Classifying Classical Piano Music Into Time Period Using Machine Learning

Bas van 't Spijker
University of Twente
P.O. Box 217, 7500AE Enschede
The Netherlands
b.c.vantspijker@student.utwente.nl

ABSTRACT

The combination of Music and Machine Learning is a popular topic. Much research has been done attempting to classify pop music in different genres, but classical music has been looked at less, mainly because of the lack of available datasets. In this paper, classical piano music dating from the 17th to the 20th century is classified into the category period of composition. For this the MAESTRO dataset from Magenta is used, consisting of approximately 200 hours of piano performance recordings. Mel-Frequency Cepstral Coefficients and Chromagrams are evaluated as features, and they are tested with Convolutional Neural Networks, Long Short-Term Memory (a type of Recurrent Neural Networks) and Support Vector Machines as Machine Learning algorithms.

Keywords

Machine Learning, Music Classifying, Classical Music, Piano, Period

1. INTRODUCTION

Within the field of music there exist many genres, such as pop, rock, metal and classical. Each genre has its own musical, tonal and harmonious properties, and different subgenres. Classifying music can be useful for a multitude of reasons: for indexing, or for research purposes on for instance its popularity, or its cognitive effects [36]. While these genres are defined and identified by humans, computers can classify music pieces using Machine Learning, usually performing just as well as a human would; this already has been done quite often [19, 1, 33, 35, 25]. It is usually referred to as (Automatic) Music Information Retrieval, or MIR for short. Some types of music are investigated more than others, as there has been done much research on modern, popular music, but less so the classical variant. Reasons for this might revolve around the music's impopularity itself, diminished commercial interest, or a lack of widely available datasets.

Classical music can be placed in different subcategories, based on time of composition. In this paper, I will attempt to classify classical piano music dating from the 17th to the 20th century into their respective time periods 'Baroque', 'Classical', 'Romantic' and 'Modern'. This goal is reflected in the following research questions:

- **RQ1:** What makes piano music unique in terms of:
  - **RQ1.1:** Music as a time series
  - **RQ1.2:** Feature selection & Machine learning
- **RQ2:** Can we explore and extend previously used machine learning methods in order to obtain accurate classification of piano music based on into time period of composition

First, RQ1 is defined because classical music can be performed with different musical instrument setups (piano, quartet, orchestra, vocal) and the dataset used in this paper solely contains piano music recordings. It will then be useful to go over the characteristics and methods that are important to this specific classification problem.

Music Information Retrieval is usually performed in two parts: first, a database is selected, and features are extracted from the data. Second, one or more classification algorithms are applied and the data is classified. The choice for features and algorithm heavily depends on the source material and the type of classes. In this paper RQ2 will be answered by selecting and testing two different features and three different Machine Learning methods based on previous research on Music Information Retrieval, and the best performing combination will be identified.

1.1 Main Contribution

The main contribution of this work is the comparison of the performance between two popular Extraction Features in music: Mel-Frequency Cepstrum Coefficients and Chromagrams, and three Machine Learning algorithms: Convolutional Neural Networks, Long-Short Term Memory, and Support Vector Machines.

2. RELATED WORK

Automatic music classification is a type of Music Information Retrieval (MIR), a term first mentioned by Kassler (1966) [18], naming the programming language he created for extracting information from audio files. Since then, MIR has split into many different branches. Fu et al. [10] defines 5 of them directly related to music classification in his 2011 survey: Genre and Mood Classification, Artist Identification, Instrument Recognition, and Music Annotation. The research presented in this paper is complementary with their work and focuses on Period Classification, which is not part of that list, but comes closest to Artist (composer) Identification. I will shortly go over a history of research both Composer Identification and Instrument Classification to give an overview of the progress made over the years. I chose the latter because it is one

32^{th} Twente Student Conference on IT Febr. 2^{nd}, 2020, Enschede, The Netherlands.

of the most straight-forward of MIR tasks and completely audio-based, while the former introduces complications as described in Section 2.2. Still, most attention will be paid to Composer Identification. In 2011, Fu et al [10] described feature types extracted from source data during the Feature Extraction part of classification. They lists low-level Timbre and Temporal features, and mid-level as Pitch, Rhythm and Harmony. Researchers having been tending to use combinations of these features, selecting them based on the type of task. Some develop their own specialized features, often derived from the ones above.

2.1 Instrument Classification

Instruments are most often classified by extracting Timbre and Spectral (a type of Timbre) features. One of the earliest works here is performed by Marques et al [22] in 1999. Eight instruments were classified from 16 CDs, using Mel-Frequency Cepstrum Coefficients (MFCC) and Cepstrum features, with a Support-Vector Machine (SVM) and Gaussian Mixture Model (GMM) tested as classifiers. The best combination was MFCC and SVM, resulting in an accuracy of 70%. This is one of the first papers to use an SVM for MIR.
Eronen [9] classified 29 instruments using different features and a k-NN classifier. The best performing feature was MFCC with an accuracy of 32%.
Eggink and Brown [7] was the first to identify two instruments being played at the same time out of a pool of 5, using Timbre features, a-priori masks for instrument separation, and a GMM classifier, achieving 74% accuracy.
Simmermacher et al. [34] classified 4 instruments using a mixture of features among which "MPEG-7". Testing on a k-NN, Naïve Bayes and SVM classifier, the SVM came out best with 97% accuracy. It is noteworthy to mention that just using MFCC features already gave 93% accuracy. The same researchers published a comprehensive study on features used in Instrument Classification two years later (2008) [5], concluding that MFCC features performed best individually in this task, especially when combined with SVMs. Since then, there have been many more papers on instrument classification, usually achieving similar results. Neural Networks have also been researched as a classifier. For example, in 2018 Chakraborty [2] stated that Convolutional Neural Networks need much more training data to achieve performance similar to the conventional MFCC and SVM.
To conclude, one of the main points that can be deducted from these papers is that the the task of classifying instruments often involved MFCCs and SVMs for the best results, especially before deep learning came around.

2.2 Composer Identification

Classifying composers is the task that comes closest to classifying the period in which a piece was composed, since a period could be considered an aggregate of musical characteristics exhibited by composers during a certain timeframe.
Before I continue, it is important to note here that this task of MIR brings to light an important distinction between two types of classification, namely classifying based on symbolic data, and classifying based on audio source material. Symbolic data usually comes in the form of MIDI files or (digital) sheet transcriptions, of which MIDI files are most commonly used. MIDI is "a data communications protocol that allows for the exchange of information between software and musical equipment through a digital representation of the data that is needed to generate the musical sounds" (Rothstein, 1992) [29]. In other words, it is a format that is easily processed and analyzed by a computer. As a consequence, there has been done a lot of research into classifying composers using MIDI files. I will summarize some of it in the next section. However, for research part of this paper I want to focus on audio source files instead, which pose a different challenge in terms of selecting feature extraction methods.

Pollastri and Simmencelli [28] first classified 4 composers and the Beatles from a collection of 605 MIDI files. They used melodies coded as intervals together with a Hidden Markov Model, achieving 42% accuracy. Interestingly, they also conducted a listening expirement where experts and non-experts were asked to classify the music, observing accuracies of 48% and 24% for each group respectively.
Kranenburg et al. [37] classified 5 composers using a set of self-defined 'style markers', representing mostly low-level features. Using Nearest Neighbour they achieved accuracies between 73 and 91% depending on the classification problem.
Wolkowicz et al. [45] classified 5 composers as well, on a set 251 piano MIDI files using a novel N-Grams method describing melodic and rhythmic features. A similarity measure formula achieved 84% accuracy.
Hillewaere et al. [14] created a system that could differentiate between string quartets of Haydn and Mozart. They used 3 subsets of features and statistics derived from Alicante, Jesser and Mcay, and combined them with an SVM classifier. They also implemented the N-Grams model for comparison; both models achieved about 75% accuracy maximum.
Kaliakatsos-Papakostas et al. [17] are one of the first to use Neural Networks in Composer Identification in 2010. They classified 7 composers by calculating Dodecaphonic Trace Vectors from Chroma Profiles, and entered those into a sequence of a Probabilistic Neural Network and a Feedforward Neural Network. The reported accuracy was 48%.
Hontanilla et al. [16] used melody N-Grams on the same dataset as Kranenburg [37], resulting in an average accuracy of 79%.
For a set of 5 composers Wołkowicz et al. [44] calculated N-Grams, and compared the feature vectors using a variety of statistical methods. Accuracies ranged between 70% and 75%.
Herlands et. al [12] classified two composers using the same dataset as Hillewaere [14], this time testing melodic and rhythmic features. Together with an SVM classifier they increased the accuracy to 80%.
SVMs were once again used by Herremans et al. [13], classifying 3 composers by extracting 12 selected interval and pitch features provided by software called 'jSymbolic', getting 80-86% accuracy. Sadeghia et al. [30] then tackled the same problem using the same features but a (Enhanced) Fuzzy Min-Max Neural Network. The accuracy only got up to 70% this time.

In short, when dealing with symbolic data, classification accuracy (for about 5 composers) often goes above 80% when using n-grams, or SVMs with melodic or pitch-features. But while there is plenty of research on symbolic data, there has been little research on audio data. In the next section I will describe a few of the only works that have been done this way.

One of the first attempts classifying raw audio data was done by Zwan et al. in 2011 [46]. They selected music of 60 different composers from 5 different genres, one being Classical. 171 features were extracted, such as MPEG-7 and MFCC means. An SVM classifier achieved 72% accuracy, a Neural Network 70%. It is important to recognize

here that the music selected came from a wide variety of genres with different instrumental occupations. That type of data might contain a lot of timbre information not present in a single genre/instrument dataset.
In a rather striking recent work, Micchi [24] managed to classify 320 pieces of piano music from 6 Classical composers using just Short-Time Fourier Transforms and a Convolutional Neural Network, with the accuracy going as high as 70%.
Lastly, Velarde et al. [38] classified several audio sets, some of which were actually generated from MIDI files, into two composers. They used spectograms and SVM and k-NN techqniques for this. Accuracies reached 72%, but they concluded "in multi-class composer identification, methods specialised for classifying symbolic representations of music are more effective."

To close this section out, I wanted to note that there exists a yearly Music Information Retrieval Evaluation eXchange, or MIREX [6] for short, in which participants compete in several MIR subjects. One of these subjects is Composer Identification, where a dataset of 11 classical composers is provided. However, in the 15 years of its existence, there have only been two attempts at this challenge. Of those just one case from 2015 is still documented. It only reached an accuracy of 35% [31] using a variety of features together with a Neural Network.

2.3 Period Classification

On the specific subject of Period Classification, according to my best knowledge, there has really only been work done by one researcher: Christof Weiß. In 2017 he published a PhD thesis on analyzing Classical Music Audio Recordings [39]. In my opinion, it is by far the most interesting work on the topic yet, containing some novel papers and detailed descriptions of methods and related work performed by other researchers. Besides that, he also published some papers over the span of a few years, which I will shortly go over. In all of his works, he used a handcrafted dataset containing 1600 audio recordings of classical music, half for piano, half for orchestra; 200 of both for each Period. The dataset does not contain works from transitional composers that lived and created works in two Periods.
In [41], Weiß proposed a method to extract audio features tailored to the task of Period Classification. First he calculated Chroma Vectors using a Nonnegative Least Squares formula. From those vectors, 'Pitch Classes' were extracted. When using a SVM method, classification accuracy reached 86% for piano.
In [40], he attempted a second method in the same setup, extracting Complexity Features instead of Pitch Classes from the Chroma Vectors. Accuracy for piano was lower this time at 64%. Besides this, spectral features (such as MFCCs) were also tried. They only provided 29% accuracy on piano audio, barely above random guessing.
A third paper [43] researched key detection features with a Random Forest classifier. Accuracies went above 70%. Moreover, inserting simple raw Chromagrams as features still gave 53% accuracy.
Lastly, in his most recent paper on the topic [42], he investigated the historical lines drawn by musicologists in a computational way, "using automated feature aggregation, tonal complexity as well as the ratio of plagal and authentic transitions arised as style markers in an unsupervised fashion". He concluded that the unsupervised statistics, interestingly, mostly agreed with musicologists.

2.4 Remarks

To summarize, MIR has become a popular field of research over the past two decades. While categories such as Instrument Classification and Composer Identification have continuously been researched and improved upon, accuracies still rarely go above 90%, which leaves room for improvement. And while SVMs have been used many times, there is still a lot to explore regarding Neural Networks. Classical Composer Identification has mostly been researched in the form of symbolic data, with audio data largely left untouched.
Closing this section, I want to mention that Sharma et al [32] recently published a very useful survey, describing all popular features used in the Feature Extraction part of Music Classification up until now.

3. METHODOLOGIES

The experiments are conducted in python using keras [3] and scikit-learn [27]. The code can be found in on github[1].

3.1 Dataset

For the data I use the MAESTRO dataset [11], a relatively new dataset released by Google in October 2018 as part of their Magenta project. It contains "over 200 hours of paired audio (WAV) and MIDI recordings from ten years of International Piano-e-Competition", divided into 1282 different pieces from 37 composers, although not every composer is equally well-represented. Figure 5 in the Appendix shows eight example boxplots and Table 1 gives some statistical information about the data. Figure 6 in the Appendix shows the signal and Fourier transforms for the same example music pieces as in Figure 1. Since these statistics do not show any interesting patterns they will not be used in the rest of the paper.

	Mean (μ)	**Standard dev.** (Σ)
Baroque	-2.97e-05	0.050336
Clasical	-3.18e-05	0.049551
Romantic	-2.44e-05	0.053910
Modern	-1.93e-05	0.046481

Table 1: Statistical details of the MAESTRO dataset: the average mean and standard devation of the discrete audio signal per period

3.2 Feature Selection and Extraction

I opted to test two commonly-used features: Mel-Frequency Cepstrum Coefficients (MFCCs) and Chroma Features. MFCCs contain much information about timbre. I expect this to be less useful since the data used solely contains piano music, which does not differ much in timbre from piece to piece. Chroma Features on the other hand contain mainly pitch information, which is the main component that differentiates piano pieces and their styles from each other. For this research, I sampled the first 30 seconds of each piece at a sampling rate of 8000 Hz, and extracted the MFCCs and Chromagrams.

3.2.1 Mel-Frequency Cepstrum Coefficients

MFCCs were first introduced by Davis and Mermelstein in 1980 [4]. They have been one of the most popular features in this research field for years now; they are used for every type of classification (see also Section 2). They are a short-term representation of the power spectrum of a signal, with frequency weights adjusted to the range of frequencies humans can perceive. MFCCs are calculated in five

[1]https://github.com/vantspijker/periodclassification

steps [20], using the python "python_speech_features" [21] library.

1. *Framing the signal*

The signals used in the MAESTRO dataset are sampled at 44100Hz. This is more than I need for accurate feature extraction, so I subsampled to 8000Hz. The signal is then framed with framing windows of 0.25 ms, creating 0.025*16000 = 200 samples per frame. Frame step is 10ms (80 samples), allowing some overlap in the frames.

2. *Calculating the Periodogram estimate*

First the Discrete Fourier Transform (DFT) is calculated using the following formula:

$$S_i(k) = \sum_{n=1}^{N} s_i(n)h(n)e^{\frac{-j2\pi kn}{N}} \quad 1 \leq k \leq K \qquad (1)$$

where $s_i(n)$ denotes the time domain signal on the ith frame number, n ranges over the 200 samples, $h(n)$ is an N sample long hamming window, and K is the length of the DFT, which I set to 256.
The Periodogram estimate of the power spectrum spectrum is then calculated by:

$$P_i(k) = \frac{1}{N}|S_i(k)|^2 \qquad (2)$$

3. *Computing the Mel-spaced filterbank*

This step is performed so simulate human hearing, which is more sensitive to changes in the lower frequencies than in the higher ones. The filterbank is "a set of 20-40 (26 is standard) triangular filters that we apply to the periodogram power spectral estimate from step 2. To calculate filterbank energies we multiply each filterbank with the power spectrum, then add up the coefficents. Once this is performed we are left with 26 numbers that give us an indication of how much energy was in each filterbank." [20] Figure 1 displays a simple visual example of a Mel-spaced filterbank.

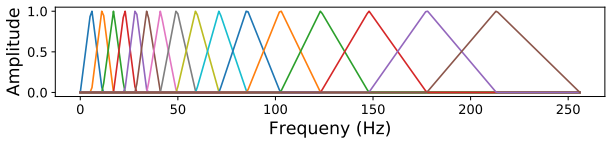


Figure 1: Example of a Mel-spaced filterbank

4. *Taking the log of each of the 26 energies from the previous step*

This is also done in conjunction with the theory of human hearing: for the human ear to perceive something as twice as loud, the amplitude has to be about eight times as large.

5. *Taking the Discrete Cosine Transform (DCT) of the filterbank*

This step decorrelates the 26 coefficients. Only the lower 13 of the 26 MFCCs that are calculated are kept. Using more than 13 coefficients usually does not increase classifier accuracy in Machine Learning.
After performing these steps, I end up with a MFCC matrix of 13 by 2999.

3.2.2 *Chromagrams*

"Chroma features are an interesting and powerful representation for music audio in which the entire spectrum is projected onto 12 bins representing the 12 distinct semitones (or chroma) of the musical octave."[8] The chromagrams in this research paper are calculated according to the following steps using the python "librosa.feature.chroma_stft" library [23]:

1. *Framing the signal*

The signal is framed with a framing window of 2048 samples, and a frame step of 512 samples. This creates $\frac{8000*30}{512} \approx 469$ frames (the last frame is padded with zeros).

2. *Calculating the Discrete Fourier Transform*

The DFT is calculated the same way as in (1).

3. *Mapping to tone bins*

For every frame, the Fourier Transform values are then mapped to 12 bins, each representing one tone in the musical 12-tone scale. Each bin represents all octaves (doubling in frequencies).
These three steps output a Chroma matrix of 12 by 469.

3.3 Machine Learning algorithms

Since MFCCs and Chromagrams are both image representations of the signals, I suspect Convolutional Neural Networks might use the extracted features well. Furthermore, because more modern music pieces change notes and chords over time more than the older, traditional ones, I will also apply a Long Short Term Memory algorithm (LSTM), which is a form of a Recurrent Neural Network (RNN) first introduced by Hochreiter and Schmidhuber in 1997 [15]. Lastly, one of the most popular methods in the field of music classification is Support Vector Machine learning. This is a much simpler form of Machine Learning (SVM), as it only attempts to separate groups of classes in their regions based on coordinates (value vectors) it is given. The SVM here is used with a radial basis kernel.

3.3.1 *Convolutional Neural Networks*

The Convolutional Neural Network used in this research is built from different layers: first a set convolution and activation layers, then a pooling layer, followed by a dropout layer, and two dense layers. Because the Chromagrams were of a smaller size than the MFCC spectograms, larger networks could be built for the first. In the convolution layers, a 3x3 convolution window slides through the Chroma or MFCC matrix with step size (1,1) and calculates a filter dot product to identify relevant features. The amount of filters increases in each new layer, so that more detailed features are identified along the way. After each convolution layer, an activation 'Leaky ReLu' layer follows, that is meant to only keep active neuron values. The formula for this Leaky ReLu implementation is given by:

$$f(x) = \begin{cases} x & \text{if } x > 0, \\ 0.3x & \text{otherwise} \end{cases}$$

At the end of the convolution layers, a max pooling layer is added to compress data size. A 2x2 window with step size 2 slides over the feature maps and keeps only the highest value, reducing the total data size by a factor of four.

This is followed by a dropout layer, where half of the neurons are randomly dropped. This layer is meant to reduce overfitting and force the network to recogize probabilistic patterns. After this, the output is flattened to one dimension and two consecutive "Dense" layers are added: one to be trained to recognize patterns, and a final one to determine the most likely class (using a "softmax") function. Figure 2 provides a visualization of the layer setup. The differences between setups for the MFCC and the Chroma data can be found in Table 2.

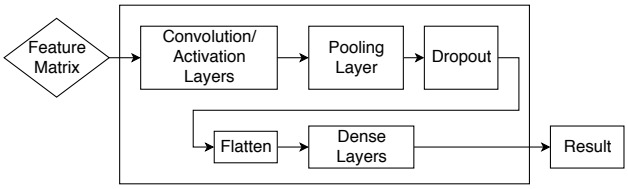


Figure 2: Convolutional Neural Network Setup

	MFCC	Chroma
Convolution Layers	4	6
Convolution Filter Maximum	128	512
Dense Neurons	64	128

Table 2: Differences between the Convolutional Neural Networks used for classifying. Since Chromagrams are less dense than MFCC spectograms, a larger neural network could be used.

3.3.2 *Long Term Short Term Memory*

The LSTM implementation is like the CNN, with a few key differences: instead of convolution layers, there are two consecutive LSTM layers of size 128. This is followed by the dropout layer (there is no pooling layer), a 64-sized dense layer (Time-distributed), an activation layer, a flatten, and finally one more dense layer of size four.

3.4 Metrics assessing the classification accuracy

Classification based on time period is a multi-class classification problem by definition. Measuring its algorithm accuracy can be done in multiple ways. Confusion matrices on the test split of the data will be showed for illustration, and I will calculate the accuracy and logarithmic loss for each combination of feature and algorithm. The formula for accuracy is given below:

$$Accuracy = \frac{\text{True positives} + \text{True negatives}}{\text{Total number of predictions}}$$

The logarithmic loss is calculated using the class probabilities the machine assigns to every single case. It especially punishes heavy confidence in false positive and false negative cases. The close to zero the score is, the higher the accuracy.

Because the dataset is heavily imbalanced (see section 4.1), I also calculated the weighted accuracy, which takes this imbalance into account. It is calculated here by taking the average of the recall for evrey class. Given a confusion matrix M, the recall of class x can be calculated by

$$Recall(x) = \frac{M_{ii}}{\sum_j M_{ij}}$$

where i is the row of x and j ranges over the columns of M. The weighted accuracy W of matrix M can then be calculated by taking the average of the recall for every class X:

$$W(M) = \frac{\sum_x^X Recall(x)}{X}$$

where X is the total number of classes.

4. RESULTS

4.1 Pre-processing

Since the MAESTRO dataset does not contain information about year or period of composition by itself, I added this manually. Music historians generally identify four distinct periods when starting from the 1600s, with music pieces from each of them having uniquely identifiable characteristics. As a rule of thumb, the later the century, the more dissonant and less harmonious compositions become, as composers experiment with new chords and scales. There is no hard line between the periods and the definitions can vary slightly from source to source. I used [26] here, which keeps track of the lifespan of most known classical composers. The four periods then are:

1. 1600-1750: Baroque
2. 1750-1820: Clasical
3. 1820-1910: Romantic
4. 1910-present: Modern

I categorized every piece in the dataset into its period based on the canonically established year of its composition. This resulted in a division of periods as shown in Figure 3. Evidently, the Romantic period is rather overrepresented. During the Machine Learning part this aspect proved to be difficult to completely counterbalance, so I decided to reduce the amount of Romantic Pieces by two thirds to compensate. The distribution after reduction is given in table 3. It also shows the split of data: Train was used to train the networks, Validation was used to test accuracy after each epoch, and Test was used for a final accuracy measure.

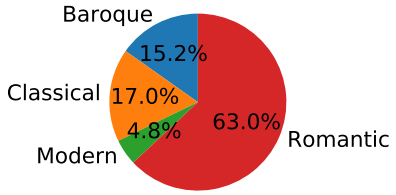


Figure 3: Database piece distribution.

	Train	**Validation**	**Test**
Baroque	151	14	30
Clasical	160	20	38
Romantic	211	38	20
Modern	46	4	11
Total	522	76	99

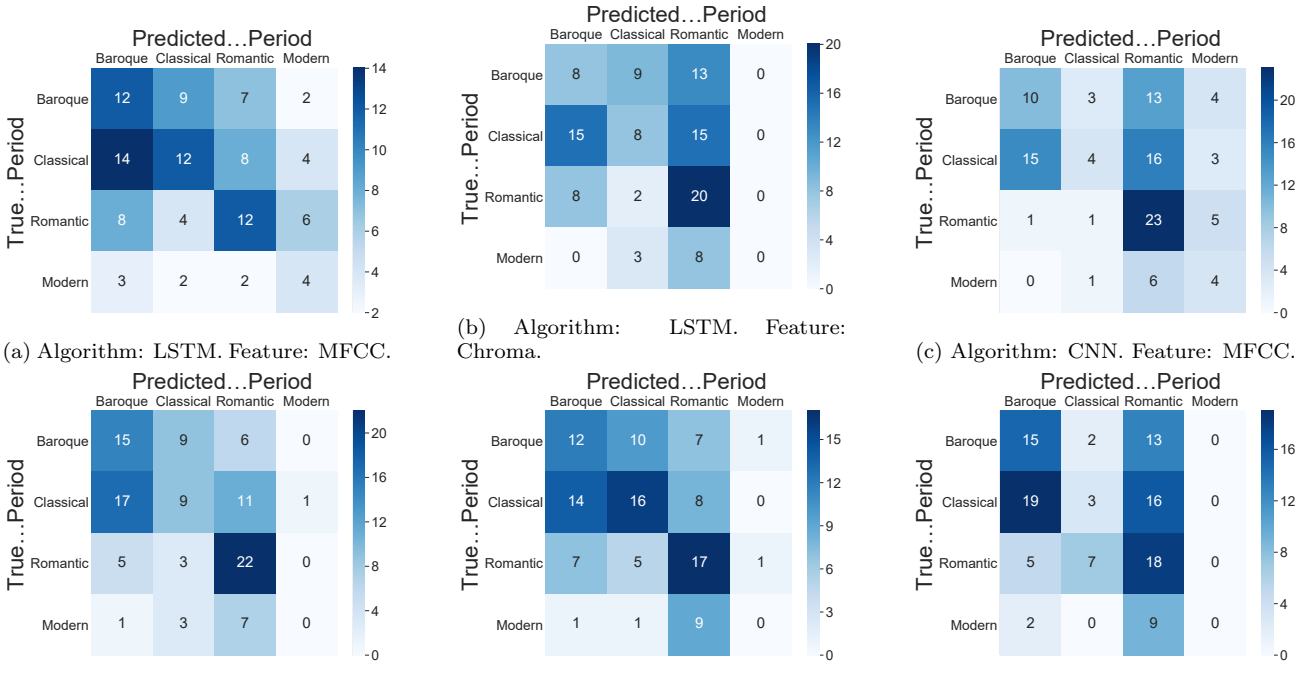
Table 3: Distribution of the split of data across the four periods in the MAESTRO dataset after reducing the number of Romantic pieces by two thirds. The data is split into Train, Validation and Test according to the labels the database provided.

Algorithm	Subset	Metric	MFCC	Chroma
LSTM	Validation	Accuracy	0.3	0.49
		Loss	2.89	1.14
	Test	Accuracy	0.37	0.33
		Loss	3.04	1.4
		Accuracy(W)	0.37	0.29
CNN	Validation	Accuracy	0.5	0.58
		Loss	1.43	1.33
	Test	Accuracy	0.38	0.42
		Loss	1.5	2.19
		Accuracy(W)	**0.39**	**0.37**
SVM	Test	Accuracy	0.41	0.33
		Accuracy(W)	0.35	0.29

Table 4: Numerical Summary of the classification results. The accuracy and loss are displayed for the validation and test set of each algorithm. Accuracy(W) is the most important statistic and denotes the Weighted Accuracy. Since SVMs are only split in a train and test set, no validation accuracy is given.

4.2 Machine Learning

The numerical results of the tests can be found in table 4. The numbers for the best performing performing algorithm (CNN) are in bold. The confusion matrices are shown in Figure 4. Figure 7 in the Appendix displays examples of MFCC and Chroma Feature graphs, two for each period, along with their signal transform. The Filterbank values, which are the result of performing every step until the Discrete Cosine Transform while calculating MFCCs (see also Section 3.2.1, step 4), are also shown.
The performance of algorithms differed quite substantially. What table 3 does not display is the time efficiency of the algorithms and features: as explained in section 3, MFCC spectograms are much bigger (about 15 times) and contain more information than Chromagrams. LSTMs are slower than CNNs, so the slowest performing combination was LSTM and MFCCs. The fact that Chroma feature vectors are smaller also means that the CNN could be given an extra layer of neurons, which helped the accuracy.
The most important statistic is the weighted accuracy. The unweighted accuracy gives a warped impression of the models robustness, as only guessing "Romantic" on the validation set would give an accuracy of 50%, and guessing "Classical" on the test set would give an accuracy of 38%. Weighted accuracy gives exactly 25% in these cases.
Judging from the weighted accuracy, the best combination was an MFCC spectogram combined with a Convolutional Neural Network, at 39% accuracy and one of the lowest logarithmic losses. Contrary to what I expected, the Chromagrams perform worse on every classifier compared to MFCC spectograms. This might be because of the larger information density of MFCCs. While Chromagrams clearly do not perform better directly here, there still is potential in them left untapped in this research. For example, Weiß [41] computed more specific pitch-related information based on Chromagrams, which increased accuracy as opposed to using flat Chromagrams as reported in [43].
It appears the LSTM algorithm performed worse than the CNN. Although the LSTM did well on MFFCs in terms of Weighted Accuracy, the logarithmic loss was rather high. The SVM performed the worst. From the confusion matrices in Figure 4 it becomes clear the algorithms have the much trouble correctly identifying Modern pieces. There are two explanations for this: first, the Modern dataset is by far the smallest (see table 3). Second, the modern pieces are not very recent: they are mostly composed in the early 1900s, shortly after the Romantic period ended, half of them written by transitional composers that also wrote music for the Romantic Period (such as Debussy). Furthermore, what the numerical summary cannot represent accurately is the distance of periods next to each other. For example, it is much worse for the machine to classify a Baroque piece as Modern or Romantic than Classical, since the former is more closely related to the Baroque. The confusion matrices give a better idea of this,



(a) Algorithm: LSTM. Feature: MFCC.

(b) Algorithm: LSTM. Feature: Chroma.

(c) Algorithm: CNN. Feature: MFCC.

(d) Algorithm: CNN. Feature: Chroma.

(e) Algorithm: SVM. Feature: MFCC.

(f) Algorithm: SVM. Feature: Chroma.

Figure 4: Confusion Matrices of the test sets for each of the Algorithm and Feature combinations presented in table 3.

but it would be better to devise an alternative metric that takes this Period distance into account.

5. CONCLUSIONS AND FUTURE WORK

In this paper, I attempted to classify audio of classical piano music from a variety of composers into four different Time Periods. I tested two feature extraction methods, MFCCs and Chroma Vectors, on three different classifiers: CNNs, LSTMs and SVMS. The highest test-set accuracy came from the combination of MFCCs Vectors with CNNs. Contrary to what I expected, raw Chroma features do not perform better when using audio source material from a single instrument. While Convolutional Neural Network classifiers performed best here, the accuracy numbers attained in this work leave much room for improvement.

Part of the accuracy deficiency could be attributed to the large data imbalance in the MAESTRO dataset. Future work could correct this during the pre-processing stage in several ways: the small Modern subset could be discarded (so that only 3 Periods can be classified), or transitional composers could be left out to establish clearer distinction lines between Periods. The data-gap between Periods could also be corrected by taking more than one 30-second sample per piece in the underrepresented Periods, although this might introduce bias into the system.

Furthermore, the parameters set for the Neural Networks have been chosen rather arbitralily as of now; it would be interesting to research if there exist more optimal values that increase accuracy when applied.

Although two of the more commonly-used features have been applied in this research, there are still many more to be compared. Sub-features could also be extracted from MFCCs and Chroma Vectors, reducing data size and specializing the information input for the task. Some research papers reviewed in Related Work section describe the combining of multiple features into a Feature Vector, something also not yet attempted here. Combining features usually, though not always, results in higher accuracy.

Lastly, the MAESTRO dataset contains both MIDI and WAV data for every piano piece. This opens up opportunities for directly comparing the performance of both Period and Composer classification between audio and symbolic representations of music.

6. ACKNOWLEDGEMENTS

I would like to thank dr. E. Mocanu very much for all her help and feedback (and patience with me) while supervising the writing of this paper.

7. REFERENCES

[1] A. C. Bickerstaffe and E. Makalic. Mml classification of music genres. In T. T. D. Gedeon and L. C. C. Fung, editors, *AI 2003: Advances in Artificial Intelligence*, pages 1063–1071, Berlin, Heidelberg, 2003. Springer Berlin Heidelberg.

[2] S. S. Chakraborty and R. Parekh. Improved musical instrument classification using cepstral coefficients and neural networks. In J. K. Mandal, S. Mukhopadhyay, P. Dutta, and K. Dasgupta, editors, *Methodologies and Application Issues of Contemporary Computing Framework*, pages 123–138, Singapore, 2018. Springer Singapore.

[3] F. Chollet et al. Keras. `https://keras.io`, 2015.

[4] S. Davis and P. Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 28(4):357–366, August 1980.

[5] J. Deng, C. Simmermacher, and S. Cranefield. A study on feature analysis for musical instrument classification. *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society*, 38:429–38, 05 2008.

[6] J. S. Downie. Mirex home. https://www.music-ir.org/mirex/wiki/MIREX_HOME, 2019. (Accessed on 01/26/2020).

[7] J. Eggink and G. J. Brown. A missing feature approach to instrument identification in polyphonic music. In *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, volume 5, pages 553–556, 2003.

[8] D. P. Ellis. Chroma feature analysis and synthesis. `http://labrosa.ee.columbia.edu/matlab/chroma-ansyn/`, 04 2007. (Accessed on 01/26/2020).

[9] A. Eronen. Comparison of features for musical instrument recognition. In *Proceedings of the 2001 IEEE Workshop on the Applications of Signal Processing to Audio and Acoustics (Cat. No.01TH8575)*, pages 19–22, 2001.

[10] Z. Fu, G. Lu, K. M. Ting, and D. Zhang. A survey of audio-based music classification and annotation. *IEEE Transactions on Multimedia*, 13(2):303–319, April 2011.

[11] C. Hawthorne, A. Stasyuk, A. Roberts, I. Simon, C.-Z. A. Huang, S. Dieleman, E. Elsen, J. Engel, and D. Eck. Enabling factorized piano music modeling and generation with the MAESTRO dataset. In *International Conference on Learning Representations*, 2019.

[12] W. Herlands, Y. Greenberg, R. Der, and S. Levin. A machine learning approach to musically meaningful homogeneous style classification. *Proceedings of the National Conference on Artificial Intelligence*, 1:276–282, 01 2014.

[13] D. Herremans, D. Martens, and K. Sörensen. Composer classification models for music-theory building. In D. Meredith, editor, *Computational Music Analysis*, pages 369–392, Cham, 2016. Springer International Publishing.

[14] R. Hillewaere, B. Manderick, and D. Conklin. String quartet classification with monophonic models. In *Proceedings of the 11th International Society for Music Information Retrieval Conference, ISMIR 2010*, pages 537–542, 01 2010.

[15] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997.

[16] M. Hontanilla, C. Pérez-Sancho, and J. M. Iñesta. Modeling musical style with language models for composer recognition. In J. M. Sanches, L. Micó, and J. S. Cardoso, editors, *Pattern Recognition and Image Analysis*, pages 740–748, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[17] M. Kaliakatsos-Papakostas, M. Epitropakis, and M. Vrahatis. Musical composer identification through probabilistic and feedforward neural networks. In *Lecture Notes in Computer Science*, volume 6025, pages 411–420, 04 2010.

[18] M. Kassler. Toward musical information retrieval.

Perspectives of New Music, 4(2):59–67, 1966.

[19] Y. Lo and Y. Lin. Content-based music classification. In *Proceedings - 2010 3rd IEEE International Conference on Computer Science and Information Technology, ICCSIT 2010*, volume 2, pages 112–116, 2010.

[20] J. Lyons. Mel frequency cepstral coefficient (mfcc) tutorial. `http://www.practicalcryptography.com/miscellaneous/machine-learning/guide-mel-frequency-cepstral-coefficients-mfccs/`, 2009. (Accessed on 01/26/2020).

[21] J. Lyons, D. Y.-B. Wang, , Gianluca, H. Shteingart, E. Mavrinac, Yash Gaurkar, Watcharapol Watcharawisetkul, S. Birch, L. Zhihe, J. Hölzl, J. Lesinskis, H. Almér, Chris Lord, and A. Stark. jameslyons/python_speech_features: release v0.6.1, 2020.

[22] J. Marques and P. J. Moreno. A study of musical instrument classification using gaussian mixture models and support vector machines. Technical report, Compaq Corporation, Cambridge Research laboratory, USA, 1999.

[23] McFee et al. librosa/librosa: 0.7.2, 2020.

[24] G. Micchi. A neural network for composer classification. In *International Society for Music Information Retrieval Conference (ISMIR 2018)*, Paris, France, 2018.

[25] A. Nasridinov and Y. . Park. A study on music genre recognition and classification techniques. *International Journal of Multimedia and Ubiquitous Engineering*, 9(4):31–42, 2014.

[26] J. Paterson. Composer timelines for classical music periods. `https://www.mfiles.co.uk/composer-timelines-classical-periods.htm`. (Accessed on 01/26/2020).

[27] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[28] E. Pollastri and G. Simoncelli. Classification of melodies by composer with hidden markov models. In *Proceedings of the First International Conference on WEB Delivering of Music (WEDELMUSIC'01)*, WEDELMUSIC '01, page 88, USA, 2001. IEEE Computer Society.

[29] J. Rothstein. *MIDI: A Comprehensive Introduction*. A-R Editions, Inc., USA, 1992.

[30] P. Sadeghian, C. Wilson, S. Goeddel, and A. Olmsted. Classification of music by composer using fuzzy min-max neural networks. In *2017 12th International Conference for Internet Technology and Secured Transactions (ICITST)*, pages 189–192, Dec 2017.

[31] L. P. Sempere. Audio classical composer identification in mirex 2015: Submission based on structural analysis of music. Master's thesis, Polytechnic University of Valencia, 2015.

[32] G. Sharma, K. Umapathy, and S. Krishnan. Trends in audio signal feature extraction methods. *Applied Acoustics*, 158:107020, 2020.

[33] A. C. M. D. Silva, M. A. N. Coelho, and R. F. Neto. A music classification model based on metric learning applied to mp3 audio files. *Expert Systems with Applications*, 144, 2020.

[34] C. Simmermacher, D. Deng, and S. Cranefield. Feature analysis and classification of classical musical instruments: An empirical study. In P. Perner, editor, *Advances in Data Mining. Applications in Medicine, Web Mining, Marketing, Image and Signal Mining*, pages 444–458, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[35] J.-H. Su, C.-Y. Chin, T.-P. Hong, and J.-J. Su. Content-based music classification by advanced features and progressive learning. In N. T. Nguyen, F. L. Gaol, T.-P. Hong, and B. Trawiński, editors, *Intelligent Information and Database Systems*, pages 117–130, Cham, 2019. Springer International Publishing.

[36] G. Subramaniam, J. Verma, N. Chandrasekhar, C. NarendraK., and K. George. Generating playlists on the basis of emotion. *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 366–373, 2018.

[37] P. Van Kranenburg and E. Backer. Musical style recognition - a quantitative approach. *Handbook of Pattern Recognition and Computer Vision, 3rd Edition*, pages 15–18, 05 2004.

[38] G. Velarde, C. Cancino Chacón, D. Meredith, T. Weyde, and M. Grachten. Convolution-based classification of audio and symbolic representations of music. *Journal of New Music Research*, pages 1–15, 05 2018.

[39] C. Weiß. *Computational methods for tonality-based style analysis of classical music audio recordings*. PhD thesis, Ilmenau, Aug 2017.

[40] C. Weiss and M. Müller. Tonal complexity features for style classification of classical music. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 688–692, April 2015.

[41] C. Weiß, M. Mauch, and S. Dixon. Timbre-invariant audio features for style analysis of classical music. In *Proceedings - 40th International Computer Music Conference, ICMC 2014 and 11th Sound and Music Computing Conference, SMC 2014 - Music Technology Meets Philosophy: From Digital Echos to Virtual Ethos*, pages 1461–1468, 2014.

[42] C. Weiß, M. Mauch, S. Dixon, and M. Müller. Investigating style evolution of western classical music: A computational approach. *Musicae Scientiae*, 23(4):486–507, 2019.

[43] C. Weiß and M. Schaab. On the impact of key detection performance for identifying classical music styles. In *Proceedings of the 16th International Society for Music Information Retrieval Conference, ISMIR 2015*, pages 45–51, 2015.

[44] J. Wołkowicz and V. Kešelj. Evaluation of n-gram-based classification approaches on classical music corpora. In J. Yust, J. Wild, and J. A. Burgoyne, editors, *Mathematics and Computation in Music*, pages 213–225, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[45] J. WoŁkowicz, Z. Kulka, and V. Kešelj. N-gram-based approach to composer recognition. *Archives of Acoustics*, 33(1):43–55, 2008.

[46] P. Zwan, B. Kostek, and A. Kupryjanow. Automatic classification of musical audio signals employing machine learning approach. In *130th Audio Engineering Society Convention 2011*, volume 2, pages 1254–1264, 2011.

8. APPENDIX

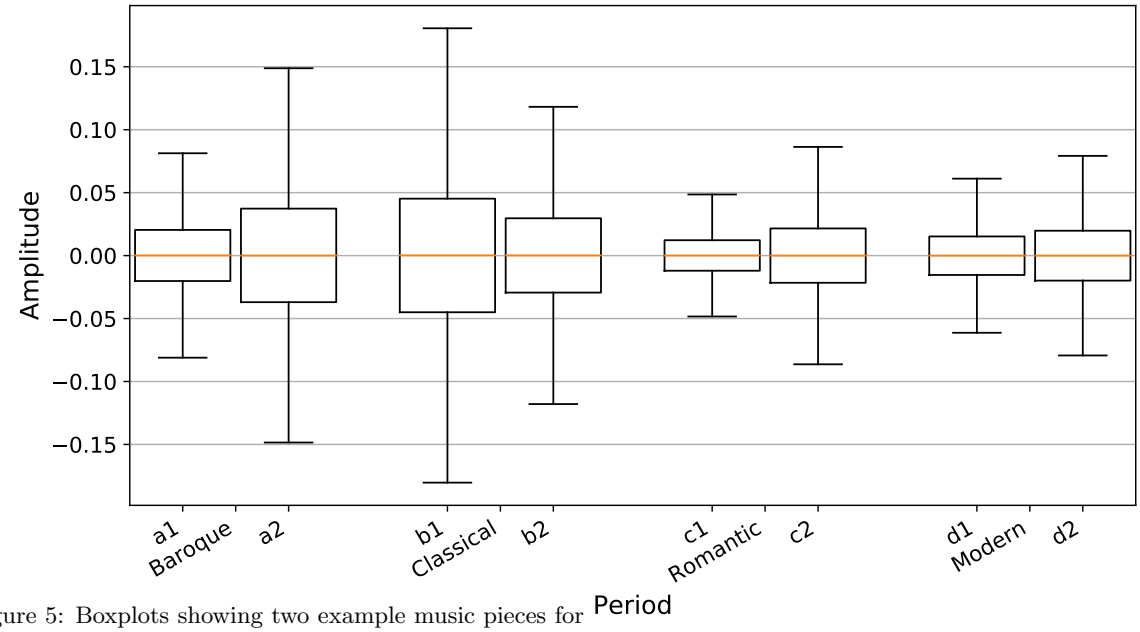


Figure 5: Boxplots showing two example music pieces for each of the classes, from two different composers per period. Pieces (a1) and (a2) are from the Baroque, (b1) and (b2) are Classical, (c1) and (c2) are Romantic, (d1) and (d2) are Modern

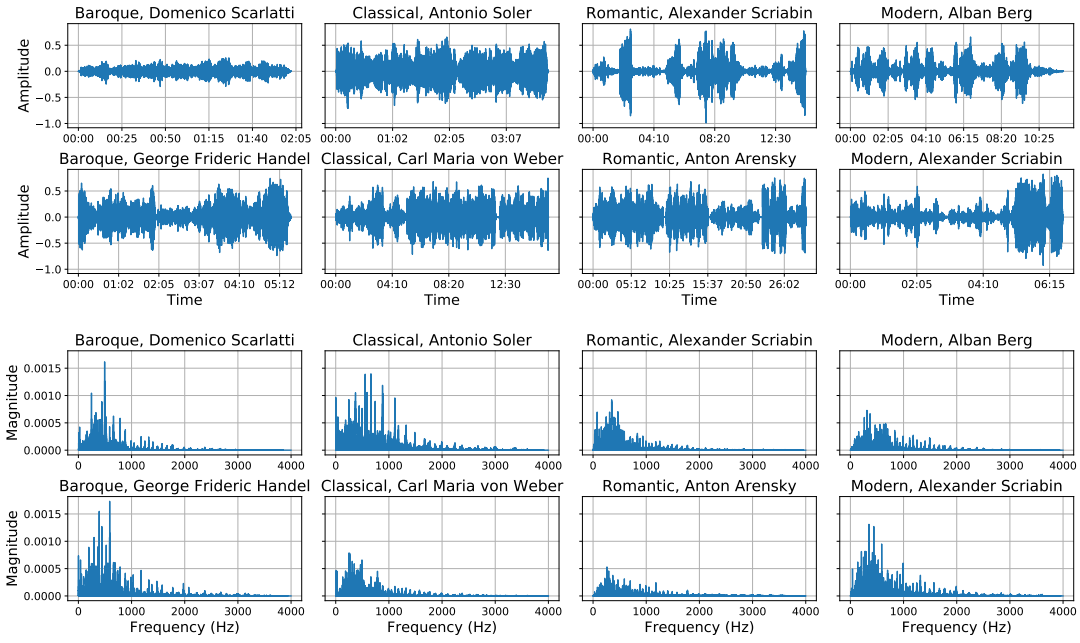


Figure 6: Two example signal waveforms for each period (the first two rows), and their corresponding Fourier transforms (the last two rows).

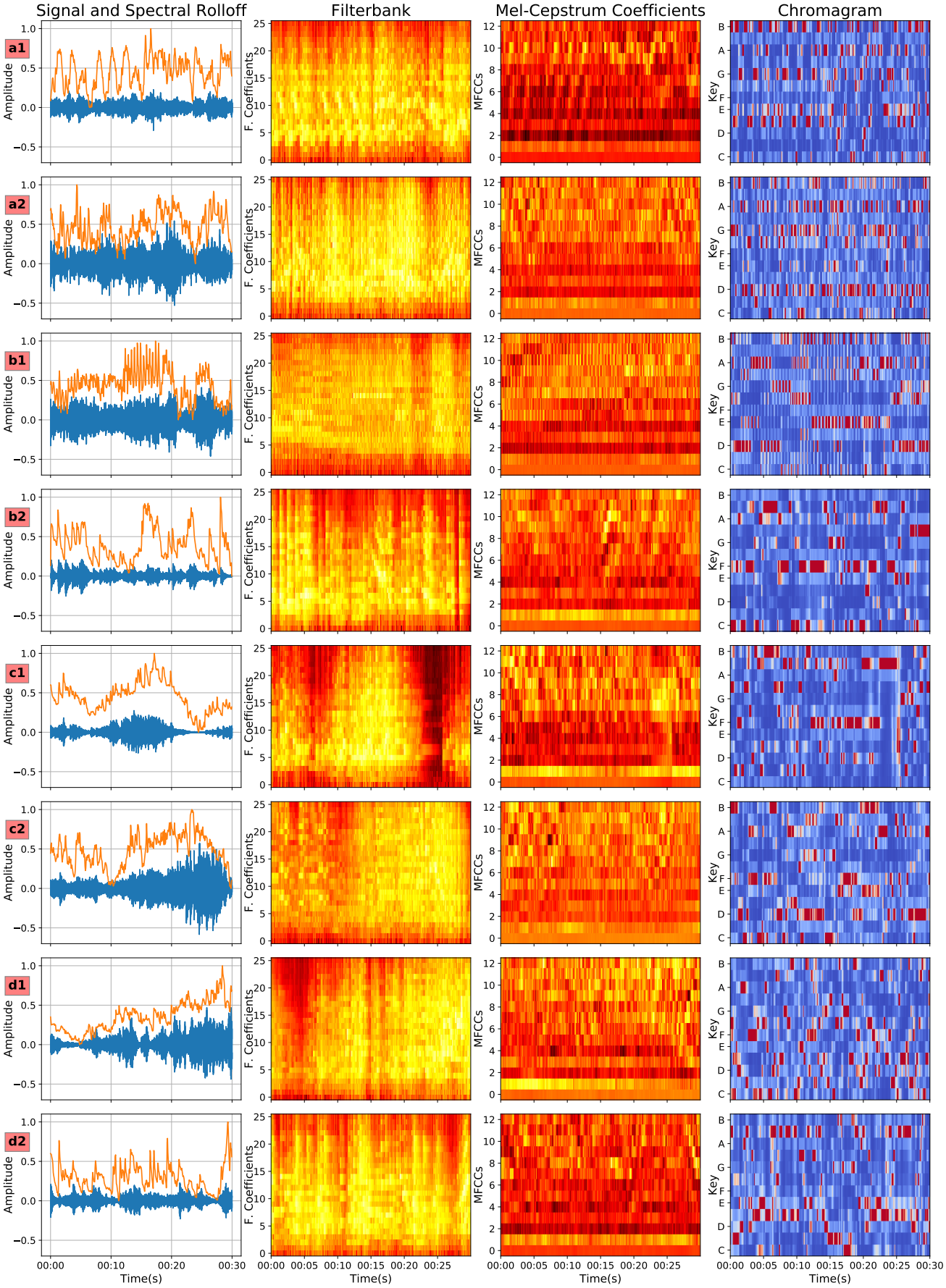


Figure 7: Spectral analyses: (a1) and (a2) are pieces from the Baroque, (b1) and (b2) are Classical pieces, (c1) and (c2) Romantic, and (d1) and (d2) Modern.