ISSN 2007-9737

Multi-Instrument Based N-Grams for Composer Classification Task

Alexander Gelbukh[1], Daniel Alejandro Pérez Alvarez[1],
Olga Kolesnikova[1], Liliana Chanona-Hernandez[*,2], Grigori Sidorov[1]

[1] Instituto Politécnico Nacional,
Centro de Invetigación en Computación,
Mexico

[2] Instituto Politécnico Nacional,
Escuela Superior de Ingeniería Mecánica y Eléctrica Unidad Zacatenco,
Mexico

lchanona@gmail.com

Abstract. In this research, we address the composer classification supervised problem from a Natural Language Processing perspective. Starting from digital symbolic music files, we build two representations: a class representation and other based on MIDI pitches. We use the technique of n-grams to build feature vectors of musical compositions based on their harmonic content. For this, we extract n-grams of size 1 to 4 in harmonic direction, differentiating between all possible subsets of instruments. We populate a term-frequency matrix with the vectors of compositions and we classify by the means of Support Vector Machines (SVM) classifier. Different classification models are evaluated, e.g., using feature filters and varying hyperparameters such as TF-IDF formula, among others. The results obtained show that n-grams based on MIDI pitches perform slightly better than n-grams based on class representation in terms of overall results, but the best result of each one of these representations is identical. Some of our best models reach accuracy results that exceed previous state of the art results based on a well-known dataset composed of string quartets by Haydn and Mozart.

Keywords. Composer classification, composer attribution, composer recognition, composer identification, composer style, n-grams, harmonic n-grams, string quartet, mozart, haydn.

1 Introduction

Starting from a list of composers and a list of compositions, the task of composer classification is defined as automatically assign each composition to the correct composer. It can be approached in unsupervised or supervised manner, but the last its a more known task. There are two types of formats to represent music digitally.

These are audio files and symbolic files. Audio files store recorded sound and are used in the field of Signal Processing. Our approach is more close to Natural Language Processing field. That's why we use symbolic files, which store information similar to a stave, as the composer intended it.

Different names have been given to this task. For example, composer style [30, 29], composer identification [21, 22], composer recognition [44, 42] and composer attribution [38, 40].

While several datasets have been tested for the symbolic domain, comparing Mozart and Haydn remains a challenge due to the similarities between these composers [39, 21]. These two composers admired and influenced each other [36, 4, 11].

Datasets involving Mozart and Haydn are often the most difficult to classify [39, 9, 43]. According to Hillewaere et al. and Kaliakatsos-Papakostas et al. [15, 22] the models for solving classification tasks related to music (in symbolic format) can be grouped into two large categories.

The first category includes models based on global features or statistical descriptors which express each piece as a vector of features. Each feature or descriptor represents the measurement of a certain musical element throughout the entire piece, for example, frequency of major second intervals, average pitch of notes, etc.

Among the works that are part of this category we can mention [2, 38, 23]. The second category involves predictive models such as n-grams or Markovian models, which are based on the counting or prediction of local events. An event can be the interval between two consecutive notes or the duration ratio between two consecutive notes, etc. Examples of studies in this category include [44, 16, 19].

In recent years, deep learning based models for composer classification [43, 45, 7, 8, 24] may also worth a category on his own. If we look at the more abroad music classification task we find deep learning approaches based on Convolutional Neural Network (CNN) [43], Residual Neural Network [24, 8] and transformer-based architectures [45, 7, 47]. This open exciting new possibilities to represent music but this types of models usually need a lot of samples to train and many resources to process.

For example, when trying to apply CNN to a Mozart and Haydn dataset, the efforts by Verma and Thickstun [43] were unsuccessful, probably due to the small size of the dataset, which could introduce overfitting in deep learning models, and also the need for applying Leave One Out (LOO) cross validation in order to compare with previous works, which multiplies the processing resources needed.

Our approach is based on n-grams, but instead of the more common melody-based methods [44, 16, 19], we use harmony-based n-grams. Our goal is to investigate whether harmonic content can be a good predictor for the composer classification task. This document is structured as follows.

In Subsection 1.1, we present the dataset used. In Section 2, we discuss related work. In Section 3, we show our method. In Section 4, we discuss our results. Finally, in Section 5, we give our conclusions.

1.1 Dataset

The dataset we use is composed by 107 movements from string quartets by Haydn and Mozart. It is a balanced dataset, since there are 54 movements by Haydn and 53 by Mozart. This dataset was collected by Van Kranenburg and Backer [39] in **kern format. This format store musical information about pitch, duration, dynamics etc., in a similar fashion to a musical staff.

The **kern files can be converted into other popular formats of symbolic music such as MIDI [20]. The data was gathered from two different sources: Musedata[1] and KernScores[2] and generated at the Center for Computer Assisted Research in the Humanities at Stanford University[3]. According to our research, 16 more works have been built around this dataset or similar, using accuracy as the measure in all cases.

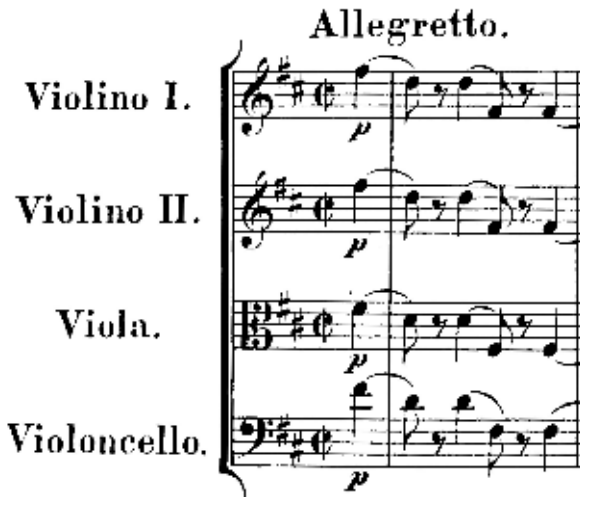
2 Related Work

In 2005, Van Kranenburg and Backer made an important contribution to the field of Music Information Retrieval MIR [39].

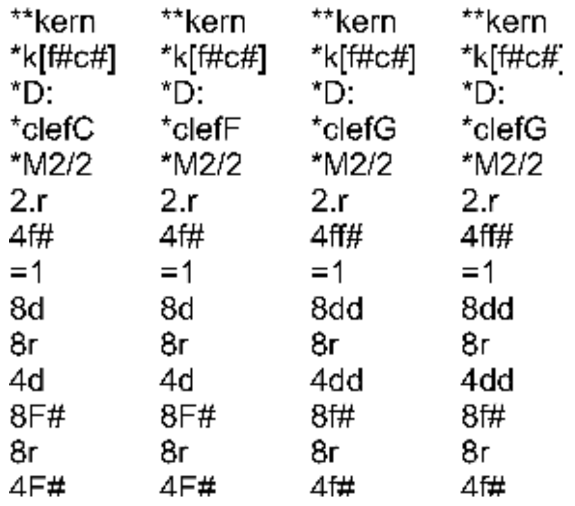
Instead of characterizing the style of composers from different periods [33] or differentiating the work of a composer among a group of composers [6], these researchers posed the problem of differentiating two contemporary and very similar composers such as the case of Mozart and Haydn.

Several research has been made in this regard, but solutions to this problem still has room for improvement.

[1]musedata.org
[2]kern.ccarh.org
[3]www.ccarh.org

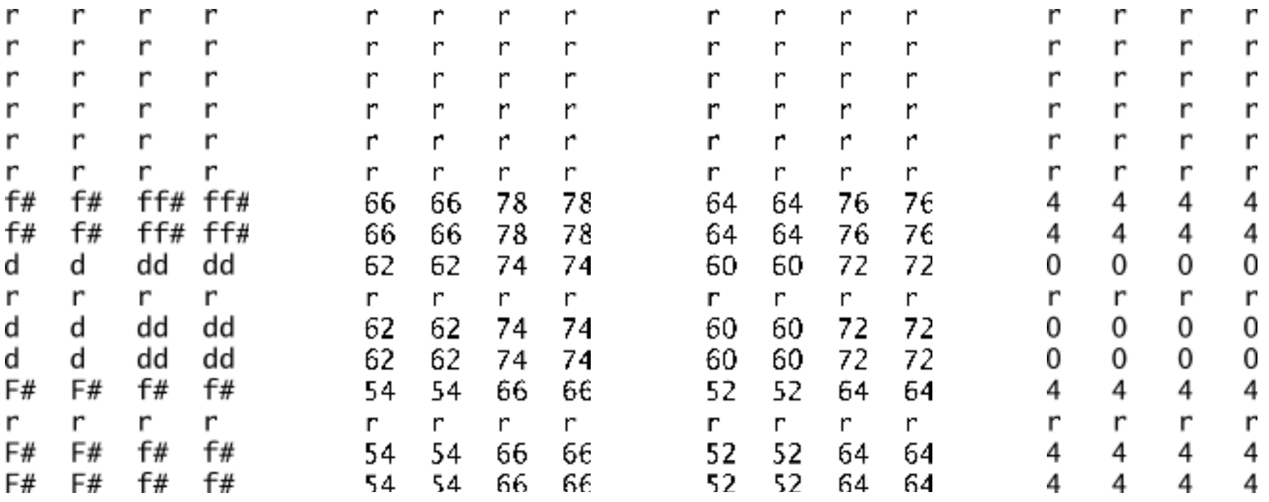


(a)



```
**kern      **kern      **kern      **kern
*k[f#c#]    *k[f#c#]    *k[f#c#]    *k[f#c#
*D:         *D:         *D:         *D:
*clefC      *clefF      *clefG      *clefG
*M2/2       *M2/2       *M2/2       *M2/2
2.r         2.r         2.r         2.r
4f#         4f#         4ff#        4ff#
=1          =1          =1          =1
8d          8d          8dd         8dd
8r          8r          8r          8r
4d          4d          4dd         4dd
8F#         8F#         8f#         8f#
8r          8r          8r          8r
4F#         4F#         4f#         4f#
```

(b)



```
r   r   r   r        r  r  r  r        r  r  r  r        r r r r
r   r   r   r        r  r  r  r        r  r  r  r        r r r r
r   r   r   r        r  r  r  r        r  r  r  r        r r r r
r   r   r   r        r  r  r  r        r  r  r  r        r r r r
r   r   r   r        r  r  r  r        r  r  r  r        r r r r
r   r   r   r        r  r  r  r        r  r  r  r        r r r r
f#  f#  ff# ff#      66 66 78 78       64 64 76 76       4 4 4 4
f#  f#  ff# ff#      66 66 78 78       64 64 76 76       4 4 4 4
d   d   dd  dd       62 62 74 74       60 60 72 72       0 0 0 0
r   r   r   r        r  r  r  r        r  r  r  r        r r r r
d   d   dd  dd       62 62 74 74       60 60 72 72       0 0 0 0
d   d   dd  dd       62 62 74 74       60 60 72 72       0 0 0 0
F#  F#  f#  f#       54 54 66 66       52 52 64 64       4 4 4 4
r   r   r   r        r  r  r  r        r  r  r  r        r r r r
F#  F#  f#  f#       54 54 66 66       52 52 64 64       4 4 4 4
F#  F#  f#  f#       54 54 66 66       52 52 64 64       4 4 4 4
```

(c) (d) (e) (f)

Fig. 1. Steps to final representations

2.1 Composer Classification

Van Kranenburg and Backer [39] evaluate the effectiveness of harmony and counterpoint-based features for detecting the style of the composers Bach, Handel, Telemann, Haydn and Mozart.

To do this, they extract 20 features, among which are: fraction of dissonant sounds, average number of active voices, number of different harmonic intervals between pairs of voices, number of parallel thirds, fourths and sixths, etc.

The k-means clustering, k-nearest neighbor and decision tree classifiers are used on this feature vector.

The researchers divide the dataset into several subsets, of which the most difficult to classify turn out to be those containing the composers Haydn and Mozart.

The best results are obtained with the Nearest neighbor classification algorithm in conjunction with the Fischer transformation for dimension reduction.

ISSN 2007-9737

Table 1. Accuracy best results of MIDI representation for each subset of instruments

insts	results	num feat	insts	results	num feat	instrums	results	num feat
1	65.42	48	1-3	86.92	1,208	1-2-3	**88.79**	8,685
2	70.09	38	1-4	82.24	519	1-2-4	84.11	9,982
3	66.36	38	2-3	75.7	574	1-3-4	83.18	2,102
4	78.5	44	2-4	84.11	650	2-3-4	85.98	7,946
1-2	77.57	1,247	3-4	82.24	566	1-2-3-4	87.85	25,526

Kempfert and Wong [23] explore the use of features derived from the sonata form, particularly the use of primary and secondary themes in the melodic development of the pieces.

As a complement, they add to the feature vector other style markers based on the rhythm and melody of individual voices and harmonic features that capture the interaction between voices, as well as global features of the pieces, such as average and standard deviation of pitch and duration of notes or the ratio of notes and rests during the piece. Some of these features are derived from previous studies by other researchers.

They apply a selection of features based on the Bayesian Information Criterion and classify using Bayesian Logistic Regression, obtaining state of the art results for the widely used dataset of string quartets by Haydn and Mozart. Their work demonstrates the usefulness of incorporating features that take into account the sonata structure.

2.2 Harmony Based Classification

Ogihara and Li [30] seek to characterize the style of composers through the chord progressions of their works. To do this, they use n-grams of previously simplified chords and assign weight to these n-grams depending on the duration of the chords.

These researchers also use a transposition system to ensure that the key is the same for all works. With the n-grams obtained, the researchers create profiles of composers and compare them with each other using the cosine similarity measure. Using this method, the researchers managed to automatically group jazz composers hierarchically, according to a chronological order.

Something that remained to be demonstrated in this research is whether the cosine similarity measure is the most appropriate for comparing composer profiles because the n-grams that make up the profiles suffer from dispersion.

Pérez-Sancho et al. [34] face the task of classifying musical genres. For this, they collect a corpus of pieces from three genres: popular, jazz and classical and three subgenres for each of these genres. These researchers use two methods to ensure the homogeneity of the data: to transpose the entire dataset to the same key or to represent the chords as degrees of the key.

They also use feature selection procedures based on Average Mutual Information (AMI) on the chord list of the training set. To classify, they compare the performance of the n-gram technique with the Naïve Bayes Classifier, obtaining slightly better results with n-grams for the data set of three genres and with Naïve Bayes for the data set of nine subgenres.

According to [13], the representation of musical structure can have a significant influence in the quality of the results of a computational analysis on a given dataset. In most of the reviewed research that faces musical classification tasks based on harmonic content, the classic symbolic nomenclature of chord representation is used (e.g., Cmaj, Cmin, Cdim etc.), or some representation variant.

An example of this is the use of classical chord labels [30, 34, 46, 13] or a binary notation based on the present notes [46]. Other transformations include chord simplification [30, 34, 13], enharmonic representation of chords [35] or the replacement of chords by degrees or tonal functions (1st degree, 5th degree etc.) [34].

Table 2. Accuracy best results of class representation for each subset of instruments

instrums	results	num feat	instrums	results	num feat	instrums	results	num feat
1	63.55	13	1-3	76.64	148	1-2-3	86.92	1,016
2	66.36	13	1-4	71.03	169	1-2-4	75.7	1,725
3	69.16	13	2-3	79.44	140	1-3-4	79.44	1,704
4	52.34	5	2-4	68.22	150	2-3-4	86.92	638
1-2	64.49	135	3-4	62.62	121	1-2-3-4	**88.79**	2,136

Some researchers also process the duration of the chords [30, 35, 13] and the vast majority use transposition to the same key [30, 34, 46, 13]. Our representation specifies not only the notes that are included in a given chord but also the order in which these notes are produced. That is, our representation includes information about which of the instruments produces a given note within the chord. In many cases, our representation even specifies which octave each note is in.

In this way our classifier can know at all times which instrument produces the tonic note, which instrument produces the 5th degree, etc. This information could be important to characterize the style of a certain composer. The representations based on chord labels discussed in the previous paragraph do not offer access to that information.

3 Method

Some aspects of our methodology are shared with our previous work [1], for example, setting a minimum note length, which aims to transform duration information into tonal information. But the way we generate n-grams is different because we base our method in harmony instead of melody. Other aspects are similar to those that have been observed in prior studies, including feature filtering and using popular machine learning classifiers such as SVM.

3.1 Preprocessing

We found some errors in **kern files and we fix them manually (see subsection 4.1). We also remove multiple stops keeping the highest note at any moment for each instrument, as is common practice [44, 26]. Besides this, we extend the representation, we convert the pitches from **kern to MIDI format, we transpose the compositions and we transform MIDI pitches into classes in an optional fashion. We explain all this steps in detail below.

3.2 Extended Representation

Because notes can have different duration is very hard to generate a n-gram model using more than one instrument simultaneously. That's why we establish a minimum note length, thus transforming long notes into many notes of minimum length.

Our goal is to convert duration information into pitch information. For instance, we can represent a quarter note as two quavers, a half note as four quavers and a whole note as eight quavers, if we define the quaver as minimum note length. However, with such election would be impossible to represent shorter length notes such as semiquavers.

That's why a much smaller base note must be established to avoid losing too much information. The downside to this is the high number of computational resources that a small minimum note length can take to process. We define as extended representation the action of converting a regular staff with regular note lengths into a representation with only minimum length notes.

Several researchers have tried somewhat similar procedures, for instance Pape et al. [31] (they also add one hot encoding representation) and Velarde et al. [42] (they add multi hot encoding).

ISSN 2007-9737

Table 3. Comparison with the state of art (in the last 9 works the same composers were used but with different datasets)

Comparison with SOTA	
4-grams (Class representation)	88.79
Alvarez et al. (2024) [23]	86.92
Kempfert and Wong (2020) [23]	84.11
Lostanlen (2018) [26]	82.2
Velarde et al. (2016) [42]	80.4
Van Kranenburg and Backer (2005) [39]	79.4
Hillewaere et al. (2009) [16]	79.4
Velarde et al. (2018) [41]	79.4
Hajj et. al. (2018) [10]	82.9
Kempfert and Wong (2020) [23]	82.46
Herlands et al. (2014) [14]	80.0
Hillewaere et al. (2010) [17]	75.4
Hontanilla et al. (2013) [19]	74.7
Dor and Reich (2011) [9]	73.75
Pape et al. (2008) [31]	73.5
Taminau et. al. (2010) [37]	73.0
Kaliakatsos et al. (2011) [22]	70.0

3.3 Transposition

Bringing all compositions of the dataset to the same key it's required to avoid being conditioned when classifying by the diverse original tonalities of the pieces. Otherwise, instead of classifying by composers, we probably would be classifying by tonalities.

This follows Wolkowsky's view [44], which calls for independence between the features of the feature vector and the tonalities of the pieces of dataset. There are ways of avoiding transposition, for instance using intervals between consecutive notes [44, 15, 16, 19, 10].

However, we could lose some information with this type of representation. For example, the intervals C-G and F-C are both perfect fifth intervals, so they are represented in the same way in a interval-based representation.

On the other hand, in a transposed based representation, they are represented differently. The interval C-G being equivalent to going from the tonic to the fifth degree and the interval F-C being equivalent to going from the fourth degree to the tonic.

3.4 Class Pitch vs MIDI Pitch

As an optional step in the representation, we add a class representation. To do so, we use 12 different symbols representing each of the musical notes and one special symbol for rests. We use the modulo 12 operation in order to normalize MIDI pitch into classes. This results in a reduction of the MIDI values in degrees or functions (fifth, fourth, third etc). For instance, after normalization, MIDI values 48, 60 and 72 would be represented as class 0, corresponding to tonic C, and MIDI values 55, 67 and 79 would be equivalent to class 7, corresponding to fifth degree G. This representation is compared with a representation that preserves the original MIDI values in Section 4.

3.5 Example of Feature Generation

We summarise the above steps with an example. Figure 1 shows the processing of the first two bars of the first movement of Mozart's string quartet k499. Clause a) shows the representation on the staff, as conceived by the composer. In clause b) the same piece is shown, but now in **kern format.

At the top of this format, some metadata can be observed (character '*') and at the bottom the notes can be seen. Each instrument is represented by a column and the '=' character denotes the beginning of a new measure. The length of each note is represented with a number and the pitch with a combination of letters and symbols.

Clause c) shows how the piece looks after "extending" the representation, we use a quaver as the minimum base note in this example. We discard unused information contained in the **kern format. The piece in this example begins with an anacrusis, hence the first thing represented is the dotted half note rest that precedes the quarter note f#.

Table 4. Most important features for class Haydn and class Mozart based on our two best models

MIDI	H	60-52-52, 65-62-53, 67-59-59, 83-r-r, 64-55-55, 64-58-55, r-71-67, 62-53-53, 62-57-57, 65-59-62
rep	M	60-57-50, 72-57-52, 71-59-55, 72-57-54, 74-62-r, 62-47-43, 62-50-47, 76-67-60, r-64-55, 72-60-r
Class	H	5-2-5-2, 0-4-4-0, 2-9-2-5, 0-4-4-7, 7-0-4-4, 11-5-11-7, 6-9-0-7, 2-9-9-5, 6-2-0-r, 9-0-0-5
rep	M	9-5-0-5, 2-7-11-5, r-r-2-2, 10-4-0-0, 4-7-0-r, 5-11-7-7, 2-7-11-2, 5-2-9-2, 11-7-7-4, 2-9-0-6

It can be seen that in this extended representation the eighth notes are not modified, the quarter notes are doubled and the dotted half note rest is replaced by 6 eighth note rests. Clause d) shows a representation similar to that in clause c), only the pitches in **kern format have been replaced by MIDI values.

Clause e) shows the transposition of these MIDI values to the scale of C. Since the original scale is D, we subtract 2 from each MIDI value. The rests remain unchanged. Finally, clause f) shows the optional step of converting the already transposed MIDI values into class values. To achieve this, we calculate each MIDI value by its modulo 12. Once again, the rests are not modified.

3.5.1 Instrument Filtering

Once we have all preprocessing done we can start to generate features. We use a simple n-gram alike method, but instead of words we use groups of minimum length notes in harmonic direction.

Since we have reduced the content of each instrument to a single note, in our dataset of string quartets must be a maximum of 4 voices playing (or at rest) at any given moment (see 1). Instead of always using all of these 4 voices, we build models for all combinations of instruments and we create n-grams based solely on the current subset of instruments.

For example, we can have models that use only viola, models that use only viola and cello, models that use all voices except for viola etc. In this way we can analyse based on results if there is a particular instrument, or combination of instruments that make the results stand out.

3.5.2 N-grams Generation

Continuing with the example of feature generation, in figure 1 after building MIDI representation in clause e) and class representation in clause f), we can generate $2^4 - 1 = 15$ n-grams in harmonic (horizontal) direction for each line (time).

Each one of these n-grams will be made from notes solely from a subset of instruments, so if the current model uses only first violin and viola, then we only use columns 1 and 3 to generate a 2-gram and if the current model uses all instruments except for second violin then we only use columns 1, 3 and 4 to generate a 3-gram.

We use symbol '-' to concatenate from left to right the notes within the n-gram. For instance, for a model using all instruments and class representation the following 4-grams are generated: r-r-r-r, r-r-r-r, r-r-r-r, r-r-r-r, r-r-r-r, r-r-r-r, 4-4-4-4, 4-4-4-4, 0-0-0-0... etc.

For a model based on MIDI representation using only first violin and viola the following 2-grams are generated: r-r, r-r, r-r, r-r, r-r, r-r, 64-76, 64-76, 60-72, ..., etc. and we ignore the information from remaining columns (for that particular model).

Counting the number of occurrences of each of the n-grams generated we fill the feature vector that identifies each composition.

3.6 Feature Filtering

Once each sample is represented as a vector of occurrences of n-grams of groups of instruments, we can optionally apply a simple feature selection criterion. To do this, we set 3 thresholds, a threshold that filters the most frequent n-grams, a threshold that filters the most infrequent n-grams

Table 5. Misclassified scores

MIDI	H	op103-01, op20n3-01, op20n6-04, op50n1-04, op64n1-03, op64n4-04, op71n1-04, op76n4-01, op76n4-03
rep	M	k138-03, k159-02, k168-04
Class	H	op1n0-05, op20n3-03, op20n6-04, op33n6-02, op50n2-01, op64n1-02, op64n4-04, op76n4-01
rep	M	k159-03, k168-02, k168-03, k465-03

and a threshold that limits the number of features. This is somewhat similar to the way text is processed by removing stop words and rare words. We can also alter the TF-IDF formula or the type of normalization of the feature vector.

Thus, we can compare models with different degrees of filtering, different norm and different variations of TF-IDF formula. For more on this, see subsection 4.1.

3.7 Classification Parameters

We accommodate some NLP concepts to the field of music with the goal of building the Term-document matrix [27, 28]. These concepts are enumerated below:

1. Term frequency (TF): $\mathrm{tf}(t, c)$, the frequency of occurrence of term t in composition c.
2. Inverse document frequency (IDF): $\log[n/\mathrm{df}(t)] + 1$, where n is the total number of compositions and $\mathrm{df}(t)$ is the number of compositions in which term t is present.
3. TF-IDF: $\mathrm{TF}(t, c) \times \mathrm{IDF}(t)$, TF multiplied by IDF.
4. Sublinear scaling: $1 + \log(\mathrm{TF})$, is used as optional replacement for TF.
5. IDF smoothing: $\log\Big((1+n)/\big(1+\mathrm{df}(t)\big)\Big)+1$, is used as optional replacement for IDF.
6. Normalization L1: set the sum of values of the feature vector equal to 1.
7. Normalization L2: set the sum of squares of values of the feature vector equal to 1.

As can be observed, we have replaced documents (for which the original formula was created) for musical compositions. I the case of terms, we have defined them as minimum length groups of notes in harmonic direction, but it also could be intervals between notes, note duration or any other element taken from compositions.

4 Results and Discussion

4.1 Quartet Classification

We selected SVM with linear kernel, a classifier commonly used for text classification, for quartets classification. Given the computationally expensive process resulting from vectors with high dimensionality, we avoided SVM with RBF kernel.

To ensure that the vector size is uniform for all samples, n-grams which have not been observed in a particular composition are added to the vector with an occurrence of zero. We chose to transpose all the movements to the key of C major.

Changes in tonality may occur within each movement, that is a peculiarity of string quartet format. These changes were carefully considered during the transposition process. The **kern files were found to contain a certain amount of encoding errors.

For example, a file whose lines do not match the humdrum encoding, files or some bars of files with more instruments than the established ones (we removed the extra column), files with duplicate instruments (we ignored the extra columns) and files with duration errors (see [16, 17].

We use Python as the programming language for our method.

We use scikit-learn [32] and nltk [3] libraries, as well as Support Vector Classifier (SVC), CountVectorizer and TfidfTransformer methods of these libraries. We established the values **C** for SVC as **C=1,000** and minimum note length **g** as **g=192**. We created models for both MIDI and class representation.

For each representation, we developed models for each of the 15 subsets of instruments (first violin, second violin, viola, cello, first violin and second violin, first violin and viola etc.).

For each subsets of instruments, we tested models with and without IDF, sublinear scaling and IDF smoothing, different types of norms (L1 and L2), values of 0, 5 and 10 for minimum document frequency and values of L, L-5 and L-10 for maximum document frequency, where L is the length of the dataset. We use leave one out cross validation to compare our results with previous research (See Table 3).

4.2 Analysis of Results

The code and the results for the present work can be seen at the following address[4]. Here we present the best results for each each subset of instruments and each representation. Table 1 shows the best results obtained for each instrument subset using the MIDI pitch representation. Columns of the table show in order the instruments used, the results and the average number of features.

Most of the models listed in this table obtained good results, above 80% of accuracy. The best result derived from the MIDI representation (88.79) is based on a model formed by the union of the first and second violin and viola. The parameters of this model are as follows:

L2 norm with sublinear scaling in conjunction with IDF smoothing, a minimum document frequency equal to 0 and a maximum document frequency equal to 97.

On the other hand, the cello was the most predictive instrument among the models made up of individual instruments, with a wide difference over the rest. It is worth highlighting the result of the model formed by the first violin and viola (86.92) with just over 1200 features. Several of the models presented in this table outperform previous state of the art results. Table 2 shows the best results obtained for each instrument subset using the class representation. Compared to the MIDI pitch representation, most results are inferior.

[4]github.com/dapalvarez/harmonic_ngrams

The best result of this representation (88.79) equals the best result of the MIDI representation but with a smaller number of features. Among the parameters of this best result, we used the L2 norm and sublinear scaling in conjunction with the inverse document frequency, and we also set the minimum document frequency to 5 and the maximum document frequency to 107.

Among other notable results, we can mention the mixture of second violin and viola (79.44) with only 140 features and the union of second violin, viola and cello (86.92) with only 638 features. Several models based on this representation also surpass the results of the state of the art.

Table 3 shows the state of the art results for the task of composer classification. The first part of the table shows the results of previous research using the same dataset we use. The second part of the table shows the research in which the same composers (Haydn and Mozart) are classified but using datasets different than ours. The results obtained in the present work outperform the results of the state of the art.

4.3 Musicological Analysis

Table 4 presents the 10 most important features for Haydn and Mozart based on our two best models. This was calculated using ELI5 Python library and taking into account all iterations of the cross validation process. The features are sorted by importance. First part of the table shows the features derived from trigrams of all instruments except for violoncello and MIDI representation.

It's difficult to make conclusions about the chords involved, since we are missing the lower voice but we can recognize in the Mozart's row a likely A minor in second position, a G major in sixth position and a C major in eighth position. In the case of features derived from the class representation model, it's easier to identify chords.

ISSN 2007-9737

So, in the case of Haydn, we recognize a couple of D minor chords in first inversion, a couple of C major chords in second and first inversion, and an F major. In the case of Mozart, we can see an F major, a G major 7 in third inversion, a C major, a G major in second inversion, a D minor, an E minor and a D major 7 in first inversion.

Most of this chords are shared by both composers but differences arise in the order of the notes. When we compare the features from the two models, applying modulo 12 operation to the MIDI pitches and comparing them with the three first notes of the class derived features, we find some coincidences. So, for Haydn, feature '60-52-52' matches '0-4-4-0' and '0-4-4-7', feature '65-62-53' matches '5-2-5-2' and feature '62-57-57' matches '2-9-9-5'. For Mozart, feature '76-67-60' matches '4-7-0-r'. Besides, there are no coincidences between MIDI-based features belonging to one composer and class-based features belonging to the other composer, so we can glimpse points of contact between the procedures of both models.

Table 5 lists the compositions that were misclassified by the two models with which we obtained the best results. The first part of the table shows the pieces misclassified by the model formed by the union of first violin, second violin and viola, based on the MIDI pitch representation.

Most of the pieces listed in this segment are composed by Haydn. Among these is the fugal finale (4th movement) opus 20 number 6. This movement has similarities with another Mozart fugue, the fourth and final movement of the k168 quartet, also misclassified.

Heartz [12] asserts that Mozart's inclusion of a fugue at the conclusion of the K168 quartet was a result of his familiarity with Haydn's opus 20. Brown [5] refutes this, noting out that composing fugal ends is not just a Haydn practice, but rather a Viennese one.

On the other hand, Hontanilla [18] points out that fugues are popular throughout the Baroque era, which makes it difficult to classify because both composers use them to the same extent. The second part of the table 5 shows the compositions misclassified by the model formed by the union of all the instruments and based on the class representation.

As with the MIDI-based model, most of the misclassified pieces belong to Haydn. Several compositions were misclassified by both models. These are the fourth movement opus 64 number 4, the first movement opus 76 number 4 and the fourth movement opus 20 number 6, all by Haydn.

The third movement of the quartet k465, also misclassified, belongs to the "Haydn Quartets", a group of 6 string quartets that Mozart composed in honor of Haydn. Bonds [4], suggests that with the "Haydn Quartets", Mozart intended to show himself as the master Haydn's successor rather than attempting to copy his style.

Maybe that's why only one movement turned out to be misclassified among the 18 movements in the dataset derived from the "Haydn Quartets". Regarding this 3rd movement k465, La Rue [25] identifies a direct influence of Haydn on it.

The movements k168-02, k168-03, k168-04 included in the "Viennese Quartets" were composed by Mozart in 1973 in the city of Vienna. There are studies on similarities between the "Viennese Quartets" and other contemporary quartets by Haydn [5]. For example Heartz [12], identifies the quartets opus 9 and opus 17 as probable sources of inspiration for Mozart.

5 Conclusions

In the present manuscript, we expose a new approach to the supervised problem of composer recognition. We adapt to music field some concepts of NLP domain to create a vectorial representation of musical pieces based on n-grams extracted in harmonic direction. We compare the pros and cons of representing pitches using MIDI values or class values. We use the SMV classifier to achieve state of the art results on a dataset of string quartets by Mozart and Haydn.

We compare models with different subsets of hyperparameters such as norm, feature filtering values etc. and we give some musical insight about the misclassified scores. We think it would be interesting, as future work, to evaluate models which can combine different n-gram sizes. That is, models which integrate unigrams with trigrams, bigrams with 4-grams, etc.

Another proposal would be to apply our method to datasets used in other music classification tasks, such as emotion recognition or genre recognition. Finally, we propose to adapt recent discoveries from NLP field, for example transformer models such as BERT, to tackle the problem of composer classification.

Acknowledgments

The present work was done with partial support from the Mexican Government through the grant A1-S-47854 of CONACYT, Mexico, grants 20232138, 20231567 and 20232080 of the Secretaría de Investigación y Posgrado of the Instituto Politécnico Nacional, Mexico.

The authors thank the CONACYT for the computing resources brought to them through the Plataforma de Aprendizaje Profundo para Tecnologías del Lenguaje of the Laboratorio de Supercómputo of the INAOE, Mexico and acknowledge the support of Microsoft through the Microsoft Latin America PhD Award.

References

1. **Alvarez, D. A. P., Gelbukh, A., Sidorov, G. (2024).** Composer classification using melodic combinatorial n-grams. Expert Systems with Applications, Vol. 249. DOI: 10.1016/j.eswa.2024.123300.

2. **Backer, E., van-Kranenburg, P. (2005).** On musical stylometry—a pattern recognition approach. Pattern Recognition Letters, Vol. 26, No. 3, pp. 299–309. DOI: 10.1016/j.patrec.2004.10.016.

3. **Bird, S., Klein, E., Loper, E. (2009).** Natural language processing with Python: Analyzing text with the natural language toolkit. O'Reilly Media, Inc.

4. **Bonds, M. E. (2007).** Replacing haydn: Mozart's "Pleyel" quartets. Music and Letters, Vol. 88, No. 2, pp. 201–225. DOI: 10.1093/ml/gcl150.

5. **Brown, A. P. (1992).** Haydn and Mozart's 1773 stay in Vienna: Weeding a musicological garden. Journal of Musicology, Vol. 10, No. 2, pp. 192–230. DOI: 10.2307/763612.

6. **Buzzanca, G. (2002).** A supervised learning approach to musical style recognition. Additional Proceedings of the Second International Conference on Music and Artificial Intelligence, Vol. 2002, pp. 167.

7. **Chou, Y. H., Chen, I., Chang, C. J., Ching, J., Yang, Y. H. (2021).** MidiBERT-piano: Large-scale pre-training for symbolic music understanding. arXiv. DOI: 10.48550/arXiv.2107.05223.

8. **Deepaisarn, S., Buaruk, S., Chokphantavee, S., Chokphantavee, S., Prathipasen, P., Sornlertlamvanich, V. (2022).** Visual-based musical data representation for composer classification. 17th International Joint Symposium on Artificial Intelligence and Natural Language Processing, pp. 1–5. DOI: 10.1109/iSAI-NLP56921.2022.9960254.

9. **Dor, O., Reich, Y. (2011).** An evaluation of musical score characteristics for automatic classification of composers. Computer Music Journal, Vol. 35, No. 3, pp. 86–97.

10. **Hajj, N., Filo, M., Awad, M. (2018).** Automated composer recognition for multi-voice piano compositions using rhythmic features, n-grams and modified cortical algorithms. Complex and Intelligent Systems, Vol. 4, No. 1, pp. 55–65. DOI: 10.1007/s40747-017-0052-x.

11. **Hatten, R. S. (2017).** Mozart's music of friends: Social interplay in the chamber works, Vol. 39. DOI: 10.1093/mts/mtx010.

12. **Heartz, D. (1995).** Haydn, Mozart, and the Viennese School, 1740-1780. WW Norton New York.

13. **Hedges, T., Roy, P., Pachet, F. (2014).** Predicting the composer and style of jazz chord progressions. Journal of New Music Research, Vol. 43, No. 3, pp. 276–290. DOI: 10.1080/09298215.2014.925477.

14. **Herlands, W., Der, R., Greenberg, Y., Levin, S. (2014).** A machine learning approach to musically meaningful homogeneous style classification. Vol. 28, No. 1. DOI: 10.1609/aaai.v28i1.8738.

15. **Hillewaere, R., Manderick, B., Conklin, D. (2009).** Global feature versus event models for folk song classification. Proceedings of the International Society for Music Information Retrieval Conference, Vol. 2009, pp. 729–733.

16. **Hillewaere, R., Manderick, B., Conklin, D. (2009).** Melodic models for polyphonic music classification. Proceedings of the Second International Workshop on Machine Learning and Music, pp. 19–24.

17. **Hillewaere, R., Manderick, B., Conklin, D. (2010).** String quartet classification with monophonic models. 11th International Society for Music Information Retrieval Conference, pp. 537–542.

18. **Hontanilla, M., Pérez-Sancho, C., Inesta, J. M. (2017).** Music style recognition with language models–beyond statistical results. Proceedings of the 10th International Workshop on Machine Learning and Music, pp. 31–36.

19. **Hontanilla, M., Pérez-Sancho, C., Iñesta, J. M. (2013).** Modeling musical style with language models for composer recognition. Iberian Conference on Pattern Recognition and Image Analysis, Pattern Recognition and Image Analysis, pp. 740–748. DOI: 10.1007/978-3-642-38628-2_88.

20. **Huron, D. (2002).** Music information processing using the humdrum toolkit: Concepts, examples, and lessons. Computer Music Journal, Vol. 26, No. 2, pp. 11–26. DOI: 10.1162/014892602760137158.

21. **Kaliakatsos-Papakostas, M. A., Epitropakis, M. G., Vrahatis, M. N. (2010).** Musical composer identification through probabilistic and feedforward neural networks. Proceedings of the European Conference on the Applications of Evolutionary Computation, pp. 411–420. DOI: 10.1007/978-3-642-12242-2_42.

22. **Kaliakatsos-Papakostas, M. A., Epitropakis, M. G., Vrahatis, M. N. (2011).** Weighted Markov chain model for musical composer identification. European Conference on the Applications of Evolutionary Computation, pp. 334–343. DOI: 10.1007/978-3-642-20520-0_34.

23. **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, Vol. 49, No. 5, pp. 457–476. DOI: 10.1080/09298215.2020.1814822.

24. **Kim, S., Lee, H., Park, S., Lee, J., Choi, K. (2020).** Deep composer classification using symbolic representation. Proceedings of the International Society for Music Information Retrieval Conference, pp. 1–3. DOI: 10.48550/arXiv.2010.00823.

25. **La-Rue, J. (2001).** The haydn-dedication quartets: Allusion or influence?. Journal of Musicology, Vol. 18, No. 2, pp. 361–373. DOI: 10.1525/jm.2001.18.2.361.

26. **Lostanlen, V. (2018).** Eigentriads and eigenprogressions on the tonnetz. arXiv. DOI: 10.48550/arXiv.1810.00790.

27. **Manning, C., Schutze, H. (1999).** Foundations of statistical natural language processing. MIT press.

28. **Manning, C. D., Raghavan, P., Schütze, H. (2009).** Introduction to information retrieval. Cambridge University Press.

29. **Mearns, L., Tidhar, D., Dixon, S. (2010).** Characterisation of composer style using high-level musical features. Proceedings of 3rd International Workshop on Machine Learning and Music, pp. 37–40. DOI: 10.1145/1878003.1878016.

30. **Ogihara, M., Li, T. (2008).** N-gram chord profiles for composer style representation. Proceedings of the 9th International

Conference on Music Information Retrieval, pp. 671–676.

31. **Pape, L., de-Gruijl, J., Wiering, M. (2008).** Democratic liquid state machines for music recognition. Speech, Audio, Image and Biomedical Signal Processing using Neural Networks, pp. 191–215. DOI: 10.1007/978-3-540-75398-8_9.

32. **Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E. (2011).** Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, Vol. 12, No. 85, pp. 2825–2830. DOI: 10.48550/arXiv.1201.0490.

33. **Pollastri, E., Simoncelli, G. (2001).** Classification of melodies by composer with hidden Markov models. Proceedings of the 1st International Conference on WEB Delivering of Music, pp. 88–95. DOI: 10.1109/wdm.2001.990162.

34. **Pérez-Sancho, C., Rizo, D., Iñesta, J. M. (2009).** Genre classification using chords and stochastic language models. Connection Science, Vol. 21, No. 2–3, pp. 145–159. DOI: 10.1080/09540090902733780.

35. **Rohrmeier, M., Graepel, T. (2012).** Comparing feature-based models of harmony. Proceedings of the 9th International Symposium on Computer Music Modelling and Retrieval, pp. 357–370.

36. **Schmid, E. F., Sanders, E. (1956).** Mozart and Haydn. The Musical Quarterly, Vol. 42, No. 2, pp. 145–161. DOI: 10.1093/mq/XLII.2.145.

37. **Taminau, J., Hillewaere, R., Meganck, S., Conklin, D., Nowé, A., Manderick, B. (2010).** Applying subgroup discovery for the analysis of string quartet movements. Proceedings of 3rd International Workshop on Machine Learning and Music, Association for Computing Machinery, pp. 29–32. DOI: 10.1145/1878003.1878014.

38. **van-Kranenburg, P. (2006).** Composer attribution by quantifying compositional strategies. The International Society for Music Information Retrieval, pp. 375–376.

39. **van-Kranenburg, P., Backer, E. (2005).** Musical style recognition—a quantitative approach. Handbook of Pattern Recognition and Computer Vision, World Scientific, pp. 583–600. DOI: 10.1142/9789812775320_0031.

40. **van-Nuss, J., Giezeman, G. J., Wiering, F. (2017).** Melody retrieval and composer attribution using sequence alignment on RISM incipits. Proceedings of 9th International Conference on Technologies for Music Notation and Representation, pp. 1–7.

41. **Velarde, G., Cancino-Chacón, C., Meredith, D., Weyde, T., Grachten, M. (2018).** Convolution-based classification of audio and symbolic representations of music. Journal of New Music Research, Vol. 47, No. 3, pp. 191–205. DOI: 10.1080/09298215.2018.1458885.

42. **Velarde, G., Weyde, T., Chacón, C. E. C., Meredith, D., Grachten, M. (2016).** Composer recognition based on 2D-filtered piano-rolls. Proceedings of the 17th International Conference on Music Information Retrieval, pp. 115–121.

43. **Verma, H., Thickstun, J. (2019).** Convolutional composer classification. arXiv. DOI: 10.48550/arXiv.1911.11737.

44. **Wołkowicz, J., Kulka, Z., Kešelj, V. (2008).** N-Gram-based approach to composer recognition. Archives of Acoustics, Vol. 33, No. 1, pp. 43–55.

45. **Yang, D., Ji, K., Tsai, T. J. (2021).** A deeper look at sheet music composer classification using self-supervised pretraining. Applied Sciences, Vol. 11, No. 4, pp. 1387. DOI: 10.3390/app11041387.

46. **Yoshii, K., Goto, M. (2011).** A vocabulary-free infinity-gram model for nonparametric bayesian chord progression analysis.

Proceedings of the International Society for Music Information Retrieval Conference, pp. 645–650.

47. **Zeng, M., Tan, X., Wang, R., Ju, Z., Qin, T., Liu, T. Y. (2021).** MusicBERT: Symbolic music understanding with large-scale pre-training. Findings of the Association for Computational Linguistics, pp. 791–800. DOI: 10.48550/arXiv. 2106.05630.

Article received on 29/09/2023; accepted on 14/12/2023.
** Corresponding author is Liliana Chanona-Hernandez.*