order to grasp the correlations and patterns of data, is that the value of a specific entry of the network output depends on the entire network input. The causal convolution is only able to look back at history linearly in the depth of the network. That is, given a fixed kernel size, the number of layers required for full history coverage is linear in the length of the input tensor, which results in very deep networks. This leads to models with a large number of parameters and degradation problems related to the gradient of the loss function [4]. A way to reduce the number of layers lies in utilizing dilated convolutions (See Fig. 4) with an exponentially increasing dilation alongside the network's depth (See Fig. 5): this also enables an exponentially growing receptive field with the number of layers. The combination of dilated and causal layers results in the temporal convolution layer, which is the core of TCNN architectures.

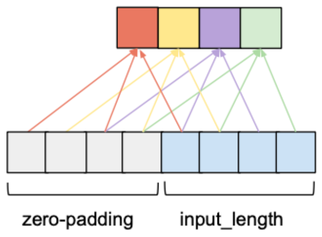


Fig. 4: Causal dilated convolution with kernel size 3 and dilation 2 acting on a 1D input of length 4.

Besides the filter size and dilation factor, the minimum number of layers required for full history coverage is still significant for large inputs. Thus, the stabilization of the learning process of deeper TCNNs is essential. For this purpose, the already mentioned residual learning can be exploited for accelerating convergence and allowing much deeper models to be trained [4] [5]. We therefore decided to modify the "identity block" and "convolutional block" by replacing the classical convolutions with dilated causal layers (See Fig.6).

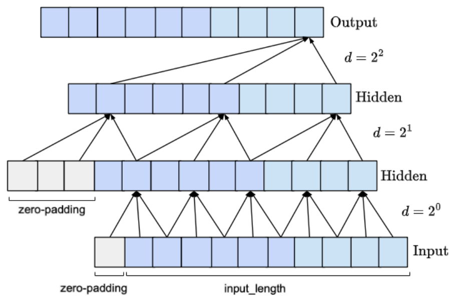
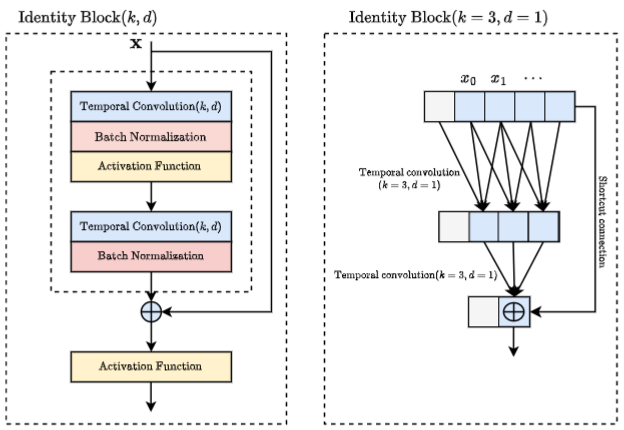


Fig. 5: A dilated causal convolution with dilation factors $d = 1, 2, 4$ and filter size $k = 3$. The receptive field is able to cover all values from the input sequence.

C. *Bottleneck residual block*

He et al. [4] proposed a structural improvement known as the bottleneck block, designed to enhance RNN architectures by addressing the challenge of constructing deeper and more accurate networks with reduced complexity compared to standard RNNs. Thus, mainly due to practical considerations, we



Identity Block(k, d)
$\mathbf{x}$
Temporal Convolution(k, d)
Batch Normalization
Activation Function
Temporal Convolution(k, d)
Batch Normalization
Activation Function
Identity Block$(k = 3, d = 1)$
x_0 x_1 $\cdots$
Temporal convolution $(k = 3, d = 1)$
Temporal convolution$(k = 3, d = 1)$
Shortcut connection

Fig. 6: **Left:** TCNN residual identity block. The convolutional version differs from the identity in the shortcut. **Right:** Example of a TCNN's residual connection applied to an input.

remodel the residual block to a bottleneck design because of concerns on the computational resources and on training time that we can afford. In our design, the bottleneck block deviates from the previous "identity" and "convolutional" blocks by introducing three layers instead of two in the main path. These layers consist of 1×1, 3×3, and 1×1 convolutions (See Fig.7). The main characteristic is the role of the 1×1 layers that reduces and increases dimensions, while the 3×3 layer serves as a bottleneck with smaller input/output dimensions. Again, temporal convolutions have been employed instead of the standard ones.

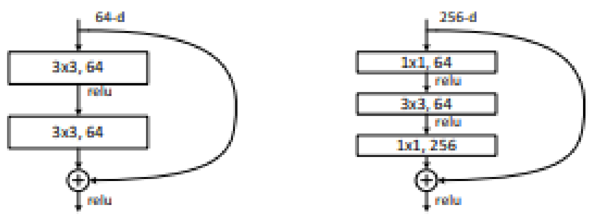


Fig. 7: **Left:** "Classic" residual block. **Right:** Bottleneck residual block.

D. Mixed pooling

The mixed pooling method is the combination of max pooling and average pooling [7], where the underlying idea is to provide more statistical information to the following layers [1]. In fact the experimental results of Yu et al. [14] showed that it can better address the overfitting problem and that it performs better than max pooling, average pooling and some other state-of-the-art methods known in the literature.

Given their nature, the causal dilated convolutions return output vectors that preserve their size in the temporal dimension. The mixed pooling operation therefore also allows us to resize the output of the temporal convolutions and consequently reduce the computation for upper layers. In our case, the max and average pooling combination is implemented as a simple summation.

E. Majority Voting

As reported in Abdoli et al. [15], when an audio input X is splitted into S chunks $X_1, X_2, ..., X_S$ in the temporal coordinate, during the classification we need to aggregate the network predictions to come up to a decision on X, as illustrated in Fig. 8. For this purpose, we employed the majority voting method, described as follows:

$$\text{Choose } \tilde{k}^{th} \text{ class s.t. } \quad \tilde{k} = \arg\max_{k} \sum_{j=1}^{S} o_{jk} \tag{1}$$

where o is the prediction for the $j = 1, ..., S$ segment of the audio waveform X, K is the number of classes and $k = 1, ..., K$ is the predicted class.

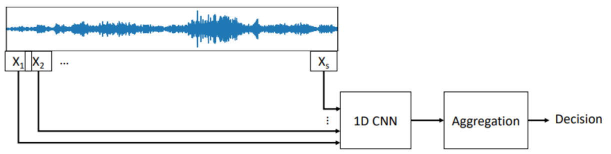


Fig. 8: Scheme for aggregating the predictions of audio frames.

F. Sinc Convolutional Filter

CNNs are the most popular architecture for processing raw waveform. However, Ravanelli et al. [16] showed that one of the most critical part of current waveform-based CNNs is the first convolutional layer. In fact, the kernels learned by CNNs often take noisy and incongruous multi-band shapes, especially when few training samples are available. These filters certainly make some sense for the neural network, but they do not seem meaningful to human intuition. In standard CNNs, convolution is defined by:

$$y[n] = x[n] * h[n] = \sum_{l=0}^{L-1} x[l] \cdot h[n-l] \tag{2}$$

where $x[n]$ is a chunk of the speech signal, $h[n]$ is the filter of length L, and $y[n]$ is the filtered output. All the L elements of each kernel are learned directly during the training process.

In [16], the Sinc filter is introduced as a novel approach to extract interpretable and meaningful kernels. The Sinc filter learned by a CNN has a constrained shape that is the result of the convolution performed between the waveform and a predefined sinc-function g that implements band-pass filters and depends on few learnable parameters θ:

$$y[n] = x[n] * g[n, \theta] \tag{3}$$

In a classical 1D CNN, each filter has L learnable parameters. In contrast, each Sinc filter has only two parameters, $\theta = \{\theta_1, \theta_2\}$, representing the low and high cut-off frequencies

Dataset	Clips	Genres	Lenght [s]	Size [GiB]
small	8,000	8	30	7.4
medium	25,000	16	30	23
large	106,574	161	30	98
full	106,574	161	278	917

TABLE 1: Subsets of the FMA dataset.

of the band-pass filter. These parameters are learned during training, significantly reducing the overall number of parameters compared to the conventional 1D CNN. The consequence is a more efficient optimization process, facilitating faster convergence during the training. Furthermore, the network is forced to focus only on high-level tunable parameters that have a clear physical meaning.

IV. Signals and Features

The Free Music Archive (FMA) [8] is a vast Creative Commons-licensed music dataset with over 106000 tracks, designed to support Music Information Retrieval (MIR) research. It provides high-quality audio data, pre-computed features, and comprehensive metadata for music classification and other MIR applications. All tracks are mp3-encoded, most of them with sampling rate of 44100 Hz, bit rate 320 kbit/s (263 kbit/s on average), and in stereo format. This archive also provides three subsets (small, medium, large) whose features are listed in the table 1.

We chose the balanced FMA small subset: it consists of 1000 clips from each of the eight most popular genres (Rock, Electronic, Experimental, Hip-Hop, Folk, Instrumental, Pop, International) of the medium set. The subset is thus composed of 8000 - 30s clips from 8 genres, balanced with 1000 clips per genre.

Each subset is already divided into $80/10/10\%$ training, validation and test sets to make research on the FMA reproducible.

Firstly, we preprocessed the data by decreasing the sampling rate to 22050 Hz and converting to mono channel.

Consequently, we conducted a duration analysis and identified anomalous tracks of varying lengths, including some that are exceptionally short. To address this issue, we have decided to replace the shorter tracks and standardize the duration to 28 seconds for all entries.

In our analysis, we work with three different primary feature vectors:

- **Waveform:** we segment each clip in 4s chunks with a 50% overlap, yielding 13 chunks per track. This segmentation strategy serves multiple purposes. On one hand, it functions as a data augmentation technique. Simultaneously, it significantly reduces the computational time required for processing individual samples and enhances resource management, preventing potential resource related issues. Regarding our data augmentation process, we undertake several critical steps. Initially, we normalize the data to the $(-1, 1)$ range, ensuring data consistency. Subsequently, we probabilistically introduce a Normal noise for each entry of the waveform and a fade effect.
- **Mel Spectrogram:** The Mel feature extraction process initiates with Short-Time Fourier Transform (STFT) analysis on each 4s chunks, in order to capture the audio's frequency components overtime. Mel filterbanks are then applied to the STFT output to group spectral components into mel-frequency bins. Parameters like the number of filterbanks, fast Fourier trasform window size and hop size influence the granularity of Mel features. We respectively set them to 128, 1024, 512. We then apply the following normalization for each spectrogram:

$$spect_{norm} = \frac{spect - spect_{mean}}{spect_{std}} \tag{4}$$

  Regarding the augmentation, we introduce probabilistic frequency and time masking, randomly obscuring frequency and time frames within Mel spectrograms.
- **Image:** The normalized Mel spectrograms are converted into PIL (Python Imaging Library) RGBA images. Augmentation includes probabilistic affine transformations, such as rotations, random horizontal and vertical flips.

V. Learning Framework

In order to develop the three previously mentioned strategies (see Sec. I) to improve music genre classification accuracy, it was inevitable to implement multiple network structures. Therefore, this section is organized in three different parts, each corresponding to one of these tasks.

In general, the proposed network structures share common elements: LeakyReLU activation functions, categorical cross-entropy loss function, Kaiming uniform weights initialization and Adam optimizer. Moreover, we employ batch normalization technique before each activation to make the model more robust and dropout was used during the training process in order to prevent overfitting problems.

The table 2 shows the input shapes for each type of data.

Type	Shape [Channel, *Size]
Waveform	[1, 88200]
Mel spectrogram	[1, 128, 513]
Image	[4, 128, 513]

TABLE 2: Input data type and shape.

The networks' output consists in a probability vector for each music chunk. This vector contains the probabilities of each chunk to belong to the different genres. Instead, the model prediction for each track is carried out using the majority voting technique as specified in Sec. III.

A. *TCNN vs RNN*

First, we want to construct a TCNN and compare its performance with the RNN design. The learning architecture (See Tab. 3) is inspired by the network proposed by Allamy et al. [3], specifically designed for the waveform as 1D-input data. The only difference between the TCNN and the RNN

Layer	Filters	Kernel Size	Stride
Input	-	-	-
Conv1D	128	3	3
IB1D	128	3	1
MixPool	-	3	3
IB1D	128	3	1
MixPool	-	3	3
CB1D	256	3	1
MixPool	-	3	3
IB1D	256	3	1
MixPool	-	3	3
IB1D	256	3	1
MixPool	-	3	3
IB1D	256	3	1
MixPool	-	3	3
IB1D	256	3	1
MixPool	-	3	3
IB1D	256	3	1
MixPool	-	3	3
CB1D	512	3	1
MixPool	-	3	3
Conv1D	8	1	1
Dropout	-	-	-
Output	-	-	-

TABLE 3: Waveform Network

lies in the use of temporal convolutions instead of classical convolutions. The learning rate value chosen is 0.004, the dropout rate is 0.5, while the weight decay associated with the regularization term is 10^{-6}.

B. Comparing different data types

Secondly, we focused on the behaviour and performance of TCNNs with respect to the different data types chosen.

1) Waveform: we improved the previous network by replacing the first convolutional layer with the Sinc filter with 80 channels, 251 kernel size and stride 3 as reported in Ravanelli et al [16]. The hyperparameters chosen are the same as those listed above (Sec. V-A).

2) Mel Spectrogram and Image: The network designed for both 2D-input data (See Tab. 4) is inspired by resnet34, reported in the article by He et al. [4]. The learning rate value chosen is 0.004, the dropout rate is 0.2, while the weight decay value is 10^{-4}.

C. Bottleneck configuration

Finally, given the performance results, we construct an alternative learning architecture for the Mel spectrogram data type, using the same structural idea but with bottleneck blocks that allow a much deeper model (Tab. 5).

Layer	Filters	Kernel Size	Stride
Input	-	-	-
Conv2D	64	7	2
MixPool	-	3	2
IB2D	64	(3,3)	1
IB2D	64	(3,3)	1
CB2D	128	(3,3)	1
MixPool	-	3	2
Dropout	-	-	-
IB2D	128	(3,3)	1
IB2D	128	(3,3)	1
CB2D	256	(3,3)	1
MixPool	-	3	2
Dropout	-	-	-
Linear	8	-	-
Output	-	-	-

TABLE 4: Mel spectrogram and Image Network.

The learning rate value chosen is 0.0026, the dropout rate is 0.2, while the weight decay is $1.65 \cdot 10^{-4}$. The search for hyperparameters was conducted by means of a Bayesian optimisation algorithm.

VI. Results

Table 6 shows the genre classification accuracies for the three different goals outlined in Sec. I.

Observing the result, the TCNN exhibits a slightly lower accuracy compared to the RNN, but this outcome appears to depend on the data type used with the TCNNs. Specifically, the Mel Spectrogram demonstrates better results with respect to waveform and image data type. Lastly, concerning the network's structure, the marginal accuracy improvement shows how the bottleneck configuration achieves a better classification compared to prior attempts. Overall, the accuracies are not satisfactory.

Recurring patterns in the four metrics considered - precision, recall, F1 score and ROC curve - can be identified in a deeper analysis of the results for each class for the different architectures used. Specifically, it emerges that the "pop" class has the lowest F1 score, with a higher precision value than recall. Although the other classes have minimal misclassifications in the "pop" class, a significant number of "pop" tracks are classified in other genres. The categories "folk", "experimental", and "instrumental" immediately follow, with slightly improved metrics and F1 scores around 0.3. Then we have the "rock" class, which performs better than the previous ones, but it tends to behave inconsistently across different networks. Finally, the best ranked genres are "international",

Layer	Filters	Kernel Size	Stride
Input	-	-	-
Conv2D	64	7	2
MixPool	-	3	2
IB2D	64	(3,3)	1
IB2D	64	(3,3)	1
CB2D	128	(3,3)	1
MixPool	-	3	2
Dropout	-	-	-
IB2D	128	(3,3)	1
IB2D	128	(3,3)	1
CB2D	256	(3,3)	1
MixPool	-	3	2
Dropout	-	-	-
IB2D	256	(3,3)	1
IB2D	256	(3,3)	1
CB2D	512	(3,3)	1
MixPool	-	3	2
Dropout	-	-	-
Linear	8	-	-
Output	-	-	-

TABLE 5: Bottleneck Network.

Strategies	Network	Accuracy
A	RNN	46.12%
	TCNN	42.87%
B	Waveform + Sinc Filter	44.12%
	Mel Spectrogram	46.75%
	Image	36.25%
C	Bottleneck	48.25%

TABLE 6: Accuracy results

"electronic" and lastly "hip hop" with the best value overall: its F1 score lies around 0.7. Figure 9 provides a representative example of the general trend of the different metrics.

The described common trend among the different architectures leads us to hypothesise a potential issue related to the nature of the dataset. The main problem may be the assignment of genres and sub-genres to each track. On one hand, artists make this choice arbitrarily without any limitations on the number of classes and sub-classes assigned to a song. On the other hand, an increase in the number of sub-genres for a given track may correspond to a greater blurring of the boundaries between one genre and another, making accurate classification more difficult.

	F1 Score	Precision	Recall
Pop	0.2	0.29	0.15
Folk	0.27	0.25	0.29
Experimental	0.4	0.4	0.4
Intrumental	0.38	0.33	0.45
Rock	0.61	0.64	0.58
International	0.58	0.69	0.5
Electronic	0.62	0.66	0.58
Hip Hop	0.74	0.62	0.91

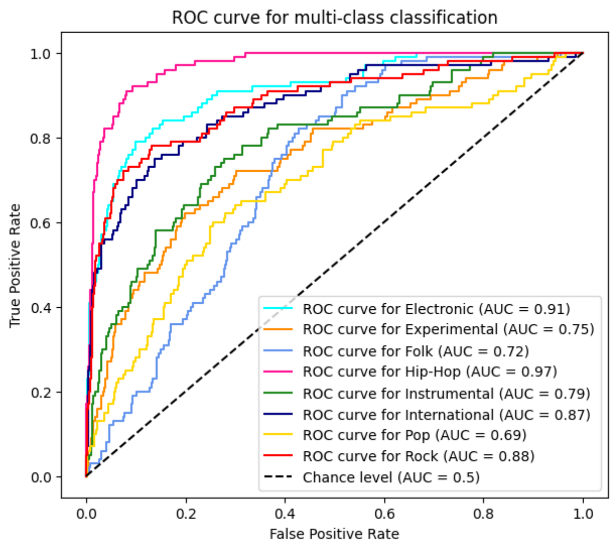


Fig. 9: Classification metrics and ROC curves for the bottleneck configuration.

VII. Concluding Remarks

The purpose of the paper was to test the application of temporal convolutions in conjunction with residual learning for a MGC task. Firstly, we implemented a TCNN and compared it with a RNN. Secondly, we analysed how the TCNNs respond to different data types. Finally, we constructed a bottleneck version of a TCNN.

Although the initial results are not in line with our expectations, further analysis may provide valuable insights and refinements regarding the innovative use of TCNNs composed of bottleneck blocks. This approach could potentially address the complex temporal patterns specific to music. Their versatility in handling various forms of temporal data could enhance not only the classification of music genres, but also improve the TCNNs state-of-the-art in tasks involving sequential data or time series, such as speech recognition, video analysis, sequence modelling and forecasting.

Due to the complexity of this structure and the time required for the training process, it will be necessary, on one hand, to use data processing techniques to handle less heavy data and, on the other, to increase the available resources in order to improve the results obtained and consolidate the emerged trends.

To conclude, we consider necessary to conduct a further analysis on the dataset used, in particular to establish whether each track actually belongs to the respective class. The FMA dataset, although versatile, is still relatively young and constantly being updated. The doubts raised could be addressed by using a more established dataset such as GTZAN.

References

[1] W. Zhang, W. Lei, X. Xu, and X. Xing, "Improved music genre classification with convolutional neural networks," pp. 3304–3308, 09 2016.

[2] A. oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," 09 2016.

[3] S. Allamy and A. L. Koerich, "1d CNN architectures for music genre classification," *CoRR*, vol. abs/2105.07302, 2021.

[4] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015.

[5] S. Bai, J. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," 03 2018.

[6] C. Pelletier, G. Webb, and F. Petitjean, "Temporal convolutional neural network for the classification of satellite image time series," *Remote Sensing*, vol. 11, p. 523, 03 2019.

[7] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, and T. Chen, "Recent advances in convolutional neural networks," *Pattern Recognition*, vol. 77, pp. 354–377, 2018.

[8] 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)*, 2017.

[9] X. Luo, "Automatic music genre classification based on cnn and lstm," *Highlights in Science, Engineering and Technology*, vol. 39, pp. 61–66, 04 2023.

[10] D. Kostrzewa, P. Kaminski, and R. Brzeski, *Music Genre Classification: Looking for the Perfect Network*, pp. 55–67. 06 2021.

[11] C. Zhang and Y. Zhang, "Songnet: Real-time music classification," 2018.

[12] W. Bian, J. Wang, B. Zhuang, J. Yang, S. Wang, and J. Xiao, "Audio-based music classification with densenet and data augmentation," 2019.

[13] C. El Achkar, R. Couturier, T. Atechian, and A. Makhoul, *Combining Reduction and Dense Blocks for Music Genre Classification*, pp. 752–760. 12 2021.

[14] D. Yu, H. Wang, P. Chen, and Z. Wei, "Mixed pooling for convolutional neural networks," pp. 364–375, 10 2014.

[15] S. Abdoli, P. Cardinal, and A. Koerich, "End-to-end environmental sound classification using a 1d convolutional neural network," *Expert Systems with Applications*, vol. 136, 06 2019.

[16] M. Ravanelli and Y. Bengio, "Speaker recognition from raw waveform with sincnet," 08 2018.