Extraction of Musical Motifs from Handwritten Music Score Images

Benammar Riyadh[1], Véronique Eglin[1] and Christine Largeron[2]
[1]*Université De Lyon CNRS INSA-Lyon, LIRIS, UMR5205, F-69621, France*
[2]*UJM-Saint-Etienne, CNRS, Institut d'Optique Graduate School, Laboratoire Hubert Curien UMR 5516,*

Keywords: Musical Motifs Extraction, Transcription, Handwritten Music Scores Analysis.

Abstract: A musical motif represents a sequence of musical notes that can determine the identity of a composer or a music style. Musical motifs extraction is of great interest to musicologists to make critical studies of music scores. Musical motifs extraction can be solved by using a string mining algorithm when music data is represented as a sequence. When music data is initially produced in XML or MIDI format or can be converted into those standards, it can be automatically represented as a sequence of notes. So, in this work, starting from digitized images of music scores, our objective is twofold: first, we design a system able to generate musical sequences from handwritten music scores. To address this issue, one of the segmentation-free R-CNN models trained on musical data have been used to detect and recognize musical primitives that are next transcribed into XML sequences. Then, the sequences are processed by a computational model of musical motifs extraction algorithm called CSMA (**C**onstrained **S**tring **M**ining **A**lgorithm). The consistency and performances of the framework are then discussed according to the efficiency of the R-CNN (**R**egion-proposal **C**onvolutional **N**eural **N**etwork) based recognition system through the estimation of misclassified primitives relating to the detailed account of detected motifs. The carried-out experiments of our complete pipeline show that it is consistent to find more than 70% of motifs with less than 20% of average detection/classification R-CNN errors

1 INTRODUCTION

The study of musical scores using automated image analysis and data mining tools offers new perspectives for musical analysis and critical edition of scores. Our work is devoted to the study of musical scores with the main objective to assist musicologists in their understanding of the musical background (i.e. composer and musical style characterization, plagiarism detection, expert reading grid through motif discovery). To assist musicological researcher in his everyday work (critical edition, musical influence study, history of music...), we propose here the development of a complete search engine for musical motifs detection, defining the motifs as frequent successions of identical notes including also some meaningful melodic information.

This project is a part of a pluridisciplinary collaboration supported by the *Région Rhône Alpes* (France) between Computer Sciences laboratories (through computer vision and text mining fields) and the musicologists staff of the MSH of Lyon in France (Pardoen, 2012).

We present in the paper the overall scheme of a

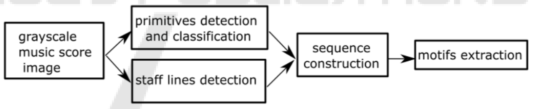
grayscale music score image
primitives detection and classification
staff lines detection
sequence construction
motifs extraction

Figure 1: Musical motifs extraction from music score images pipeline.

musical motifs detection based, for its first part, on a pixel-wise system dedicated to musical primitives recognition and then on a string mining algorithm for the exact motifs extraction.

Music data can be presented in a variety of formats: audio, transcription and image. The CSMA algorithm that has been proposed in (Benammar et al., 2017) is dedicated to motifs extraction on the audio data (MIDI) and XML encoding transcription (MusicXML).

In this manuscript, we present the general framework of the generation of a musical sequence starting from a segmentation-free binary objects detection and an efficient musical primitives classification. In this work, we need to answer two main questions. First, how can we generate a sequence of primitive objects from R-CNN (*Region-proposal Convolutional Neural*

Riyadh, B., Eglin, V. and Largeron, C.
Extraction of Musical Motifs from Handwritten Music Score Images.
DOI: 10.5220/0007577404280435
In *Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2019)*, pages 428-435
ISBN: 978-989-758-354-4

Network) output model and extract meaningful motifs from it. Secondly, what is the impact of R-CNN classification errors on the quality of the extracted motifs.

To address these questions, we propose a complete pipeline allowing to make a transcription of the R-CNN model output. This transcription is used to build a sequence that is processed by CSMA to capture variant-sized motifs from it as illustrated in Figure 1. For the evaluation of errors impact, a set of experiments are also proposed. They consist in the initial evaluation of the transcription quality, by comparing the generated sequence from the R-CNN output with a reference sequence built from a correct XML transcription. Then they are finally based on the evaluation of the motifs delivered from the R-CNN output compared to the reference ones extracted from the ground-truth XML transcription.

This paper is structured as follows: In the section 2, we briefly remind the recent works related to music primitives and symbols recognition required as the first stage of our pipeline. Section 3 describes the different annotated datasets and ground truths that are offered by the community for the experiments and evaluations and also the data preparation that is required for our own needs and experiments. Then, in Section 4, we present next stages of the pipeline dedicated to sequences generation through the primitives encoding. Next, in section 5, the motifs extraction algorithm from sequences (CSMA) is presented. In Section 6, the evaluation protocol is established and different experiments of motifs extraction are presented, concluding by the nature of relations existing between performances of the R-CNN primitives detector and the efficiency of the CSMA approach to extract meaningful motifs of music score images.

2 OVERVIEW OF MUSIC PRIMITIVES DETECTION AND CLASSIFICATION

This last decade, different approaches have been proposed for handwritten music symbol recognition. Some works are based on classical classification schemes that consist in a first step of features extraction and classification procedure as described in (Tardón et al., 2009). In their work, the authors used K-NN, the Mahalanobis distance, and the Fisher discriminant as classification approach. As well as Hidden Markov Models (*HMMs*) prove their efficiency for character recognition models, some works, like in (Lee et al., 2010) (Mitobe et al., 2004), tried to use those models for printed music notation classification.

More recently, in (Rebelo et al., 2010), authors show that Neural Network models outperform HMMs. It is in this way, that in the last few years, the convolutionnal neural networks (CNN) outperforms most of the state-of-the-art classification systems for character recognition (Chen et al., 2015). The research community working on handwritten music recognition turned to those approaches. Indeed, the neural architectures are able to make decision without prior knowledge on the data (i.e. without features extraction). However, they need to be designed through very specific network schemes, dealing in particular with the number of layers, the layers layout and the hyper-parameters initialization (e.g. batch size, number of features per layer, etc.).

One of the first attempts was made by A. Rebelo et al. in (Wen et al., 2015), using hierarchical classification based on two combined neural networks models and requiring a sliding window to correct the classification process. In the same class of methods, Lee et Al., in (Lee et al., 2016), tried to classify handwritten music symbol of HOMUS dataset, which have been proposed in (Calvo-Zaragoza and Oncina, 2014), using different deep convolution network architectures. They study the efficiency of famous deep neural networks like CifarNet, AlexNet and GoogleNet to recognize music symbols.

More recently, Calvo-Zaragoza et al. proposed a pixel-wise binarization of musical documents with convolution neural networks (Calvo-Zaragoza et al., 2017b). They proposed a CNN based approach for the segmentation of handwritten music scores into staff lines, music symbols and text regions (Calvo-Zaragoza et al., 2017a). This last work is the very first attempt to make automatic detection of musical primitives without using any heuristics for the steps of primitives detection and recognition.

This trend continued in 2018 by Pacha in (Pacha et al., 2018a). In their work, the authors used the deep learning library TensorFlow and tested a set of object detection models (called Region proposal CNN) on musical data by considering almost all the music vocabulary. The best performing detector is the Faster Region-proposal Convolutional Neural Network (Faster R-CNN) using the Inception-Resnet V2 feature extractor pre-trained on the COCO dataset(Szegedy et al., 2017). Their model produces a mean average precision of 80% on a set of hundred visual musical primitives of the MUSCIMA++ dataset(Fornés et al., 2012) (Hajič and Pecina, 2017). Other neural region-proposal architectures like U-Net (Ronneberger et al., 2015) have also shown very accurate results on similar data (Pacha et al., 2018b). Currently, their deep approaches outperform all other of the state-of-art.

So as to show the efficiency of a recognition system, A. Fornes et. al. proposed in 2012 a first dedicated dataset called MUSCIMA. This dataset was firstly dedicated to writer identification and staff lines detection. This dataset contains 1,000 music sheets written by 50 different musicians where each writer transcribed the same 20 music pages, using the same pen and the same kind of music paper (Fornés et al., 2012). As a consequence, accurate approaches were proposed in ICDAR 2013 competition for writer identification and staff lines detection (Louloudis et al., 2013).

Several other approaches use private datasets making it impossible to establish proper comparisons. The advantage of making an annotated dataset for symbols recognition available to the researcher community is obvious. So as to make the primary MUSCIMA dataset (dedicated dataset for writer identification and staff lines detection(Fornés et al., 2012)) also usable for music primitives recognition, Jan Hajic jr. and Pavel Pecina, in (Hajič and Pecina, 2017), proposed an annotation tool (Muscimaker) to build a primitive level annotation of music symbols over the 140 MUSCIMA score images. This dataset is called MUSCIMA++. In MUSCIMA++ dataset, each primitive is described by its class, top and left position into the original MUSCIMA image, its width/height and a mask to reconstruct the image and outlinks (Fornés et al., 2012)(Hajič and Pecina, 2017).

MUSCIMA++ dataset provides enough information to setup an end-to-end trainable object detector for music symbols in handwritten music scores.

3 FASTER REGION-PROPOSAL CNN FOR PRIMITIVES RECOGNITION

In this work, we propose to adapt the meta deep-learning architecture *Faster Region-proposal CNN (Fast R-CNN)* as mentioned in (Pacha et al., 2018b) to detect and classify music primitives and to transform them into textual sequences to properly extract musical motifs. To show the efficiency of our overall pipeline, we consider two datasets. *Dataset1* is built from MUSCIMA and MUSCIMA ++ and only consider correctly segmented and identified visual primitives. *Dataset2* is built from the outputs of Pacha's primitives detection model applied on MUSCIMA dataset (Pacha et al., 2018a) which provide the current best performing detection and recognition scores over hundred classes of primitives. A primitive is defined as a part of a musical composed symbols. There are around 100 types of primitives in a music score (also including handwritten textual annotation).

In both cases of datasets, we only consider one-voice music scores with a single instrument (corresponding to a total of 9 pages and 11 primitives) to enable the representation of the score into a single (one dimensional) music sequence. This selection avoids confusing the chords and the arrangement of voices when building the musical sequences. In addition, the use a basic algorithm of detection (based on vertical projections) of staff lines pushed us to leave out some score images (those with a significant slant) even if this can be avoided by a deskewing operation.

For both datasets, staff lines images from MUSCIMA dataset are used to detect and encode the staff lines for each score image. The MUSCIMA ++ dataset annotations are used to encode the primitives that compose the first dataset *dataset1* and the Faster R-CNN output, based on Inception ResNet v2 (for features extraction) is used as annotation tools to encode the primitives of the second dataset *dataset2*.

Table 1 and Figure 3 resume the performances in mAP (mean Average Precision) of primitives detection and recognition obtained by the Faster R-CNN with the same configuration as mentioned in (Pacha et al., 2018b). The architecture has been applied on a selection of significant musical primitives of the MUSCIMA dataset, carefully chosen to fully address the melodic information of the scores. These primitives are chosen among the variety of the initial hundred ones: notehead-full, notehead-empty, flat, natural, sharp, measure bars, additional lines (ledger lines) and clefs. In this scenario, we have disregarded primitives with minor importance such as uncommon numerals and letters and other objects appearing less than 50 times in the dataset. We note that most primitives are well detected and recognized with an average accuracy greater than 87%, except for the ledger lines primitives (additional bars in the noteheads beside staff lines), that are very sensitive to handwritten execution (only 61 % of correct recognition) as shown in Table 1. In Figure 2, we show two output samples of the Fast R-CNN, the first one illustrates a situation where the network fails to detect some targeted primitives and the second one shows an example where it is totally successful. There are many reasons of the difficulties of the Fast R-CNN to fully recognize primitives: most often it is due to a clumsiness of their graphical execution. In Figure 3, we show statistics about the average accuracy of the R-CNN considering eleven classes of primitives relative to our selection of pages in MUSCIMA dataset. Here, we show that each score image has its own complexity (overall layout and primitive position, achieved primitive execution,

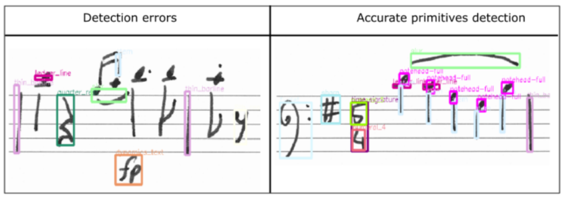

Figure 2: Faster R-CNN outputs.

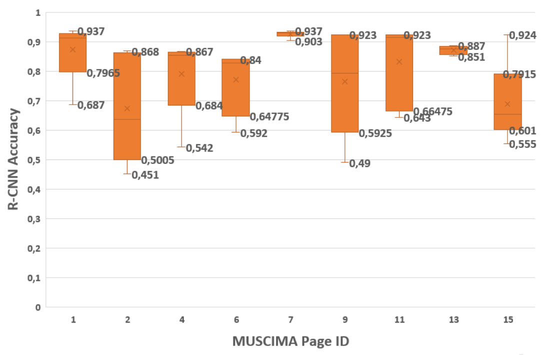

Figure 3: Faster R-CNN outputs accuracy per page.

Table 1: Faster R-CNN accuracy statistics in mean Average Precision on Test set considering 11 classes of primitives.

Primitive	Min	avg	std deviation	max
notehead-full	0.426	0.888	0.188	1
notehead-empty	0,263	0,871	0,235	1
stem	0,624	0,889	0,142	0,994
flat	0,4	0,893	0,204	1
sharp	0,4	0,942	0,151	1
natural	0,25	0,88	0,217	1
f-clef	1	1	0	1
g-clef	1	1	0	1
c-clef	1	1	0	1
ledger line	0,203	0,613	0,23	0,883
thin barline	0,714	0,920	0,105	1

quality of the writing...) that impacts the accuracy of the R-CNN output. The later experimental study presented in section 6 will show the impact achieved by recognition errors on the transcription quality and the detection of motifs.

4 PRIMITIVE ENCODING AND SEQUENCE GENERATION

In the MUSCIMA dataset, a score image relative to an author is presented in three forms: binary with staff lines, grey level with staff lines, and binaries without staff lines.

Although there is a large diversity of staff line detection approaches even for distorted images (Visani et al., 2013), we have opted for the horizontal projection algorithm that is effective enough on our selection of MUSCIMA images. It should be noted that we can easily employ skew corrector algorithm to adjust lines inclination. This step allows to detect very accurately the five lines forming the stave. Accordingly, the primitives are encoded with their position (in the lines or in-between), such as the first line on the top is encoded 0, the first staff line is encoded 2 and the space in-between is encoded 1 and so on. Figure 4 illustrates the counting mechanism of lines (space numbers are not drawn for readability).

Each primitive image is defined by its class and its bounding box whether it concerns MUSCIMA++ primitives dataset or the faster R-CNN outputs. As explained before, we only consider a significant subset of primitives that we judge efficient to cover most of the melodic information present in a score. This primitive-level encoding allows to build sequences from music scores such that the notes are firstly ordered by associated vertical stave position, and then they are ordered left to right following the reading direction at the stave level, see Figure 4.

The first step consists in associating stems to staves by considering the maximal shared space ranges between stem and stave, otherwise by affecting the stem to the nearest stave, as illustrated in Figure 5. Then, noteheads and accidentals are associated to staves of the nearest stem (cf. links represented by arrows in Figure 5. Then, we store the line number (or the space) which is the closest to the center of the primitive. Based on this analysis, the sequence of primitives is ordered along the x-axis to ensure the temporality, except for the chord notes that are played at the same time, and ordered from the highest to the lowest according to y-axis.

In Figure 5, the primitives are encoded following the regular expression:

?*(Alteration) (NoteType) (Position)*

Such that ? for 0 or 1 occurrence, *Alteration* can be (F: Flat, N: Natural, ♯: sharp), *NoteType* can be (NF: Notehead-full, NE: Notehead-Empty) and *Position* refers to the associated staff line index.

The position of the notes, respectively above or under the stave, depends on the apparent number of ledger lines (primitives) in the stem of the note, acting as extra-lines for the stave. For example, in Figure 5, the empty notehead on the second stave is above 4 ledger lines, and following the same encoding as used for staff lines, this position corresponds to -9.

In order to complete the melodic information we consider the projection of accidentals with respect to the staff lines. First, clefs must be identified and staff lines encoding must be adjusted: for example, for a C-Clef located on line index 2, the line index should be updated with an offset of +12, converting a line index value of 0 to 12. For more explanation on the

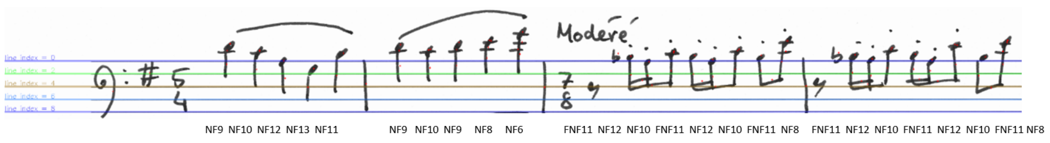

Figure 4: Staff lines numbering for the primitives encoding.

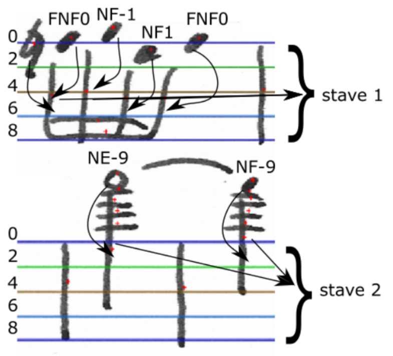

Figure 5: Zoom in on the notehead primitives encoding.

meaning of clefs and their use we invite the reader to see music theory lessons or simply visit the Wikipedia web page https://en.wikipedia.org/wiki/Clef. This justify the values of the sequence in Figure 4.

After that, depending on the type of clef, we count the number of accidentals next to the clef to determine the key signature of a note and we associate minor / major accidentals to it according to their positions. Inside measures, each accidental is associated to the notes that directly follow it. Following barline cancels the effect of an accidental.

Accordingly, the sequences are generated from the primitives position (the octave number and its type) in the score image and from the concomitant presence of accidentals when they exist. It is then straightforward to associate each primitive-level encoding to a MIDI encoding and also a XML encoding (transcription) by using simple matching rules between representations. For example, #NF0 (a black on the first line of range with accidental on #) corresponds to the note 78 of the MIDI encoding. The sequence of primitives can easily be expressed by the same vocabulary as a MIDI sequence. It becomes then obvious to establish the transcription accuracy (on the dataset of the study) by using the MUSCIMA++ annotated primitives (*dataset1*) as ground-truth and the tested one (*dataset2*), corresponding to the R-CNN output primitives. This second dataset will reveal the impact of an error of recognition on the consistency of the transcription. For the transcription evaluation, each music score is represented by two MIDI sequences:the ground-truth MUSCIMA++ sequence and the R-CNN based one. The evaluation of the R-CNN derived transcriptions lies on its comparison with the ground-truth MUSCIMA++ sequences. Results are presented in Section 6.2. In the next sections, we show and evaluate the motifs extraction from a transcribed sequence of varying sizes.

5 MOTIFS EXTRACTION

CSMA(**C**onstrained **S**tring **M**ining **A**lgorithm) has been designed for discovering all frequent motifs in a string (Benammar et al., 2017). CSMA performs motifs search according to constraints related to frequency, gaps between motifs, minimal and maximal length of motifs.

A motif m_i is defined by three elements $m_i = (X, freq(X), P_i = [(p_{i1}, len_{i1}), ..., (p_{in}, len_{in})])$ such that X corresponds to the motif value (ordered list of items), $freq(X)$ corresponds to its frequency and P_i its positions and lengths. In the set of positions, called P_i, the j^{th} position of the i^{th} motif is denoted p_{ij} and its length at this position is denoted len_{ij}.

The pseudo-code of **CSMA** is given in Algorithm 1. This algorithm takes in input a sequence S, a minimum frequency threshold $minFreq$, a maximum allowed gap length inside motifs $maxGap$, a minimum motif length $minLength$ and a maximum motif length $maxLength$.

The first step of CSMA (Line 4, Algorithm 1) consists in computing the set $\mathcal{F}_1$ containing the frequent motifs of length one using **COMPUTE** function. The items, from the sequence S, with frequency greater than or equals to $minFreq$ are added to $\mathcal{F}_1$. In order to get the set $\mathcal{F}_K$ containing the motifs of length equals to ($K = 2$), a joining operation **JOIN** is considered (line 7) between each element m_i of $\mathcal{F}_{K-1}$ and each item m_j belonging to $\mathcal{F}_1$. The joining operation is $O(|P_i| \times |P_j|)$. So, in order to prune the search space, we compute the position on which the motif m_i is considered as frequent (line 8). This position, called fp for frequent position, corresponds to the sum of the index of $m_i \in \mathcal{F}_{K-1}$ at the $minFreq^{th}$ position and the length of m_i for the same position. Then, candidate motifs are generated using the **GEN_CAND** function. The interest reader is referred to the original article for a detailed description of this function (Benammar et al., 2017). Once the selection of candidate motifs is done, the joining operation is performed for the selected motif m_i with each element $m_j \in \mathcal{C}$. The motif joining (concatenation) is defined as follows:

Algorithm 1: Constrained String Mining Algorithm (CSMA).

```
Input  : Sequence S, minFreq,maxGap,
         minLength and maxLength
Output: $\mathcal{F}$: The set of frequent motifs respecting
         constraints
1  begin
2      $K = 1$;
3      $\mathcal{F}_1 = \emptyset$;
4      COMPUTE(S, minFreq, $\mathcal{F}_1$);
5      while $\mathcal{F}_K \neq \emptyset$ do
6          $K = K + 1$;
7          for $m_i = (X, freq(X), P_i) \in \mathcal{F}_{K-1}$ do
8              $fp = p_{iminFreq} + len_{iminFreq}$;
9              $C = GEN\_CAND(fp, \mathcal{F}_1)$;
10             for $m_j = (Y, freq(Y), P_j) \in C$ do
11                 $m_l = JOIN(m_i, m_j,$ maxGap,
                     maxLength);
12                 if $freq(Z) \geq minFreq$ then
13                     $\mathcal{F}_K = \mathcal{F}_K \cup \{m_l\}$;
14                 end
15             end
16         end
17     end
18     $\mathcal{F} = \bigcup_{k \leq K} \mathcal{F}_k$
19     $FILTER(\mathcal{F}, minLength)$;
20     return $\mathcal{F}$;
21 end
```

Let be two motifs $m_1 \in \mathcal{F}_{K-1}$ and $m_2 \in \mathcal{F}_1$ defined as $m_1 = (X, freq(X),\ P_1 = [\bigcup_{i \leq freq(X)} (p_{1i}, len_{1i})])$ and $m_2 = (Y, freq(Y), P_2 = [\bigcup_{j \leq freq(Y)} (p_{2j}, len_{2j})])$, m_1 join m_2 gives $m_3 \in \mathcal{F}_K$ defined as $m_3 = (Z, freq(Z), P_3)$ such that Z is the concatenation of (X,Y) and P_3 is a set of positions p_{3k} and lengths len_{3k}. A position $p_{3k} \in P_3$ equals to p_{1i} if and only if $\exists j \leq freq(Y)$ such that the three conditions are verified:

$$\begin{cases} 0 \leq p_{2j} - (p_{1i} + len_{1i}) \leq maxGap & (1) \\ i = \arg\min_{l \leq freq(X)} (p_{2j} - (p_{1l} + len_{1l})) & (2) \\ p_{2j} + len_{2j} - p_{1i} \leq maxLength & (3) \end{cases}$$

The positions p_{1i} from m_1 and p_{2j} from m_2 verifying the three conditions allow to define the position p_{3k} corresponding to p_{1i} for m_3, and the length len_{3k} is equal to $p_{2j} + len_{2j} - p_{1i}$. The frequency of m_3 is equal to the number of positions in P_3. It can be noticed that the frequency of each new motif is lower or equal to its sub-motifs. This means that the joining operation verifies the anti-monotony property which allows to prune the search space. Once $\mathcal{F}_2$ is obtained, the other sets $\mathcal{F}_K$ of length $K > 2$, are computed and the while loop stops when no new motif is generated. In the next step, (line 19 in Algorithm1), all frequent motifs of order $k \leq K$ are put in $\mathcal{F}$. Then, motifs that do not respect the *minLength* constraint are removed from $\mathcal{F}$. The **FILTER** function scans each motif $m \in \mathcal{F}$ and if it finds a position for which len_i is lower than *minLength* it removes it from the set P. In the end, the value $freq(X)$ is updated and if it is lower than *minFreq* the motif is removed from $\mathcal{F}$. As Algorithm 1 makes a breath first search to build motifs, it needs an exponential running time which is estimated to $O((max(|P|) \times |F_1|)^{maxLength})$; with $max(|P|)$ the maximal size of positions sets.

6 EVALUATION

The goal of this section is to show and discuss the impact of errors of primitives recognition on the transcription accuracy and then on the efficiency of the motifs detection pipeline. On both cases, the evaluations are based on the two datasets containing the musical primitives (introduced in section 3):*dataset1* for the ground-truth annotated primitives of MUSCIMA++ dataset and *dataset2* for the R-CNN output primitives. For both datasets the transcriptions of musical scores are produced according to the method presented section 4.

6.1 Evaluation of the Transcription

The evaluation of the transcription consists in the estimation of the Levenshtein distance between the two sequences and the correct MIDI sequence (deduced from the ground-truth XML and denoted as XML-GT). The distance consists in the removal, the insertion, or the substitution of a character in the string (Navarro, 2001). In this definition of edit distance, errors of insertion, suppression and substitution have the same weight. The final distance is then normalized with respect to the size of the reference sequence (XML-GT).

Figure 6 shows the statistics of the computed Levenshtein distance between each sequence from *dataset1* (MUSCIMA ++ dataset) and *dataset2* (R-CNN output) with its corresponding MIDI reference sequence (XML-GT). We can notice that Lenvenshtein distances are mostly greater in the case of R-CNN sequences. This observation is expected due to the detection and/or classification errors. However, in the case of sequences derived from MUSCIMA++ dataset, even if Lenvenshtein distances are smaller, they are not null. This can arise from a lack or an incompleteness of an efficient manual ground-truth annotation in the MUSCIMA++ dataset.

Lastly, we can note that R-CNN errors in the detection of noteheads, ledger lines, and accidentals (especially for page 9) belonging to key signatures (set

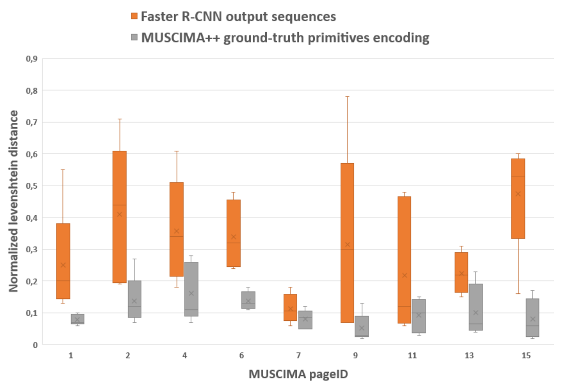

Figure 6: Normalized Edit Distance stats between XML-GT sequences and those provided by the R-CNN output (orange) and the MUSCIMA++ ground-truth primitives (grey).

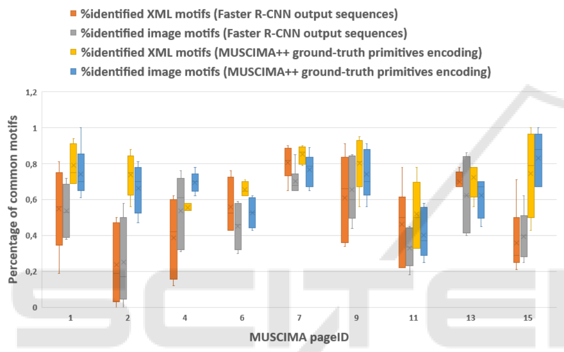

Figure 7: Box-and-whisker plots for the motifs alignment. Estimate made between XML-GT & R-CNN sequences (orange & grey bars) and XML-GT & MUSCIMA++ sequences (yellow & blue bars).

of sharp or flat symbols placed together on the staff) spread mistakes over the sequence.

6.2 Evaluation of the Motifs Extraction

As the sequences share the same vocabulary, the evaluation of motifs extraction is based on the estimation of the number of common motifs between sequences (i.e. one from *dataset1* or *dataset2* and the second from the ground-truth MIDI transcription, still denoted as XML-GT in figure 7). It should be noted that this work is based on MUSCIMA++ dataset which have been proposed for handwriting processing tasks, reflecting a data diversity on the writing styles and music symbols. Thus, only short motifs will be found through the music scores (with an average length of 4 notes). The efficiency of CSMA algorithm on real music data has been demonstrated in (Benammar et al., 2017).

In Figure 7, we show statistics on common motifs between each dataset and the associated MIDI transcription (XML-GT). The '%identified XML motifs' refers to the percentage of common motifs according to the total number of MIDI motifs (used as reference in the calculation). By analogy, '% identified image motifs' refers to the percentage of common motifs divided by the total number of primitives motifs (used as reference in the calculation).

We observe that CSMA is able to find an average of 70% of common motifs between the primitive sequence and the reference sequence in the case of *dataset1*. We note that, at the best CSMA is able to find 100% of motifs in the case of page 15. Nevertheless, some sequences share only 25% of motifs. Theses sequences are made from images containing lot of annotation oblivion. This refers to annotated pages with maximal edit distance values and motifs membership probability of wrongly encoded primitives (cf. Figure 6).

In the case of *dataset2* (R-CNN output), CSMA is able to find in average 50% of common motifs. At worst case CSMA is not able to find motifs when the distance between R-CNN based sequence and the reference MIDI sequence is maximal (c.f. Figure 6). However, when the edit distance is minimal (minimal errors occurrences), CSMA is able to identify about 76% of common MIDI motifs with about 27% of remaining primitive motifs. This point out that motif extraction process is highly sensitive to errors but, with certain level of mistakes, can find most of musical motifs.

In future works, we will study how CSMA can be more accurate by allowing gaps into motifs. The theorical part of the gap introduction can be found in (Benammar et al., 2017).

7 CONCLUSION

In this work, we detail the pipeline of a complete motif extraction system dedicated to handwritten music scores images, starting from the very low level steps of primitives recognition to the exaction of motifs from the generation of musical transcriptions. For the detection and the recognition of musical primitives, we focus on one of the most accurate pre-trained R-CNN on the MUSCIMA++ dataset. For each score, the identified primitives are encoded into a sequence uses as input of our string mining algorithm (CSMA) to retrieve musical motifs.

This end-to-end process is evaluated on MUSCIMA dataset and shows the real impact of misclassification on the mined motifs. We also show that it is not obvious to retrieve accurate motifs (common motifs between the correct XML sequence and the primitives sequence) when primitives sequence is proned to errors. In most cases, with less than 20% of average detection/classification R-CNN errors, the mining algorithm is able to find more than 70% of motifs. This can be considered as efficient from a musicologist viewpoint to target the major motifs but we believe that these performances can still be improved.

In our future work, we will try to use the gap constraint of CSMA in order to see how we can reduce the impact of errors on the extracted motifs.

ACKNOWLEDGEMENTS

The funding for this project was provided by a grant from *la Région Rhône Alpes*. We acknowledge Mylene Pardoen, the musicologist of the project for her sound advices and expert recommendations.

REFERENCES

Benammar, R., Largeron, C., Eglin, V., and Pardoen, M. (2017). Discovering motifs with variants in music databases. In *Intern. Symposium on Intelligent Data Analysis*, pages 14–26. Springer.

Calvo-Zaragoza, J. and Oncina, J. (2014). Recognition of pen-based music notation: the homus dataset. In *22nd Intern. Conf. on Pattern Recognition (ICPR)*, pages 3038–3043. IEEE.

Calvo-Zaragoza, J., Pertusa, A., and Oncina, J. (2017a). Staff-line detection and removal using a convolutional neural network. *Machine Vision and Applications*, pages 1–10.

Calvo-Zaragoza, J., Vigliensoni, G., and Fujinaga, I. (2017b). Pixel-wise binarization of musical documents with convolutional neural networks. In *Fifteenth IAPR Intern. Conf. on Machine Vision Applications (MVA)*, pages 362–365. IEEE.

Chen, L., Wang, S., Fan, W., Sun, J., and Satoshi, N. (2015). Reconstruction combined training for convolutional neural networks on character recognition. In *Intern. Conf. on Document Analysis and Recognition*, pages 431–435. IEEE.

Fornés, A., Dutta, A., Gordo, A., and Lladós, J. (2012). Cvc-muscima: a ground truth of handwritten music score images for writer identification and staff removal. *Intern. Conf. on Document Analysis and Recognition*, pages 1–9.

Hajič, jr., J. and Pecina, P. (2017). In Search of a Dataset for Handwritten Optical Music Recognition: Introducing MUSCIMA++. *CoRR*.

Lee, K. C., Phon-Amnuaisuk, S., and Ting, C. Y. (2010). Handwritten music notation recognition using hmma non-gestural approach. In *Intern. Conf.on Information Retrieval & Knowledge Management,(CAMP), 2010*, pages 255–259. IEEE.

Lee, S., Son, S. J., Oh, J., and Kwak, N. (2016). Handwritten music symbol classification using deep convolutional neural networks. In *Intern. Conf. on Information Science and Security (ICISS)*, pages 1–5. IEEE.

Louloudis, G., Gatos, B., Stamatopoulos, N., and Papandreou, A. (2013). Icdar 2013 competition on writer identification. In *Intern. Conf. on Document Analysis and Recognition*, pages 1397–1401. IEEE.

Mitobe, Y., Miyao, H., and Maruyama, M. (2004). A fast hmm algorithm based on stroke lengths for on-line recognition of handwritten music scores. In *Ninth Intern. Workshop on Frontiers in Handwriting Recognition, 2004. IWFHR-9*, pages 521–526. IEEE.

Navarro, G. (2001). A guided tour to approximate string matching. *ACM computing surveys (CSUR)*, 33(1):31–88.

Pacha, A., Choi, K.-Y., Coüasnon, B., Ricquebourg, Y., Zanibbi, R., and Eidenberger, H. (2018a). Handwritten music object detection: Open issues and baseline results. In *Inter. Workshop on Document Analysis Systems*.

Pacha, A., Hajič, J., and Calvo-Zaragoza, J. (2018b). A baseline for general music object detection with deep learning. *Applied Sciences*.

Pardoen, M. (2012). Projet Le Magasin de Musique BIS. http://ladehis.ehess.fr/index.php?592. Accessed: 2018-11-14.

Rebelo, A., Capela, G., and Cardoso, J. S. (2010). Optical recognition of music symbols. *Inter. journal on document analysis and recognition*, 13(1):19–31.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, pages 234–241. Springer.

Szegedy, C., Ioffe, S., Vanhoucke, V., and Alemi, A. A. (2017). Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, volume 4, page 12.

Tardón, L. J., Sammartino, S., Barbancho, I., Gómez, V., and Oliver, A. (2009). Optical music recognition for scores written in white mensural notation. *EURASIP Journal on Image and Video Processing*, 2009(1):843401.

Visani, M., Kieu, V. C., Fornés, A., and Journet, N. (2013). Icdar 2013 music scores competition: Staff removal. In *Intern. Conf. on Document Analysis and Recognition*, pages 1407–1411. IEEE.

Wen, C., Rebelo, A., Zhang, J., and Cardoso, J. (2015). A new optical music recognition system based on combined neural network. *Pattern Recognition Letters*, 58:1–7.