CAMERA-PRIMUS: NEURAL END-TO-END OPTICAL MUSIC RECOGNITION ON REALISTIC MONOPHONIC SCORES

Jorge Calvo-Zaragoza
PRHLT Research Center
Universitat Politècnica
de València, Spain
jcalvo@prhlt.upv.es

David Rizo
Instituto Superior de Enseñanzas Artísticas
de la Comunidad Valenciana (ISEA.CV)
Universidad de Alicante, Spain
drizo@dlsi.ua.es

ABSTRACT

The optical music recognition (OMR) field studies how to automate the process of reading the musical notation present in a given image. Among its many uses, an interesting scenario is that in which a score captured with a camera is to be automatically reproduced. Recent approaches to OMR have shown that the use of deep neural networks allows important advances in the field. However, these approaches have been evaluated on images with ideal conditions, which do not correspond to the previous scenario. In this work, we evaluate the performance of an end-to-end approach that uses a deep convolutional recurrent neural network (CRNN) over non-ideal image conditions of music scores. Consequently, our contribution also consists of Camera-PrIMuS, a corpus of printed monophonic scores of real music synthetically modified to resemble camera-based realistic scenarios, involving distortions such as irregular lighting, rotations, or blurring. Our results confirm that the CRNN is able to successfully solve the task under these conditions, obtaining an error around 2% at music-symbol level, thereby representing a groundbreaking piece of research towards useful OMR systems.

1. INTRODUCTION

The optical music recognition (OMR) discipline was born several decades ago [28], and nowadays there are still too many open problems to consider it a solved task. This applies not only for handwritten notation but also for the case of printed scores [4]. Unfortunately, unlike other automatic content transcription domains, such as speech recognition [23] or optical character recognition [24], the latest advances in pattern recognition and machine learning—namely deep learning—have not definitively broken the long-term glass ceiling.

Actually, other computer music domains are taking advantage of these advances, but quite often, especially in symbolic music research, the lack of big enough datasets block their improvement. If OMR technologies were able to convert the massive printed scores libraries [1] into structured, symbolic scores, all those fields would obtain interesting corpora to work on.

Furthermore, out of the scientific community, the availability of tools that transcribe sheet music without errors into symbolically-encoded music would help professional and amateur musicians to take advantage of the plenty of computer music tools at hand that cannot work directly with digital images.

Following the steps of other aforementioned disciplines, we claim that the problem can be appropriately addressed with holistic approaches, i.e., end-to-end, where systems learn with just pairs of inputs and their corresponding transcripts. Here, these pairs consists of sheet music and their symbolic encoding.

In this work, we extend previous proposals that applied neural network models over monodic digitally-rendered music scores [8]. However, we evaluate here their performance with a set of scores that are rendered simulating camera-based conditions. Our objective is to study whether the approach is feasible for non-ideal image conditions. Although we do not experiment with fully-fledged scores yet, we believe that this avenue is promising for reaching the final objective of dealing with any kind of input score. Thus, in this work we introduce the so-called *Camera-Printed Images of Music Staves* (Camera-PrIMuS) dataset of monodic single-staff printed scores, that have been distorted to resemble photographed scores and encoded in such a way a neural network recognizer can manage.

Our experiments demonstrate that the considered neural models are able to learn even in difficult situations where none of the current commercial OMR systems might be successful. The results reflect that an error rate below 2%, at symbol level, can be attained.

The paper is organized as follows: first, a brief background about OMR is given in Sect. 2; then, the construction of Camera-PrIMuS dataset is detailed in Sect. 3; the neural end-to-end framework is described and formalized in Sect. 4; the experimental results that demonstrate the suitability of the approach are reported in Sect. 5; and finally, the conclusions are discussed in Sect. 6.

[1] Libraries such as http://imslp.org

2. BACKGROUND

Most of the existing OMR approaches work in a multi-stage fashion [38]. These systems typically perform an initial processing of the image that consists of several steps of document analysis, not always strictly related to the musical domain. Examples of this stage comprise the binarization of the image [10], the detection of the staves [11], the delimitation of the staves in terms of bars [45], or the separation among the different sources of information [5].

The staff-line removal stage requires a special mention. Although staff lines represent a very important element in music notation, their presence hinders the automatic segmentation of musical symbols. Therefore, much effort has been devoted to successfully solving this stage [14, 15, 18]. Recently, results have reached values closer to the optimum over standard benchmarks [7, 17].

In the next step, remaining symbols are classified into music-notation categories. A number of works can be found in the literature that deal with this task [30, 37], including deep learning classification as well [6, 32].

Recently, it has been demonstrated that the traditional pipeline up to symbol classification can be replaced by deep region-based neural networks [31], which both localize and classify music-notation primitives from the input image. Either way, once graphical symbol are identified, they must be assembled to eventually obtain actual music notation. Previous attempts to this stage proposed the use of heuristic strategies based on graphical and syntactical rules [13, 36, 40, 43].

Full approaches are more common when recognizing mensural notation, where the OMR challenge is more restricted than that of modern Western notation because of the absence of simultaneous written voices in the same staff and a lower number of symbols to be recognized [9, 33, 44].

3. THE CAMERA-PRIMUS DATASET

The training of a machine learning based system requires a good quality training dataset with enough size to statistically include a representative sample of the problem to be solved. The *Camera-based Printed Images of Music Staves* (Camera-PrIMuS) dataset has been devised to fulfil both requirements [2] . Thus, the objective pursued when creating this ground-truth data is not to represent the most complex musical notation corpus, but to collect the highest possible number of scores readily available to be represented in formats suitable for heterogeneous OMR experimentation and evaluation.

Camera-PrIMuS is an extension of a previously published PrIMuS dataset [8]. It contains 87 678 real-music incipits, [3] each one represented by six files: the Plaine and Easie Code (PAEC) source [3], an image with the rendered score, the same image distorted resembling a camera-based scenario, the music symbolic representation of the incipit both in Music Encoding Initiative format (MEI) [39] and in an on-purpose simplified encoding (semantic encoding), and a sequence containing the graphical symbols shown in the score with their position in the staff, without any musical meaning (agnostic encoding). These two agnostic and semantic representations, that will be described below, are especially designed to be considered in our framework.

Order	Filter	Ranges of used parameters
1	`-implode`	$[0, 0.07]$
2	`-chop`	$[1, 5]$, $[1, 6]$, $[1, 300]$, $[1, 50]$
3	`-swirl`	$[-3, 3]$
4	`-spread`	-2
5	`-shear`	$[-5, 5]$, $[-1.5, 1.5]$
6	`-shade`	$[0, 120]$, $[80, 110]$
7	`-wave`	$[0, 0.5]$, $[0, 0.4]$
8	`-rotate`	$[0, 0.3]$
9	`-noise`	$[0, 1.2]$
10	`-wave`	$[0, 0.5]$, $[0, 0.4]$
11	`-motion-blur`	$[-7, 5]$, $[-7, 7]$, $[-7, 6]$
12	`-median`	$[0, 1.1]$

Table 1. GraphicsMagick filter sequence

Pursuing the objective of considering real music, and being restricted to use short single-staff scores, an export in PAEC format of the RISM dataset [29] has been used as source. The PAEC is then formatted to be fed into the musical engraver Verovio [34], that outputs both the musical score in SVG format—that is posteriorly converted into PNG format (Fig. 1(a))—and the MEI encoding containing the symbolic semantic representation of the score in XML format. Verovio is able to render scores using three different fonts, namely: Leipzig, Bravura, and Gootville. This capability has been used by randomly choosing one of the those fonts in the rendering of the different incipits, leading to a higher variability in the dataset. The on-purpose semantic and agnostic representations (Figs. 1(c) and 1(d)) have been obtained as a conversion from the MEI files. Finally, the PNG image file is distorted, as described below, in order to simulate imperfections introduced by taking a picture of the sheet music from a (bad) camera (Fig. 1(b)).

To simulate distortions, the GraphicsMagick image processing tool [4] has been used. Among the huge amount of filters this tool contains, a number of them have been used and tweaked empirically. Table 1 contains the filters used and the ranges considered for each parameter, from which random values are selected at each instance. Filters have been applied using the order shown in the table.

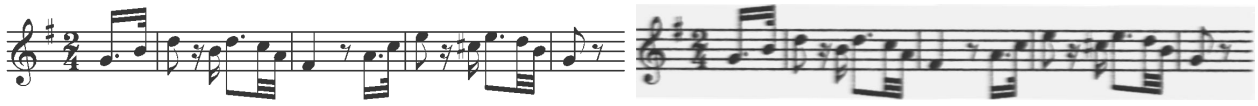
3.1 Semantic and agnostic representations

The suitable encoding of input data for the neural network determines the scope of its performance. Most of the available symbolic representations [41], being devised for other purposes such as music analysis (e.g. `**kern`), or music

[2] The dataset is freely available at `https://grfia.dlsi.ua.es/primus/`.

[3] An incipit is a short sequence of notes from the beginning of a melody or musical work usually used for identifying it

[4] `http://www.graphicsmagick.org`



(a) Clean image.

(b) Distorted image.

```
clef-G2, keySignature-GM, timeSignature-2/4, note-G4_sixteenth., note-B4_thirty_second, barline, note-D5_eighth, rest-sixteenth,
note-B4_sixteenth, note-D5_eighth., note-C5_thirty_second, note-A4_thirty_second, barline, note-F#4_quarter, rest-eighth,
note-A4_sixteenth., note-C5_thirty_second, barline, note-E5_eighth, rest-sixteenth, note-C#5_sixteenth, note-E5_eighth.,
 note-D5_thirty_second, note-B4_thirty_second, barline, note-G4_eighth, rest-eighth
```

(c) Semantic encoding.

```
clef.G-L2, accidental.sharp-L5, digit.2-L4, digit.4-L2, note.beamedRight2-L2, dot-S2, note.beamedLeft3-L3, barline-L1,
note.eighth-L4, rest.sixteenth-L3, note.sixteenth-L3, note.beamedRight1-L4, dot-S4, note.beamedBoth3-S3, note.beamedLeft3-S2, barline-L1,
note.quarter-S1, rest.eighth-L3, note.beamedRight2-S2, dot-S2, note.beamedLeft3-S3, barline-L1,
note.eighth-S4, rest.sixteenth-L3, accidental.sharp-S3, note.sixteenth-S3, note.beamedRight1-S4, dot-S4,
note.beamedBoth3-L4, note.beamedLeft3-L3, barline-L1, note.eighth-L2, rest.eighth-L3
```

(d) Agnostic encoding.

Figure 1. Example of a short item in the corpus: Incipit RISM ID no. 000100367, Incipit 28.1.1 *30 Canons*, Luigi Cherubini. MEI and Plaine and Easie Code files are also included in the corpus but omitted here.

notation (such as MEI [39] or MusicXML [20])—for naming just a few—do not encode a self-contained chunk of information for each musical element. This is why two representations devised on-purpose compliant with this requirement were introduced in [8], namely the semantic and the agnostic ones. For practical issues, none of the representations is musically exhaustive, but representative enough to serve as a starting point from which to build more complex systems.

The semantic representation contains symbols with musical meaning, *e.g.*, a G Major key signature (see Fig. 1(c)); the agnostic encoding (see Fig. 1(d)) consists of musical symbols without musical meaning that should be eventually interpreted in a final parsing stage [16], e.g. a D Major key signature is represented as a sequence of two *sharp* symbols. This way, the alphabet used for the agnostic representation is much smaller, which allows to study the impact of the alphabet size and the number of examples shown to the network for its training. Note that in the agnostic representation, a *sharp* symbol in the key signature is the same pictogram as a *sharp* accidental altering the pitch of a note. A complete description of the grammars describing these encodings can be found in [8].

More specifically, the agnostic representation contains a list of graphical symbols in the score, each of them tagged given a catalogue of pictograms without a predefined musical meaning, and located in a position in the staff (*e.g.*, third line, first space). The Cartesian plane position of symbols has been encoded relatively, following a left-to-right, top-down ordering (see encoding of fractional meter in Fig. 1(d)). In order to represent beaming of notes, they have been vertically sliced generating non-musical pictograms (see elements with prefix `note.beamed` in Fig. 1(d)).

As mentioned above, this new way of encoding complex information in a simple sequence allows us to feed the network in a relatively easy way. Note that the agnostic representation is different from a primitive-based segmentation of the image, which is the usual internal representation of traditional OMR systems [12, 25].

The agnostic representation has an additional advantage: in other less known music notations, such as the early neumatic and mensural notations, or in the case of non-Western notations, it might be easier to transcribe the manuscript through two stages: one stage performed by any non-musical expert that only needs to identify pictograms (agnostic representation), and a second stage where a musicologist, maybe aided by a computer, interprets them to yield a semantic encoding.

4. NEURAL END-TO-END APPROACH FOR OPTICAL MUSIC RECOGNITION

As introduced above, some previous work have proved that it is possible to successfully accomplish the recognition of monodic staves in an end-to-end approach by using neural networks [8]. This section contains a brief description of such framework.

A single-voice monophonic staff is assumed to be the basic unit; that is, a single monodic staff will be processed at each instance. Formally, let $\mathcal{S} = \{(x_1, y_1), (x_2, y_2), ...\}$ be our end-to-end application domain, where x_i represents a single staff image and y_i is its corresponding sequence of music symbols, each of which belongs to a fixed alphabet set Σ.

Given an input staff image, the OMR problem can be solved by retrieving its most likely sequence of music symbols $\hat{y}$:

$$\hat{y} = \arg\max_{y \in \Sigma^*} P(y|x) \tag{1}$$

A graphical scheme of the considered framework is given in Figure 2. The input image depicting a monodic staff is fed into a Convolutional Recurrent Neural Network (CRNN), which consists of two sequential parts: a convolutional block and a recurrent block. The convolutional block is in charge of learning how to deal with the input image [47]. In this way, the user is prevented from performing a pre-processing of the image because this block is able to learn adequate features from the training set. These

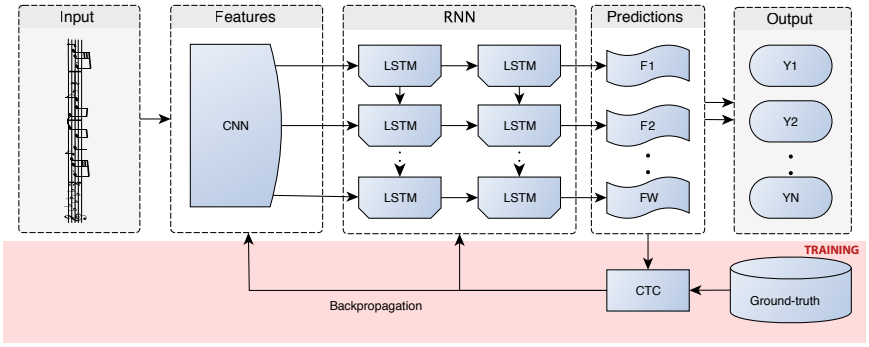


Figure 2. Graphical scheme of the end-to-end neural approach considered.

extracted features are provided to the recurrent block [21], producing the sequence of musical symbols that approximates Eq. 1.

Since both convolutional and recurrent blocks are configured as feed-forward models, the training stage can be carried out jointly. This scheme can be easily implemented by connecting the output of the last layer of the convolutional block with the input of the first layer of the recurrent block, concatenating all the output channels of the convolutional part into a single image. Then, columns of the resulting image are treated as individual frames for the recurrent block.

The traditional training mechanisms for a CRNN need a framewise expected output, where a frame is a fixed-width vertical slice of the image. However, as the goal is to not recognize frames but complete symbols, either semantic or agnostic, and Camera-PriMuS does not contain sequences of labelled frames, a Connectionist Temporal Classification (CTC) loss function [22] has been used to solve this mismatch.

Basically, CTC drives the CRNN to optimize its parameters so that it is likely to give the correct sequence y given an input x. As optimizing this likelihood exhaustively is computationally expensive, CTC performs a local optimization using an Expectation-Maximization algorithm similar to that used for training Hidden Markov Models [35]. Note that CTC is only used for training, while at the decoding stage the framewise CRNN output can be straightforwardly decoded into a sequence of music symbols (details are given below).

4.1 Implementation details

The specific organization of the neural model is given in Table 2. As observed, variable-width single-channel (grayscale) input image are rescaled at a fixed height of 128 pixels, without modifying their aspect ratio. This input is processed through a convolutional block inspired by a VGG network, a typical model in computer vision tasks [42]: four convolutional layers with an incremental number of filters and kernel sizes of 3×3, followed by a 2×2 max-pool operator. In all cases, Batch Normalization [27] and Rectified Linear Unit activations [19] are considered.

$\text{Input}(128 \times W \times 1)$
Convolutional block
$\text{Conv}(32, 3 \times 3)$, $\text{MaxPooling}(2 \times 2)$
$\text{Conv}(64, 3 \times 3)$, $\text{MaxPooling}(2 \times 2)$
$\text{Conv}(128, 3 \times 3)$, $\text{MaxPooling}(2 \times 1)$
$\text{Conv}(256, 3 \times 3)$, $\text{MaxPooling}(2 \times 1)$
Recurrent block
$\text{BLSTM}(256)$
$\text{BLSTM}(256)$
$\text{Dense}(\lvert\Sigma\rvert + 1)$
$\text{Softmax}()$

Table 2. Instantiation of the CRNN used in this work, consisting of 4 convolutional layers and 2 recurrent layers. Notation: $\text{Input}(h \times w \times c)$ means an input image of height h, width w and c channels; $\text{Conv}(n, h \times w)$ denotes a convolution operator of n filters and kernel size of $h \times w$; $\text{MaxPooling}(h \times w)$ represents a down-sampling operation of the dominating value within a window of size $(h \times w)$; BLSTM(n) means a bi-directional Long Short-Term Memory unit of n neurons; Dense(n) denotes a dense layer of n neurons; and Softmax() represents the *softmax* activation function. Σ denotes the alphabet of musical symbols considered.

At the output of this block, two bidirectional recurrent layers of 256 neurons, implemented as Long Short-Term Memory (LSTM) units [26], try to convert the resulting filtered image into a discrete sequence of musical symbols that takes into account both the input sequence and the modelling of the musical representation. Note that each frame performs an independent classification, modelled with a fully-connected layer with as many neurons as the size of the alphabet plus 1 (a *blank* symbol necessary for the CTC function). The activation of these neurons is given by a *softmax* function, which allows interpreting the output as a posterior probability over the alphabet of music symbols [2].

The learning process is carried out by means of stochastic gradient descent (SGD) [1], which modifies the CNN parameters through back-propagation to minimize the

CTC loss function. In this regard, the mini-batch size is established to 16 samples per iteration. The learning rate of the SGD is updated adaptively following the Adadelta algorithm [46].

Once the network is trained, it is able to provide a prediction in each frame of the input image. These predictions must be post-processed to emit the actual sequence of predicted musical symbols. Thanks to training with the CTC loss function, the final decoding can be performed greedily [22]: when the symbol predicted by the network in a frame is the same as the previous one, it is assumed that they represent frames of the same symbol, and only one symbol is concatenated to the final sequence. There are two ways to indicate that a new symbol is predicted: either the predicted symbol in a frame is different from the previous one, or the predicted symbol of a frame is the *blank* symbol, which indicates that no symbol is actually found.

Thus, given an input image, a discrete musical symbol sequence is obtained. Note that the only limitation is that the output cannot contain more musical symbols than the number of frames of the input image, which in our case is highly unlikely to happen.

5. EXPERIMENTS

5.1 Experimental setup

Once introduced the Camera-PrIMuS dataset, and a model able to learn the OMR task from it, some experiments have been performed whose results may serve as a baseline to which other works can be compared. [5]

Currently, there is an open debate on which evaluation metrics should be used in OMR [4]. This is especially arguable because of the different points of view that the use of its output has: it is not the same whether the intention of the OMR is to automatically play the content or to archive it in a digital library. Here we are only interested in the computational aspect itself. Hence, we shall consider metrics focused on the symbol and sequence recognition, avoiding any music-specific consideration, such as:

- Sequence Error Rate (ER) (%): ratio of incorrectly predicted sequences (at least one error).
- Symbol Error Rate (SER) (%): the average number of elementary editing operations (insertions, deletions, or substitutions) needed to produce the reference sequence from the one predicted by the model, normalized by its length.

Note that the length of the agnostic and semantic sequences are usually different because they are encoding different aspects of the same source. Therefore, the comparison in terms of Symbol Error Rate, in spite of being normalized, may not be totally fair. On the other hand, the Sequence Error Rate allows a more reliable comparison because it only takes into account the perfectly predicted sequences (in which case, the outputs in different representations are equivalent).

[5] For the sake of reproducible research, source code and trained models are available at `https://github.com/calvozaragoza/tf-deep-omr`.

5.2 Performance

We show in this section the results obtained in our experiments. We consider three different data partitions: 80% of the data is used as training set, to optimize the network according to the CTC loss function; 10% of the data is used as validation set, which is used to decide when to stop the optimization to prevent over-fitting; the evaluation results are computed with the remaining 10%, which constitutes the test partition.

In order to study the ability of the system to learn in different situations, four scenarios have been evaluated depending upon which set of images are used for training and testing, either the clean original files or the synthetically distorted ones. We report in Table 3 the whole evaluation.

The results show that the system, trained with the appropriate set, is able to correctly recognize in almost all scenarios, with error rates at symbol level below 2%. In an ideal scenario, where only clean images are given, the semantic encoding outperforms the agnostic one. The behaviour is different when distorted images are used, for which the agnostic representations behave much better. What seems most interesting from these results is the ability of the system to learn from distorted images and correctly classify both distorted and clean versions. This leads us to conclude that the networks are being able to abstract the content from the image condition. As a qualitative example of the performance attained, the sample of Figure 1 was correctly classified using both encodings.

In an informal analysis, we observed that the most repeated error, both in agnostic and semantic encodings, is the incorrect classification of the ending bar line. In addition to it, no other repeating mistake has been found. Also, we checked that most of the wrongly recognized samples only failed at 1 symbol. Another interesting feature to emphasize is that we observed an independence of the mistakes with respect to the length of the ground-truth sequence, i.e., errors are not accumulated and, therefore, the number of mistakes do not necessarily increase with longer sequences. Figures 3 and 4 depict two examples of wrongly recognized sequences.

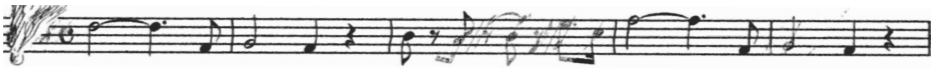
6. CONCLUSIONS

The suitability of a neural network approach to solve the OMR task in an end-to-end fashion has been evaluated on realistic single-staff printed monodic scores from a real world dataset. To this end, the new Camera-PrIMuS dataset has been introduced, containing 87 678 images synthetically distorted to resemble a camera-based scenario.

The neural network model considered consists of a CRNN, in which convolutions process the input image and recurrent blocks deal with the sequential nature of the problem. In order to train this model directly using symbol sequences, instead of fine-grained annotated images, the so-called CTC loss function has been utilized.

		Evaluation			
		Clean		Distortions	
		Agnostic	Semantic	Agnostic	Semantic
Training	Clean	1.1 / 21.7	0.8 / 12.5	44.3 / 94.1	59.7 / 97.9
	Distortions	1.4 / 24.9	3.3 / 44.6	1.6 / 24.7	3.4 / 38.3

Table 3. Average SER (%) / ER (%) reported in all possible combinations of training and evaluation conditions.



(a) Distorted image file of Incipit RISM ID no. 000104754, Incipit 1.1.1 *Achille in Sciro. Excerpts*. Niccolò Jommelli.

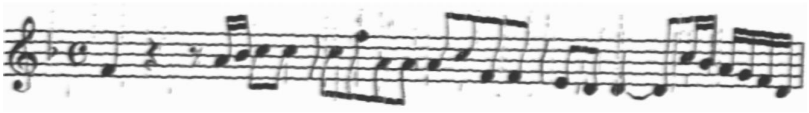
clef-G2, keySignature-DM, timeSignature-C, note-D5_half, tie, note-D5_quarter., note-F#4_eighth, barline, note-G4_half, note-F#4_quarter, rest-quarter, barline, note-B4_eighth, rest-eighth, note-A4_eighth, rest-eighth, *note-B4_half*, **[rest-eighth-L3]** note-E5_eighth., note-C#5_sixteenth, barline, note-F#5_half, tie, note-F#5_quarter., note-F#4_eighth, barline, note-G4_half, note-F#4_quarter, rest-quarter, barline

(b) Semantic encoding network output. The symbol in italics should be classified as `note-B4_eighth`, and the bold symbol between brackets has been omitted by the network.

clef.G-L2, accidental.sharp-L5, accidental.sharp-S3, metersign.C-L3, note.half-L4, slur.start-L4, slur.end-L4, note.quarter-L4, dot-S4, note.eighth-S1, barline-L1, note.half-L2, note.quarter-S1, rest.quarter-L3, barline-L1, note.eighth-L3, rest.eighth-L3, note.eighth-S2, rest.eighth-L3, *fermata.above-S6*, *note.quarter-L3*, note.beamedRight1-S4, dot-S4, note.beamedLeft2-S3, barline-L1, note.half-L5, slur.start-L5, slur.end-L5, note.quarter-L5, dot-S5, note.eighth-S1, barline-L1, note.half-L2, note.quarter-S1, rest.quarter-L3, barline-L1

(c) Agnostic encoding network output. Wrong symbols have been highlighted in italic face symbols. They should be `note.eighth-L3` and `rest.eighth-L3`, respectively.

Figure 3. This incipit contains distortions that are very hard to recognize, such as the scratch at the beginning of the staff and some overlapped ink. Despite these difficulties, just two symbols in each encoding have been wrongly recognized.



(a) Distorted image file of Incipit RISM ID no. 000100170, Incipit 1.1.1 *Trios*. Joseph Haydn.

clef-G2, keySignature-FM, timeSignature-C, note-F4_quarter, rest-quarter, rest-eighth, note-A4_sixteenth, note-Bb4_sixteenth, note-C5_eighth, note-C5_eighth, barline, note-C5_eighth, note-F5_eighth, note-A4_eighth, note-A4_eighth, note-A4_eighth, note-C5_eighth, note-F4_eighth, note-F4_eighth, barline, note-E4_eighth, note-D4_eighth, note-D4_quarter, tie, note-D4_eighth, note-C5_sixteenth, note-Bb4_sixteenth, note-A4_sixteenth, note-G4_sixteenth, note-F4_sixteenth, *note-D4_thirty_second*, barline

(b) Semantic encoding network output. The italic font face symbol should be classified as a sixteenth note.

clef.G-L2, accidental.flat-L3, metersign.C-L3, note.quarter-S1, rest.quarter-L3, rest.eighth-L3, note.beamedRight2-S2, note.beamedLeft2-L3, note.beamedRight1-S3, note.beamedLeft1-S3, barline-L1, note.beamedRight1-S3, note.beamedBoth1-L5, note.beamedBoth1-S2, note.beamedLeft1-S2, note.beamedRight1-S2, note.beamedBoth1-S3, note.beamedBoth1-S1, note.beamedLeft1-S1, barline-L1, note.beamedRight1-L1, note.beamedLeft1-S0, note.quarter-S0, slur.start-S0, slur.end-S0, note.beamedRight1-S0, note.beamedBoth2-S3, note.beamedLeft2-L3, note.beamedRight2-S2, note.beamedBoth2-L2, note.beamedBoth2-S1, note.beamedLeft2-S0, barline-L1

(c) Agnostic encoding network output. All symbols are correctly detected.

Figure 4. Incipit correctly recognized using the agnostic representation but with one mistake using the semantic encoding.

Our experiments have reflected the correct construction and the usefulness of the corpus. The end-to-end neural optical recognition model has demonstrated its ability to learn from adverse conditions and to correctly classify both perfectly clean images and imperfect pictures. In regard to the output encoding, the agnostic representation has been shown to be more robust against the image distortions, while semantic encoding maintains a fair performance.

Given these promising results, from the musical point of view, the next steps seem obvious: first, we would like to complete the catalogue of symbols, thus including chords and multiple-voice polyphonic staves. In the long-term, the intention is to consider fully-fledged real piano or orchestral scores. Concerning the most technical aspect, it would be interesting to study a multi-prediction model that uses all the different representations at the same time. Given the complementarity of the agnostic and semantic representations, it is feasible to think of establishing a synergy that ends up with better results in all senses.

7. ACKNOWLEDGEMENT

This work was partially supported by the Spanish Ministerio de Economía, Industria y Competitividad through HispaMus project (TIN2017-86576-R) and Juan de la Cierva - Formación grant (Ref. FJCI-2016-27873), and the Social Sciences and Humanities Research Council of Canada.

8. REFERENCES

[1] L. Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[2] H. Bourlard and C. Wellekens. Links between markov models and multilayer perceptrons. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(11):1167–1178, 1990.

[3] B. Brook. The Simplified 'Plaine and Easie Code System' for Notating Music: A Proposal for International Adoption. *Fontes Artis Musicae*, 12(2-3):156–160, 1965.

[4] D. Byrd and J. G. Simonsen. Towards a Standard Testbed for Optical Music Recognition: Definitions, Metrics, and Page Images. *Journal of New Music Research*, 44(3):169–195, 2015.

[5] J. Calvo-Zaragoza, F. J. Castellanos, G. Vigliensoni, and I. Fujinaga. Deep neural networks for document processing of music score images. *Applied Sciences*, 8(5):654–674, 2018.

[6] J. Calvo-Zaragoza, A.-J. Gallego, and A. Pertusa. Recognition of handwritten music symbols with convolutional neural codes. In *14th IAPR International Conference on Document Analysis and Recognition*, pages 691–696, 2017.

[7] J. Calvo-Zaragoza, A. Pertusa, and J. Oncina. Staff-line detection and removal using a convolutional neural network. *Machine Vision & Applications*, 28(5-6):665–674, 2017.

[8] J. Calvo-Zaragoza and D. Rizo. End-to-end neural optical music recognition of monophonic scores. *Applied Sciences*, 8(4):606–629, 2018.

[9] J. Calvo-Zaragoza, A. H. Toselli, and E. Vidal. Early handwritten music recognition with hidden markov models. In *15th International Conference on Frontiers in Handwriting Recognition*, pages 319–324, 2016.

[10] J. Calvo-Zaragoza, G. Vigliensoni, and I. Fujinaga. Pixel-wise binarization of musical documents with convolutional neural networks. In *Fifteenth IAPR International Conference on Machine Vision Applications*, pages 362–365, 2017.

[11] V. B. Campos, J. Calvo-Zaragoza, A. H. Toselli, and E. Vidal-Ruiz. Sheet music statistical layout analysis. In *15th International Conference on Frontiers in Handwriting Recognition*, pages 313–318, 2016.

[12] L. Chen, E. Stolterman, and C. Raphael. Human-Interactive Optical Music Recognition. In *17th International Society for Music Information Retrieval Conference*, pages 647–653, 2016.

[13] B. Couasnon. Dmos: A generic document recognition method, application to an automatic generator of musical scores, mathematical formulae and table structures recognition systems. In *6th International Conference on Document Analysis and Recognition*, pages 215–220, 2001.

[14] C. Dalitz, M. Droettboom, B. Pranzas, and I. Fujinaga. A comparative study of staff removal algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(5):753–766, 2008.

[15] J. Dos Santos Cardoso, A. Capela, A. Rebelo, C. Guedes, and J. Pinto da Costa. Staff Detection with Stable Paths. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(6):1134–1139, 2009.

[16] H. Fahmy and D. Blostein. A graph grammar programming style for recognition of music notation. *Machine Vision and Applications*, 6(2-3):83–99, March 1993.

[17] A. Gallego and J. Calvo-Zaragoza. Staff-line removal with selectional auto-encoders. *Expert Systems with Applications*, 89:138–48, 2017.

[18] T. Géraud. A morphological method for music score staff removal. In *21st International Conference on Image Processing*, pages 2599–2603, Paris, France, 2014.

[19] X. Glorot, A. Bordes, and Y. Bengio. Deep sparse rectifier neural networks. In *Fourteenth International Conference on Artificial Intelligence and Statistics*, pages 315–323, 2011.

[20] M. Good and G. Actor. Using MusicXML for File Interchange. *International Conference on Web Delivering of Music*, page 153, 2003.

[21] A. Graves. *Supervised sequence labelling with recurrent neural networks*. PhD thesis, Technical University Munich, 2008.

[22] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *23rd International Conference on Machine Learning*, pages 369–376, 2006.

[23] A. Graves, A.-R. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6645–6649, 2013.

[24] A. Graves and J. Schmidhuber. Offline handwriting recognition with multidimensional recurrent neural networks. In *Advances in neural information processing systems*, pages 545–552, 2009.

[25] J. Hajic and P. Pecina. The MUSCIMA++ dataset for handwritten optical music recognition. In *14th IAPR International Conference on Document Analysis and Recognition*, pages 39–46, 2017.

[26] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[27] S. Ioffe and C. Szegedy. Batch normalization: accelerating deep network training by reducing internal covariate shift. In *32nd International Conference on Machine Learning*, pages 448–456, 2015.

[28] M. Kassler. Optical character-recognition of printed music: A review of two dissertations. *Perspectives of New Music*, 11(1):250–254, 1972.

[29] K. Keil and J. A. Ward. Applications of RISM data in digital libraries and digital musicology. *International Journal on Digital Libraries*, 50(2):199, January 2017.

[30] S. Lee, S. J. Son, J. Oh, and N. Kwak. Handwritten music symbol classification using deep convolutional neural networks. In *3rd International Conference on Information Science and Security*, 2016.

[31] A. Pacha, K.-Y. Choi, B. Coüasnon, Y. Ricquebourg, R. Zanibbi, and H. Eidenberger. Handwritten music object detection: Open issues and baseline results. In *13th IAPR Workshop on Document Analysis Systems*, 2018.

[32] A. Pacha and H. Eidenberger. Towards a universal music symbol classifier. In *12th International Workshop on Graphics Recognition*, pages 35–36, 2017.

[33] L. Pugin. Optical music recognition of early typographic prints using hidden markov models. In *7th International Conference on Music Information Retrieval*, pages 53–56, 2006.

[34] L. Pugin, R. Zitellini, and P. Roland. Verovio - A library for Engraving MEI Music Notation into SVG. In *International Society for Music Information Retrieval*, 2014.

[35] L. Rabiner and B.-H. Juang. *Fundamentals of speech recognition*. Prentice hall, 1993.

[36] C. Raphael and J. Wang. New Approaches to Optical Music Recognition. In *12th International Society for Music Information Retrieval Conference*, pages 305–310, 2011.

[37] A. Rebelo, A. Capela, and J. S. Cardoso. Optical recognition of music symbols: A comparative study. *International Journal on Document Analysis and Recognition*, 13(1):19–31, March 2010.

[38] A. Rebelo, I. Fujinaga, F. Paszkiewicz, A. R. S. Marçal, C. Guedes, and J. S. Cardoso. Optical music recognition: state-of-the-art and open issues. *International Journal of Multimedia Information Retrieval*, 1(3):173–190, 2012.

[39] P. Roland. The music encoding initiative (MEI). In *Proceedings of the First International Conference on Musical Applications Using XML*, pages 55–59, 2002.

[40] F. Rossant and I. Bloch. Robust and adaptive omr system including fuzzy modeling, fusion of musical rules, and possible error detection. *EURASIP Journal on Advances in Signal Processing*, 081541, 2007.

[41] E. Selfridge-Field. *Beyond MIDI: The handbook of musical codes*. MIT Press, 1997.

[42] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[43] M. Szwoch. Guido: A musical score recognition system. In *9th International Conference on Document Analysis and Recognition*, pages 809–813, 2007.

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

[45] G. Vigliensoni, G. Burlet, and I. Fujinaga. Optical measure recognition in common music notation. In *14th International Society for Music Information Retrieval Conference*, pages 125–30, 2013.

[46] M. D. Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[47] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *13th European Conference on Computer Vision — Part I*, pages 818–833, 2014.