Instruments as a key parameter for composer classification models

By

Nathan Hoche
Kent ID: _____
_____

School of Computing, University of Kent

Supervisor: Dr. Dominique Chu

Date: September the 1st, 2023

Number of words: 8357

ABSTRACT

Composer classification is mainly made using feature extraction or deep learning techniques, such as image neural networks. None of these techniques consider instruments as a parameter for the classification. This dissertation focuses on the hypothesis that instruments can be a key to improve composer classification. In order to find evidence for this hypothesis, an RNN model for each instrument, trained to classify composers, is created. These models served as input for a Decision Tree which studies which instruments can be used to classify composers based on its model prediction. The training of the models is done on the MusicNet, a MIDI file dataset. The results show that composers are mainly classified by a unique instrument, which differs for each composer. It also points out that other instruments model predictions manage to isolate the misclassified music, whereas some instrument models are left aside. Several assumptions can be made from these results: composers can be recognized based on a unique specific instrument, but not all of their music. This results in a need for the combination of several instruments in order to classify all the music. The presence or absence of an instrument can be a key to separate two composers. Some instruments do not have any impact on the composer classification. All these assumptions prove to be interesting for the study of music using the instruments.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my supervisor Dr. Dominique Chu for his help and guidance throughout the elaboration of this project.

CONTENT

LIST OF FIGURES AND TABLES

1 INTRODUCTION

Parallel to the appearance of commercial music platforms over the last 10 years, the problem of music classification has arisen. This is mainly due to the commercial needs of increasing the customer's appetite for more by proposing genres and titles they could like. To achieve this goal, several software programs have been developed based on audio files, of which the most known is probably « Shazam ».

The goal of this dissertation is to study the interest of instruments in the field of classification of composers.

While there are already multiple ways to categorise music through genres and composers, our aim is to concentrate on the importance of the use of each instrument by different composers. In order to do so, we will apply a number of deep learning methods on MIDI file data. The choice of this type of source is linked to the fact that it allows a convenient way to study the impact of a given instrument on the accuracy of the classification. We will use a MusicNet dataset containing 330 MIDI files of 10 different composers. This will enable us to measure the impact of each instrument by its relevance of prediction for each composer. This relevance outcome will then determine the instrument's position in the final decision tree.

The dissertation will be structured as follows : after an enumeration of useful terms and definition, as well as a review of related scientific work, we will explain the methodology used, before diving into the implementation and the actual deep learning process. Then, the results will be analysed and evaluated, and finally, a general conclusion will summarise the outcomes.

2 BACKGROUND

In order to provide a better understanding of the basic tools and techniques employed in this project, a short description of their main qualities will be outlined in this chapter.

2.1. MIDI FILES

A MIDI file, or Musical Instrument Digital Interface, contains important information such as the notes, the different instruments needed, the tempo, and so on. But it does not play any role of storage for the music itself. In other words, a MIDI file contains everything a musician needs to reproduce the music it describes except the actual sounds produced. It goes without saying that each one of these informations is extremely interesting when it comes to deep learning computation. As a matter of fact, this specific type of data is convenient for the processing of deep learning, since the music therein is represented by a sequence of events in the form of the notes played by each instrument. Every single one of these events also informs about the moment in time it is played, the instrument which plays it, the note itself, the velocity and duration of the note played. Also, the extraction of all this information is easy and thus turns MIDI files into convenient sources for deep learning.

Each one of the files is divided into 2 parts: the Header and the Tracks. The header contains the general information about the music stored, such as its title, the name of its composer, and so forth. It is obvious that this part will not be of any use in this project whose aim is to predict exactly that through a deep learning process.

It is therefore that our sole interest will go toward the information stored in the tracks. In the case of an orchestra with multiple instruments, a MIDI file contains obviously multiple tracks, one for each instrument. Each one of these tracks is allocated to a specific channel, holding the event sequences. Then the sum of the different channels, representing a stream of musical events, creates a polyphonic system.

Each channel being formed by multiple events, our program will use 2 MIDI instructions, the first one being ‘program change‘ and the second ‘note on’. The program change instruction will define a specific instrument by an encoding system using numbers from 0 to 127.

PC#	Family	PC#	Family
1-8	Piano	65-72	Reed
9-16	Chromatic Percussion	73-80	Pipe
17-24	Organ	81-88	Synth Lead
25-32	Guitar	89-96	Synth Pad
33-40	Bass	97-104	Synth Effects
41-48	Strings	105-112	Ethnic
49-56	Ensemble	113-120	Percussive
57-64	Brass	121-128	Sound Effects

Figure 1: Midi file’s instrument families[OBJ]

Furthermore, the instruments will be organised in 16 ‘families’ as shown in figure 1. By family we understand the different forms of each instrument, for instance the family ‘piano’ contains: acoustic grand piano, bright acoustic piano, electric grand piano, Honky-Tonk piano, Rhodes piano, chorused piano, harpsichord piano, and clavinet.

As for the note on instruction, it will define each note by allocating it with 3 parameters : the type of note, its velocity and time. The type of note is defined by its number in a series from 0 to 127, organised in a grid of octaves and the notes contained in each octave, as shown in figure 2. It is important to indicate that the velocity parameter does not express the rapidity by which a given note is to be played but its intensity which also influences the volume produced. Low velocity means a softer touch on the key , producing a more quiet sound whereas high velocity has the inverse effect. Finally, the time parameter indicates the lapse of time separating a note from a preceding one.

Octave #	Note Numbers											
	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
-1	0	1	2	3	4	5	6	7	8	9	10	11
0	12	13	14	15	16	17	18	19	20	21	22	23
1	24	25	26	27	28	29	30	31	32	33	34	35
2	36	37	38	39	40	41	42	43	44	45	46	47
3	48	49	50	51	52	53	54	55	56	57	58	59
4	60	61	62	63	64	65	66	67	68	69	70	71
5	72	73	74	75	76	77	78	79	80	81	82	83
6	84	85	86	87	88	89	90	91	92	93	94	95
7	96	97	98	99	100	101	102	103	104	105	106	107
8	108	109	110	111	112	113	114	115	116	117	118	119
9	120	121	122	123	124	125	126	127				

Figure 2: Midi file’s encoded note values

The way a composer organises the exact interactions of the instruments he uses being very personal, these interactions included in the MIDI file therefore allow us to aim for a higher accuracy of classification.

On another level, due to the preprocessing of audio files, any attempt to extract an equivalent amount of usable data from this type of file would be an impossible task. Indeed, the preprocessing has for effect to remove data such as the instruments used, the notes played, or their duration, replacing them with the sounds produced by all of the instruments at a given moment without distinction other than the listener’s capacity to distinguish them. In other words, the precision of classification would be blurred and by far more difficult to obtain through deep learning processes.

2.2. DEEP LEARNING

Deep learning is a form of artificial intelligence and machine learning. It consists of the construction and the use of complex neural network architectures. Its general goal is to automatically learn and represent patterns and features within data. This technique being inspired by the human brain, the neural network is composed of a multitude of interconnected layers. Each one of these layers are made of a multitude of artificial neurons which reproduce the work of human brain cells. Artificial Neural Networks can be constructed using different types of layers, among which Convolutional and Recurrent layers.

Convolutional layers apply filters on the input data, this has for consequence to only bring out certain aspects of data features. For instance, a great number of image filters, such as Blur filters, Sharpening filters, Embossing filters and so on, can be used for image processing. This type of layer structure allows a neural network to focus only on relevant aspects in an otherwise complex dataset. Convolutional layers are frequently combined with pooling layers. A pooling layer is a process whose role is to reduce the number of features studied by applying a mathematical operation on a group of input. This results in a reduction of the input size, which accelerates the process and simplifies the operations. The combination of these layers is generally called a Convolutional Neural Network (CNN).

Another interesting type of neural network is the Recurrent Neural Network (RNN), which is used specifically to process a sequence of events. Indeed, whilst in a standard neural network the input is composed of independent features, in the case of a sequence of events, the features are all dependent one from another. Consequently, standard neural networks are not appropriate in this particular case. When it comes to studying music, each note played at a certain moment and with a certain duration is dependent from the notes preceding it and those that follow. Consequently, this sequential functionality is best studied by a recurrent layer. In the domain of music classification, different recurrent layers can be used, such as GRU, Bi-GRU, Pointer Network, and so on. The most frequently employed is LSTM or Long-short Term Memory, whose major quality is the

capture of long sequence information, which makes it particularly effective on sequence studies with complex dependencies.

2.3. DECISION TREE

Decision Tree is a supervised learning technique mimicking human decision taking. Graphically, it has a tree-like structure with branches, nodes and leafs. In which branches stand for the decision rules, decision nodes for the departure point of new branches and leaf nodes represent the final outcome of a decisional branch.

Generally speaking, a decision tree evaluates features through conditions which can be either mathematical or logical. The cascading architecture of the process enables a step-by-step evaluation, in which the more initial a decision node is, the more impact it has. As a consequence, this is a convenient way to increase the accuracy of the predictions. Another quality of decision trees is their ease of interpretability.

3 LITERATURE REVIEW

Scientific literature about deep learning techniques in use of the categorization of music is already a rather rich field, as shown by the result of a search for « deep learning for music classification » on Google scholar which lists more than one million articles. Consequently, we concentrated on articles that seemed relevant in 3 main topics: models for music classification, datasets for music classification, and data preprocessing.

3.1. MODELS FOR MUSIC CLASSIFICATION

Shazam is perhaps the best known automatic classifying software. In his article « An Industrial-Strength Audio Search Algorithm » [15], A.Wang describes the use of reproducible hash tokens extracted from audio files as a very accurate way of identifying

music in a large database. But, while the process is very accurate, its elaboration is both time and labour consuming.

Another paper [2], by Elbir and Aydin, also based on acoustic data, depicts the creation of an automatic music genre classifier using MusicRecNet (MRN), a deep neural network model, on a dataset called GTZAN. This dataset contains 10 different music genres with 100 samples for each one. The application of MRN on 5-second long Mel-spectrogram images of each sample allowed a standalone accuracy of 81.8%. Combined with a support vector machine classifier (SVM), this result is even increased to 97.6%.

Nanni et al. [12] propose to combine both visual and acoustic feature extraction on audio files in order to increase accuracy for their genre classification program.

One research team [1] used MIDI files for genre classification. In fact, Cataltepe et al. compared classification methods on Audio files, MIDI files, and on a combination of both, as a matter of fact, they converted MIDI to audio in order to employ audio features. They also compared their results to those achieved by another team who used MIDI features only and showed the superiority of the latter.

None of these genre classification processes studied the impact of one or more instruments.

In terms of composer classification, likewise to genre classification, a large number of classifying models can be used among which certain are traditional machine learning models and others are part of the catalogue of deep learning (DL) methods.

Among the traditional machine learning methods, one can mention the study done by Herrera et al. [5] who cites K-nearest neighbours, Naive Bayesian, Binary tree, Support Vector Machine. They conclude that none of these techniques have sufficient accuracy to be applied in real-world audio conditions for instrument identification.

In "Composer Classification Models for Music-Theory Building" [4], Herremans et al. compare different techniques, and show that the most efficient one is SVM with 86% of accuracy when applied on a generated dataset of Midi files from the KernScores database. This dataset contains Midi files of 3 composers: Haydn (254 files), Beethoven (196 files) and Bach (595 files) [4].

Generally speaking, the most frequently used traditional classifiers in composer classification found in this literature review were: SVM or support vector machine [2,4,5,6,12,13,14], KNN or K-nearest neighbours [1,5,6,13,14], Naive Bayes [4,5,6,7],

and Logic Regression [2,4,6]. Other models like decision trees were less representative [2,4,5].

As already announced, this review of related work predominantly searched for deep learning models in the domain of music classification [2,5,7,8,10,11,15,16,17,18] as they seem more effective. This is shown in J.Zhang's paper, "Music Feature Extraction and Classification Algorithm Based on Deep Learning" [18], who compares SVM algorithms with deep learning techniques. They used a GTZAN dataset of audio files destined to music genre classification and containing 10 different genres, SVM obtained an accuracy of 52% while the deep learning techniques obtained accuracies between 70% and 87%. Although this paper compares SVM and deep learning techniques under the aspect of Music genre classification and not composer classification, it nevertheless shows that deep learning techniques are more efficient than SVM in this field.

Another aspect to consider before deciding which technique to choose is the type of data used. Indeed, the exact type of neural network to use can depend on the content of the chosen dataset. Music being composed as sequences of notes, a convenient way to store it are Midi files. Studying the information in this type of dataset is not the same problem as working on images or text. After going through several articles [8,10,11,18] the use of recurrent neural networks (RNNs), which are a type of neural network used specifically to process a sequence of events, seems to be the most logical choice to make. The main reason for this is its capacity to study music under the aspect of each note being played at a certain moment and with a certain duration, but also being dependent from the notes preceding it and those that follow. This functionality that we call sequence of events is of great importance and must not be underestimated. In fact, the use of recurrent layers shows better results than the use of standard neural networks, as shown in "Large-Scale MIDI-based Composer Classification" [10]. Kong et al. find that the use of recurrent layers shows an accuracy of 0.648 while the use of standard neural networks only obtained an accuracy of 0.605 for samples of 10 composers of the GiantMIDI-Piano dataset with 30s of music as data. This result is confirmed when working on 100 composers with the same conditions : 0.341 for the standard neural networks and 0.385 with recurrent layers [10].

However, the use of recurrent layers is not the only deep learning method used to study music. In the case of audio files, due to the data encoded as frequency, spectrograms

are frequently used to reduce the data complexity [2,10]. As shown in the paper "Music Genre Classification and Music Recommendation by Using Deep Learning" from A. Elbir and N. Aydin [2], these spectrograms can be studied like images with a convolutional neural network. In fact, the use of this technique on the GTZAN dataset shows an accuracy of 81.8% using only MusicResNet for the image process, and can be boosted if a SVM is combined to the MusicResNet up to 97.9% of accuracy.

3.2. Datasets for music classification

In the field of computer based study of music, the selection of an appropriate dataset is of great importance. As explained on the article "Convolutional Composer Classification" [14], small datasets can create problems due to their lack of diversity for each composer, by example if all data from Bach are chorales and all data from Mozart are string quartets, the classification will be biased by this specificity and not consider a composer's style elements in their entirety. In order to avoid this problem, many authors decide to use different datasets for specific types of classifications.

For the classification of musical genre, the papers "Music Genre Classification and Music Recommendation by Using Deep Learning" [2] and "Music Feature Extraction and Classification Algorithm Based on Deep Learning" [18] are both based on the GTZAN dataset, which contains 1000 audio files of 10 different genres. This dataset is composed of 100 audio files for each genre, and each audio file is 30s long. The 10 genres are: blues, classical, country, disco, hiphop, jazz, metal, pop, reggae and rock [2,18].

For the classification of composers a large panel of datasets is available. In "Large-Scale MIDI-based Composer Classification" [10], the authors use the GiantMIDI-Piano dataset, which contains a total of 10,854 MIDI files composed by 2786 composers. This dataset is generated by scraping the names of pieces and composers out of the International Music Score Library Project (IMSLP). The thus obtained data will be used to find piano solos on Youtube videos which were finally converted to MIDI files [10]. Consequently, this dataset is only focused on piano solos which makes it incompatible with the goal of this dissertation (the study of the impact of instruments on the accuracy of a composer classification). In order to obtain a big dataset of multiple

instruments, some research papers generate their own datasets. This is the case in "Composer Classification Models for Music-Theory Building" [4] and in "Convolutional Composer Classification" [14], where the authors use the KernScores database to generate their own datasets. This database contains 108,703 files which can be downloaded as MIDI files [4], as well as being converted into audio files.

An alternative to the use of self-generated datasets is the use of composer-specific datasets, such as HM107 [6,13], HM207 [13] and HM285 [6] for the classification of Mozart and Haydn. These datasets are respectively composed of 107, 207 and 285 string quartets of Mozart and Haydn. They contain the data concerning 3 instruments: violin, viola and cello [6,13].

Another and larger example of this type of dataset is MAESTRO [8,16], which contains subunits with several composers. For instance, in "Deep Composer Classification Using Symbolic Representation" [8], the authors use the MAESTRO dataset V2.0.0 from which they decided to remove the music of several composers and to select only composers represented with a minimum of 16 pieces. This resulted in a dataset with 505 pieces from 13 composers. Another example of a big dataset is BHTHM [13], which is composed of 312 audio files by 5 composers: Bach, Handel, Telemann, Haydn and Mozart.

In some cases, the dataset used was generated without specifying the sources [1,11,15].

All of these datasets are mainly based on MIDI files, but this type of file can be easily converted to audio ones, which allows all the previously presented models to stay compatible with them [1].

3.3 Data preprocessing

In order to manipulate a dataset via a learning model, preprocessing is needed. Both in the case of audio and in Midi files, the number of parameters is too large to be studied without any adaptation. However, multiple solutions to this problem exist.

The downsizing of music dataset information is generally done by selecting short clips of a music piece [2,6,10], the length of these clips can differ from 5 seconds [2], 20 seconds [8], 30 seconds [1,10,13], up to 60 seconds [1], or even 120 seconds [1]. In some

cases however entire samples can be used [1,10]. In “Music genre classification using MIDI and audio features” [1], the comparison of the accuracy when working on 30s, 60s, 120s and entire MIDI file samples is undertaken. The outcome of this comparison is that for music genre classification, the length of the sample is to be considered. In fact, the accuracy is better for Classical music if the clips are shorter, while the opposite is true when it comes to studying pop music samples [1]. This seems to indicate that the duration of samples is a parameter to be taken seriously for the classification for each particular genre of music.

Frequently, the preprocessing of audio files consists of converting sound signals into image data. The conversion into spectrograms is an example of this type of conversion. A spectrogram is an image representation of spectral information over time [13], and is applied on short random samples of a music file. Multiple types of spectrograms can be used, such as the Mel spectrogram [2,10], Mel Frequency Cepstral coefficients (MFCC) [18] or VQT (variable-Q transform) spectrograms [13]. The technique of converting the music to an image is also frequently used in the case of Midi files. An example of sound-to-image conversion for Midi files are piano rolls techniques [8,10,13]. Piano rolls (PR) are a means to represent the number of pitches activated at a given moment in a sample [10]. This technique can be employed to create three different type of PR: *frame rolls* which represent the activated notes in each temporal frame [10], *onset rolls* which only provide the moment a note is activated without its duration [10] and *velocity/piano rolls* which is similar to the frame roll but contains in addition a colour translating the intensity (‘velocity’) of each note [8,10,13].

Kong et al. [10], used both spectrograms and piano-rolls techniques to compare the performance of Audio files and Midi files for composer classification in the case of piano solos. In order to do the comparison, they applied a Convolutional Neural Network (CNN) and a Convolutional Recurrent Neural Network (CRNN) on both representations. The results show a higher accuracy of the MIDI files for both networks when classifying whole music pieces from 10 composers and 100 composers. However, in the case of the study of 30 seconds clips, MIDI files prove to be more accurate for 10 composer sets, while Audio files show better results for 100 composers [10].

An alternative to converting the music data to an image, is to extract features from the music. In the case of audio files, different types of features can be extracted: Timbral features, Rhythmic content features and Pitch content features [1]. Timbral features can take multiple forms, such as Spectral centroids, Spectral Flux, Spectral Contrast, Mel Frequency Cepstral coefficients (MFCC) and so on [1,18]. Each one of these features has a different role: Spectral centroids characterise a frequency spectrum by its centre of mass, in order to transmit the brightness of a sound, Spectral Flux measure the rate of change of the signal spectrum, Spectral Contrasts measure the difference in decibels between peaks and valleys in a frequency spectrum and MFCC is a frequency representation considered close to the sound level perceived by the human ear [18].

Rhythmic content features represent the movement of music signals over time. They contain multiple information such as the beat, the tempo, the time signature and the regularity of the rhythm [1]. Finally, Pitch content features represent the features obtained with pitch detection techniques, aiming to obtain melody and harmony information [1].

Feature selections are slightly different when working with MIDI files. Just as in the domain of audio files, a multitude of types exist, such as Monophonic and Polyphonic features, Segment Features, Lengths of Segment features, Transforming Pitch for Sonata-Inspired Segment Features or Fraction of Overlap for Sonata-Inspired Segment features [6]. More than 100 different features can be extracted, such as Melodic Fifth Frequency, Melodic Octaves Frequency, and so on [4].

An alternative or even upgrade of the feature extraction is score encoding. In "Convolutional composer classification" [15], Verma and Thickstun generate a score, defined by a mathematical calculus based on the number of pitch / note-value, the range of note pitches and the number of distinct note values. This has for advantage to reduce the data to a single resultant feature, which can then be easily treated by simple classification model, such as SVM or KNN [14]

4 PROBLEM DEFINITION

The aim of this dissertation is to establish if the use of specific instruments as a means of identifying the composer in deep learning processes can be more efficient than

the use of the general sound of music in its globality. For this purpose, the goal is to find out if the prediction of a composer based on a unique instrument is accurate enough, or if the help of other instruments is necessary in order to develop accuracy.

In order to answer this question, we need to study the impact of each instrument on the accuracy of the prediction. For the purpose of doing so, the data must first be preprocessed in order to be compatible with a classifying model. Then an algorithm needs to be developed allowing us to measure the impact of each instrument.

A positive answer to this question could significantly change our way of studying music. Not only would it represent a new way of classifying music, but it could also lead to a better understanding of how a composer develops his personal style.

5 METHODOLOGY

Based on the literature review different approaches to the problem have come to evidence. In the following chapters, these approaches will be discussed, before stating the retained solution.

5.1. Approach selection

The creation of a composer classification program can be approached from different angles and with a multitude of tools, as seen in the previous sections. In the following sections we will briefly summarise these tools and justify the choices made for the project of this dissertation.

5.1.1. Data type selection

As explained in the literature review, audio files are frequently used for music classification [1,2,5,11,12,13,15,16]. In fact, they are the most intuitive way to study music, and are mainly used for composer classification. However, these audio files contain a huge amount of data represented by the sound signals of the music. This has for consequence a decrease of the interpretability of the data by creating a blur effect on the data. In the aim of studying the impact of each instrument on the accuracy of composer

classification, the use of audio files is therefore not the most appropriate choice. The main reason for this lies in the fact that the distinction between the different instruments in action on an audio file would be an extremely difficult task. Whereas the use of MIDI files is a far better approach. In fact, due to the MIDI files structure, the separation of notes for each instrument is already done, since each track is dedicated to a single instrument . This has for consequences to facilitate the study of each instrument and to avoid wrong interpretations.

For all of these reasons, MIDI files dataset impose themselves as the best choice, given the purpose of this dissertation.

5.1.2. Dataset

As seen previously, in order to study the impact of instruments for composer classification, different rules need to be respected for the choice of a dataset. To summarise, the set must contain several instruments which means that single-instrument datasets like GiantMIDI-Piano [10] cannot be used. Secondly, the dataset should contain multiple composers, in order to assure a good diversity of music styles, so HM107, HM207 and HM285 [6,12] cannot be used either. Finally, the dataset should hold a sufficient number of musical works for each composer, in order to ensure a wide range of music for each one. As seen in [8], the MAESTRO dataset is composed of 13 composers, 505 musics, but only 3 instruments: violin, viola and cello, which does not assure this kind of wide range.

Another solution could have been the generation of a new dataset based, for instance on the KernScores database with its 108,703 MIDI files.

But, our final choice fell on the MusicNet dataset [3]. This dataset contains 303 musical works from 10 composers: Brahms, Dvorak, Cambini, Faure, Haydn, Bach, Mozart, Schubert, Beethoven and Ravel [3]. During the preprocess, 11 instruments was found: violin (40), viola (41), cello (42), acoustic grand piano (0), clarinet (71), flute (73), bassoon (70), french horn (60), contrabass (43), pizzicato strings (45) and oboe (68), as seen in figure 1. This ‘reasonable’ number of instruments is enough to study, while simultaneously simplifying the preprocessing through a manageable list of composers.

5.1.3. Data preprocessing

As previously explained, the data needs to be preprocessed in order to allow the handling of the great number of parameters included and the different lengths of the studied pieces of music. Departing from the standpoint that the chosen dataset is composed of MIDI files, only MIDI-specific preprocessing techniques will be considered. The first one is the conversion to piano-rolls [8,10,15], which is a representation of music in the form of a grid of time and pitch. This technique is frequently used and shows good results and could therefore be interesting to use.

Another possibility is to extract certain global features for each instrument, such as the ratio of notes or the number of notes, which could be interesting for the study of the impact of each instrument on the accuracy of a classifier, but this technique is a source of data feature loss.

The last technique considered is the conversion of all instruments into separate audio files, which would allow the use of audio composer classification models for the impact study. Considering the blur effect of converted audio files, as well the lower accuracy of global features, it seems natural to point out piano-rolls based on the MIDI data as the most appropriate choice. Yet, this technique creates a problem when it comes to the computation of instruments with a low activity in a given music piece. In fact, in this case the instrument will induce a considerable amount of empty space, thus diminishing the accuracy of the process and introducing error within the training process.

Consequently, the piano rolls technique needs to be adapted in order to have a more appropriate representation. The chosen preprocess for this dissertation are therefore onset rolls, which, for each activation of a note, set an array containing the value of the note pressed, the time-lapse which separates it from the previous one and its intensity. This has for consequence to consider the notes as a succession of events rather than a grid of notes, thus removing all the empty moments.

The next problem to be handled is the duration of the music. In fact, the temporal length of the music can be different for each piece, which leads to having a different number of notes for each one of them. For most models it would be difficult to cope with this case. In the reviewed literature, most papers mention that they cut down their samples to specific duration [1,2,6,10,13], such as 30 seconds.

In this dissertation, this is not a problem, since our technique uses a fixed number of notes rather than a duration, avoiding therefore any empty space. This also allows us to study the pieces of music in their entirety and not only through small parts.

5.1.4. Classification model

Due to the chosen preprocessing method, the choice of a classification model was dictated by the need of a technique capable of coping with event sequences. Consequently, the use of a Recurrent Neural Network was the most appropriate, rather than traditional machine learning models, like SVM, KNN, Logistic Regression, Naive Bayes, Decision Tree, and so on.

Some RNN contain convolutional layers or pooling layers, mainly to reduce data complexity. However, in our case, the data is already reduced by the grouping preprocess. The use of convolutional layers and pooling layers is therefore useless.

The solution retained in this project is a RNN using LSTM (Long Short Term Memory) layers, due to the capability of LSTM to use long term and short term memory for classification, which seems the best possible choice for sequences of notes.

5.1.5. Instrument evaluation

In order to evaluate the impact of each instrument on the accuracy of the classification, the composers need to be classified under the aspect of each separate instrument used. For this purpose, a classification model will be created for each instrument. Once all the instrument models are able to classify composers, a submodel will be defined in order to evaluate the impact of each instrument on the classification. This submodel will use the predictions of each instrument as input. In order to avoid overfitting, this submodel will have a part of his training set composed of music which has not been used previously. As the submodel objective is to identify the interest of the instrument for the classification of each composer, the selection of the submodel will focus on the interpretability. In order to do so, the decision tree is the most appropriate, as it reproduces the human scheme of classification.

5.2. Description of the solution

In order to train the instrument networks and the submodel, the dataset will be divided into multiple subsets. For the purpose of maintaining the same proportion for each composer, a first subset composed of only 10 music files per composer will be used for the training of the instrument networks. During the reading of the files in the first subset, each MIDI file track will be studied to find the active instrument. After that, all the notes played on the tracks will be grouped by sequences of 20 notes. Each one of these groups will be labelled by music composers and will be attributed to a specific instrument's table. This table will contain the groups of notes played by these instruments for all the music from the current subset. After this preprocess, the instruments table will be like shown in figure 3. Also this preprocess will bring to evidence that some instruments are used by a small number of composers, as shown in figure 4.

Instrument	Group of notes by instruments
violin (40)	35701
viola (41)	21724
cello (42)	22777
acoustic grand piano (0)	30205
clarinet (71)	8001
instrument not found (-1)	5954
flute (73)	4108
basson (70)	6082
french horn (60)	4624
contrabass (43)	2400
pizzicato strings (45)	608
oboe (68)	4316

Figure 3: Group of notes by instruments after the preprocessing

Instrument	Brahms	Dvorak	Faure	Haydn	Bach	Mozart	Schubert	Beethoven	Ravel	Cambini
violin (40)	x	x	x	x	x	x	x	x	x	
viola (41)	x	x	x	x		x	x	x	x	
cello (42)	x	x	x	x		x	x	x	x	
acoustic grand piano (0)	x		x			x	x	x		
clarinet (71)	x					x				x
instrument not found (-1)	x				x					
flute (73)	x									x
basson (70)	x					x				x
french horn (60)	x					x				x
contrabass (43)	x						x			
pizzicato strings (45)		x						x		
oboe (68)										x

Figure 4: Table of Instruments by composer in the MusicNet Dataset

Once all the music files have been preprocessed to generate the instruments table, the instrument networks need to be defined. For this purpose, a network will be created for each instrument, dedicated to learning how to classify the groups of notes by composers. In the case of the MusicNet dataset, 11 instruments were found and the denomination "a negative instrument" is created for non-identified instruments. Consequently, 12 networks are created, each one of which is composed of 2 *LSTM layers* with 128 neurons, each one followed by a *Dropout layer* with a 0.2 rate. After that, 3 *Dense layers* are added with respectively 128, 32 and 10 neurons. The first two of these Dense layers will use a *sigmoidal activation function* and the last one a *softmax activation function*. The optimizer used is *Adam* and the loss function is *categorical_crossentropy*. The training will be made on 10 epochs with a batch size of 128. In order to do so the instruments table is divided into 3 subsets: 67.5% for the training set, 7.5% for the validation set and 25% for the test set. During the training, the validation set will be used to avoid overfitting.

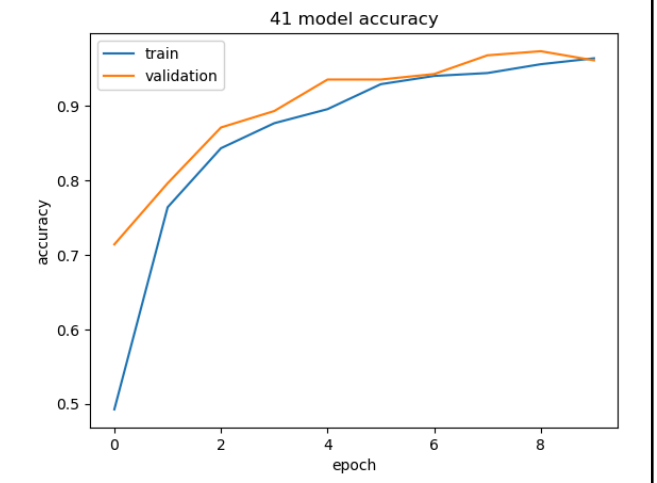

Figure 5: Training history of the viola neural network

All the instrument networks showed a high accuracy for the training set and the validation set, as shown in figure 5 for the viola network. After the training, the instruments networks were evaluated on their testing set, which showed an accuracy between 94.1% and 100%, as shown in figure 6.

Then, after all the instrument networks have been trained, the submodel is created. This submodel, which consists of a decision tree, has 12 inputs, one for each instrument, and 10 outputs, one for each composer. In order to train this model, a new training set needs to be created, this one will be composed of the 10 previously used music files and 5 new music files for each composer. These new music files are added in order to introduce cases where instrument models would not have the same result. After the reading of the music files, each music will be again preprocessed separately with the same techniques used previously.

Once the preprocess is done, the submodel will be trained. In order to do so, for each music, the instrument networks will be used to predict the composer with all groups of notes of the instruments found previously as input. As obviously not all of the 12 instruments are used by every composer, the prediction of unused instruments will be set to -1. After that, all the predictions will be added in an array for each music. These arrays will be used as input for the submodel, and the composer of each music will be used as the music's label. Once all the music files are treated, the decision tree will be trained to recognize each composer using all the prediction arrays. After the training, the submodel showed an accuracy of 98% for the training set. This decision tree created by this training is shown in figure 7.

Instrument	Test accuracy
violin (40)	95.7%
viola (41)	95.5%
cello (42)	94.1%
acoustic grand piano (0)	99.3%
clarinet (71)	100%
instrument not found (-1)	100%
flute (73)	97.7%
basson (70)	95.4%
french horn (60)	100%
contrabass (43)	98.0%
pizzicato strings (45)	100%
oboe (68)	97.7%

Figure 6: Accuracy of all instrument models measure on the testing set

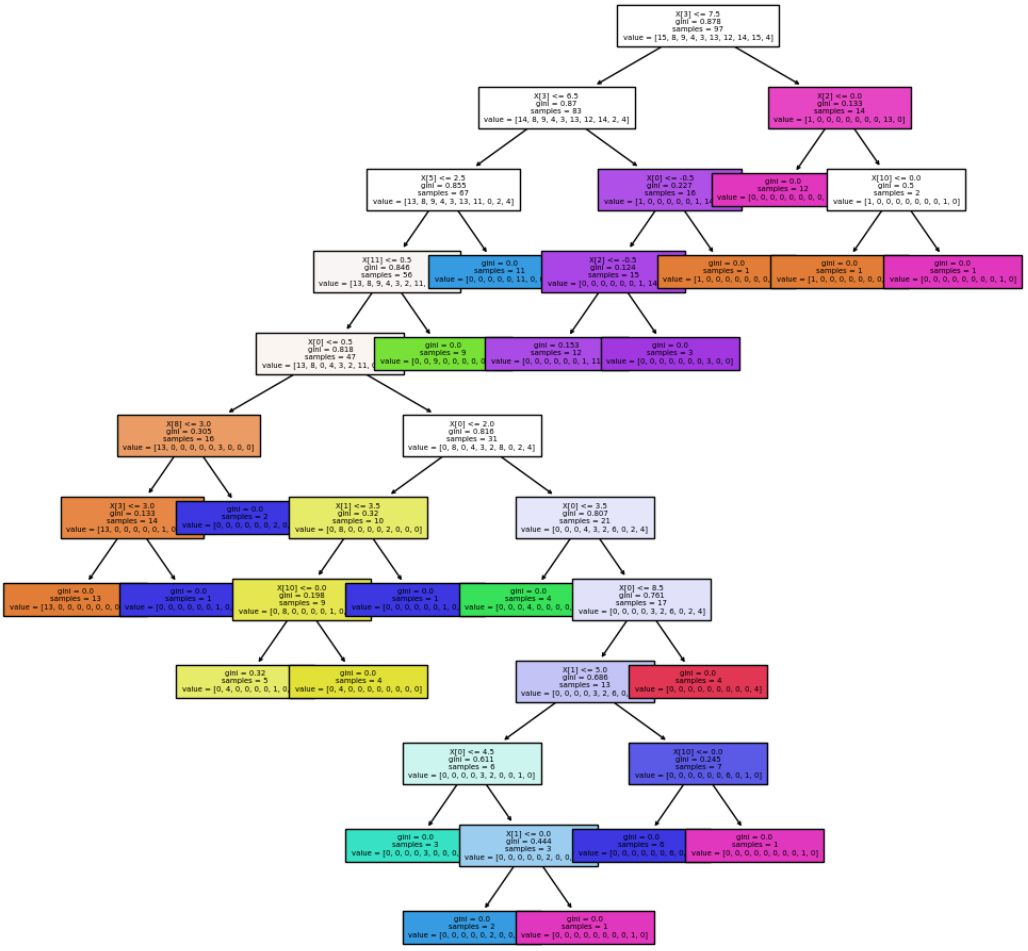

Figure 7: The Decision tree structure generated by the submodel

6 ANALYSIS & EVALUATION

6.1. Result analysis

The first element of the result to analyse is the accuracy of the instrument networks. As shown in figure 6, each instrument network has an accuracy between 94.1% and 100%, which explains that the instrument networks are actually efficient for the classification of composers using unique instruments. In parallel to this, validation sets are used to detect overfitting (Figure 5 shows the viola network validation set for example) and, being close to the accuracy of the training set, prove that the instrument networks are

not overfitted. It is important to state that the validation sets for the other instruments showed equivalent results.

As each of the instrument networks has a high accuracy, one can suppose that all of them are able to identify composers. However, what if two or more of them show contradictory results? The answer to this question implies further analysis of the submodel.

The analysis of the decision tree structure in figure 7 allows multiple assumptions: First, the composers are mainly isolated one by one using one main instrument. In fact, the following isolations are observed:

- The acoustic grand piano model's prediction withdraws 13/15 of Beethoven's music in a first step, before withdrawing 14/14 of Schubert's music in a second step
- The unknown instrument model's prediction withdraws 11/13 of Bach's music
- The oboe model's prediction withdraws 9/9 of Cambini's music
- The violin model's prediction withdraws 13/15 of Brahms's music, 8/8 of Dvorak's music, 4/4 of Faure's music, 4/4 of Ravel's music
- The viola model's prediction withdraws 5/5 of Haydn's music, 6/12 of Mozart's music

This has multiple consequences:

1. Some instruments are more important for the classification of composers, only 5 of 11 instruments are used to isolate all the composers: acoustic grand piano, unknown instrument, oboe, violin and viola.
2. All the composers are not withdrawn by the same instruments, which shows the need of multiple instruments to classify all the composers. In other words, composers cannot be classified using only one instrument.

Other assumptions can be made. Of the 93 music files used for the training of the submodel, 12 were withdrawn with music of the wrong composer. For example, one of Brahms's music was confused with Beethoven's music. In order to make up for confusion, the absence of cello and pizzicato strings is used to find the right composer. This shows once more, the need for the use of multiple instruments to isolate music from wrong

composers. It also shows that the absence or the presence of an instrument can be used as an element of identification for a composer.

Finally, some instruments turn out to be less meaningful for the classification. This is due to the fact that the predictions based on them were not able to isolate any composer or music. This is true for clarinet, flute, bassoon and contrabass which are not used by the decision tree.

6.2. Limitations

All the results shown previously seem relevant, but some limitations cannot be ignored.

First, the chosen dataset does not contain enough music. In fact, many of the composers are under-represented in order to guarantee a sufficient amount of training. This is particularly true for 5 composers: Cambini, Dvorak, Faure, Hadyn and Ravel, who do not provide the 10 music files needed for the model training of each instrument.

This has for consequence to reduce the accuracy of the instrument models for these composers. Also, for the training of the submodel, 5 new music files were needed to introduce divergence between instrument models' predictions. Nonetheless, these 5 composers will still lack the requested number of files, and consequently, will totally under-perform when it comes to producing divergence between instrument models. This creates an overfitting of the submodel for these composers, and consequently, errors in the interpretation of the submodel by submitting wrong hypotheses.

Another problem with the handling of the dataset is the testing set elaboration. As the testing set is created with notes from music files used for the training of the instrument models, the accuracy of the testing set is not a good indicator of the performance of the instrument models. In fact, the testing set can be composed of notes already seen by the instrument models. This results in an overestimation of the accuracy of the instrument models.

Yet, another problem can be found in the data preprocessing. As previously mentioned, some instruments are only active for brief moments during the music and thus create a large amount of empty space. The technique used in this dissertation to handle these empty spaces is responsible for the loss of a certain amount of information, by

blocking the management of polyphonic music. It seems therefore interesting to look for another method to handle this type of instrument.

Finally, the management of the unknown instrument remains an important problem. In fact, the label “unknown instrument” is used to support midi file tracks which do not have any specified instrument. This has for consequence to introduce a bias in the submodel, as the unknown instruments are only observed for two composers. Even though this can be used to isolate these two composers, it introduces an error in the submodel. As shown previously, unknown instrument model's predictions were used to withdraw Bach's music. This error blocks an efficient classification of Bach's music by the submodel.

CONCLUSION

In conclusion, several assumptions can be made from the results. As supposed in the introduction, the instruments have an impact on the classification of composers. In fact, the instruments' models have a high accuracy, which shows that they are efficient for the classification process. But all of them do not have the same impact. In fact, the decision tree shows that some instruments are more important than others in the final classification: only 5 of 11 instruments were used by the submodel to separate each composer. However, as multiple instruments are needed to separate all the composers, this indicates that the instruments do not have the same effect on the classification of each composer. For example, it is better to study Beethoven by the piano, while it is wiser to study Dvorak and Brahms by the violin, for their classification. Also, some instruments are not useful in the classification of composers, such as clarinet and flute, which are not used by the decision tree. On the other hand, some instruments with less impact on the classification of composers can be used to separate misclassified music, and certain instruments can be used to classify composers by their presence or absence. For instance, the absence of cello and pizzicato strings are used to isolate Brahms from Beethoven. Also, the presence of oboe was used to separate Cambini from the other composers. This shows that the composer classification can be made using single instruments, but to classify all the music, multiple instruments are needed.

To improve this study, multiple adjustments can be proposed, such as using a larger dataset, using a polyphonic preprocess for the instruments, and a better management of the unknown instruments. Also, to go further, it could be interesting to implement a classification model using instruments as features in order to compare its results with the results of previous classification models. Finally, it could be interesting to explore the impact of instruments in other fields of music classification, such as genre classification.

REFERENCES

[1] Cataltepe C.Z., Yaslan Y., and Sonmez A. (2007), "Music genre classification using MIDI and audio features," EURASIP Journal on Advances in Signal Processing, vol. 2007, pp. 1-8, 2007.

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

[3] Gupta S. (2020), “MusicNet Dataset”, online:https://www.kaggle.com/datasets/imsparsh/musicnet-dataset

[4] Herremans D., Martens D., Sörensen K. (2015), “Composer classification models for music-theory building”. Computational Music Analysis 2016: 369-392

[5] Herrera P, Amatriain X, Batlle E, Serra X. (2000) “Towards instrument segmentation for music content description a critical review of instrument classification techniques”. In: Proceedings of the 1st International Symposium on Music Information Retrieval; 2000 Oct 23-25; Plymouth, Massachusetts, USA. [Plymouth]: ISMIR; 2000. 9 p.

[6] Kempfert K.C. & Wong S.W.K. (2020) “Where does Haydn end and Mozart begin? Composer classification of string quartets”, Journal of New Music Research, 49:5, 457-476, DOI: 10.1080/09298215.2020.1814822

[7] Kher R.N. (2022) “Music composer recognition from MIDI representation using deep learning and N-gram based methods”, Dalhousie University, http://hdl.handle.net/10222/82031

[8] Kim S., Lee H., Park S., Lee L., Bytedance K., and Lab A. (2020), “Deep Composer Classification Using Symbolic Representation.” Available: https://arxiv.org/pdf/2010.00823.pdf

[9] McGill University, “Standard MIDI-File Format Spec. 1.1, updated”, Available:http://www.music.mcgill.ca/~ich/classes/mumt306/StandardMIDIfileformat.html

[10] Kong Q., Choi K., and Wang Y. (2020), “Large-Scale MIDI-based Composer Classification”, arXiv:2010.14805 [cs, eess], Available: https://arxiv.org/abs/2010.14805

[11] Micchi G. (2018), “A neural network for composer classification” Available: https://hal.science/hal-01879276/

[12] Nanni L., Costa Y.M.G., Lumini A., Kim M.O., Baek SR. (2016), “Combining visual and acoustic features for music genre classification”, in Expert Systems with Applications, Volume 45, 2016, Pages 108 - 117,j.eswa.2015.09.018

[13] Velarde G., Cancino Chacón C., Meredith D., Weyde T., and Grachten M. (2018), “Convolution-based classification of audio and symbolic representations of music,” Journal of New MusicResearch, vol. 47, no. 3, pp. 191–205, May 2018, doi: https://doi.org/10.1080/09298215.2018.1458885.

[14] Verma, H., & Thickstun, J. (2019). “Convolutional composer classification”. *arXiv preprint arXiv:1911.11737*.

[15] Wang, A. (2003) “An industrial strength audio search algorithm.” *Ismir*. Vol. 2003.

[16] Yang, D., and Tsai, T. (2021), “Composer Classification With Cross-Modal Transfer Learning and Musically-Informed Augmentation.” *ISMIR*.

[17] Yang D., Ji K., and Tsai T.J. (2021), “A deeper look at sheet music composer classification using self-supervised pretraining” . Appl.Sci.2021, 11, 1387. https://doi.org/10.3390/app11041387

[18] Zhang J. (2021), “Music Feature Extraction and Classification Algorithm Based on Deep Learning,” Scientific Programming, vol. 2021, pp.1–9, May 2021, doi: https://doi.org/10.1155/2021/1651560