Pattern Extraction on Discrete Spaces: Combining Machine Learning and Mathematical Morphology for Computational Music Analysis

M2 MLDM Internship Report

Charles FRANCHI[1]

Supervisors: Moreno ANDREATTA[2], Paul MAGRON[3], Florent JACQUEMARD[4]

[1]Université Jean-Monnet, Saint-Etienne, charles.franchi@etu.univ-st-etienne.fr
[2]IRMA, Strasbourg, andreatta@math.unistra.fr
[3]Centre INRIA de l'Université de Lorraine, Nancy, paul.magron@inria.fr
[4]Centre INRIA de Paris, Paris, Florent.Jacquemard@inria.fr

March 2025 - July 2025

Abstract

This report presents connections between machine learning, mathematical morphology, and Music Information Research (MIR) for the extraction of musical patterns from symbolic music representations. Motivated by the mathematical theory of binary morphology, we investigated the use of correlation as an analytical method for pattern variations extraction, and as optimization approaches for motive discovery in music. We introduced a new loss function — the *Correlation Loss* — for unsupervised pattern learning, and tested the training of CNNs using this criterion. We created a new dataset of fugues by J. S. Bach and D. Shostakovich for use in pattern detection tasks. Despite technical challenges and dataset limitations, the experiments showed promising directions and suggested potential for pattern extraction in MIR and beyond. All the code produced during this project can be found at https://github.com/FRANCHI-Charles/MIDI-pattern-detection.

Keywords: Binary mathematical morphology, Correlation, Neural networks, CNN, BiSE, Pattern variations, Motive extraction, MIR.

1 Introduction

Repetition of motives across time is an important criterion to understand music [Cook, 1994]. Patterns help the listener to get the music structure, creating sensation of rhythm, metrics and motion. More generally, patterns presence and/or shape can define a genre, a style and can be related with musical culture [Auerbach, 2021]. Thus, identifying motives in music is a relevant question to analyze and study music and its cultural and historical context.

Pattern extraction is an open problem in Music Information Research (MIR), and it is not a well explored topics compared with other MIR tasks, such as music separation, transcription, or generation [Collins, 2017]: the difficulty to generalize the definition of a pattern and the lack of human annotations make it harder to automatize [Melkonian et al., 2019]. Despite those difficulties, attempts to create algorithms to address this task have been made, mostly using unsupervised methods [Meredith, 2006] [Scerri, 2019].

A common approach to explore this task is to use symbolic music representation. This

format have shown strong results [Meredith, 2006] and are explored extensively for their generalization possibilities.

More recently, a new point of view on the pattern extraction task appeared with link between algorithm working with symbolic music representation and binary mathematical morphology [Lascabettes and Bloch, 2024]: mathematical morphology is a powerful tool for image processing (like filtering), and is studied as a strong theoretical field. Moreover, studies showed morphological tools could be of great interest for deep learning networks, mainly for their theoretical foundation and potential help in increasing the explicability of networks [Aouad, 2024].

This work is a tentative of merging the 3 fields of Machine Learning, Mathematical Morphology and MIR. Aside this research, we created a first version of an unsupervised dataset, in the aim of exploring pattern extraction with symbolic music.

In Section 2, we present the context of this work. Next, in Section 3, we present the main tools we use and definition of the main concepts. In Section 4, we review the current method for pattern extraction, as well as links between the three different domains we explore. Then, in Section 5, we introduce the dataset we created on which we performed our experiments, before presenting our main contributions in Sections 6 & 7. Finally, we have a small discussion on our work in Section 8 and conclude in Section 9.

2 Context of the project

This work have been conducted as a second year master's degree internship. It was supervised by Moreno Andreatta, a researcher in the Structural Music Information Research team (SMIR)[1], from Institut de Recherche Mathémathiques Avancées (IRMA), as well as Paul Magron and Florent Jacquemard, two researchers from Institut national de recherche en sciences et technologies du numérique (INRIA).

The SMIR project aims at applying mathematical methods like algebra, geometry or topology to music theory, and at using these tools to solve MIR tasks, but also to approach in a new way some open mathematical problems. This team creates a collaboration context between mathematicians, computer scientists, but also musicians and musicologists.

We based our research on the work of Lascabettes [2023] and the clear link he made between mathematical morphology and geometric algorithm for pattern detection.

Our primary objective was to find possible research directions between this domain and machine learning methods.

3 Theoretical background

In this section, we introduce the different mathematical objects we worked with.

3.1 Convolution

Convolutions are powerful mathematical tools used in image and signal processing. More precisely, a convolution is a mathematical operator between two functions f and g, often associated to images, and generalizes the idea of sliding average. In our case, we will focus on the discrete formulation and use a similar definition as in Aouad [2024] :

Definition 1 (Convolution)
Let $f : \Omega_f \to \mathbb{R}$ and $g : \Omega_g \to \mathbb{R}$ be two functions with $\Omega_{\{f,g\}} \subset \mathbb{Z}^D$.
The convolution between f and g, denoted $f*g$ is defined by :

$$\forall x \in \Omega_f, (f * g)(x) = \sum_{t \in \Omega_g} f(x-t)g(t)$$

using the convention $f(x-t) = 0$ if $x-t \notin \Omega_f$.

The dimension D is equal to 2 or 3 in the image processing case, representing the height, the width and the number of color channels. The convolution can thus be seen as a reversed window g that we slide over the representative image f, x representing the translation.

[1] http://repmus.ircam.fr/moreno/smir

We can also defined the correlation of two functions with the following definition.

Definition 2 (Correlation)
With the same notations as definition 1, the correlation between f and g, denoted $f \star g$ is defined by :

$$\forall x \in \Omega_f, (f \star g)(x) = \sum_{t \in \Omega_g} f(x+t)g(t)$$

using the convention $f(x+t) = 0$ if $x+t \notin \Omega_f$.

Both definitions are similar up to a flip of the function g, which can be useful in certain contexts.

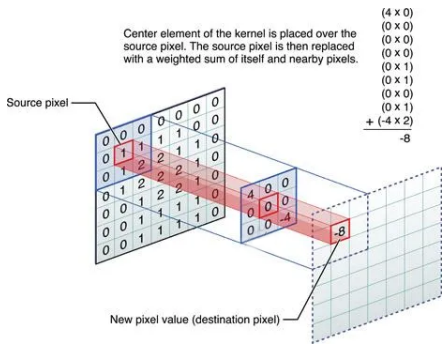


Figure 1: Correlation over an image for a single x with $g \in \mathbb{R}^{3\times3}$. [2]

This tool is the foundation of Convolutional Neural Networks (CNN), a specific type of deep neural network often used in image processing, where each layer is a convolution with the output of the previous layer (corresponding to f) and a kernel $g \in \mathbb{R}^{h_g \times w_g}$, with h_g the height and w_g the width of the kernel (as illustrated in Figure 1). In that case, the parameters learned during the training of the network are the coefficients of g.

3.2 Binary morphology

Morphology is a mathematical field that studies non-linear and non-reversible operators that modify the shape, size and properties of images and objects. The two major operations are dilation and erosion, because most morphological operations are equivalent to a combination of dilations and erosions.

Binary morphology focus on object transformations, that is to say operations on set of points. Dilation and erosion for binary morphology can be defined as follows [Lascabettes, 2023]:

Definition 3 (Dilation)
Let D be an integer.
Let $I \subset \mathbb{R}^D$, $S \subset \mathbb{R}^D$ be two sets.
The dilation of I by S, denoted $\delta_S(I)$ is defined by :

$$\delta_S(I) = \{x + s | x \in I, s \in S\}$$

S is called the structuring element, and $\delta_S(I)$ the dilated object.

Definition 4 (Erosion)
Let D be an integer.
Let $I \subset \mathbb{R}^D$, $S \subset \mathbb{R}^D$ be two sets.
The erosion of I by S, denoted $\epsilon_S(I)$ is defined by :

$$\epsilon_S(I) = \{t \in \mathbb{R}^D | S + t \subseteq I\}$$

with $S + t = \{s + t | s \in S\}$.

S is called the structuring element, and $\epsilon_S(I)$ the eroded object.

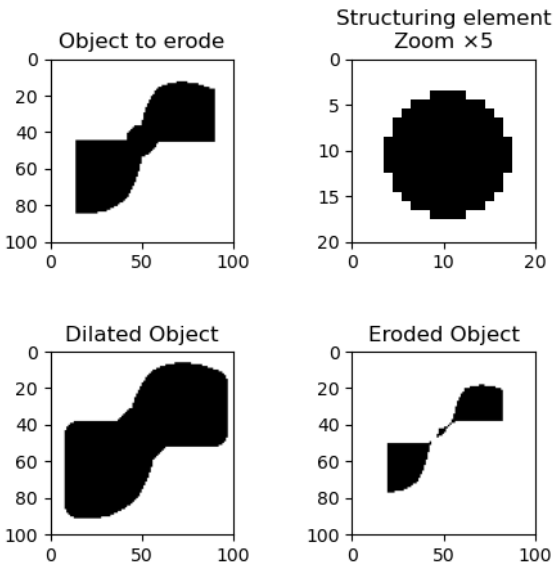


Figure 2: An example of erosion and dilation.

The dilation is as an augmentation of the object I by the object S, while the erosion can be seen as the set of points where we can place the structuring element S such as it fits in I.
These operators have several strong properties

[2] Source : https://medium.com/@bdhuma/6-basic-things-to-know-about-convolution-daef5e1bc411

and lead to important tools in image processing [Heijmans and Ronse, 1990] [Najman and Talbot, 2013].

Similar definition for binary image $I \in \{0,1\}^{h_i \times w_i}$ can be introduced [Aouad, 2024], and an important link between convolution and binary morphology operator is pointed out in Section 4.2.

3.3 Symbolic music representation

In this section, we introduce basic notions about symbolic music.

Most MIR tasks are primary based on processing the audio signal, as machine learning methods such as deep neural network can effectively process such representations. However, major breakthrough in the specific case of pattern detection were often made on symbolic music data, like geometric (point-set) approach or sequence of notes [Ren et al., 2020]

With symbolic data, music can be seen as a superposition of notes over time. A note is an event that represents a sound which has several properties. Important ones are:

- Onset : starting time when the note is played.
- Duration : the length of the note.
- Pitch : how low or high the sound will be heard.
- Velocity : the strength of the note, how loud it will be.
- Timbre : the texture of the sound, which includes properties such as the type of instrument or the technique used to play the notes.

A common format for symbolic music is `MIDI` files. `MIDI` contains information of notes, instrument, sound deformation, sound speed (tempo) and more. It is a common format to record digital instruments and to create ground truth or precise notations regarding audio. For the sake of ground truth representation, `MIDI` files are often quantized, i.e. the onset and offset times of notes are rectified to match a maximum beat division. Quantized `MIDI` can be created algorithmically to correspond to sheet music, or done greedily on recordings.

Sheet music is another common representation of music, where notes are represented using symbols of time division, and are written in such a way that we easily get time duration of the music. There is no universal computer format for this traditional representation, because companies have their own standards. For these reasons, even if this format can offer ground truth, data is less accessible and more difficult to process than `MIDI` files.

In pattern discovery, two simpler representations are commonly used : string or geometric representation [Janssen et al., 2014]. We focus on the geometric approach for their recent new perspectives given by Lascabettes [2023].

In the geometric approach, also called point-set representation, a piece of music is represented as a set of notes, and each note is a point in a multidimensional space. Each note dimension corresponds to a property (i.e. onset, duration, pitch). This format creates a more flexible representation for polyphony[3] and pattern variations [Meredith et al., 2002]. For simplicity, we often focus on only two or three features : onset, pitch, and sometimes duration. Here, we will only look at onset and pitch.

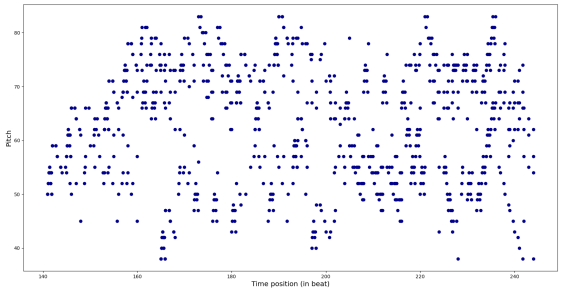


Figure 3: An example of geometric representation with J. S. Bach BWV 850.

The binary images representation is derived from this point-set representation: given a music piece, we define a minimum duration to quantized the music with, and represent the

[3]Music where several notes could be played simultaneously.

music as a tensor where each dimension represents a note feature. Every values of the tensor is 1 if it exists a note with the corresponding coordinates, 0 else. This representation will have several advantages regarding convolution and morphological operators.

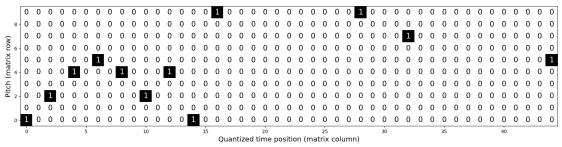


Figure 4: An example of matrix representation with the first notes of J. S. Bach BWV 850.

3.4 Patterns and pattern extraction

We use the definitions given for the Music Information Research Evaluation eXchange (MIREX) competition [Collins, 2017], because it is nowadays the definition and benchmark used to evaluate several MIR task and precisely patterns extraction.

According to Collins [2017], "a pattern is defined as a set of ontime-pitch pairs that occurs at least twice (i.e., is repeated at least once) in a piece of music". Even if other type of patterns can be obtained using other notes features [Lartillot and Toiviainen, 2007], they decided to focus on onset-pitch for the sake of simplicity.

In their definition, they also authorize pattern variations, i.e. the repetition is inexact to a certain (undefined) extent, including different distance between notes, shifted pitch or others.

Given a piece of music or a corpus of musical pieces, pattern extraction consists in extracting a reference pattern as a set of onset-pitch notes, and its repetitions as other onset-pitch notes sets, equivalent to the reference set up to translation and optionally some variations. It is important to notice these two tasks - to find a reference pattern, or to find the corresponding repetition given the reference - are often referred to as Musical Pattern Discovery and Musical Pattern Matching respectively [Janssen et al., 2014].

In Musical Pattern Matching, we sometimes do not focus on finding all the notes of the repeated patterns, but only the patterns onsets, i.e. the starting points of the occurrence of the patterns (thus, we do not aim at detecting the difference between two repetitions).

Pattern extraction can be conducted on a single musical piece, or on a whole music corpus. In the last case, there is no need for a motive to be repeated inside a single music. This task can be relevant for style or genre analysis.

In our study, we will work on single musical pieces.

4 State of the art

In this section, we briefly present recent research on pattern detection and connection between morphology and Machine Learning methods.

4.1 Pattern extraction

4.1.1 Pattern detection common issues

Before presenting some state of the art method, it is fundamental to understand current main challenges in pattern detection research.

The first problem is the lack of a formal definition of a pattern and its repetitions. As presented in Section 3.4, we define a pattern as a repetition of notes across times, but this definition is not totally consistent with a musical point of view: notions of notes proximity, pattern lengths, degree of variations or even musical "textures" and "shapes" are not determined, and create a huge gap between the listener level and the algorithm level [Tomašević et al., 2023]. This lack of concise and accurate definition makes it difficult to establish a ground truth against which evaluating pattern recognition algorithms.

The second problem is connected to the first one and is the lack of human annotations. Aside the difficulty to define what a pattern is, creating labels on a music piece is a bigger challenge. To this day, there exist only two

datasets with expert annotations: the JKU-PDD database [Collins, 2013] and the MTC-ANN dataset [van Kranenburg et al., 2016].
The JKU-PDD database is a set of 5 classical musical pieces in audio and `MIDI` format, both monophonic and polyphonic, with human annotations of 3 to 11 patterns per music. The small size of the dataset and the weakness of the annotation make the dataset not suitable for an evaluation task.
The MTC-ANN dataset is a corpus of 360 monophonic melodies of Dutch Folk Song with human annotations of patterns that occurs through all the corpus. This bigger dataset deal with the case of inter-music patterns, which is not the topic we focus on.
Moreover, researchers have discussed and criticized these two datasets' annotations, having strong disagreement on them [Ren et al., 2017]. Consequently, using those datasets may not be relevant.

Finally, other challenges have been pointed out, like the strong influence of the metrics used to measure similarity between patterns during the extraction or evaluation process [Rolland and Ganascia, 2002].

4.1.2 Geometric algorithm

Geometric or point-set algorithms refer to pattern extraction algorithms based on the geometric representation of music. This representation is detailed in Section 3.3.

The first trail with this representation is Meredith et al. [2002]. In this paper, the authors introduced the SIA algorithm: given a set of points S, SIA tests all possible translation vectors[4] $v = x_2 - x_1$ between two points $(x_1, x_2) \in S^2$, and regroup the points $x \in S$ such that the translation of x by v yields to an existing point in the set: $\{x \in S \mid \exists y \in S, y = x + v\}$. In the end, we obtain a big quantity of subsets, each one corresponding to a specific translation.

SIA lays the foundations of several algorithms to improve its results. In fact, SIA has several problems that make the musical interpretation barely relevant. For instance, SIA detects too many patterns, like noise coincidence. It also regroups all distant patterns as one single scattered motive, which is not relevant with temporal proximity in music [Collins et al., 2010].

In the same paper, the authors develop the SIATEC algorithm, a post process to SIA that also finds the corresponding exact repetition of a pattern: given a subset $P \subset S$, the algorithm searches for patterns P' such that we can find a translation $v = x_2 - x_1$ so P' is the translation of P by v.

SIA answers the Musical Pattern Discovery task, while SIATEC answer the Musical Pattern Matching task. From these algorithms, the same authors and others developped more complex algorithms to find more interesting patterns, based on musical or compression criteria [Meredith, 2006] [Ren et al., 2020]. Most of these algorithms are still considered as the state of the art of pattern extraction [Collins, 2017].

Recently, it has been shown that SIA and algorithms derivative can be expressed with morphological operators [Lascabettes, 2023]. From this observation, the authors built new technics and theorems to find more musical patterns in a more efficient way. This trail encourages to continue in this domain to find new applications and tools for pattern extraction.

With these new morphological tools, Quaetaert [2025] find a new way to discover pattern variation regarding Musical Pattern Matching, with erosion operators and duplicated data operation. Regarding classical music, this is one of the most promising results for pattern variation nowadays.

4.1.3 Other approaches for pattern detection

Other approaches exist for Musical Pattern Discovery, based on analytical methods [Lartillot, 2014], greedy approaches [Nieto and Farbood, 2014], clustering [Velarde and Meredith, 2014], or machine learning [Pesek et al., 2017] [Wang et al., 2015].

[4] Without computing both a vector v and its opposite $-v$.

To the best of our knowledge, only one deep learning based method was proposed [Scerri, 2019], based on using an LSTM on the MTC-ANN dataset, and demonstrating promising results. However, it is not suitable for processing one single musical piece at a time.

The limited use of deep learning can be explained by the issues presented in Section 4.1.1. Since there is no properly labeled dataset, supervised methods cannot be applied. Moreover, the difficulty in extracting a good definition, for instance, makes it harder to design effective unsupervised criteria.

4.2 Binary morphology, convolution & machine learning

4.2.1 Links between morphology and convolution

An important result that helps us finding new research directions is a property that connects morphological operators and thresholded convolution [Mazille, 1989]:

Property 1
Let D be an integer.
Let $I \subset \mathbb{R}^D$, $S \subset \mathbb{R}^D$ be two sets of points.
Then :

$$\epsilon_S(I) = (\mathbb{1}_I \star \mathbb{1}_S \geq |S|)$$
$$\delta_S(I) = (\mathbb{1}_I * \mathbb{1}_S \geq 1)$$

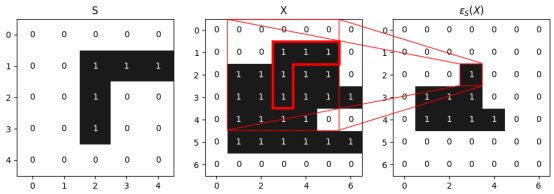


Figure 5: Example of an erosion as a correlation.

This property shows that we can interpret erosions as thresholded correlations, and dilations as thresholded convolutions (see Figure 5). Since the major operators of morphology can be framed as combinations of dilations and erosions, a large part of mathematical morphology can be expressed using thresholded convolutions, enabling us to reformulate many theorems and properties in terms of thresholded convolution.

[5]We present a simplified definition, see Aouad [2024].

4.2.2 Morphology and Machine Learning

Mathematical morphology has been explored for a long time in the context of machine learning [Ritter and Sussner, 1996]. Motivations to implement morphological operators for Machine Learning and Neural Network are of two kinds: first, Morphological operators have been studied extensively and can bring a strong theory and potential explicability to neural networks ; second, composition of mathematical operators have a strong approximation power as shown by the following result [Giardina and Dougherty, 1988]:

Theorem 1
Let $f : \mathcal{P}(\mathbb{R}^n) \to \mathcal{P}(\mathbb{R}^n)$ be any increasing, translation invariant mapping. Then:

$$\forall A \subset \mathbb{R}^n,\ f(A) = \bigcup_{S \in \text{Ker } f} \epsilon_S(A)$$

with Ker f *the kernel of* f.

This theorem shows that increasing translation invariant functions can be expressed as a union of erosion, making the erosion equivalent to the perceptron for Deep learning.

Traditional morphological methods focus on Grey-scale morphology, whereas we focus on binary morphology. A review of the evolution of morphological neural network for Grey-scale images can be found in Aouad [2024].

Only the recent thesis of Aouad [2024] explores binary morphology for neural networks. Based on the property 1, they design the Binary Morphological Neural Network (BiMoNN), a specific type of CNN based on the Binary Structuring Element (BiSE) neuron[5]:

Definition 5 (BiSE neuron)
Let ξ be a sigmoidal function.
Let $\mathcal{W} : \mathbb{R}^{h_\omega \times w_\omega} \to \mathbb{R}^{h_\omega \times w_\omega}$ and $\mathcal{B} : \mathbb{R} \to \mathbb{R}$ be reparameterization functions.
Let $p \in \mathbb{R}$, $\omega \in \mathbb{R}^{h_\omega \times w_\omega}$, $\beta \in \mathbb{R}$ be some parameters.
Let $I \in [0,1]^{h_i \times w_i}$ be an almost binary images, that is to say :

$\exists t \in [0, 0.5],\ \forall x \in I,\ x \in [0, 0.5 - t[\cup]0.5 + t, 1]$.
The BiSE neuron with parameters p, ω, β *is defined by:*

$$BiSE_{\omega,\beta,p}(I) = \xi[\ p\ (I * \mathcal{W}(\omega) - \mathcal{B}(\beta))]$$

applying ξ, p *and* $\mathcal{B}(\beta)$ *element-wise.*

The BiSE neuron is similar to a convolution where learnable weights correspond to the structuring element. However:

1. It systemized the use of a sigmoidal function as activation function.
2. The reparameterization $\mathcal{W}$ and $\mathcal{B}$ are set to force the parameters to satisfy certain properties (e.g. positiveness of the bias, rectified weights)
3. The parameter p increases the values of ω to reach the limits of the sigmoidal function.
4. Under some conditions on the parameters if the training succeed, the BiSE neuron can be strictly equivalent to an erosion or a dilation.

In practice, a combinations of a few number of BiSE neurons can successfully learn a composition of erosion/dilation on a set of images[6], and BiMoNN yields promising performance on a disease detection task [Aouad, 2024].

We are interested in this type of architecture because, as current geometric approaches are nearly all equivalent to morphological operators with a specific structuring element, we assume we can obtain interesting results by learning the structuring elements statistically. Furthermore, this architecture deals with binary images, which is convenient in our scenario.

5 Fugues dataset

In this Section, we present the dataset we created and used in our work.

J. S. Bach is a well-studied German composer of the baroque period. He composed more than a thousand music pieces through all his career, among which several fugues. The fugue is a specific compositional technique where a theme, named *subject*, is repeated all along the piece, respecting several rules - for instance, the piece always starts with the subject, making it identifiable. The subject is also transposed[7] to a specific degree (the dominant of the main scale), which is called the *response* (or *answer*), and is repeated after the subject. The response could have some variations as well as the pitch-translation, and part of the subject with potential modification can be repeated inside the piece.

The easily identifiable pattern and the known presence of variations make this musical structure suitable for both Musical Pattern Discovery and Musical Pattern Matching tasks.

As there is no dataset of ground truth `MIDI` files of fugues, we built one from different `MIDI` files available under Creative Commons BY-NC license from two sources : The Tobi's score archive[8] and The Shostakovich Opus 87 Page[9]. The Tobi's score archive gathers `MIDI` files and other music format of J. S. Bach's and W. A. Mozart's work with quantized recordings not perfectly line up on the ground truth sheet music. Among other, it provides "The Well tempered Clavier", "Preludes and Fugues", "Fantasia and Fugues", and "Fugues", BWV 846-909, 944-962[10], for a total of 83 fugues. The Shostakovich Opus 87 Page regroup manual `MIDI` encoding by José Oscar de Almeida Marques of Shostakovich Opus 87 work. D. Shostakovitch is a composer of the twentieth century who composed a series of 24 Preludes and Fugues (Opus 87) following the model of Bach's Well Tempered Clavier.

We look more specifically into the content of 111 `MIDI` files. 64 files of Bach were longer pieces, and we had to remove the non-fugue part by hand. 5 files were of poor quality and

[6] From our own tests, we can confirm the reproducibility of the results.
[7] Translated in pitch
[8] https://tobis-notenarchiv.de/wp/en/startseite-english/
[9] https://unicamp.br/~jmarques/mus/opus87/
[10] Identification number of Bach work.

were therefore excluded.

Then, we had to compute a geometric representation from these files. To do so, we quantized each file to some precision using fractions notation and round notes to their nearest division. We obtained a list of onset-pitch pairs for all the pieces. As the `MIDI` files where correctly quantized, this basic method succeeded for most of the files. However, with greater division as preprocessing, we were able to look into the potential issues of this trivial quantization. Some `MIDI` files were actually not precise enough regarding the ground truth sheet musics. We solved these issues by preprocessing the files with Musescore quantization, a license-free music editor, and deleting by hand the extra tempo indicators on most of the files.

Finally, we looked into basic information of the dataset, like maximum duration, minimum and maximum pitches, or number of notes distribution. As this dataset will be used to train machine learning models, the size of the data will have an impact on training, especially as all the data must have the same shape as binary image. For this reason, we decided to remove some specific pieces[11] because they were too long or to wide in height.

As a result, there are two versions of our dataset: a normal version containing all 106 pieces with no constraint on their lengths, and a reduced version containing 96 pieces with shorter/same length.

6 Correlation for patterns and variation

In this section, we present our contribution for correlation to detect patterns variations for the Musical Pattern Matching task.

6.1 Relaxed Erosion

As seen in Section 4.2, a morphological erosion is similar to a correlation with a maximum threshold. Based on this observation, we decided to study the effect of the threshold on the retrieved pattern's onsets. In particular, to allow a threshold to be less than the total size of the pattern is equivalent to allow occurrences of partial patterns.

To do so, we define the Correlation Map as follows:

Definition 6 (Correlation Map)
Let D be an integer.
Let $I \subset \mathbb{R}^D$, $S \subset \mathbb{R}^D$ be two sets of points.
The Correlation Map is the function $\text{corr}_S(I) : \mathbb{R}^D \to [0, 1]$ *such that:*

$$\forall x \in \mathbb{R}^D,\ \text{corr}_S(I)(x) = \frac{(\mathbb{1}_I \star \mathbb{1}_S)(x)}{|S|}$$

In fact, an erosion corresponds to the points of the correlation map where the value is 1:

$$\epsilon_S(I) = \{x \in \mathbb{R}^D \mid \text{corr}_S(I)(x) = 1\}$$

So the Correlation Map relaxes the erosion and creates a more continuous heatmap with high points where a great part of the pattern S is located.

Further theoretical investigation may help to understand the meaning of mean-hot area, like stretched patterns, patterns superposition or variations.

6.2 Test on Bach Fugues

In order to first evaluate this method, we decided to test the Correlation function on the J. S. Bach fugues, as presented in Section 5. We tested several pieces of the Well Tempered Clavier, using the annotations from Giraud et al. [2015] to extract the subject, freely available on Dezrann[12], and compared our results with the ones of Quaetaert [2025].

6.2.1 Method

For our first test, we directly looked at the values of the thresholded correlation. Actually, it means we focused on cropped patterns, because a threshold of t means we only have $100 \times t$ % of the original pattern's notes.

Cropped patterns is a specific kind of variation. However, as a pattern's variation often

[11] From Opus 87: 4, 10, 12, 15, 20, 21, 23, 24. For Bach : BWV 909 and 944.
[12] https://www.dezrann.net/explore/bach-fugues

does not concern the entire pattern, detecting the onset of cropped pattern's should coincide with some pattern's other type of variation, like non-uniform translations.

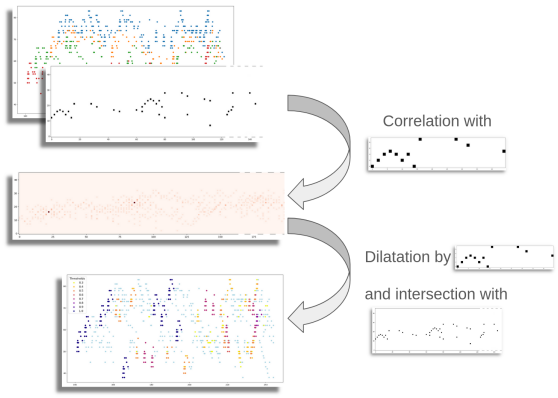


Figure 6: The detection method. On top, the music piece as a point-set, and as binary image (black dots image above it). In the middle, the correlation map. At the bottom, the threshold map.

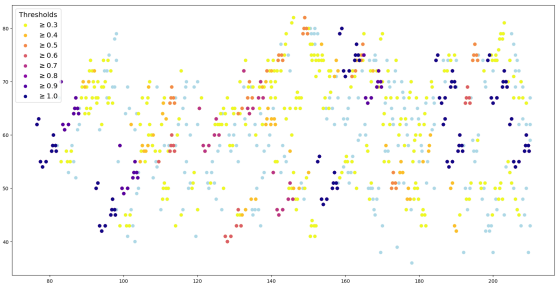
The method is illustrated in Figure 6. For every pieces, we extracted the subject with the known annotations as a binary image convolution kernel and applied the Correlation Map operator on the whole music with the subject. From the obtained heat-map, we extracted the points/onsets with correlation greater than a defined threshold $0 < t \leq 1$ and found the corresponding notes in the original music by applying the dilation of the onsets by the pattern [Lascabettes, 2023], and intersected it with the original image (to only have existing notes).

6.2.2 Results

We tested our method on several pieces, including BWV 846, BWV 847, BWV 850, BWV 861 and BWV 900. Figures 7a & 7b present the results on two pieces: BWV 850 and BWV 861.

The different tests we have conducted show promising results. In all cases, with a threshold around 0.4-0.5 (around 5 notes per pattern), we discover at least as much patterns onsets as Quaetaert [2025]. The extra onsets are consistent with the annotation of Giraud et al. [2015], so we do not obtain clear irrelevant instances. However, even if the retrieved points are consistent regarding the extracted notes, it does not mean the pattern's onsets are well retrieved: for instance, it may appear 1 full pattern with variations made with 2 onsets with a threshold of 0.5 (meaning we have 100% of the notes at the end). In that case we do not know at the moment how to select the true pattern onsets among the 2 ones or a combination of both.

(a) BWV 850. The subject is composed of 12 notes.



(b) BWV 861. The subject is composed of 10 notes.

Figure 7: Thresholded correlation with different thresholds. The light-blue points are the remaining points.

As expected, the selection of the threshold is crucial to obtain good results, and it is not constant from one piece to another. For instance, with a threshold of 0.3, we still obtain notes that might be interesting for BWV 850, whereas it is only noise on BWV 861 ; on the other hand, the first detected variation/cropped pattern occurs with a threshold of 0.8 for BWV 850, whereas we obtain the first ones (the answer of the fugues) with 0.9 on BWV 861. On the tested instances, a consistent threshold seems stable around 0.4-0.5 to find pattern variations, but we do not exclude this result to be a coincidence. A more general result might be obtained by expressing the threshold in terms of the pattern size, but as music is an art, exception will always occur.

7 Correlation as an unsupervised criterion

In this section, we present a new loss function based on the correlation, as well as first results on a basic optimization problem and a first attempt to train a CNN for Musical Pattern Discovery.

7.1 Correlation Loss function

From the Correlation Map and the experiments made for variation detection, we assumed representative patterns should have a great correlation on the corresponding piece. For this reason, we tried to design an unsupervised criterion with correlation.

To do so, we adapt our Correlation Map to generalize it for a non binary kernel S:

Definition 7 (Correlation Map)
Let D be an integer.
Let $I \subset \mathbb{R}^D$ be a set of point.
Let $S : \mathbb{R}^D \to \mathbb{R}_+$ be a weighted object such that $\|S\| = \int_{\mathbb{R}^D} S(x)dx < +\infty$.
In this more general context, the Correlation Map is the function $\mathrm{corr}_S(I) : \mathbb{R}^D \to [0,1]$ such that:

$$\forall x \in \mathbb{R}^D,\ \mathrm{corr}_S(I)(x) = \frac{(\mathbb{1}_I \star S)(x)}{\|S\|}$$

This definition allows a smoother version for optimization work.
We can discretize S as a convolution's kernel $S \in [0,1]^{h_s \times w_s}$, and thus $\|S\| = \sum_{s \in S} s$, for use in CNN.

With this more general definition, we define a loss function we called *Correlation Loss* to maximize the sum of the correlation between an image and a convolution kernel:

Definition 8 (Correlation Loss)
Let $P : [0,1] \to [0,1]$ be an increasing differentiable function.
Let D be an integer.
Let $I \subset \mathbb{R}^D$ be a set of points.
Let $S \in [0,1]^{h_s \times w_s}$ be a weighted object expressed as a kernel.
The Correlation Loss, noted $\mathrm{CorrLoss}_P(I,S)$ is defined by:

$$\mathrm{CorrLoss}_P(I,S) = -\frac{1}{|I|}\sum_{x \in I} P(\mathrm{corr}_S(I)(x))$$

Minimizing the Correlation Loss regarding S is equivalent to maximizing the sum of the correlation heat-map.

In practice, P is a polynomial function $P(x) = x^n,\ n \in \mathbb{N}$ which reduces the impact of small values and highlights points of S with nearly 1 value.

In this context, minimizing the Correlation Loss with respect to S yields several already known global minima: the trivial pattern with only one note (see Figure 8b) is a global minimum, because all notes of I have a perfect correlation, so $\mathrm{CorrLoss}_P(I,S) = -1$, which is minimal[13].

In fact, the amplitude of this single note can be anything in $[0,1]$, and this note can be anywhere in the kernel. Thus, there is an infinite amount of trivial global minima.

This result means that without regularization on the size of S, we will probably not obtain any non-trivial patterns.

Furthermore, we will see in Section 7.2 the Correlation Loss without regularization has a great probability to learn patterns that only correspond to regular rhythm.

To address these problems, we added a regularization term to the loss to constrain the pattern size.

In this smooth context, using the sum of all coefficients of S as a measure of size is not meaningful, since S would have the same size whether all its coefficients are small but non-zero, or if there is a single large coefficient $s \in S,\ s \gg 0$ while the remaining coefficients are zero.

We then focus on the large elements of the kernel, that is to say coefficients greater than 0.5, denoted $S_{0.5} \in \mathbb{R}^{h_s \times w_s}$:

$$\forall a \in [\![1, h_s]\!],\ \forall b \in [\![1, w_s]\!],$$
$$S_{0.5\ a,b} = \begin{cases} S_{a,b} \text{ if } S_{a,b} \geq 0.5 \\ 0 \text{ else} \end{cases}$$

[13] As the maximum value of I is 1, $\mathrm{corr}_S(I)(x) \leq 1$ and $\mathrm{CorrLoss}_P(I,S) \geq -1$

By doing so, we avoid reducing the coefficient amplitude uniformly and promote maximizing a few coefficients among others.

We then define the *Regularized Correlation Loss*:

Definition 9 (Regularized Correlation Loss) *Let $P : [0,1] \to [0,1]$ be an increasing differentiable function.*
Let $\beta \in \mathbb{R}_+$ and $m \in \mathbb{R}_+$ be hyper-parameters.
Let D be an integer.
Let $I \subset \mathbb{R}^D$ be an object.
Let $S \in [0,1]^{h_s \times w_s}$ be a weighted object expressed as a kernel.
The L1-Correlation Loss, noted $\text{L1-CorrLoss}_{P,\beta,m}(I,S)$ *is defined by:*

$$\text{L1-CorrLoss}_{P,\beta,m}(I,S) = \text{CorrLoss}_P(I,S) + \beta \left| \|S_{0.5}\| - m \right|$$

The L2-Correlation Loss, noted $\text{L2-CorrLoss}_{P,\beta,m}(I,S)$ *is defined by:*

$$\text{L2-CorrLoss}_{P,\beta,m}(I,S) = \text{CorrLoss}_P(I,S) + \beta \left(\|S_{0.5}\| - m \right)^2$$

We made some comments on the definitions of these loss functions:

- In these new definitions, m acts like a center-point of the number of high coefficients. A large β force the number of high coefficients in the kernel to be as close as possible to m, while a small β do not penalize the size so much.
- The difference between the L1 and L2 regularizations is in the smoothness of the regularization: the L1 regularization will have a more uniform penalization of the size whereas the L2 regularization will penalize more a strong deviation while not focusing on small errors.
- In practice, we obtained a batch version for several musical piece with distinct kernel S of this loss looking for correlation image by image and summing the losses. We also added scales on the different terms of the loss to have classical values for β (between 0 and 1).

[14] Because a convolution is not convex in general.

The derivative of this loss has not been theoretically studied for now: the optimization is performed in practice using a classical back-propagation process as for a neural network.
Finally, for the extraction of several patterns on a same music piece, we believe another regularization can be done to ensure the difference of the extracted patterns by penalizing the similarity between 2 patterns. We did not test this regularization yet.

7.2 Tuning the loss on a basic optimization problem

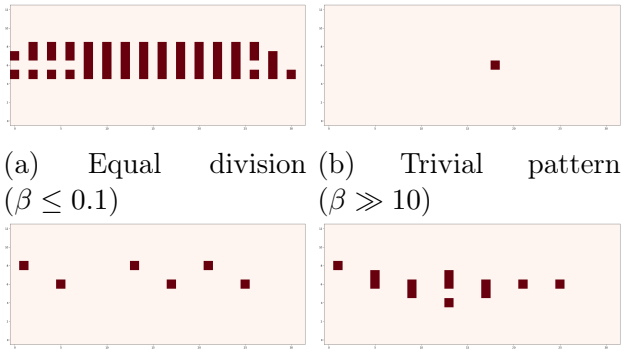
We studied a first simple optimization problem to explore potential good hyper-parameters, to rectify issues, and to have a first insight on the performance of the loss.

The problem is: given a binary image I, we search $\hat{S} \in [0,1]^{h_s \times w_s}$ the optimal kernel that maximizes the correlation:

$$\hat{S} = \underset{S}{\arg\min}\, \text{L}\{1,2\}\text{-CorrLoss}_{P,\beta,m}(I,S)$$

To do so, we performed small test on J. S. Bach Fugues, as presented in Section 5. As this problem is non-convex[14], we cannot solve this problem formally. Consequently, we used a gradient descent approach and look for local minima.
In practice, we searched for $\hat{S} = \sigma(p \times W)$ and optimize $W \in \mathbb{R}^{h_s \times w_s}$ and $p \in \mathbb{R}$, with σ the sigmoid function $\sigma(x) = \frac{1}{1+e^{-x}}$. The parameter p acts as a boost to reach the sigmoid limits.

We tested it on several pieces from BWV 846 to BWV 908, with several quantization (with a minimum duration corresponding to 8^{th} or 16^{th} note) and tried to find the best hyper-parameters.



(a) Equal division ($\beta \leq 0.1$)

(b) Trivial pattern ($\beta \gg 10$)

(c) Common pattern with $1 \leq \beta \leq 10$

(d) Common pattern with $0.1 \leq \beta \leq 1$

Figure 8: Different $\hat{S}$ by tuning the Correlation Loss.

We originally started without regularization, expected the optimization to naturally gives the trivial pattern (Figure 8b). As shown in Figure 8a, we quickly noticed the opposite effect was happening: the model optimization leads to regular spaced points. It can be interpreted as a sequence of regular notes with pitch variations on the minimum rhythm division (Tatum). It is a consistent result as rhythm is an important aspect of music ; however, it is not an interesting musical pattern.

We suggest that the reason of the large size of the kernel could be due to the difficulty to privilege a specific point in the kernel during gradient descent.

We decided to add a regularization (Definition 9) to decrease the pattern size. We tested the L1 regularization with $m = 0$ first. With this setup, we found more interesting results. By setting a β whose value is too large, we sometimes obtained the trivial pattern (Figure 8b), which is the expected behavior. In this scenario, increasing β gave a smooth decrease of the number of notes in the pattern, without strong gaps.

To tune the size of S in a more flexible way, we start looking at the m hyper-parameter and test the L2 regularization. From now, we easily find more pattern-like results, as show in Figure 8c. As we could imagine, we did not obtain one single pattern with maximum correlation, however part of this pattern often occurs. Other interesting results include muddled pattern with several sub-patterns, or superposed variations of a pattern which occur in the piece, as in Figure 8d - we never have two notes at one in height of distance like in this pattern, but the global shape of the pattern is relevant several times in the musical piece.

However, even with a big β, a small change of ± 1 on m does not necessarily change the number of points in the final pattern in a noticeable way.

We can make several additional comments.

The interaction between β and the smooth function P is sensitive. A change in the exponent of P may conduct to a reevaluation of the range of values of β.

Also, β does not seem to be highly correlated with the input, but may vary a bit. An adaptive β could lead to better results.

Regarding the training, the loss value is nearly stable before finding a local minimum hole and having a quick decrease before stopping.
The training step on the loss is pretty low, meaning small gradient may provoke vanishing gradient in a more complex scenario.

Finally, we also scaled the different terms of the regularization to have a consistent and easily tunable hyperparameter β in our implementation (the values in Figure 8 take into account this scaling).

In the end, we had a good view on the impact of the hyperparameters, and decided to test the loss function in a deep learning setup.

7.3 A first CNN attempt

In this section, we present an experiment where we train a CNN to extract patterns using the Correlation loss.

7.3.1 Experiments

Based on the work presented in Section 7.2, we tried to learn a CNN that extracts a relevant pattern on the Bach and Shostakovich Fugues, presented in Section 5.

The architecture consist of a CNN with batch-norm regularization and ReLU activation function, with a final Dense Layer, that amount to about 100,000 parameters. The detailed architecture can be found in Appendix B.

The input is a 2 dimensional binary image that represents a fugue, and the output is a smooth pattern S expressed as a kernel of size (h_s, w_s) with values in $[0, 1]$. We apply the Correlation Loss between S and the input and minimize it using gradient descent with ADAM optimizer method [Kingma and Ba, 2014]. We used a learning rate scheduler, where the learning rate decreases by 10 if the loss on the validation set does not yield a new minimum after 5 epochs. Training stops when a reduced learning rate also does not yield a lower validation loss.

We trained on the Fugues Dataset (Section 5) and used a data augmentation method: for every musical piece, we applied a horizontal flip in order to reverse the note pitches. In this way, we doubled the size of the dataset without introducing a strong bias. This augmentation is relatively consistent for music. Doing it on the all piece imply reversing the music scale and going outside common major and minor scales.
Other augmentations could be implemented in future works, like time translation or pitch transposition.

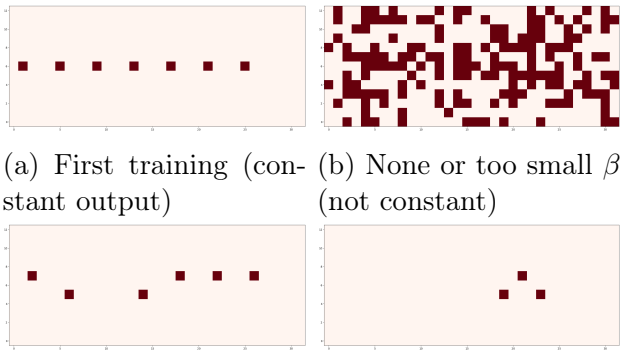
All the pieces have been standardized to tensors using an 8^{th} note quantization, with the maximum time length as number of rows and the maximum pitch amplitude as number of columns. In this way, we were able to batch the inputs and did not need to deal with size variation for our network.

We divided the dataset as 72% for training, 18% for validation, and 10% for testing.

We implemented our model with PyTorch[15] in Python language and ran our training on a local cluster[16]. Different hyperparameters and small modification on the architecture have been tested.

7.3.2 Results

We now present some preliminary results of our first trainings. We tried different hyperparameters configuration, and all of the training last at maximum 10 minutes.



(a) First training (constant output)
(b) None or too small β (not constant)
(c) Rhythm variations (constant output)
(d) Pitch variations (constant output)

Figure 9: Different Results of the first CNN trainings.

The first experiment we ran already gave an interesting outcome. The model was able to learn a pattern from the dataset, however it was a constant sequence of equally spaced notes of same pitch for all input (Figure 9a). The model was able to learn using the loss - attesting it can be actually relevant -, and the model learned a musical pattern and not a random pattern, even if it did not bring new knowledge about the music. This showed promising possibilities.

Regarding the constant output, we tried using a batch size of 1 instead of a mini-batch optimization of several pieces so we performed a more noisy optimization and could help to go through more unexpected minimum. At the end, it was still constant ; however, we obtained more refined patterns, like irregular rhythm or small pitch variation of 1 or 2 pitches (Figure 9c & 9d).

Next, we assumed that a constant output may happen because of biases. We ran several other experiments removing partial and finally all biases. We tested it with the same other hyper-parameters, without more success.

Finally, since we previously observed that the loss is rather β-sensitive in Section 7.2, we tested different values for β. A small β

[15] https://pytorch.org/
[16] Details of the software used can be found in Appendix C.

gives non constant but noisy maps (Figure 9b), while big β can lead to the trivial pattern (as in Figure 8b). With the current other hyper-parameters, there was no other outcome that the first experiment scenario or the noisy maps by tuning β.

The issue of constant output could come from a loss drawback: even by considering a more general pattern than the trivial one, given a pattern S and a subpart of it $S' \subset S$, we observe that: $\text{CorrLoss}_P(I, S) \geq \text{CorrLoss}_P(I, S')$. That is to say, using this loss does not encourage to learn a pattern that maximize the size when a pattern is found. By fixing a specific size with the regularization, it may be easier for the model to learn a mean pattern resulting of the intersection of better patterns across the entire corpus instead of distinct patterns for each music piece. We will investigate more in this direction in future works.

We assume other several issues could be addressed to improve the results:

- The smooth function P could have a great impact on the results, a grid search along with β could help to find better results.
- The parameters initialization may not be optimal.
- As we used ReLU, some parts of the network could be deactivated if all values are negative before activation.
- The constant issue looks like an underfitting scenario, thus we may have too many regularization (the batch-norms), or not enough parameters.
- The ADAM optimizer used some momentum that may not be optimal in our setup: when tuning the loss, the momentum inertia sometimes seems to block the progression, so more stochastic descent could help.

8 Discussion & future work

In this section, we review our different contributions and point out some drawbacks. We also present some possible trails for future research directions.

First, at this point, the original `MIDI` files are not fully reliable for the Fugues dataset, as it is not ground truth annotations and the quantization is not perfect. Moreover, our own additional quantization leads to approximation errors that may cause issues when used in more exacting tasks. We should do more data augmentation and find better sources to improve it.

Regarding Correlation for pattern variations detection, we only look at preliminary results and a lot of work should be done to improve and look deeper into the method. Correlation is a known powerful tool, and there must be state of the art methods in other domains that may inspire us to improve our results. Also, the link between fuzzy morphology [Bloch and Maître, 1995] and correlation should be explored.

Moreover, we want to outline that we only did visual tests to compare our discovery with the work of Quaetaert [2025] or to compare with the annotations on Giraud et al. [2015], because the algorithm were not easily available in the first case, and because the data where difficult to extract in the second case. Since we consider music represented as binary images, visual data analysis is relevant because the data is readable and easily interpretable. However, statistical tests should be done to ensure the reliability of the results.

We also did not conduct tests on a large amount of samples (only about 10 fugues), which is not enough to draw strong conclusions.

This first insight encouraged us to look deeper into the correlation map to extract more information. Future works may focus on retrieving the remaining notes that are varied in the pattern occurrence, on better extracting the exact pattern's onsets and on finding better method to get the variations from the Correlation map. It could include analytical and greedy methods as well as machine learning approaches like deep learning.

Next, the Correlation loss function should be improved. The loss is still sensitive to a lot of phenomena and we cannot extract them all. Even with the added regularization terms to reduce the impact of input and kernel size, other terms and scaling may be included to improve it.

As pointed out in Section 7.3.2, the motive size is not taken into account in an optimal way for now, and it may have a crucial impact during the optimization process.

Also, we trained on one example at a time to tune the hyper-parameters. This example could include noises and errors due to quantization. It may lead to learning error and may impact our deduction on the loss behavior.

Finally, we do not have an expert point of view on the results on the obtained patterns. Even if they are consistent with our own musical knowledge, we cannot attest the musicality relevance.

For the pattern extraction with CNN, several trails should be studied, as the impact of the optimizer or the activation functions. The optimal hyper-parameters and optimization are not known, and we should explore deeper their impact on training.

In future works, we will explore these different scenarios with the aim to find more interesting patterns. We also plan to use the BiSE neurons (Section 4.2) instead of the normal convolution to see if we get better results, even if the risk of vanishing gradient increases with the use of sigmoid activation functions instead of ReLU.

Finally, in a more general way, we want to explore more the correlation tool as well as their connection with mathematical morphology, using known results of convolution and erosion. We think the Correlation Map obtained by applying a pattern on a musical piece can provide new musical knowledge, and both statistical and analytical method should be considered. We also want to explore more the possibility to use the correlation as an unsupervised criterion, by improving the Correlation loss or designing new losses.

9 Conclusion

Through this project, we reviewed and designed new connections between Binary Morphology, Machine Learning and Music. We found several results that may have an important impact for future work in this interdisciplinary domain, and explored a new way to deal with patterns using the correlation.

The Correlation Loss function we have designed offers a new point of view on the interaction between Machine Learning and Symbolic music. We also think this new function could be generalized to other domain, like in image processing.
Future work will help to improve this trail to use unsupervised criteria for Musical Pattern Extraction and Musical Pattern Matching.

Generally, we think our work could be extrapolated to other pattern detection tasks and more general point-set scenario, and we hope to establish new connections in the future.

References

T. Aouad. *A Foundation For Binary Morphological Neural Networks*. PhD thesis, Université Paris-Saclay, 2024.

B. Auerbach. *Musical Motives: A Theory and Method for Analyzing Shape in Music*. Oxford University Press, 2021.

I. Bloch and H. Maître. Fuzzy mathematical morphologies: a comparative study. *Pattern recognition*, 28(9):1341–1387, 1995.

T. Collins. Music information retrieval evaluation exchange (mirex), 2013. URL https://www.music-ir.org/mirex/wiki/2013:Discovery_of_Repeated_Themes_%26_Sections.

T. Collins. Discovery of repeated themes & sections, 2017. URL https://www.music-ir.org/mirex/wiki/2017:Discovery_of_Repeated_Themes_%26_Sections.

T. Collins, J. Thurlow, R. Laney, A. Willis, and P. Garthwaite. A comparative evaluation of algorithms for discovering transla-

tional patterns in baroque keyboard works. 2010.

N. Cook. *A guide to musical analysis*. OUP Oxford, 1994.

C. R. Giardina and E. R. Dougherty. *Morphological methods in image and signal processing*. Prentice-Hall, Inc., 1988.

M. Giraud, R. Groult, E. Leguy, and F. Levé. Computational fugue analysis. *Computer Music Journal*, 39(2):77–96, 2015.

H. J. Heijmans and C. Ronse. The algebraic basis of mathematical morphology i. dilations and erosions. *Computer Vision, Graphics, and Image Processing*, 50(3):245–295, 1990.

B. Janssen, W. B. De Haas, A. Volk, and P. Van Kranenburg. Finding repeated patterns in music: State of knowledge, challenges, perspectives. In *Sound, Music, and Motion: 10th International Symposium, CMMR 2013, Marseille, France, October 15-18, 2013. Revised Selected Papers 10*, pages 277–297. Springer, 2014.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

O. Lartillot. Patminr: In-depth motivic analysis of symbolic monophonic sequences. In *Music Information Retrieval Evaluation eXchange*, 2014.

O. Lartillot and P. Toiviainen. Motivic matching strategies for automated pattern extraction. *Musicae Scientiae*, 11(1_suppl):281–314, 2007.

P. Lascabettes. *Mathematical models for the discovery of musical patterns, structures and for performances analysis*. PhD thesis, Sorbonne Université, 2023.

P. Lascabettes and I. Bloch. Discovering repeated patterns from the onsets in a multidimensional representation of music. In *International Conference on Discrete Geometry and Mathematical Morphology*, pages 192–203. Springer, 2024.

J. Mazille. Mathematical morphology and convolutions. *Journal of Microscopy*, 156 (1):3–13, 1989.

O. Melkonian, I. Y. Ren, W. Swierstra, and A. Volk. What constitutes a musical pattern? In *Proceedings of the 7th ACM SIGPLAN International Workshop on Functional Art, Music, Modeling, and Design*, pages 95–105, 2019.

D. Meredith. Point-set algorithms for pattern discovery and pattern matching in music. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, 2006.

D. Meredith, K. Lemström, and G. A. Wiggins. Algorithms for discovering repeated patterns in multidimensional representations of polyphonic music. *Journal of New Music Research*, 31(4):321–345, 2002.

L. Najman and H. Talbot. *Mathematical morphology: from theory to applications*. John Wiley & Sons, 2013.

O. Nieto and M. Farbood. Music segmentation techniques and greedy path finder algorithm to discover musical patterns. In *Music Information Retrieval Evaluation eXchange (MIREX)*. 2014.

M. Pesek, A. Leonardis, and M. Marolt. Symchmmerge: An extension to the compositional hierarchical model for pattern discovery in symbolic music representations. In *18th International Society for Music Information Retrieval Conference*, 2017.

N. Quaetaert. Discovery of musical patterns and their variations with mathematical morphology. 2025.

I. Ren, A. Volk, W. Swierstra, and R. C. Veltkamp. A computational evaluation of musical pattern discovery algorithms. *arXiv preprint arXiv:2010.12325*, 2020.

I. Y. Ren, H. V. Koops, A. Volk, and W. Swierstra. In search of the consensus among musical pattern discovery algorithms. In *Proceedings of the 18th International Society for Music Information Retrieval Conference*, pages 671–678. ISMIR press, 2017.

G. X. Ritter and P. Sussner. An introduction to morphological neural networks. In *Proceedings of 13th International Conference on Pattern Recognition*, volume 4, pages 709–717. IEEE, 1996.

P.-Y. Rolland and J.-G. Ganascia. Pattern detection and discovery: The case of music data mining. In *Pattern Detection and Discovery: ESF Exploratory Workshop London, UK, September 16–19, 2002 Proceedings*, pages 190–198. Springer, 2002.

E. Scerri. An approach for automated pattern discovery in symbolic music with long short-term memory neural networks. Master's thesis, 2019.

D. Tomašević, S. Wells, I. Y. Ren, A. Volk, and M. Pesek. Exploring annotations for musical pattern discovery gathered with digital annotation tools. In *Pattern in Music*, pages 100–113. CRC Press, 2023.

P. van Kranenburg, B. Janssen, and A. Volk. The meertens tune collections: The annotated corpus (mtc-ann) versions 1.1 and 2.0. 1. *Meertens Online Reports*, 2016(1), 2016.

G. Velarde and D. Meredith. A wavelet-based approach to the discovery of themes and sections in monophonic melodies. In *International Symposium on Music Information Retrieval: ISMIR*, 2014.

C.-i. Wang, J. Hsu, and S. Dubnov. Music pattern discovery with variable markov oracle: A unified approach to symbolic and audio representations. In *ISMIR*, pages 176–182, 2015.

Appendix A Abbreviations and notations

BiMoNN : Binary Morphological Neural Network. See Section 4.2.

BiSE : Binary Structuring Element (neuron). See Section 4.2.

CNN : Convolutional Neural Network. A specific type of Deep Neural Network where each layer use a convolution, and the parameters learn are the weights of the kernel. See Section 3.1.

MIR : Music Information Research. The interdisciplinary science of retrieving information from music like at different levels, like style, genre, notes onsets, patterns, ...

MIREX : Music Information Retrieval Evaluation eXchange. A competition and benchmark for several MIR tasks. https://www.music-ir.org/mirex/wiki/MIREX_HOME

ReLU : Rectified Linear Unit. A specific type of activation function.

In all the paper, D is an integer.

$f * g$: the convolution of f by g. See Definition 1.

$f \star g$: the correlation of f by g. See Definition 2.

$\delta_S(X)$: the dilation of X by S. See Definition 3.

$\epsilon_S(X)$: the erosion of X by S. See Definition 4.

$\text{corr}_S(X)$: the correlation map of X by S. See Definition 6 & 7.

$\text{CorrLoss}_P(I, S)$: the Correlation loss between I and S. See Definition 8.

$\text{L1-CorrLoss}_{P,\beta,m}(I, S)$ and $\text{L2-CorrLoss}_{P,\beta,m}(I, S)$: regularized Correlation loss. See Definition 9.

σ : the sigmoidal function : $\sigma(x) = \frac{1}{1+e^{-x}}$

$\mathcal{P}(A)$: the set of subset of A: $\{X \mid X \subset A\}$.

$\mathbb{1}_X : \mathbb{R}^D \to \{0, 1\}$: function such that $\mathbb{1}_X(x) = 1$ if $x \in X$, 0 else.

$X_{0.5}$: the function X with a 0.5 threshold on the values. See Section 7.

$|S|$: the size of an object S (area, volume or hyper-volume).

$\|S\|$: the size of a weighted object S, $\|S\| = \int_{\mathbb{R}^D} S(x)dx$. If $S \in \mathbb{R}^{h_s \times w_s}$ is a kernel, $\|S\| = \sum_{s \in S} S$.

$[\![a, b]\!]$: the set of integers between a and b, boundaries included.

Appendix B Architecture of the CNNs

We present here the detailed architecture use for the experiments described in Section 7.3.1.

The network is built as follow:

1. A convolution layer with a kernel of size $(9, 13)$ and 3 output channels.
2. A maximum pooling layer with a window of size $(6, 1)$.
3. A ReLU activation function.
4. A batch-norm regularization.
5. A convolution layer with a kernel of size $(9, 13)$ and 6 output channels.
6. A maximum pooling layer with a window of size $(6, 1)$.
7. A ReLU activation function.
8. A batch-norm regularization.
9. A convolution layer with a kernel of size $(9, 13)$ and 12 output channels.
10. A maximum pooling layer with a window of size $(6, 1)$.
11. A ReLU activation function.
12. A batch-norm regularization.
13. A convolution layer with a kernel of size $(9, 13)$ and 24 output channels.
14. A maximum pooling layer with a window of size $(6, 4)$ and a dilation of $(1, 13)$.
15. A ReLU activation function.
16. A batch-norm regularization.
17. A Dense Layer with as output shape the desire kernel size (h_s, w_s).
18. A learning parameter p with which we multiply the output.
19. A sigmoid function $\sigma(x) = \frac{1}{1+e^x}$.

In all convolution layer, we use a padding such that we have the same image dimension in input and output except the number of channels.

The convolution kernel size corresponds to a view of 1 complete music beat on a full music scale. The maximum pooling window's size corresponds to a reduction in time of 3/4 of beat.

Appendix C Experiments software

For the pattern variation detection presented in Section 6:

- Python 3.12.2
- Mido 1.3.3
- Numpy 1.26.4
- Pretty midi 0.2.10
- Tensorboard 2.19.0
- PyTorch 2.3.1
- Torchvision 0.18.1a0

For unsupervised correlation experiments presented in Section 7:

- Python 3.10.12
- Amdsmi 24.7.1+da5dcf6
- Numpy 1.21.5
- PyTorch 2.7.1+rocm6.3
- Torchaudio 2.7.1+rocm6.3
- Torchvision 0.22.1+rocm6.3