SN Computer Science (2023) 4:106
https://doi.org/10.1007/s42979-022-01490-6

ORIGINAL RESEARCH

Genre Recognition from Symbolic Music with CNNs: Performance and Explainability

Edmund Dervakos[1] · **Natalia Kotsani**[1] · **Giorgos Stamou**[1]

Received: 25 July 2021 / Accepted: 3 November 2022 / Published online: 17 December 2022

Abstract
In this work, we study the use of convolutional neural networks for genre recognition in symbolically represented music. Specifically, we explore the effects of changing network depth, width and kernel sizes while keeping the number of trainable parameters and each block's receptive field constant. We propose an architecture for handling MIDI data that makes use of multiple resolutions of the input, called Multiple Sequence Resolution Network (MuSeReNet). These networks accept multiple inputs, each at half the original sequence length, representing information at a lower resolution. Through our experiments, we outperform the state-of-the-art for MIDI genre recognition on the topMAGD and MASD datasets. Finally, we adapt various post hoc explainability methods to the domain of symbolic music and attempt to explain the predictions of our best performing network.

Keywords Artificial intelligence · Machine learning · Music information retrieval · Genre classification · Explainability · Genetic programming · XAI

Introduction

Music is a domain that in recent years has facilitated important research on machine learning and artificial intelligence, including information retrieval and generative models. The two most common forms of musical data are symbolic representations—such as MIDI messages, pianorolls and scores—and audio recordings. Each of the two representations is most suited for different tasks of interest for the artificial intelligence community. For instance, algorithmic composition of music is almost exclusively tackled with symbolic representations of music [2, 5, 23], while music classification tasks, such as genre recognition [26, 29, 51, 52], or mood classification [30, 33, 50] are typically approached in the audio domain.

There are certain aspects of musical data which make it difficult to develop new methodologies for processing and extracting information. One of the main challenges in handling musical data is the existence of information across multiple time scales. In audio recordings, for example, there exists information that ranges from very high frequencies (> 10 kHz) mainly affecting the timbre of the sound to very low frequency information which may pertain to structural elements of a piece of music. This is also true, to a lesser extent, for symbolic representations of music, and is one of the main reasons for which some deep learning-based composition models cannot generate pieces of music longer than a few bars while maintaining musicality, prosody, and structure. On the other hand, symbolic music does not fully capture information that might be important for solving a task such as genre recognition, for instance the different sounds of various instruments or nuanced aspects of human performance. These difficulties, among others, have led to the development of new approaches for processing music in both the audio and the symbolic domain, many of which take the form of specialized neural network architectures such as WaveNets

This article is part of the topical collection "Evolutionary Art and Music" guest edited by Aniko Ekart, Juan Romero and Tiago Martins.

✉ Natalia Kotsani
nkotsani@corelab.ntua.gr

Edmund Dervakos
eddiedervakos@islab.ntua.gr

Giorgos Stamou
gstam@cs.ntua.gr

1 School of Electrical and Computer Engineering, National Technical University of Athens, Athens, Greece

[28]. In this work, one of our main goals was to develop a methodology for classification of symbolic music into genres, based on neural networks, which has not been as extensively explored as audio, but has the potential to be useful in real-world applications, as symbolic representations (such as MIDI files) are more light-weight and are sometimes easier to acquire than audio representations. Furthermore, recent hybrid approaches that utilize both audio and symbolic representations have shown promising results for various music-related tasks [19, 48] which further motivates studying symbolic representations.

Specifically, convolutional neural networks (CNNs) have seen widespread use in multiple domains, spearheaded by their success in computer vision. However, finding an optimal architecture and set of hyper-parameters, such as number of layers, kernel size, number of kernels, etc. for a given task remains a difficult problem that is still being tackled mostly experimentally. For instance, Tan and Le in [45] determined that scaling up network depth should be accompanied by scaling up network width and resolution. This approach however, on the one hand, requires a suitable initial baseline network to be scaled up, and on the other hand, it is a method proposed for computer vision tasks and it is unclear if the conclusions transfer to symbolic music. In addition, multiple network architecture search (NAS) algorithms have been proposed in recent years, especially for computer vision, such as Amoebanets [32], but these are beyond the scope of this paper. In this work, we focus our experiments on comparing CNNs that have the same receptive field and number of trainable parameters, but differ in depth, width and kernel size to explore their effectiveness for genre recognition in symbolically represented music.

Symbolic music is typically represented as sequences for deep learning approaches, but there exist multiple ways to represent music in a tree structure to capture information across multiple time scales. For instance, hierarchical models for music analysis proposed by Schenker [41] converted to trees by Marsden in [24] and of Lerdahl and Jackendoff [20] which partially formalize Schenker's ideas, parsing musical structures into strict trees. Rizo et al. propose a non-linear representation of a melody based on trees and they study the influence of different tree representations on classification rates in three corpora with monophonic melodies, concluding that tree coding gives better results [35]. Rizo and Marsden extend the Music Encoding Initiative (MEI) [37] representation with "semantic" and "non-semantic" encodings which allows the tight association of the analytical information and the information in the score [36]. This motivates us to explore ways in which such trees may be given as input to a neural network instead of sequences. In this work, we present a fully convolutional approach called MuSeReNet, in which each level of the input tree represents the original sequence at a different resolution.

The rest of the paper is structured as follows: Section "Related Work" presents relevant literature for genre recognition, in Section "Preliminaries" we expose the reader to important concepts for 1D Convolutional Neural Networks, and for music genres, in Section "Architecture" we show the proposed neural architecture, in Sect. "Experiments" we describe the experimental setting and results, in Section "Explanations" we further analyze our results using methods for explaining black-box models from the area of explainable AI (XAI) and in Section "Conclusions and Future Work" we conclude the paper.

Related Work

Most works related to genre classification involve music in audio (raw) format. The most up-to-date techniques for the classification of music use neural networks on MFCCs or spectrograms [29, 47], some of them focus on feature selection [39, 44], or introducing new architectures [22, 26, 51, 52].

The abundant availability of MIDI files online, from multiple sources, has given rise to the challenge of automatically organizing such large collections of MIDI files. One criterion for organization is music genre, among others such as music style, similarity, and emotion. In [25] McKay and Fujinaga argue in favor of genre classification, despite inherent difficulties such as ground-truth reliability.

There are many different approaches in the literature for genre recognition in the symbolic music domain. Dannenberg et al. [4] used a machine learning approach, including a simple neural network, on a custom dataset for successful genre recognition in the symbolic domain. In [12], Karydis et al. combine pattern recognition with statistical approaches to successfully achieve genre recognition for five sub-genres of classical music. Kotsifakos et al. in [16] compute a sequence similarity between all pairs of channels of two MIDI files and then use a k-NN classifier for genre recognition, on a dataset of 100 songs and four genres. Zheng et al. [53] extract features related to the melody and the bass through a musicological perspective, incorporating text classification techniques and using Multinomial Naive Bayes as the principal probabilistic classifier, in a self-collected dataset of 273 records.

These approaches were experimentally validated on relatively small datasets compared to, for example, the openly available Lakh MIDI dataset [31]. For large scale datasets, Ferraro and Lemström [7] utilize pattern recognition algorithms SIA [27] and P-2 [46] in addition to a logistic regression classifier to solve the task. The benefit of this approach is interpretability since the authors have created a large corpus of genre-specific patterns, which could also be utilized for other music-related tasks. Duggirala and Moh

[6] apply Hierarchical Attention Networks in music genre classification, after converting the audio files into a word embedding representation. Liang et al. [21] propose four word embedding models consisting of three vocabularies (chroma, velocity, and note state) and apply these models in three MIR tasks: melody completion, accompaniment suggestion, and genre classification, concluding the robustness and effectiveness of their embeddings.

Recently in multiple domains, there is a tendency to forgo feature extraction stages of an information retrieval pipeline, instead using a more complex neural network architecture, in which the first layers act as feature extractors. In computer vision for example, this end-to-end approach has surpassed most previous feature-based works in performance, which motivates us to implement such a system for MIDI genre recognition.

Preliminaries

Genre recognition from symbolic music can be described as a multi-label sequence classification task, in which a set of labels $Y = \{y_1 \dots y_m\}$ is assigned to a sequence of vectors $X = (x_1, x_2 \dots x_t)$. Each label y_i represents a music genre, and each vector x_i represents the notes that sound at a specific timestep. The sequence of vectors X thus represents a polyphonic piece of music in the form of a pianoroll or a MIDI file.

Given a large enough dataset of n such sequences, along with their ground-truth labels $D = \{(X^1, Y^1), \dots, (X^n, Y^n)\}$, in the context of machine learning, the goal is to train an algorithm $F(X;\theta)$ to model the conditional distribution of labels with respect to input sequences.

$$F(X;\theta)_m = P(y_m|X), \tag{1}$$

where m is the index of a label.

This is approached as an optimization problem of finding parameters $\hat{\theta}$ which minimize the cross-entropy between the modeled distribution and the distribution in the train set for each label separately. Each classifier's performance is then measured on a test dataset which does not overlap with D.

1D Convnets

The state-of-the-art approach for MIDI genre classification presented by Ferraro and Lemström in [7] uses an algorithm for recognizing patterns of notes in an input sequence and then performs classification based on recognized patterns. These patterns are local, with the best results achieved from extracting four or five note patterns. This, along with the success of 1D CNNs for other tasks in the symbolic music domain [5] and their suitability for sequence pattern recognition, as has been shown in multiple domains, such as pattern recognition in DNA sequences by Lanchantin et al. in [18], motivates us to explore 1D CNNs for the task of recognizing genres in symbolic music.

A one-dimensional convolution with kernels of size k is an operation, which acts on a sequence of T vectors X of size N to produce a sequence of vectors Y, where at timestep i and channel j:

$$Y_{i,j} = \sigma\left(\sum_{n=1}^{N}\sum_{m=1}^{k} W_{j,n,m} X_{i+m-1,n}\right). \tag{2}$$

The matrix W consists of the convolution's trainable parameters, while the function σ enforces non-linearity. Note that there is also a bias term that has been omitted. Each element of the output sequence Y only depends on k consecutive elements of the input sequence X, leading to the effectiveness of the convolutional operation for capturing local structures and patterns in the data. A deep neural network may then be constructed by stacking such operations in a depth-wise fashion. This results in the first convolutional operations capturing low-level features in the data, while deeper operations capture more complex high-level features.

Receptive Field and Trainable Parameters

An important attribute of such a network is its receptive field, which is defined as the number of elements in the input sequence that affect a single element of the output sequence. A single convolution C_1 with kernels of size k_1 has a receptive field of k_1. A second convolution with kernels of size k_2 which is fed C_1's output will depend on k_2 consecutive elements of said output, leading to its dependence on $k_2 + k_1 - 1$ elements of the original input sequence. Another important attribute of a CNN to keep track of is the number of trainable parameters, equivalent to the size of all weight matrices W. The number of trainable parameters is the primary factor for the memory requirements of a network which is often a bottleneck for network design.

Efficiently increasing a network's receptive field is crucial for effectively capturing features across multiple time scales. By stacking convolutional layers, receptive field increases linearly with network depth and with kernel size k. However, increasing depth could give rise to difficulties during training such as the exploding and vanishing gradients problem (EVGP) [11, 49] among others in addition to increasing the number of trainable parameters, while increasing k by dk leads to a $f_{in} \times f_{out} \times dk$ increase in trainable parameters, where f_{in} and f_{out} are the numbers of input features and output features (number of kernels in the layer). There exist many methods for increasing a network's receptive field more efficiently, for instance using dilated convolutions such as in [28], using strided convolutions, or the most common approach: pooling layers.

Pooling

A pooling layer of stride S and kernel size K acts on a sequence of vectors X of dimensions $T \times N$ and outputs a sequence Y of dimensions $(\frac{T-K}{S} + 1) \times N$. The stride parameter S determines how many input samples are skipped in between applications of the pooling kernel. A common pooling operation is max pooling with a stride equal to kernel size $K = S$. In this case for the output sequence Y:

$$Y_{i,j} = \max_{m<K} X_{(i*K+m),j}. \tag{3}$$

Another common pooling operation is average pooling with $K = S$, for which case:

$$Y_{i,j} = \mathbb{E}_{m<K}(X_{(i*K+m),j}). \tag{4}$$

A pooling operation has a receptive field K and does not have any trainable parameters. Each of two consecutive samples in the pooling operation's output depends on K input samples; however, these samples are spaced apart by on average S samples in the input sequence. This means that feeding the pooling operation's output to another layer effectively increases receptive field by a factor of S without an increase in trainable parameters. Finally, pooling operations introduce invariance to local translations of an input sequence which could be useful for the learning process but they entail information loss which could be detrimental to learning.

Architecture

Trees are an effective representation of music since they are able to capture information, patterns, and structures at multiple different time scales. There are, however, many different ways of constructing such trees from a piece of music, in addition to different approaches for feeding tree representations to neural networks.

In this work, we set a baseline for tree representation utilization for information retrieval from symbolic music, using full binary trees as input structures and by then treating each level of the tree as a separate input. Each node of the binary tree has as a value the average of its children, thus each level of the tree is equivalent to the original sequence—which is represented by the leaves of the tree—at a lower resolution. This means that MuSeReNets presented in this paper are similar to multiple resolution CNNs which have been successful for some computer vision tasks such as skin lesion recognition by Kawahara and Hamarneh in [13].

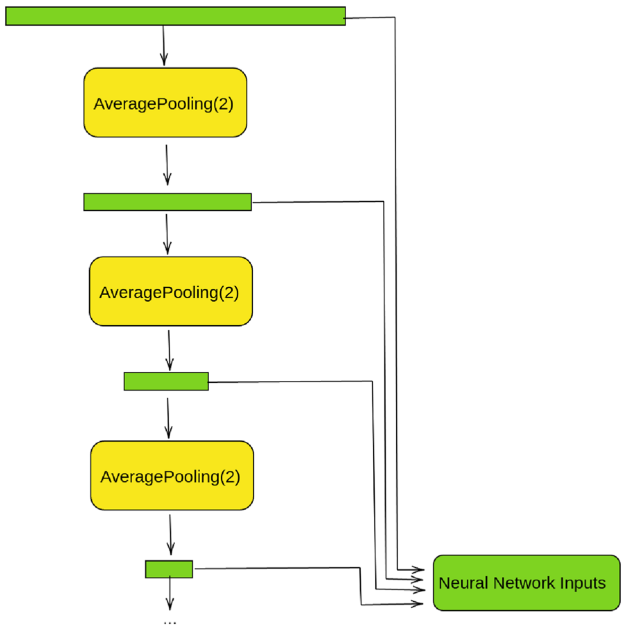

Fig. 1 Constructing a binary tree where levels are equivalent to the input sequence at lower resolutions

Intuitively, the first levels of a CNN detect low-level local features in the data, and more complex higher-level features are captured in deeper layers. However, there could exist high-level features which may be simply extracted from a lower-resolution representation of the input without requiring increasing the depth of the network. For instance, in the case of symbolic music, a simple feature that could be extracted from higher levels of a tree (closer to the root) would be the key signature of a large segment of the input—which relates to the set of notes which appear in the segment. Such features may then be combined with lower-level features extracted from levels closer to the leaves of the tree and fed to deeper layers of the network for further feature extraction and eventually solving a task, which in our case is genre classification.

The first module of a MuSeReNet is a set of average pooling operations which act on the original sequence, each producing a version of the original sequence at a different resolution (Fig. 1). Each of these is treated as a separate input for the neural network.

There are many different ways to make use of these inputs. For MuSeReNets we distinguish between two cases: When information flows from the leaves to the root (Fig. 2) and when information flows from the root to the leaves (Fig. 3).

In the first case, in which information flows from the leaves to the root (Fig. 2), each input is fed through a block which consists of convolutional layers followed by a max-pooling operation with the same stride and kernel size as

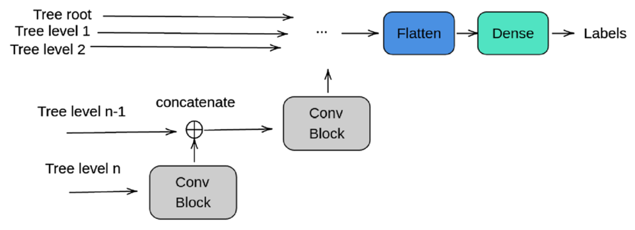

Fig. 2 A MuSeReNet where the information flows from the leaves to the root

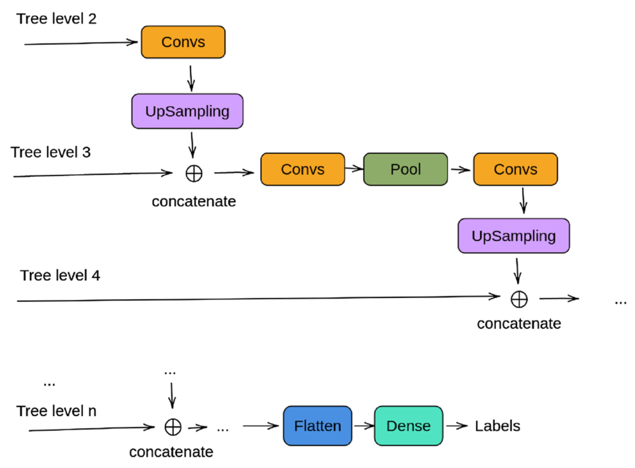

Fig. 3 A MuSeReNet where information flows from the root to the leaves

the average pooling operation which generated the specific input from its higher resolution counterpart. This way, and using 'same' padding for convolutional operations, the output of a specific block has the same sequence length as the original input at the previous resolution level and may be concatenated along their second axis, producing a sequence of the same length and with more channels. The result of concatenation is the original sequence at a lower resolution augmented with features extracted from the convolutional block which processed the input at a higher resolution. This process is repeated until we reach the root of the tree, where the sequence length is 1, and the vector consisting of the root and features extracted from the previous convolutional block is fed to a fully connected layer with the goal of solving a specific task.

In the second case, in which information flows from the root to the leaves (Fig. 3), max-pooling operations are replaced with upsampling operations, and the order with which inputs are fed to the network is reversed (the root first instead of the leaves first). In this case, the result is a sequence of length equal to the original sequence, but is augmented with features that were extracted by convolutions on lower-resolution versions of the sequence. Intuitively, via the upsampling and concatenation operations, this network could learn features that are local but are affected by the context provided by the lower-resolution version at a previous layer. Such an architecture is similar to U-nets [38] which is used for image segmentation. This resulting augmented sequence may then be fed to further neural network layers to solve a specific task.

Experiments

To explore the effectiveness of our architecture for information retrieval from symbolic music and to check the compatibility of 1D CNNs for the task, along with the effect of allocating resources to network depth or to kernel size we conducted a set of experiments.[1]

Data

For our experiments, we use the Lakh Pianoroll Dataset as presented by Dong et al. in [5], specifically the LMD-matched subset. This dataset consists of pianoroll representations of MIDI files in the Lakh MIDI Dataset presented by Raffel in [31]. The pianoroll is an array representation of music in which columns represent time at a sample rate of n samples per quarter note and rows represent pitch in the form of MIDI note numbers. The LMD-matched subset contains pianorolls that have been linked with the Million Song Dataset (MSD) [1]. The Million Song Dataset is the largest currently available collection of audio features and metadata for a million contemporary popular music tracks. We use labels acquired by MSD to construct the MASD and topMAGD datasets presented by Schindler et al. in [42], so we can compare our results with existing work. At the time of writing, Ferraro and Lemström in [7] have achieved the best results with regards to genre classification of symbolic music for the MASD and topMAGD datasets. Finally, we randomly split each dataset into a train and test set (.75/.25), we use the train set for training our models and the test set for evaluating them.

Both datasets are imbalanced with regard to the number of files corresponding to each label (Figs. 4 and 5). Methods such as over-sampling rare classes and under-sampling common classes could be used to potentially improve the generalization ability of trained models, but it is left for future work since we are interested in observing the behavior of different network configurations for this task.

[1] The code is available in the following GitHub repository: https://github.com/kinezodin/cnn-midi-genre.

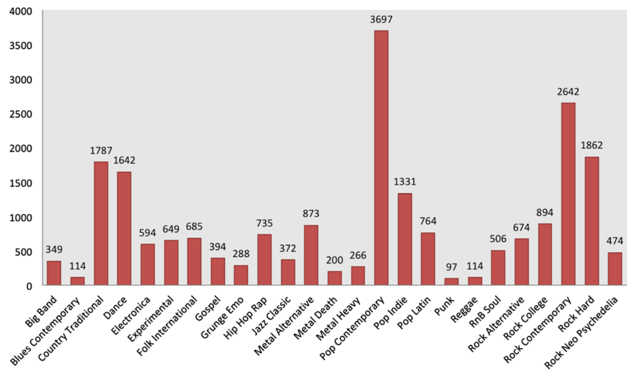

Fig. 4 Number of files in the LPD dataset per label of the MASD dataset

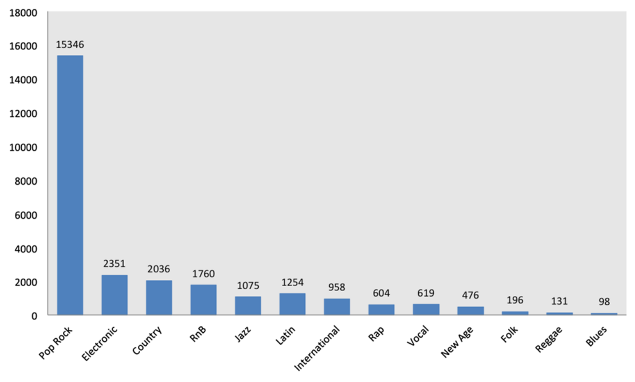

Fig. 5 Number of files in the LPD dataset per label of the topMAGD dataset

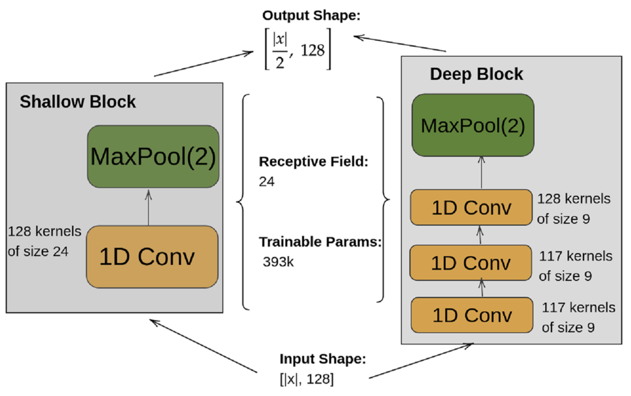

Fig. 6 The convolutional blocks used to construct the CNNs for use in our experiments

Models

We construct neural networks using blocks of 1D convolutions followed by max-pooling operations of kernel size and stride 2. Specifically, we use a shallow block, which consists of only one convolutional layer prior to the pooling operation, and a deep block which consists of three convolutional layers before each pooling operation (Fig. 6). A network built with shallow blocks will be referred to with the prefix 'shallow', and those built with deep blocks 'deep'.

In addition, each network has a 'Sequence' version in which blocks are stacked depth-wise and the first block receives as its input the original sequence, and a 'MuSeRe' (Multiple Sequence Resolution) version in which a level of a tree constructed from the input sequence is concatenated to the output of each block. The first case represents a traditional CNN architecture, while the second represents MuSeReNets in which information flows from the leaves to the root (Fig. 2).

Shallow vs Deep

All blocks are individually set to have a similar receptive field of 24 samples, thus stacking the same number of blocks will lead to CNNs with the same receptive field regardless of which type of block is used. In the shallow block case, this implies a kernel size of 24. For the deep block case, assuming all convolutions have the same kernel size k, if $k = 9$ the receptive field at the output of the third layer is 25 input samples. We arbitrarily chose the smallest kernel size $k = 9$ which has the same number of trainable parameters as a 3×3 kernel which is popular for computer vision two-dimensional CNNs.

All blocks are also set to have a similar number of trainable parameters. Given fixed input dimensions of $(|x|, f_{in})$, a single convolutional layer with f_{out} kernels of size k will have n_p trainable parameters:

$$n_p = (f_{in} * k + 1) * f_{out}.$$

We set $f_{out} = f_{in} = 128$ for all blocks, thus the number of trainable parameters for a shallow block is:

$$n_{shallow} = 393,344.$$

For the deep block, we set the number of kernels of the third layer $f_{out_3} = 128$, so that the output of each block is the same shape as with the shallow block. To satisfy the condition of having an equal number of trainable parameters to the shallow case

$$\begin{aligned} n_{deep} &= n_{shallow}, \\ 1153 f_{o_1} + 9 f_{o_1} f_{o_2} + f_{o_2} + 1153 f_{o_2} + 128 &= n_{shallow}, \end{aligned}$$

where f_{o_1} and f_{o_2} are the number of kernels for the first and second layers of the deep block. By arbitrarily setting $f_{o_1} = f_{o_2}$, we get 117 kernels per layer. This way we end up with the two blocks shown in Fig. 6.

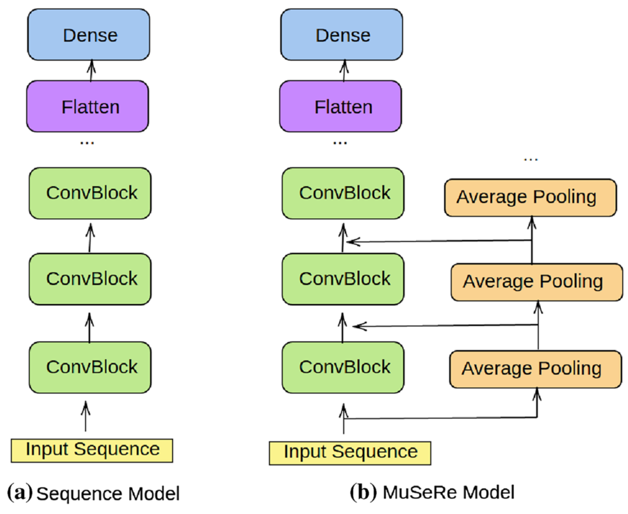

Fig. 7 **a** Sequence architecture and **b** MuSeRe architecture used in experiments

For the different models, we use powers of 2 as input sequence lengths l, ranging from $l = 64$ to $l = 2048$. In the context of our dataset, these lengths represent musical time from approximately 5 quarter notes to 170 quarter notes, or 42 bars for a $\frac{4}{4}$ time signature (around one–two minutes for typical values of a song's tempo). Then, each network will consist of $\log_2 l$ blocks stacked depth-wise, followed by a fully connected layer at the output, with as many sigmoid-activated units as there are different labels in each dataset. For all convolutional layers, we used ReLu activations.

Sequence vs MuSeRe

The two versions of each network (Fig. 7) differ with regard to the inputs of each block which are sequences of 128-dimensional vectors in the 'Sequence' case and 256-dimensional vectors in the 'MuSeRe' case, which are a result concatenation of a previous block's output with the original sequence at a lower resolution and leads to an increase of trainable parameters for the first layer of each block. The 'Sequence' networks are a typical 1D CNN architecture, which, however, to our knowledge, have not been used in an end-to-end approach for symbolic music inputs and genre recognition.

Data Preparation and Training

Every pianoroll is a fixed length sequence of vectors $\mathbf{x} = [\mathbf{x_1}, \mathbf{x_2}, ...\mathbf{x_t}]$, where each vector x_i has 128 dimensions representing MIDI note numbers. During training, before feeding a sequence to a network, we perform a random transposition by shifting elements of every vector of a sequence by a random integer in $[-6, 6]$. This corresponds to transpositions up to a tritone below or above and is done as a data augmentation step which helps to avoid bias with respect to a particular tonal center.

The way multi-track MIDI files are handled is by averaging the pianorolls of all instrumental tracks and all percussive tracks separately into two cumulative pianorolls. The downside is that we lose information pertaining to the different instruments and to some extent different voices become convoluted, but this way we are able to process a larger number of MIDI files regardless of the number of tracks.

We use a data generator to create batches for training and apply any data transformations on the fly. For finding trainable parameters of networks which minimize the cross-entropy between predicted genres and real genres we use Adam [15] as the optimization algorithm during training with a learning rate of $\alpha = 10^{-5}$ and for the other hyper-parameters of the optimizer $\beta_1 = 0.9$, $\beta_2 = 0.999$ and a batch size of 32. Before training a model, we further split the original train set into a validation set and a train set (0.2/0.8) randomly. We then trained our models for up to 300 epochs while using an early stopping criterion for each model's F1-score on the validation set.

Evaluation and Post Processing

We evaluate our models on the held-out test set for each of the MASD and topMAGD datasets by computing precision, recall, and micro f1 metric. Due to the varying sequence lengths of pianorolls in the test set, we use the post-processing procedure described below to aggregate a model's prediction across whole sequences, which are of greater length than the expected neural network inputs.

Given a pianoroll x_p of sequence length N and a model with input length $N_m < N$, we retrieve sequences of length N_m from x_p by using a sliding window of N_m samples and a hop size of $\frac{N_m}{2}$ samples on the sequence. Each window is then fed to the model for inference, which returns a vector where each element represents a probability that a specific label is assigned to x_p. Then, we assign as predicted labels those with a probability value greater than 0.5. If no labels have a probability greater than 0.5, then we assign as a single label the element of the vector which has the maximum probability, since there are no unlabeled samples in the dataset. These predictions are then used to calculate false and true positives and negatives, recall, precision, and F1 score.

Results

The results of our experiments are shown in Table 1 which lists the micro F1 scores of each trained model on the test set. In general, all CNNs which were trained on sequences longer than 256 samples surpassed Ferraro and Lemström's pattern recognition approach, as well as Liang et al. model

Table 1 Micro F1 scores on the test sets of the MASD and topMAGD dataset for each of our architectures P2–4 and P2–5 refer to the best performing configuration of those presented in [7] and PiRhDy_GM refer to the best performing configuration of those presented in [21]

Length	Block	Input	MASD	topMAGD
64	Deep	Sequence	0.258	0.620
		MuSeRe	0.265	0.622
	Shallow	Sequence	0.295	**0.623**
		MuSeRe	**0.308**	0.622
128	Deep	Sequence	0.315	0.624
		MuSeRe	0.317	0.631
	Shallow	Sequence	0.361	0.632
		MuSeRe	**0.407**	**0.639**
256	Deep	Sequence	0.411	0.654
		MuSeRe	0.404	0.639
	Shallow	Sequence	0.335	0.663
		MuSeRe	**0.491**	**0.668**
512	Deep	Sequence	0.456	0.661
		MuSeRe	0.374	0.653
	Shallow	Sequence	**0.545**	**0.711**
		MuSeRe	0.525	0.703
1024	Deep	Sequence	0.507	0.673
		MuSeRe	0.337	0.641
	Shallow	Sequence	**0.581**	**0.777**
		MuSeRe	0.526	0.737
2048	Deep	Sequence	0.456	0.696
		MuSeRe	0.264	0.627
	Shallow	Sequence	**0.593**	**0.759**
		MuSeRe	0.444	0.733
P2–4			0.468	0.662
P2–5			0.431	0.649
PiRhDy_GM			0.471	0.668

Bold represents the best performance for each sequence length and for each dataset

with regards to F1 metric [7, 21]. Increasing input length by a factor of two along with increasing the number of blocks by one in most cases improved performance, with a notable exception for the longest sequence lengths that we experimented on (1024 vs 2048). In general, MuSeRe models outperform sequence models for shorter input lengths. The poor performance of MuSeRe models for larger inputs could be a result of overfitting, but requires further experimentation.

In addition, we present precision, recall, and F1 scores for each label in the topMAGD dataset for the best performing model (Table 2). On the one hand, the effect of the imbalanced dataset is apparent in the network's performance for the most common label (Pop-Rock) when compared to those with fewer files in the dataset such as Blues, Reggae, and Folk. It is interesting that genres such as Jazz which have little representation in the dataset are better classified than genres such as Electronic which has almost double the support.

Table 2 Per label precision recall and F1-score on the test set for Shallow Sequence model with input length 1024 (best performing model) on the topMAGD dataset

Label	F1	Precision	Recall	Support
Pop-Rock	0.86	0.81	0.96	3705
Electronic	0.58	0.74	0.47	557
Country	0.67	0.83	0.56	502
RnB	0.61	0.92	0.45	432
Jazz	0.76	0.91	0.65	281
Latin	0.45	0.78	0.32	338
International	0.53	0.77	0.41	236
Rap	0.34	0.78	0.22	133
Vocal	0.65	0.90	0.51	150
New Age	0.66	0.94	0.51	116
Folk	0.48	1.00	0.32	44
Reggae	0.48	1.00	0.31	38
Blues	0.55	0.73	0.44	18
Micro avg	0.78	0.81	0.74	6550

This could be due to distinguishing musical characteristics of each genre, which are apparent in symbolic representations of music—for instance, jazz music tends to have complex harmony and utilize more notes, while electronic music tends to contain loops of very few notes.

Explanations

Even though we have shown that deep convolutional neural networks can perform better than other methods for genre recognition from symbolic music, they suffer from drawbacks that most deep learning approaches do, importantly lack of explainability and interpretability. This is not an imperative issue for genre recognition, since explainability is not critical for the method to be utilized; however, it would be a useful feature that could help improve performance in a future iteration by helping detect potential biases or flaws of the genre recognition model. Furthermore, since genres themselves are not well-defined terms, and their characteristics can vastly change over time, explanations of predictions could be valuable for understanding both the model and the dataset. In this section, we generate explanations by adapting various explainability frameworks and tools from the area of explainable AI (XAI) to the domain of symbolic music. As these are post hoc explanation methods, which treat the model as a black box, we chose to present a high-level description of multiple different methodologies, since such methods have been shown to occasionally produce misleading results [40]. We analyze and qualitatively compare the usefulness of different explanation methods when applied to symbolic music.

Table 3 Predictions for the top-4 genres for the first 1024 time-steps of each of the four songs for which local explanations were generated. The tracks are: Beethoven—Moonlight Sonata, The Beatles—Here Comes the Sun, Eminem—The Real Slim Shady, Queen—Bohemian Rhapsody

Beethoven		Beatles		Eminem		Queen	
International	0.69	Pop–Rock	0.83	Rap	0.89	Electronic	0.54
New Age	0.40	Jazz	0.06	Electronic	0.03	Pop–Rock	0.20
Rap	0.25	RnB	0.04	Jazz	0.02	Vocal	0.06
Pop–Rock	0.24	Country	0.037	Vocal	0.003	RnB	0.04

Local Explanation Methods

Methods for explaining the prediction of a black-box model on a specific sample are called local explanation methods [10]. There are many such methods in the relevant literature, for various types of data; however, none have been designed specifically for symbolic representations of music. We show results generated by Grad-CAM [43], LIME [34] and the genetic programming based GPX [8]. Grad-CAM generates visual explanations and is intended for images, LIME works on any type of data, while GPX is more suited for tabular data with a relatively small number of features. Here, we show visual explanations for Grad-CAM and LIME, where the image depicts the cumulative pianoroll of each piece of music, while for GPX we modify the explanation pipeline for it to be utilized on symbolic representations of music.

The results concern four hand-picked samples from the reddit MIDI dataset.[2] These are (a) Beethoven's *Moonlight Sonata*, (b) The Beatles - *Here Comes the Sun*, (c) Eminem—*The Real Slim Shady* and (d) Queen—*Bohemian Rhapsody*. We fed the first 1024 time-steps of each sample through the best performing (on the topMAGD dataset) CNN. The predictions are shown in Table 3. We chose Moonlight Sonata, since it is out of domain as its genre is not included in the dataset's labels. The top two predictions of *International* and *New Age* music for the specific sample are interesting and merit an explanation. We chose (b) and (c) as examples of certain predictions, while (d) shows an erroneous prediction by the CNN (Electronic), and a relatively high value of 0.06 for the vocal genre which is interesting since the introduction of the song features an *a capella* chorus.

Grad-CAM

Grad-CAM is a method proposed for generating visual explanations for large 2D CNNs which are applied in the image domain. It works by computing the gradient of the target output neuron with respect to the activation-map of the final convolutional layer of the CNN, averaging over the channels, thus leading to a coarse heatmap of the same size as the convolutional feature-maps, which in our case is always a sequence of length 4, due to the application of max-pooling operations within our networks. The resulting explanations for the top-2 predicted classes, for each of the four samples are shown in Fig. 8.

Due to the coarseness of the heatmap, these explanations are not very useful in their own right, and are used in this context as a baseline. For *Moonlight Sonata* the explanations show uniform contribution of each timestep towards the International genre and no contribution towards the New Age genre. For *Here comes the Sun* the first quarter of the pianoroll seems to contribute more towards a *Jazz* prediction, while the third quarter contributes towards a *Pop-Rock* prediction. For *The Real Slim Shady* the explanations are similar to those of *Moonlight Sonata* and are not very useful. Finally, for *Bohemian Rhapsody*, the second half of the pianoroll contributes more towards the *Electronic* genre, after the piano part is introduced.

LIME

LIME is a technique for generating explanations for any classifier, in any data domain. In order to explain a classifier $f : \mathbb{R}^d \to \mathbb{R}$, LIME searches for a function $\xi : \mathbb{R}^d \to \mathbb{R}$ which is a) Interepretable and b) Approximates f locally. It is formulated as:

$$\xi(x) = \arg\min_{g\in G} \mathcal{L}(f, g, \pi_x) + \Omega(g),$$

where G is a family of explainable functions (such as linear models or decision trees), π_x is a proximity measure that measures locality around x, Ω is a measure of how interpretable a function g is (for instance the number of weights in a linear model, or the depth of a decision tree) and $\mathcal{L}$ is a distance function showing how closely g approximates f in the locality defined by π_x. In Fig. 9 we show explanations generated by LIME using the official python package[3] provided by the authors, and specifically the lime_image object, with default parameters. For *Moonlight Sonata*

[2] https://www.reddit.com/r/WeAreTheMusicMakers/comments/3ajwe4/the_largest_midi_collection_on_the_internet/.

[3] https://github.com/marcotcr/lime.

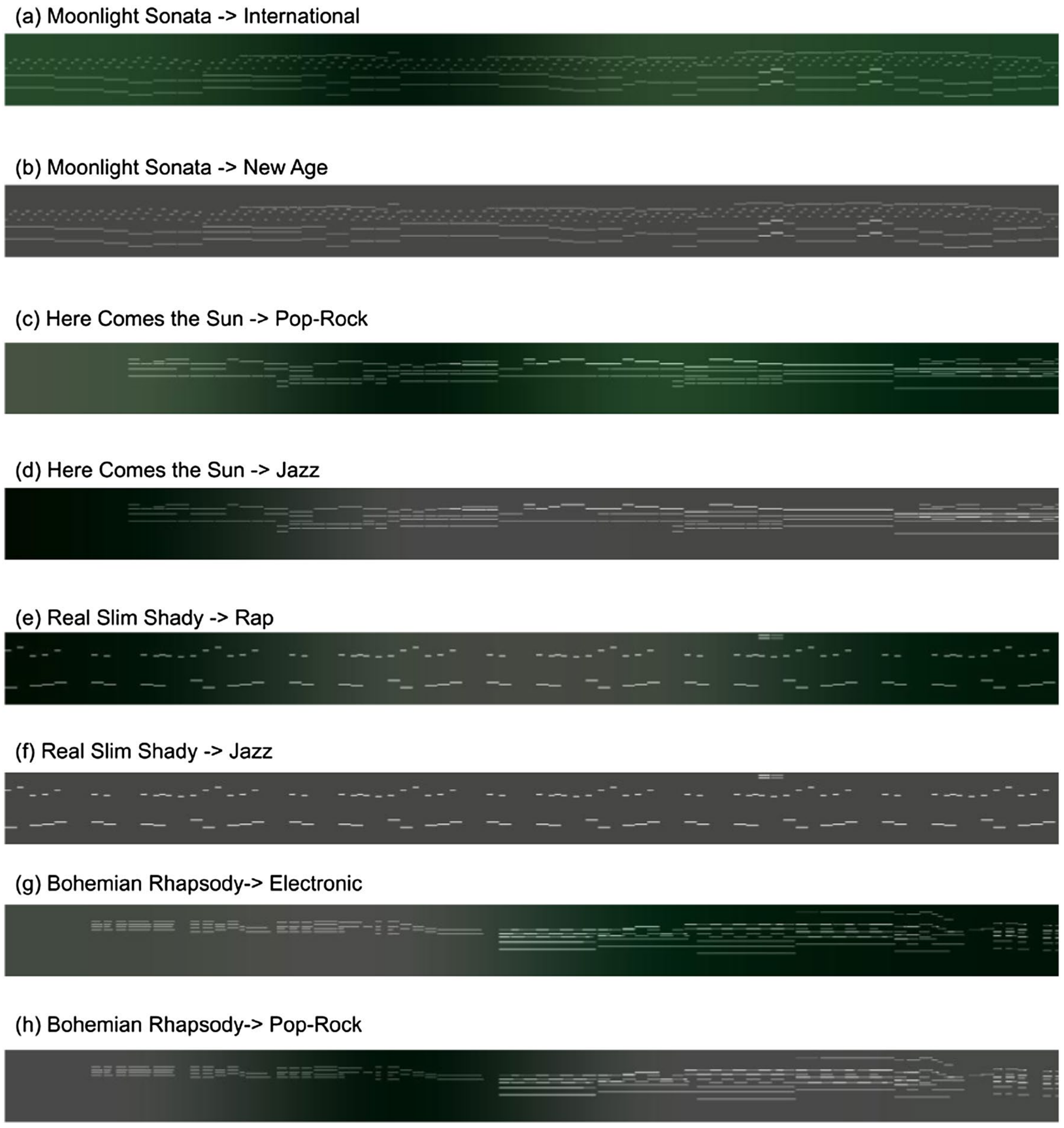

Fig. 8 Explanations generated by Grad-CAM for the top-2 predicted genres, for the best performing CNN, on the first 1024 time-steps of Beethoven's Moonlight Sonata (**a**, **b**), The Beatles—Here Comes the Sun (**c**, **d**), Eminem—The Real Slim Shady (**e**, **f**), and Queen—Bohemian Rhapsody (**g**, **h**). The highlighted area with green represents the important for the prediction segments—according to Grad-CAM

LIME has highlighted the fifth measure without the melody note, the melody of the seventh and eighth measures and the bass notes of measures 13 and 14 as contributing towards an *International* genre prediction. For *New Age* only the fourth and fifth measures are highlighted by LIME. For *Here Comes the Sun*, the fifth and sixth bars, along with their repetition four bars later contribute more towards the *Pop-Rock* genre. For *The Real Slim Shady* the same three notes have been highlighted as contributing towards the *Rap* genre across two repetitions. A different repetition of the same notes has been highlighted as contributing towards *Jazz*. Finally, the first measure of the piano part of *Bohemian Rhapsody* along with its preceding measure contribute towards *Electronic*, while the third measure after the piano is introduced contributes towards *Pop-Rock*. These explanations seem to generally agree with those generated by Grad-CAM and are not very intuitive visually. However, by changing the representation of the MIDI files to either music notation (score) or audio, the highlighted parts may be observed in more detail and listened to. It would be interesting to analyze them from a music-theoretical perspective; however, this is beyond the scope of this work.

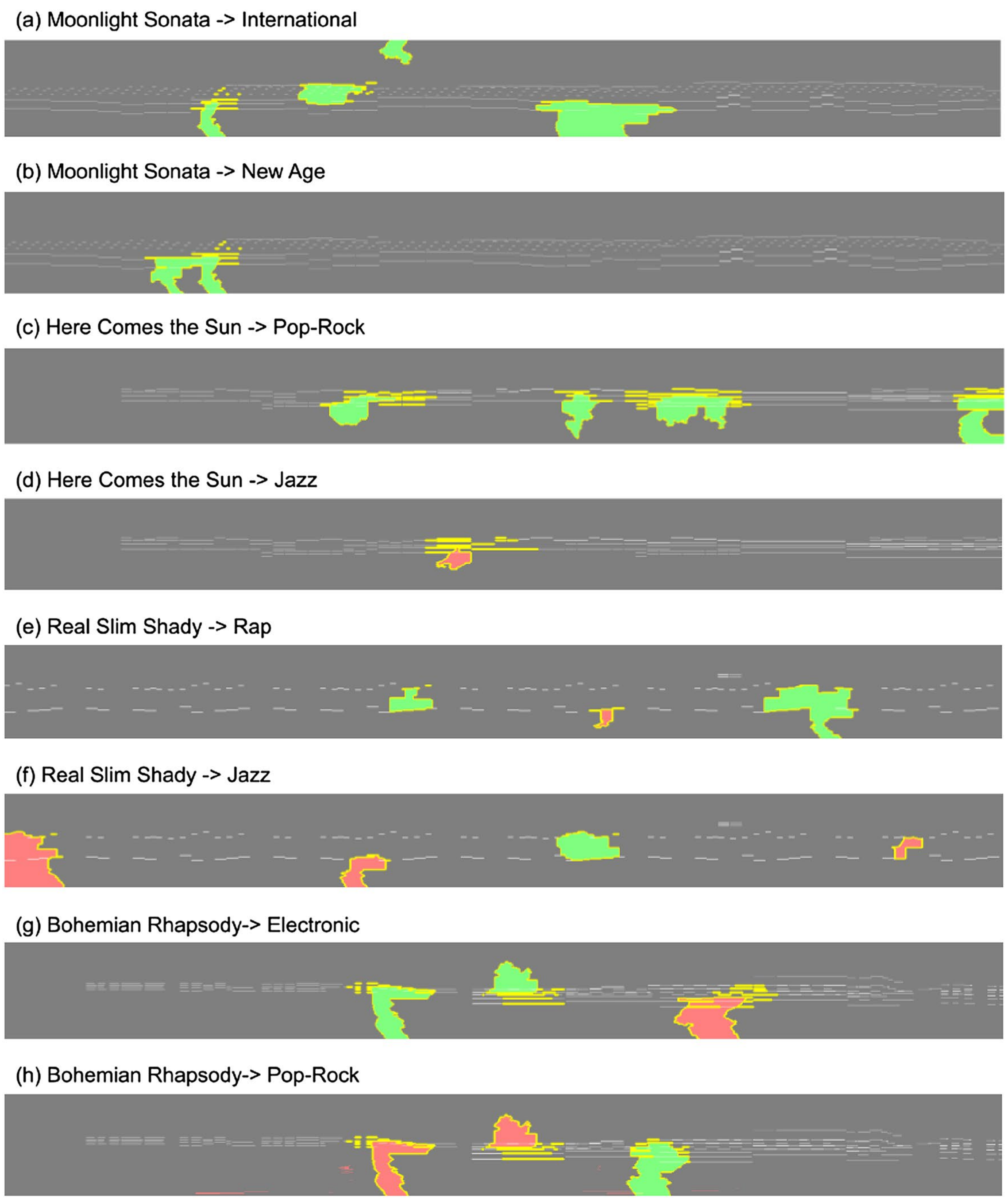

Fig. 9 Explanations generated by LIME for the top-2 predicted genres, for the best performing CNN, on the first 1024 time-steps of Beethoven's Moonlight Sonata (**a**, **b**), The Beatles—Here Comes the Sun (**c**, **d**), Eminem—The Real Slim Shady (**e**, **f**), and Queen—Bohemian Rhapsody (**g**, **h**). With green are highlighted the areas of the pianoroll that positively contribute to the label prediction, and with red those that negatively contribute, according to LIME

Modified GPX

GPX [8] is a methodology for generating local explanations by utilizing genetic programming [17]. It is formulated similarly to LIME, in that GPX searches for a function $\xi : \mathbb{R}^n \rightarrow \mathbb{R}$ which attempts to mimic the original complex model $f : \mathbb{R}^d \rightarrow \mathbb{R}$ on a given sample set η, which is typically defined locally around the sample to be explained. The goal of GPX is to find:

$$\xi = \arg \min_{g \in G, s_i \in \eta} d([g(s_1), \ldots, g(s_m)] - [f(s_1), \ldots, f(s_m)]), \quad (5)$$

where d is a distance function, such as the l_2-norm. GPX generates functions g which are non-linear algebraic expressions in the form of binary trees on the given feature set. Genetic Programming (GP), in general, generates a random population and evaluates the fitness of each individual, in terms of effectiveness in solving the problem, favoring the better individuals. In this work, the GP evolves symbolic expressions for local explanation in the genre classification task.

We modified GPX in two ways to generate more meaningful explanations for our application: a) the way the local sample set η is generated and b) what features are used within the genetic program, and for producing explanations.

Local Sample Set To generate the sample set η around the sample x to be explained, GPX samples from a multivariate Gaussian distribution centered at x with a covariance matrix computed from the training data. By experimenting with this approach we realized that on the one hand, the predictions of the classifier on such a sample set were not varying significantly, and on the other hand, the samples generated with this approach were not meaningful as pianoroll representations, and the resulting samples did not have any musical meaning. Instead, we generate each η_i by randomly choosing a set of pitches which have a value > 0 in the cumulative pianoroll of x and transposing them by a random number of semitones in $[-10, 10]$. This way η, which is supposed to consist of samples in the neighborhood of x, will contain pianorolls that are similar to x and differ only with regard to some pitches. Notably, this approach does not impact the rhythmic characteristics of the pianoroll, since the only changes are made in the pitch dimension, so even if pitches are changed drastically for some samples, they can still be considered to be “local”.

Feature Extraction Each cumulative pianoroll consists of 1024 time-steps of 128-sized vectors, which would translate to $128 \times 1024 = 131,072$ features per sample. Such a large amount of features makes the application of GPX infeasible, and even if time and memory complexity were not an issue, the explanations which would be generated would not be very informative, as GPX is a feature importance explainability method, and it is difficult to draw conclusions from such a large feature set. Instead, we incorporate a feature extraction function $h : \mathbb{R}^n \rightarrow \mathbb{R}^p$ within the explainer function, where n is the original number of features and $p < n$ is the number of extracted features, and reformulate Eq. 5 as:

$$\xi = \arg \min_{g \in G, s_i \in \eta} d([g(h(s_1)), \dots, g(h(s_m))] - [f(s_1), \dots, f(s_m)]). \tag{6}$$

For applying this idea on symbolic music, we define h to extract thirteen features, which are intuitively linked to musical harmony. Specifically, for the first twelve features, we first get the prevalence of each pitch by summing x over the time dimension, leading to a 128-sized vector. Then from this vector, we get the prevalence of each pitch class, by summing each dimension modulo 12, leading to a vector of 12 elements, each of which represents the prevalence of a pitch class within the pianoroll. Finally, we transpose this vector by rolling it, such that the largest element is at position 0. The thirteenth feature is the average pitch across the whole pianoroll. This way, the function h, acts on a pianoroll x to produce a vector $h(x)$ which shows the prevalence of each musical interval when compared to the most used pitch class. The features $h(x)$ are then utilized by the explainer which attempts to mimic the black-box classifier, and are the features that appear in generated explanations.

For the genetic programming algorithm, we used a population size of 110 evolved over 110 generations, to attempt to mimic the classifier in a local neighborhood of 11,000 samples η_i. For other hyperparameters, we used the same as those used in [8]. We ran the algorithm 5 times for each sample, and show the best results with regard to the explainer accuracy. We comment on the consistency of the results.

In Fig. 10, we show the feature importance generated by GPX for each prediction of the CNN. The feature importance is calculated as the number of appearances of each feature in the final population, which consists of 110 algebraic expressions. For Bohemian Rhapsody-Electronic and Moonlight Sonata-International, which are both erroneous predictions, GPX has shown two intervals as important features. This result was consistent for the case of Bohemian Rhapsody; while for Moonlight Sonata, the *maj_7* interval appeared in all 5 runs, but not the *fourth* interval. For the other two samples, the best explainer was a constant function; however, the existence of features in the final population can give us some musical insight. In Table 4 we show a summary of results for the top prediction for each of the four selected samples. For the local sample set generated around Here Comes the Sun and The Real Slim Shady, the CNN classified 0.75 and 0.93 as *Pop-Rock* and *Rap* respectively (percentage of positive η_i). Even though the best explainer for these two samples was (essentially) a constant function, the prevalence of the tonic note as an important feature for *Rap* classification in the final population makes sense intuitively.

In Fig. 11, we show the final program evolved by GPX with the best explainer accuracy for each sample. For Moonlight Sonata (explainer accuracy 0.8), the program is *fourth* – *maj_7*, where the intervals are based on the extracted tonic of B. However we know that Moonlight Sonata is actually in $C\sharp$ minor, so in this context, the program would be *min_third* – *sixth*. For the actual sample, the prevalence of the first interval is 0.59 while of the second it is 0.08, which would indicate that when the pitch class $B\flat$ is more prevalent, then the sample is no longer classified as

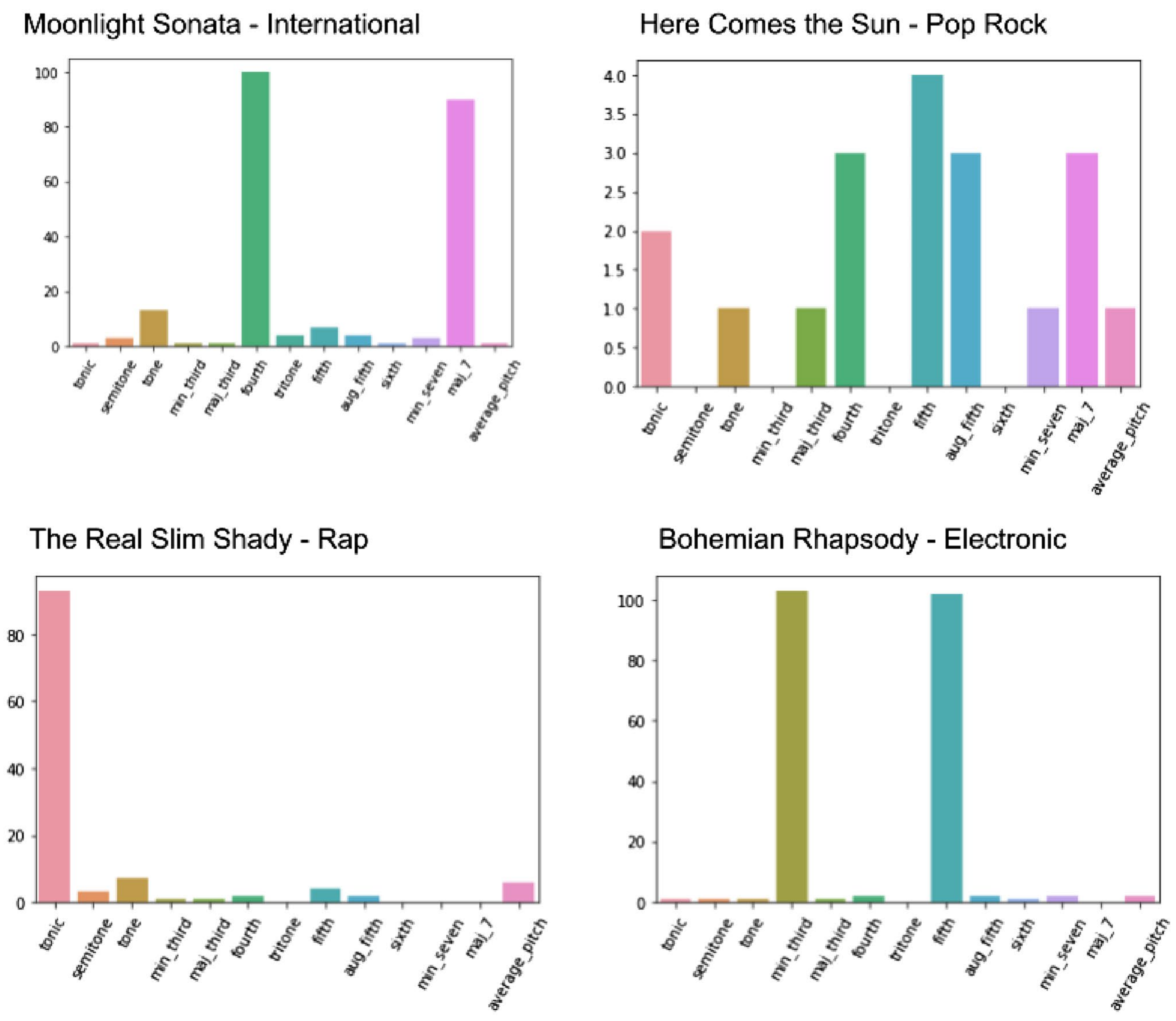

Fig. 10 Feature importance for the top-1 predicted genre as generated by the modified GPX

Table 4 Summary of results of applying GPX with the feature extraction addition, for the top prediction for each sample

	Beethoven	Beatles	Eminem	Queen
percentage of positive η_i	**0.69**	0.75	0.93	**0.62**
Explainer accuracy	**0.80**	0.75	0.93	**0.75**
Extracted tonic	B	A	E♭/D♯	B♭/A♯
Actual key	C♯ min	A Maj	C min	B♭ Maj

Bold shows the pieces of music for which explainer accuracy is different from percentage of positives in the local neighborhood

International. In music-theoretical terms, *B*♭ appears as a Dorian substitute and it would be interesting to explore the distribution of the Dorian mode within our dataset of *International* music, and determine if this is a bias learned by the classifier, or if it is an actual feature of the genre. These results are also somewhat consistent with the explanation generated by LIME (Fig 9) since for measures 13 and 14 in which the pitch class *B*♭ appears for the first time, LIME has only highlighted the bass notes (which lack the pitch class) as contributing towards the *International* genre. For Bohemian Rhapsody (explainer accuracy 0.75), the best program generated by GPX is min(*min_third*,*fifth*). In the actual sample the prevalence of the first interval is 0.63 and of the second is 0.51. This rule says that if both the minor third and the fifth appear at least half as often as the tonic, then the sample is classified as *Electronic*, which makes intuitive sense since *Electronic* music tends not to have complex harmony, and the rule represents the prevalence of a minor triad in the pitches which appear in the pianoroll. Again,

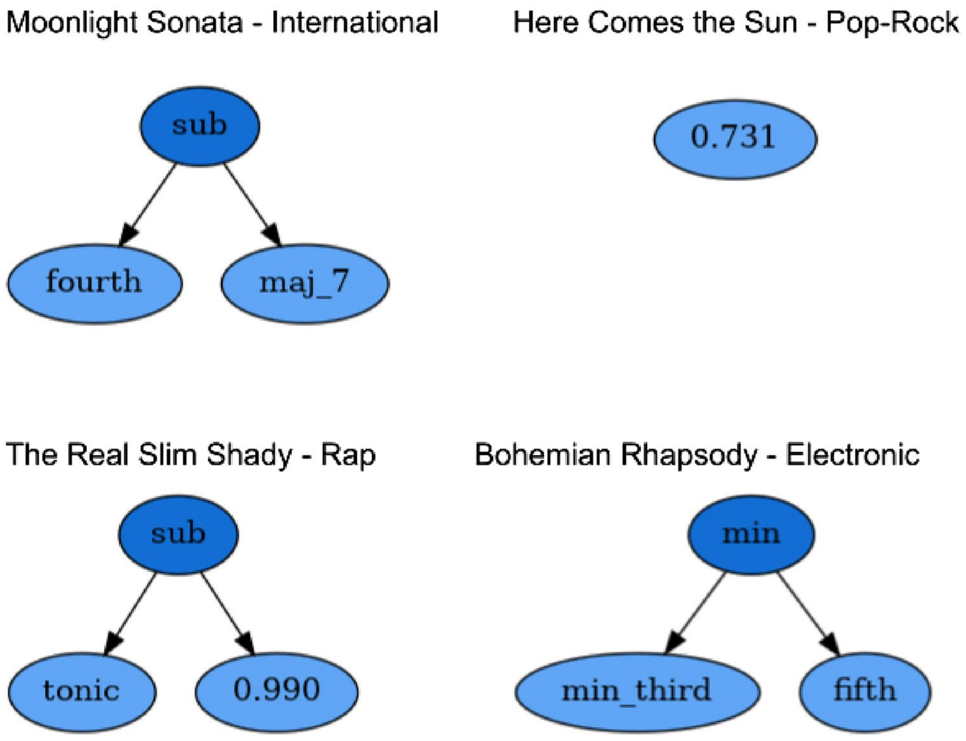
Moonlight Sonata - International
sub
fourth
maj_7
Here Comes the Sun - Pop-Rock
0.731
The Real Slim Shady - Rap
sub
tonic
0.990
Bohemian Rhapsody - Electronic
min
min_third
fifth

Fig. 11 Final programs generated by GPX as explanations for the top prediction for each sample

this merits further exploration to determine if it is a bias, or something useful that the CNN has learned. For the other two samples, the explanations generated are trivial. For Here Comes the Sun, we attribute the failure of GPX to the bias of the classifier towards the *Pop-Rock* genre which is a result of dataset imbalance. For The Real Slim Shady, we believe that all samples in the local sample set were classified as *Rap* due to the fact that the sample uses very few pitches, which does not change after randomly transposing them, and the sampling method does not affect rhythmic characteristics, which we believe to be a key feature for *Rap* classification.

Global Explanation Methods

Global explanation methods aim to explain the overall behavior of a black-box, contrary to local explanations which concern the predictions of the model on a specific instance.

MMD-critic [14]

MMD-critic is a methodology for analyzing the distribution of a dataset to find specific samples which are prototypes, and others which are criticisms. The former are samples that are characteristic for a specific distribution while the latter are outliers. MMD-critic is based on the idea of Bayesian Model Criticism [9] and produces explanations by calculating *Maximum Mean Discrepancy* (MMD).

To produce global explanations for a black-box model with MMD-critic, we first calculate prototypes and criticisms for the test set, by feeding the algorithm cumulative pianorolls. Then, for each genre, we get the set of positive examples as predicted by the black-box and compute prototypes and criticisms for each of these sets of positive examples. This way we get prototypes and criticisms according to the real distribution (test set) in addition to the distribution learned by the black-box (predictions on the test set). In Table 5 we show a prototype and a criticism for each genre, as computed by MMD-critic on the test set of topMAGD, and in Table 6 we show a prototype and a criticism for each genre, as predicted by the CNN.

Regarding the results on the test set (Table 5), we can gain some insight about the dataset and by extension the performance of the CNNs. First, this table raises the issue of ground-truth reliability, which is one of the main difficulties for genre recognition. For instance, none of the *Reggae* samples in the table are actually *Reggae*, but would probably be considered Soul/RnB/Gospel. Furthermore, the prototype for the *Latin* genre does not have any characteristics of *Latin* music besides the language and would be considered Pop. Similarly, the prototype for *Rap* could be considered RnB (which often contains Rap in modern music), while the prototype for *New Age* could be considered New Wave/Pop instead. A second issue raised by Table 5 is that of dataset imbalance. This is not only regarding the number of samples for each genre, but also the range of different music each genre encompasses. For instance, the *Pop-Rock* genre is represented by a very diverse set of samples, ranging from Hard Rock to Disco. A result of this is that almost half of the

Table 5 Prototypes and Criticisms for each genre, generated by MMD-critic on the test set of topMAGD. In parentheses are the ground-truth labels

Genre	Test set prototype	Test set criticism
Pop-Rock	Nine Inch Nails—Piggy (Pop-Rock)	Spice Girls—Wannabe (Pop-Rock)
Electronic	Toy-Box—Tarzan & Jane (Pop-Rock, Electronic)	George Michael - Fast Love (Electronic)
Country	Session Americana - John Brown (Country)	Olivia Newton-John - Everything Love Is (Pop-Rock, Country)
RnB	Mariah Carey - If It's Over (Pop-Rock, RnB)	Tina Turner - Steamy Windows (RnB)
Jazz	Lee Ritenour - Papa Was A Rolling Stone (Jazz)	Procol Harum - A Whiter Shade Of Pale (Pop-Rock, Jazz)
Latin	Yahir - Fue Ella, Fui Yo (Latin)	Os Paralamas Do sucesso - Romance Ideal (Latin)
International	Mamonas Assassinas - Pelados Em Santos (International)	Mamonas Assassinas - Pelados Em Santos (International)
Rap	50 Cent - Baby By Me (Pop-Rock, Electronic, Rap)	Bobby Brown - Don't Be Cruel (Pop-Rock, Jazz)
Vocal	Nana Mouskouri - Habanera (International, Vocal)	Salvatore Licitra - E Lucevan Le Stelle (Vocal)
New Age	Cock Robin - The Promise You Made (New Age)	Slavic Soul Party! - Never Gonna Let You Go (Jazz, New Age)
Folk	Judy Collins - Send In The Clowns (Folk)	Edison Lighthouse - Love Grows (Folk)
Reggae	Johnnie Taylor - For Your Precious Love (Pop-Rock, RnB, Reggae)	The Elgins - When A Man Loves A Woman (Pop-Rock, Country, Latin, Reggae)
Blues	Deborah Coleman - Long Time (Pop-Rock, Blues)	Jim Reeves - I Won't Forget You (Country, Blues)

Table 6 Prototypes and Criticisms for each genre, generated by MMD-critic on the positive examples as predicted by the black box, on the topMAGD test set. In parentheses are the ground-truth labels

Genre	Black-box prototype	Black-box criticism
Pop-Rock	Abba - The Winner Takes It All (Pop-Rock, Vocal)	Donny Osmond - This Guy's In Love With You (Pop-Rock)
Electronic	Siniestro Total - C'est Chic (Pop-Rock)	Crystal Waters - 100% Pure Love (Electronic)
Country	Boots Randolph - Bridge Over Troubled Water (Country)	John Fogerty - Big Train (Pop-Rock)
RnB	Stevie Wonder - You Are The Sunshine Of My Life (RnB)	Whitney Houston - So Emotional (RnB)
Jazz	Abba - Take A Chance On Me (Pop-Rock)	Vince Guaraldi Trio - Christmas Time Is Here (Jazz)
Latin	Luis Miguel - El Dia Que Me Quieras (Latin)	Os Paralamas Do Sucesso - Romance Ideal (Latin)
International	Brasilian Tropical Orchestra - Yesterday (International)	Uniting Nations - Uniting Nations (Electronic)
Rap	Queen - We Will Rock You (Pop-Rock)	Phish - Wading In The Velvet (Pop-Rock)
Vocal	Michael Crawford - The Phantom Of The Opera (Pop-Rock, Vocal)	Collin Raye - Little Rock (Country, Vocal)
New Age	Lionel Richie - Hello (New Age)	Enya - China Roses (New Age)
Folk	The Roches - Do You Hear What I Hear? (Folk)	The Roches - It Came Upon A Midnight Clear (Folk)
Reggae	The Elgins - When A Man Loves A Woman (Pop-Rock, RnB, Reggae)	Johnnie Taylor - For Your Precious Love (Pop-Rock, RnB, Reggae)
Blues	Bill Quinn - He'll Have To Go (Country, Blues)	Jim Reeves - He'll Have To Go (Country)

selected prototypes and criticisms are labeled as *Pop-Rock* among other labels. Finally, for those genres with very low support, we cannot expect MMD-critic to produce meaningful explanations since it is a statistics-based approach that requires a sufficiently large dataset.

By studying the resulting prototypes and criticisms from the predictions of the CNN (Table 6), along with the performance of the CNN on each genre (Table 2) we are able to better understand what the CNN has learned. For *Pop-Rock* the prototype chosen by MMD-critic is a power ballad by Abba which is closer to Pop than Rock, which is interesting when compared to the prototype selected from the ground-truth labels: a Nine Inch Nails song which is a lot closer to Rock. For genres with more than 100 samples in the test set, in which the CNN does not perform well (Electronic, Latin, International, Rap) the generated prototypes are, as expected, far from representative of each genre, however, they are still useful for gaining insight on what the CNN has learned. For instance, the choice of Queen—*We Will Rock You* as a prototype for *Rap* could be due to the rhythmic qualities of the vocal track, the looping music, and the repeating patterns which are prevalent in a lot of different music, including *Rap*. This could help us understand the poor performance for the specific genre (0.34 F1 Score), along with the high precision (0.78).

Discussion

In this section, we showed how various explainability methods may be used in the context of symbolic music classification. These give us some insight concerning the operation of the CNN and may be used for guiding future design decisions. In this context, visual explanations are only useful if the viewer is familiar with the piece of music, and is able to understand different parts and sections from their pianoroll representation, and even then interpreting them is not straightforward without music-theoretical analysis. Especially for results that are consistent across explainability methods such as for Moonlight Sonata we found the explanations to be particularly insightful.

Conclusions and Future Work

We have demonstrated the effectiveness of CNNs for information retrieval from symbolically represented music. From our results, it is apparent that for our definitions of ’shallow’ and ’deep’ CNNs, the ’shallow’ ones are most suited for the task at hand. We plan to further explore this idea, using network architecture search to find a good baseline network and similarly to [45] find an efficient way to scale up neural networks for MIDI classification tasks.

In addition, even though ’MuSeRe’ networks performed poorly when compared to ’Sequence’ networks in most cases, it is interesting that for the smallest models and shortest input lengths ’MuSeRe’ models tended to outperform ’Sequence’ ones. In our experiments, we used simple tree representations, which do not hold a lot of musical significance. In the future, we will incorporate more musically meaningful tree representations of music that have been proposed in literature over the years. This includes utilizing audio alongside symbolic representations for a hybrid approach for the genre recognition task.

Furthermore, increasing input sequence length effectively reduces the number of (non-overlapping) sequences in the dataset while network size increases, which makes models more prone to overfitting. This is hinted at when comparing models trained on sequences of 1024 samples and those trained on sequences of 2048 samples. For future work, we will explore ways in which our networks will dynamically accept multiple different sequence lengths and will be fed entire MIDI files instead of fixed length sequences.

In addition, even though we used one of the largest available MIDI genre annotated datasets for training and evaluating our models, the dataset is by no means representative of all available music and suffers from poor class balance. For future work, we plan to augment our dataset by including files from other sources, such as the reddit[4] MIDI dataset and automatically acquire additional labels and annotations from online sources such as The Echo Nest[5] and Spotify APIs.

Moreover, music genres themselves are part of a complex domain with hierarchical structures and different relationships between elements, such as sub-genres, fusions of genres, etc. Utilizing these relationships could lead to more robust genre recognition, similarly to how utilizing relationships between chords by Carsault et al. in [3] improved performance for chord recognition.

Finally, we gained insights about what the best performing CNN has learned by adapting post hoc explainability methods to our domain of symbolic music. Such methods could be used to guide future network design, especially since NAS is very expensive computationally; however, more work needs to be done for explainability in the symbolic music domain. Specifically, we believe that it would be useful to generate audible explanations, in addition to explanations based on terms from music-theory, in order to get a better understanding of black-box models which act on symbolic music.

Acknowledgements This research is carried out/funded in the context of the project “Automatic Music Composition with Hybrid Models of Knowledge Representation, Automatic Reasoning and Deep Machine Learning” (5049188) under the call for proposals “Researchers’ support with an emphasis on young researchers- 2nd Cycle”. The project is co-financed by Greece and the European Union (European Social Fund- ESF) by the Operational Programme Human Resources Development, Education and Lifelong Learning 2014–2020.”

Funding Open access funding provided by HEAL-Link Greece.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

[4] https://www.reddit.com/r/WeAreTheMusicMakers/comments/3ajwe4/the_largest_midi_collection_on_the_internet/.

[5] http://static.echonest.com/enspex/.

References

1. Bertin-Mahieux T, Ellis DP, Whitman B, Lamere P. The million song dataset 2011.
2. Brunner G, Konrad A, Wang Y, Wattenhofer R. Midi-vae: Modeling dynamics and instrumentation of music with applications to style transfer. In: 19th International Society for Music Information Retrieval Conference (ISMIR 2018) 2018.
3. Carsault T, Nika J, Esling P. Using musical relationships between chord labels in automatic chord extraction tasks. arXiv preprint arXiv:1911.04973 2019.
4. Dannenberg RB, Thom B, Watson D. A machine learning approach to musical style recognition 1997.
5. Dong HW, Hsiao WY, Yang LC, Yang YH. Musegan: Multi-track sequential generative adversarial networks for symbolic music generation and accompaniment. In: Thirty-Second AAAI Conference on Artificial Intelligence 2018.
6. Duggirala S, Moh TS. A novel approach to music genre classification using natural language processing and spark. In: 2020 14th International Conference on Ubiquitous Information Management and Communication (IMCOM), 2020;1–8. IEEE.
7. Ferraro A, Lemstöm K. On large-scale genre classification in symbolically encoded music by automatic identification of repeating patterns. In: 5th International Conference on Digital Libraries for Musicology. Paris 2018. https://doi.org/10.1145/3273024.3273035.
8. Ferreira LA, Guimarães FG, Silva R. Applying genetic programming to improve interpretability in machine learning models. In: 2020 IEEE Congress on Evolutionary Computation (CEC), 2020;1–8. IEEE.
9. Gelman A, Carlin J, Stern H, Rubin D. Bayesian data analysis taylor & francis. Boca Raton, FL, USA.[Google Scholar] 2014.
10. Guidotti R, Monreale A, Ruggieri S, Turini F, Giannotti F, Pedreschi D. A survey of methods for explaining black box models. ACM computing surveys (CSUR). 2018;51(5):1–42.
11. Hanin B. Which neural net architectures give rise to exploding and vanishing gradients? In: Advances in Neural Information Processing Systems, 2018;582–591.
12. Karydis I, Nanopoulos A, Manolopoulos Y. Symbolic musical genre classification based on repeating patterns. In: Proceedings of the 1st ACM Workshop on Audio and Music Computing Multimedia, 2006;53–58.
13. Kawahara J, Hamarneh G. Multi-resolution-tract cnn with hybrid pretrained and skin-lesion trained layers. In: Wang L, Adeli E, Wang Q, Shi Y, Suk HI, editors. Machine Learning in Medical Imaging. Cham: Springer International Publishing; 2016. p. 164–71.
14. Kim B, Khanna R, Koyejo OO. Examples are not enough, learn to criticize! criticism for interpretability. Advances in neural information processing systems 2016;29.
15. Kingma DP, Ba J. Adam: A method for stochastic optimization. In: Y. Bengio, Y. LeCun (eds.) 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (2015). URL http://arxiv.org/abs/1412.6980.
16. Kotsifakos A, Kotsifakos EE, Papapetrou P, Athitsos V. Genre classification of symbolic music with smbgt. In: Proceedings of the 6th international conference on PErvasive technologies related to assistive environments, 2013;1–7.
17. Koza JR, Koza JR. Genetic programming: on the programming of computers by means of natural selection, vol. 1. MIT press; 1992.
18. Lanchantin J, Singh R, Lin Z, Qi Y. Deep motif: Visualizing genomic sequence classifications. arXiv preprint arXiv:1605.01133 2016.
19. Lattner S, Grachten M, Widmer G. Learning transposition-invariant interval features from symbolic music and audio. arXiv preprint arXiv:1806.08236 2018.
20. Lerdahl F, Jackendoff R. A Generative Theory of Tonal Music. Cambridge, MA: MIT Press; 1983.
21. Liang H, Lei W, Chan PY, Yang Z, Sun M, Chua TS. Pirhdy: Learning pitch-, rhythm-, and dynamics-aware embeddings for symbolic music. In: Proceedings of the 28th ACM International Conference on Multimedia, 2020;574–582.
22. Liu C, Feng L, Liu G, Wang H, Liu S. Bottom-up broadcast neural network for music genre classification. Multimedia Tools and Applications. 2021;80(5):7313–31.
23. Mao HH, Shin T, Cottrell G. Deepj: Style-specific music generation. In: 2018 IEEE 12th International Conference on Semantic Computing (ICSC), 2018;377–382. IEEE.
24. Marsden A. Representing melodic patterns as networks of elaborations. Comput Humanit. 2001;35:37–54. https://doi.org/10.1023/A:1002705506386.
25. McKay C, Fujinaga I. Musical genre classification: Is it worth pursuing and how can it be improved? In: ISMIR, 2006;101–106.
26. Medhat F, Chesmore D, Robinson J. Masked conditional neural networks for sound classification. Appl Soft Comput. 2020;90:106073.
27. Meredith D, Lemström K, Wiggins GA. Algorithms for discovering repeated patterns in multidimensional representations of polyphonic music. Journal of New Music Research. 2002;31(4):321–45.
28. Oord Avd, Dieleman S, Zen H, Simonyan K, Vinyals O, Graves A, Kalchbrenner N, Senior A, Kavukcuoglu K. Wavenet: A generative model for raw audio. arXiv preprint arXiv:1609.03499 2016.
29. Oramas S, Barbieri F, Nieto O, Serra X. Multimodal deep learning for music genre classification. Transactions of the International Society for Music Information Retrieval. 2018;1(1): 4-21. 2018.
30. Padial J, Goel A. Music mood classification 2018.
31. Raffel C. Learning-based methods for comparing sequences, with applications to audio-to-midi alignment and matching. Ph.D. thesis, Columbia University 2016.
32. Real E, Aggarwal A, Huang Y, Le QV. Regularized evolution for image classifier architecture search. In: Proceedings of the aaai conference on artificial intelligence, 2019;33, 4780–4789.
33. Ren JM, Wu MJ, Jang JSR. Automatic music mood classification based on timbre and modulation features. IEEE Trans Affect Comput. 2015;6(3):236–46.
34. Ribeiro MT, Singh S, Guestrin C. "why should i trust you?" explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, 2016;1135–1144.
35. Rizo D, Iñesta JM, Moreno-seco F. Tree-structured representation of musical information. In: 1ST Iberian Conference on pattern recognition and image analysis. Palma De Mallorca, Spain, Vol. 2652 OF LNCS, 2003;838–846. Lecture.
36. Rizo D, Marsden A. An mei-based standard encoding for hierarchical music analyses. Int J Digit Libr. 2019;20(1):93–105. https://doi.org/10.1007/s00799-018-0262-x.
37. Roland P. The music encoding initiative (mei). In: Proceedings of the First International Conference on Musical Applications Using XML, 2002;1060, 55–59.
38. Ronneberger O, Fischer P, Brox T. U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention, 2015;234–241. Springer.
39. Rosner A, Kostek B. Automatic music genre classification based on musical instrument track separation. Journal of Intelligent Information Systems. 2018;50(2):363–84.

40. Rudin C. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nature Machine Intelligence. 2019;1(5):206–15.
41. Schenker H. Free Composition: Volume III of new musical theories and fantasies, vol. 3. Pendragon Press 2001.
42. Schindler A, Mayer R, Rauber A. Facilitating comprehensive benchmarking experiments on the million song dataset. In: ISMIR, 2012;469–474.
43. Selvaraju RR, Cogswell M, Das A, Vedantam R, Parikh D, Batra D. Grad-cam: Visual explanations from deep networks via gradient-based localization. In: Proceedings of the IEEE international conference on computer vision, 2017;618–626.
44. Senac C, Pellegrini T, Mouret F, Pinquier J. Music feature maps with convolutional neural networks for music genre classification. In: Proceedings of the 15th international workshop on content-based multimedia indexing, 2017;1–5.
45. Tan M, Le QV. Efficientnet: Rethinking model scaling for convolutional neural networks. arXiv preprint arXiv:1905.11946 2019.
46. Ukkonen E, Lemström K, Mäkinen V. Sweepline the music. In: Computer Science in Perspective, 2003;330–342. Springer.
47. Vishnupriya S, Meenakshi K. Automatic music genre classification using convolution neural network. In: 2018 International Conference on Computer Communication and Informatics (ICCCI), 2018;1–4. IEEE.
48. Wu Y, Li W. Automatic audio chord recognition with midi-trained deep feature and blstm-crf sequence decoding model. IEEE/ACM Transactions on Audio, Speech, and Language Processing. 2018;27(2):355–66.
49. Xie D, Xiong J, Pu S. All you need is beyond a good init: Exploring better solution for training extremely deep convolutional neural networks with orthonormality and modulation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017;6176–6185.
50. Xue H, Xue L, Su F. Multimodal music mood classification by fusion of audio and lyrics. In: International Conference on Multimedia Modeling, 2015;26–37. Springer.
51. Yang R, Feng L, Wang H, Yao J, Luo S. Parallel recurrent convolutional neural networks-based music genre classification method for mobile devices. IEEE Access. 2020;8:19629–37.
52. Yu Y, Luo S, Liu S, Qiao H, Liu Y, Feng L. Deep attention based music genre classification. Neurocomputing. 2020;372:84–91.
53. Zheng E, Moh M, Moh TS. Music genre classification: A n-gram based musicological approach. In: 2017 IEEE 7th International Advance Computing Conference (IACC), 2017;671–677. IEEE.