International Journal of Multimedia Information Retrieval (2023) 12:12
https://doi.org/10.1007/s13735-023-00278-5

REGULAR PAPER

Optical music recognition for homophonic scores with neural networks and synthetic music generation

María Alfaro-Contreras[1] · José M. Iñesta[1] · Jorge Calvo-Zaragoza[1]

Received: 25 May 2022 / Revised: 27 January 2023 / Accepted: 20 March 2023

Abstract
The recognition of patterns that have a time dependency is common in areas like speech recognition or natural language processing. The equivalent situation in image analysis is present in tasks like text or video recognition. Recently, Convolutional Recurrent Neural Networks (CRNN) have been broadly applied to solve these tasks in an end-to-end fashion with successful performance. However, its application to Optical Music Recognition (OMR) is not so straightforward due to the presence of different elements sharing the same horizontal position, disrupting the linear flow of the timeline. In this paper, we study the ability of the state-of-the-art CRNN approach to learn codes that represent this disruption in homophonic scores. In our experiments, we study the lower bounds in the recognition task of real scores when the models are trained with synthetic data. Two relevant conclusions are drawn: (1) Our serialized ways of encoding the music content are appropriate for CRNN-based OMR; (2) the learning process is possible with synthetic data, but there exists a *glass ceiling* when recognizing real sheet music.

Keywords Optical music recognition · Deep learning · End-to-end recognition · Music encoding

1 Introduction

Music amounts to a language used and understood worldwide. It is an art that has been crossing borders since its inception, being one of the main cultural manifestations of the human being. It is for this reason that over the centuries there has been a need to preserve the content in the best possible way, whether in cathedrals, libraries, or historical archives. However, access to these documents is often limited since continued use may end up damaging them irretrievably.

There exist multiple projects and organizations whose purpose is comprehensively documenting extant historical sources of music all over the world, such as the following ones:

- International Music Score Library Project[1]
- *Répertoire International des Sources Musicales* (RISM)[2]
- Choral Public Domain Library[3]
- Mutopia[4]
- Classical Archives Collection[5]
- OpenScore[6]
- Cantus Manuscript Database[7]

All of them are making a great effort to digitize musical scores into images, allowing their collections to be accessible as images through the Internet. But for those musical documents to be truly accessible, they must be transcribed into a digital format that enables tasks such as indexing, editing, or critical publication. This process is often done manually and is costly and tedious. Score editing tools are complex to use, which makes the process prone to introducing errors;

✉ María Alfaro-Contreras
malfaro@dlsi.ua.es

José M. Iñesta
inesta@dlsi.ua.es

Jorge Calvo-Zaragoza
jcalvo@dlsi.ua.es

[1] Instituto Universitario de Investigación Informática, University of Alicante, Ap. 99, 03080 Alicante, Spain

[1] https://imslp.org.
[2] http://www.rism.info.
[3] http://www.cpdl.org.
[4] https://www.mutopiaproject.org.
[5] https://www.classicalarchives.com.
[6] https://openscore.cc.
[7] http://cantus.uwaterloo.ca.

Published online: 26 May 2023

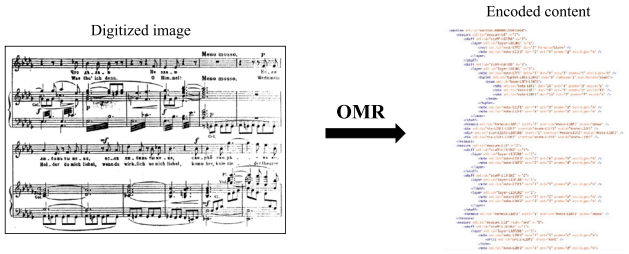
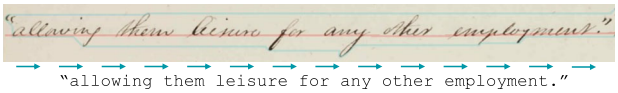
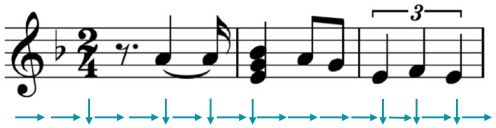


Fig. 1 By using OMR techniques, the content in a digitized image can be encoded in a symbolic format



(a) The recognition process of text does follow a linear left-to-right flow.



(b) The recognition process of music does not follow a linear left-to-right flow.

Fig. 2 Differences in reading between text and music

thereby, several rounds of review are needed to approve a transcript as a good one. In certain scenarios, such as those related to ancient musical documents, there may not even be suitable tools for that. All of this entails a great deal of work that is not feasible on a large scale. That is why it is very important to find technologies capable of revaluing all the existing musical heritage.

A promising alternative that would overcome the previous challenge is the use of automatic recognition techniques. Here is where Optical Music Recognition (OMR) comes in. OMR is the field of research that investigates how to computationally read music notation in documents [9, 16].

As seen in Fig. 1, a digitized image can be converted into encoded content automatically by means of OMR. This encoded content is the digital transcription (in terms of music notation symbols) of the score. Thus, an effective OMR system enables the study of existing musical documents for digital humanities. And not only that, but it is the only alternative capable of doing so in reasonable time and cost.

OMR has been an active research field for decades [4, 32]. Traditional approaches to OMR are based on the usual pipeline of sub-tasks that characterizes many artificial vision systems, adapted to this particular task: document pre-processing [7, 29]—including staff-line removal [10, 15]—symbol classification [28, 31], reconstruction of the music notation [27, 30], and output encoding in a suitable symbolic format.

In recent years, there has been a paradigm shift toward the use of Machine Learning (ML) techniques. These techniques make it possible to design flexible and versatile OMR systems capable of solving a wide variety of problems. This is due to the relationship between the purpose of both fields: ML studies how to make machines learn to perform certain tasks, which is exactly what OMR seeks, to teach machines to perform the task of reading musical scores.

Recent advances in ML—namely Deep Learning (DL)—which have achieved great results in several visual challenges [24], allow us to be optimistic about developing more accurate and effective OMR systems. The current trend is the use of end-to-end (or holistic) systems that treat the process as a single step, instead of explicitly performing the sub-tasks. Using this approach, training pairs only have to contain the input image and its complete transcription [5, 13], bypassing especially the need to annotate the exact positions of individual symbols.

These approaches typically rely on Convolutional Recurrent Neural Networks (CRNN), which are only able to formulate the output as one-dimensional sequences. This perfectly fits natural language tasks (text or speech recognition, or machine translation) since their outputs mostly consist of character (or word) sequences (see Fig. 2a). However, its application to music notation is not so straightforward due to the presence of different elements sharing the same horizontal position and long-term dependencies. The vertical distribution of these elements disrupts the linear flow of the timeline (see Fig. 2b). This fact is not trivial to encode and can cause significant difficulties in the performance of recognition systems that make use of the temporal relationships between the recognized elements.

The problem can be drastically simplified by considering that the process will work with each staff independently from the others—a process that could be analogous to the text recognition systems that decompose the document into a series of independent lines [21]. This is not a strong assumption as there are successful algorithms for identifying staves [17]. Even so, we still have to deal with elements that take place simultaneously in the "time" line, like the notes that make up a chord, irregular groups, or expression marks, to name a few.

Within the range of music score complexities, one possible simplification of the problem that applies to many sheet music is to assume a *homophonic* music context. In that case, there are multiple parts, but they move in the same rhythm. This way, multiple notes can occur simultaneously, but only as a single voice. Therefore, all the notes starting at the same time last the same, so the score can be segmented into verti-

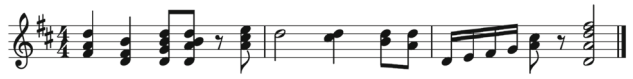


Fig. 3 Homophonic music: All the notes starting at the same time have the same duration

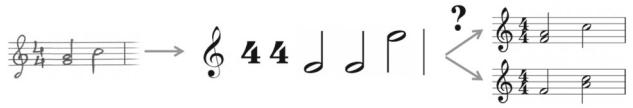


Fig. 4 Example of ambiguities that might appear when music symbols belong to a vertical distribution. The two notes that appear together, must be played at the same time, but a linear symbol sequence without specific marks cannot be interpreted unambiguously

cal slices that may contain one or more music symbols (see Fig. 3).

Even in this simplified context, there is a need for a clear and structured output coding that avoids the ambiguities that the representation of a linear output can show in presence of vertical structures in the data (see Fig. 4). This has also been stated in some previous works [6].

There already exist several structured formats for music representation and coding, like XML-based music formats [18, 22] that are focused on how the score has to be encoded to properly store all its content. This application makes it unsuitable to adopt them as output for an optical recognition system because the code is full of irrelevant markings for the system to generate when graphically analyzing the score. An OMR system is primarily interested in what symbols are there and where they are. Furthermore, these XML-based music languages do not represent sequential data, but rather hierarchical structures, so they are not suitable output formats for DL-based OMR.

Due to that, we have designed a specific coding language to represent the output of end-to-end OMR, based on serializing the music symbols found in a staff of homophonic music. The sequential nature of music reading must be compatible and unambiguous with respect to the representation of vertical distributions. In addition, this representation has to be easy to generate by the system, which analyzes the input sequentially and produces a linear series of symbols.

Preliminary research has been carried out to validate whether this represents a feasible research avenue [2]. However, the serialized ways of encoding the music content have never been tested with real data. This work aims to solve that and properly evaluate the problem. Also, we study and evaluate the possible existing boundaries of learning with synthetic data when recognizing real music scores.

The rest of the paper is organized as follows: Sect. 2 overviews the state-of-the-art recognition framework based on DL, including the ad hoc serializations for homophonic music; Sect. 3 introduces the experimental setup; Sect. 4 shows the results obtained with real homophonic sheet music; finally, Sect. 5 concludes the present work, along with some ideas for future research.

2 Recognition framework

To carry out the OMR task in an end-to-end manner, we follow the state-of-the-art approach based on Convolutional Recurrent Neural Networks (CRNN). These neural architectures permit us to model the posterior probability of generating output symbols, given an input image. Input images are assumed to be single-staff sections, analogously to text recognition that assumes independent lines [21]. As mentioned before, staves can be easily isolated by means of existing methods [17].

A CRNN consists of one block of *convolutional* layers followed by another block of *recurrent* layers [33]. The convolutional block is responsible for learning how to process the input image, that is, extracting relevant image features for the task at issue so that the recurrent layers interpret these features in terms of sequences of musical symbols. In this work, the recurrent layers are implemented as Bidirectional Long Short-Term Memory (BLSTM) units [19].

The unit activations of the last convolutional layer can be seen as a sequence of feature vectors representing the input image, $\mathbf{x}$. These features are fed to the first BLSTM layer, and the unit activations of the last recurrent layer are considered estimates of the posterior probabilities for each vector:

$$P(\sigma \mid \mathbf{x}, f), \ 1 \leq f \leq F, \ \sigma \in \Sigma \tag{1}$$

where F is the number of feature vectors of the input sequence and Σ is the set of considered symbols. Note that Σ must include a "*non-character*" symbol ϵ that acts as a separator when two or more instances of the same musical symbol appear consecutively [19].

Since both convolutional and recurrent blocks can be trained through gradient descent, using the well-known *Back Propagation* algorithm [34], a CRNN can be jointly trained. However, a conventional end-to-end OMR training set only provides, for each staff image, its corresponding transcription, not giving any type of explicit information about the location of the symbols in the image. It has been shown that the CRNN can be conveniently trained without this information by using the so-called Connectionist Temporal Classification (CTC) loss function [20]. The resulting CTC training procedure is a form of Expectation-Maximization: CTC provides a means to optimize the CRNN parameters so that it is likely to give the correct sequence given an input [20].

Once the CRNN has been trained, an input staff image can be decoded into a sequence of music symbols $\hat{\boldsymbol{s}} \in \Sigma^*$. First,

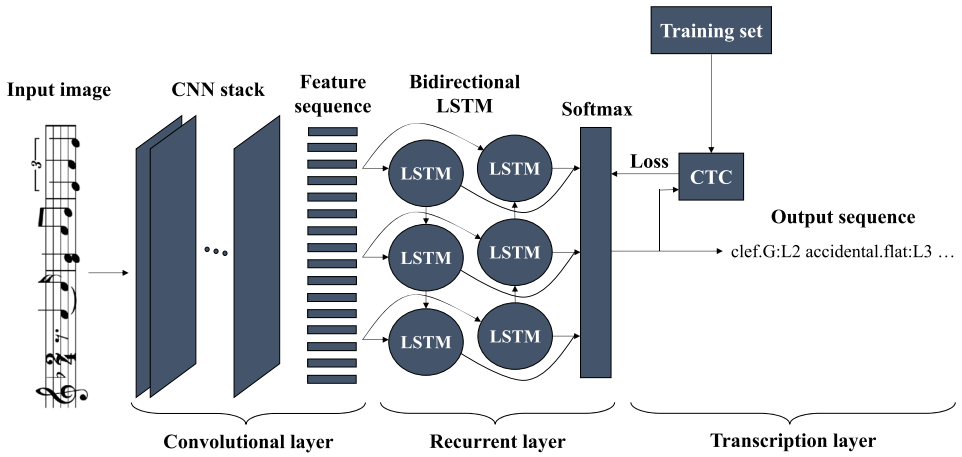


Fig. 5 Graphical scheme of the CRNN considered for the end-to-end approach. The network is trained by using the CTC loss function

the most probable symbol per frame is computed:

$$\hat{\sigma}_i = \arg\max_{\sigma \in \Sigma} P(\sigma \mid \mathbf{x}, i), \;\; 1 \leq i \leq F \tag{2}$$

Then, a pseudo-optimal output sequence is obtained as:

$$\hat{\boldsymbol{s}} = \arg\max_{s \in \Sigma^*} P(\mathbf{s} \mid \mathbf{x}) \approx \mathcal{D}(\hat{\sigma}_1, \ldots, \hat{\sigma}_F) \tag{3}$$

where $\mathcal{D}$ is a function that first merges all the consecutive frames with the same symbol and then deletes the ϵ symbols [19].

A graphical scheme of the framework explained above is given in Fig. 5.

This framework follows the architecture first applied in the work of Shi et al. [33] and later tuned by Calvo-Zaragoza et al. [11]. As stated in the latter work, its expressiveness could be sufficient when working with simple scores where all the symbols have a single left-to-right order. However, we want to extend these approaches so that they are able to model richer scores such as those of homophonic sheet music. In such a case, issues like chords may appear, where several symbols share a horizontal position. As seen in Fig. 4, a one-dimensional sequence is not expressive enough for this situation. That is why in the next section we describe our representation proposal to perform end-to-end OMR for homophonic scores.

2.1 Serialization proposals

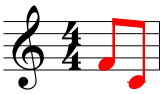
The research developed in this paper involves the study of four different deterministic, unambiguous, and serialized representations to encode the kind of scenarios that happen in homophonic music so that the OMR system becomes more effective when recognizing complex music score images. For that, we propose four different types of music representations that differ not in the encoding of the musical symbols itself, but in the way horizontal and vertical distributions of the musical symbols are represented. The grammar for these musical codifications must be unambiguous, allowing us to analyze a given document in only one way.

Our representation does not make assumptions about the musical meaning of what is represented in the document being analyzed; that is, the elements are identified in a catalogue of musical symbols by their shape and where they are placed on the staff. This has been referred to as "agnostic representation," as opposed to a semantic representation, where music symbols are encoded according to what they represent in terms of music notation [11]. This difference is illustrated in Fig. 6.

As mentioned before, the only difference between the four proposed musical codes is how to represent the horizontal and vertical dimensions. Each one of the four codes has one or two characters that indicate whether, when transcribing the score, the system should move forward, that is, from left to right, or downward, from top to bottom. These characters are referred to as *separators*.

The four different codes proposed are described as follows:

- *Remain-at-position character code* when transcribing the score, the different musical symbols are assumed to be placed left to right, except when they are in the same horizontal position. In that case, they are separated by a slash, "/". This acts as a remain-at-position character, meaning that the system has to advance downward (see Fig. 7b). This behavior is similar to the backspace of typewriters. The carriage advances after typing, and if we want to align two symbols, we need to keep the carriage in a fixed position (by moving it back to one position).
- *Advance-position character code* this type of coding uses a "+" sign to force the system to advance forward. This way, when that sign is missing, the output does not move forward and a vertical distribution is being coded (see Fig. 7c).



Semantic representation: *musical annotation*

note-F4_eighth note-C4_eighth

Agnostic representation: *graphics annotation*

note.beamedRight1:S1 note.beamedLeft1:L0

Fig. 6 Semantic and agnostic representations, respectively, of the two eighth notes forming a beam group (highlighted in red). The agnostic representation only provides graphic information—that is, the line or space of the staff on which the note is placed, or the direction of its beam—as opposed to the musical information (pitch and type of note) of a semantic representation (colour figure online)

- *Parenthesized code* when a vertical distribution appears in the score, the system outputs a parenthesized structure, like **vertical.start** musical_symbol . . . musical_symbol **vertical.end** (see Fig. 7d).
- *Verbose code* this last coding is a combination of the two first ones. It uses the "+" sign as the advance-position character to indicate that the system has to move forward, and the "/" sign as the remain-at-position character to indicate that the system has to advance downward (see Fig. 7e). So, here, every two adjacent symbols are explicitly separated by a symbol indicating whether the system must remain in the same horizontal position or has to advance to the next one.

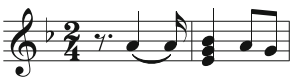
Note that the codes represent the data unambiguously; thus, it is possible to deterministically translate from any encoding to any other.

3 Experimental setup

In this section, we present the synthetic training dataset and the real RISM test dataset, the neural model architecture, and the evaluation protocols used.

3.1 Synthetic data: corpus generation

As introduced above, the current trend for the development of OMR systems is to use DL techniques able to infer the transcription from correct examples of the task, namely, set of pairs (**x**, **s**). Given the complexity of music notation, for these techniques to produce satisfactory results, it is necessary to use a set of sufficient size. To achieve this, a system of automatic generation of labeled data has been developed [1] by using algorithmic composition techniques [26]. The developed system provides two outputs: the expected transcription of the generated score in any of the encodings described in Sect. 2.1, and the corresponding score image artificially distorted as in [12]. With both outputs, the necessary pairs for the DL algorithm are obtained.



(a) Musical excerpt.

clef.G:L2 accidental.flat:S3 digit.2:L4 / digit.4:L2 rest.eighth:L3 dot:S3 note.quarter:S2 / slur.start:S2 note.sixteenth:S2 / slur.end:S2 verticalLine:L1 note.quarter:L3 / note.quarter:L2 / note.quarter:L1 note.beamedRight:S2 note.beamedLeft:L2 verticalLine:L1

(b) Remain-at-position character code.

clef.G:L2 + accidental.flat:S3 + digit.2:L4 digit.4:L2 + rest.eighth:L3 + dot:S3 + note.quarter:S2 slur.start:S2 + note.sixteenth:S2 slur.end:S2 + verticalLine:L1 + note.quarter:L3 note.quarter:L2 note.quarter:L1 + note.beamedRight:S2 + note.beamedLeft:L2 + verticalLine:L1

(c) Advance-position character code.

clef.G:L2 accidental.flat:S3 vertical.start digit.2:L4 digit.4:L2 vertical.end rest.eighth:L3 dot:S3 vertical.start note.quarter:S2 slur.start:S2 vertical.end vertical.start note.sixteenth:S2 slur.end:S2 vertical.end verticalLine:L1 vertical.start note.quarter:L3 note.quarter:L2 note.quarter:L1 vertical.end note.beamedRight:S2 note.beamedLeft:L2 verticalLine:L1

(d) Parenthesized code.

clef.G:L2 + accidental.flat:S3 + digit.2:L4 / digit.4:L2 + rest.eighth:L3 + dot:S3 + note.quarter:S2 / slur.start:S2 + note.sixteenth:S2 / slur.end:S2 + verticalLine:L1 + note.quarter:L3 / note.quarter:L2 / note.quarter:L1 + note.beamedRight:S2 + note.beamedLeft:L2 + verticalLine:L1

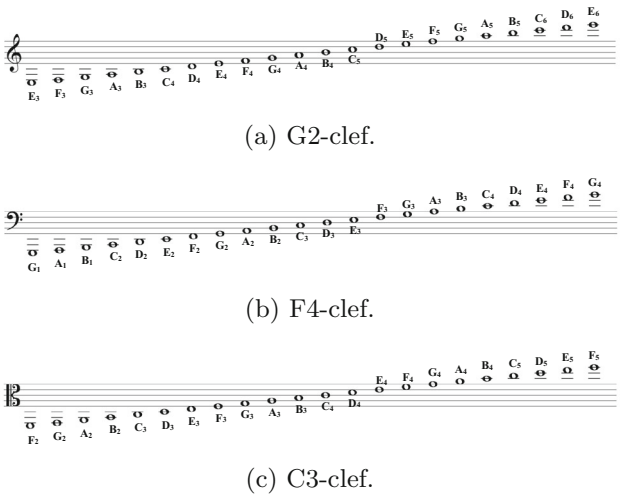
(e) Verbose code.

Fig. 7 Musical excerpt presenting a number of different situations where vertical distributions occur and its transcription using the proposed codifications

For the generation system, three different methods of algorithmic composition are used to obtain compositions with diverse musical features. Furthermore, the range of pitches that can be coded according to the clef is limited in order to achieve a score with as much musical coherence as possible. In the end, in music scores, according to the clef, there is a range of pitches that is more common than another. A range of 22 different pitches is chosen for each clef (see Fig. 8). The pitch series defined for each clef are major or minor diatonic scales in some keys. The clefs that can be encoded by the system are: G1-clef, G2-clef, F4-clef, C1-clef, C2-clef, C3-clef, and C4-clef.

The score generator creates a musical event one at a time. Such an event might be sound (by default 90% of the time) or silence (10% the time). The type of sound or silence event is chosen from a catalog of possible music symbols (from sixteenth to whole notes, as well as silences, beamed notes, or chords comprising up to a maximum of 3 notes, triplets, etc.) following a random process. The pitch of the sound events is conditioned to one of three possible algorithmic composition methods described below. Only a general outline of the score generator is given, as more specific details are beyond the scope of this work. For complete details about the implementation, please refer to [1].[8]

[8] The code developed in the work is publicly available for reproducible research at: https://github.com/mariaalfaroc/ScoreGenerator.git.



(a) G2-clef.

(b) F4-clef.

(c) C3-clef.

Fig. 8 Range of the 22 pitches coded for the 3 system clefs considered

3.1.1 Random generation according to the normal distribution

The normal distribution, Gaussian distribution, or Laplace-Gauss distribution, is a probability distribution of continuous variable. It is of great application in the fields of engineering, physics, and social sciences because it allows us to model numerous natural, social, and psychological phenomena.

The normal distribution is defined in Eq. 4, where 0 and N represent the extreme values of the domain, being, in our case, 0 and 21, respectively—integers associated with specific pitches depending on the clef used. This way, the range of pitches associated with each clef links each pitch to an integer in [0, 21] so that 0 maps to the lowest pitch of the range and 21 to the highest pitch.

$$f_{[0,N]}(x) = N(\mu, \sigma) = \frac{\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)}{\sqrt{2\pi}\sigma} \tag{4}$$

We use the $N(10.5, 6.5)$ distribution that provides a symmetric distribution centered on the mean, which corresponds to the space surrounding the central line of the staff. The mean defines the location of the peak for normal distributions, but, in our case, since the mean is a decimal value, it is not associated with any pitch, and therefore the highest probabilities are given to both rounded up (third line of the staff) and rounded down (third space of the staff) values of the mean. The probability is minimum at the extreme values, being $f_{[0,N]}(0) = f_{[0,N]}(N) = 0.0166$, i.e., 1.66% probability of occurrence for those pitches. This is illustrated in Fig. 9.

As we can see in Fig. 9, the graph of its density function has a bell shape and is symmetric with respect to the average. This curve is known as Gaussian bell and is the graph of a Gaussian function.

Fig. 10 shows an example of a musical excerpt created by the automatic generation system when the normal distribution composition method is used.

3.1.2 Random walk

A random walk is a mathematical formalization of the trajectory that results from making successive random steps. In this system, the random walk always starts at the central pitch of the pitch range determined for the system. There are three possible random steps (all equally likely) after emitting pitch:

1. One-step forward: The pitch that follows is the next higher in the defined pitch series.
2. One-step backward: The pitch that follows is the next lower in the defined pitch series.
3. No step: The pitch that follows is the same as the current one.

There are situations in which moving forward or backward will not be allowed because the pitch to be coded would be outside the established range. In these situations, two solutions are given:

- *Reflective limit* as its own name indicates, it works like a mirror, making the step to take to be the reflection of what was initially intended to be taken. If the movement was intended to be forward, now it will be backward and vice versa. That is, the pitch of the current note is the second of the range starting either from the upper limit or from the lower one, as appropriate.
- *Absorbing limit* the pitch of the current note is that corresponding to the upper or lower limit, as appropriate, of the pitch range.

The solution is chosen randomly, both being equal of probable.

Fig. 11 shows an example of a musical excerpt created by the automatic generation system when the random walk composition method is used.

3.1.3 Sonification of the logistic equation

The logistic equation is defined by Eq. 5:

$$x_{n+1} = r\, x_n(1 - x_n) \quad where\ n = 0, 1, 2, 3, \ldots \tag{5}$$

This equation defines an iteration, where x_0 is equal to 0 and the parameter r is a value between 0 and 4. The resulting value will always be in [0, 1].

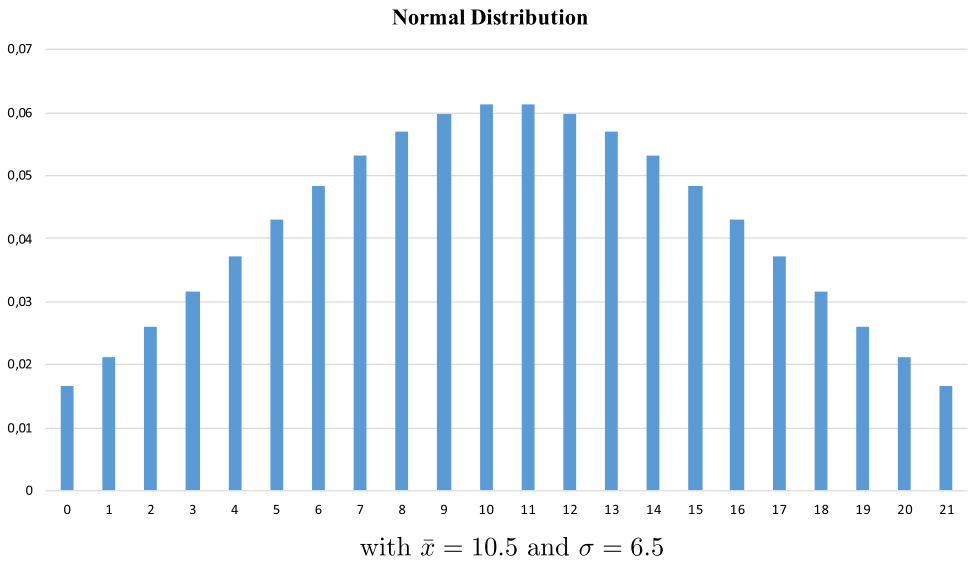


Fig. 9 Gaussian distribution for the automatic generation of music data

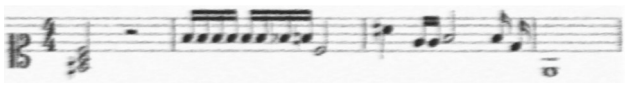


Fig. 10 Music snippet created by the automatic generation system using the normal distribution ($N(10.5, 6.5)$) composition method

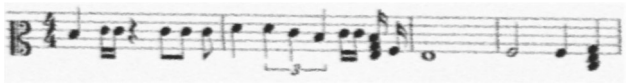


Fig. 11 Music snippet created by the automatic generation system using the random walk composition method

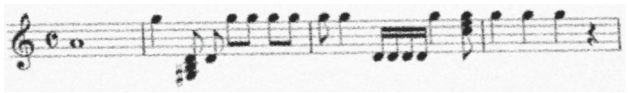


Fig. 12 Music snippet created by the automatic generation system using the sonification of the logistic equation ($r = 3.75$)) composition method

When this method is used, the value for r should be between 3.5 and 4, since it is the range of values for which the most interesting note sequences are generated. The sequences for $3 \leq r \leq 3.5$ produce repetitions of 2 or 4 pitches, with no variability. Values for $r < 3$ generate constant pitch (unison) sequences after a short transition period.

Fig. 12 shows an example of a musical excerpt created by the automatic generation system when the sonification of the logistic equation composition method is used.

The first and the last methods generate skipwise or disjunct melodic motions, characterized by frequent skips between notes, whereas the second method produces stepwise or conjunct melodic motions, where all the intervals will never be greater than four semitones. These differences lead us to consider whether the pitch generation method used to generate the training set affects the way the DL model used for the OMR task learns. Since what we want is to find the most possible optimal scenario for the OMR learning task, we consider this to be an issue that needs further investigation and decide to approach it together with the coding proposals in the next section.

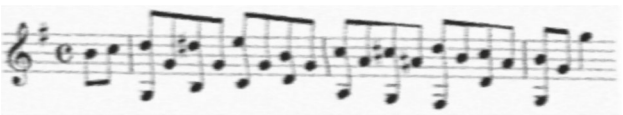


Fig. 13 Staff sample from the selected RISM corpus. *Incipit RISM ID no. 000139189111*

3.2 Real data

We aim to find the most favorable scenario for the OMR learning task when training with homophonic synthetic scores and testing with homophonic real scores. For a proper evaluation, we consider a sufficiently large set of real data taken from the RISM repository that contains the same symbols that our score generator is able to produce. The selected corpus contains 1 954 real music staves of homophonic *incipits*.[9] For each incipit, an image with the rendered score with artificial distortions—the same as those used for the synthetic data—as well as the expected transcription in any of the encodings described in Sect. 2.1 is provided. Figure 13 depicts an example of a particular staff from this corpus. It must be noted that the Camera-based Printed Images of Music Staves (Camera-PrIMuS) database [12] is not suitable for the present work since it contains only monophonic RISM-based music scores.

[9] Short sequences of notes, typically the first measures of the piece, used to index and identify a melody or musical work.

Table 1 Layer-wise description for the CRNN architecture considered. Notation: Conv($f, w \times h$) stands for a convolution layer of f filters of size $w \times h$ pixels, BatchNorm performs the normalization of the batch, LeakyReLU(α) represents a Leaky Rectified Linear Unit activation with a negative slope value α, MaxPool($w \times h$) represents the max-pooling operator of $w \times h$ dimensions and striding factors, and BLSTM(n, d) denotes a bidirectional Long Short-Term Memory unit with n neurons and d dropout value parameters

Convolutional block				Recurrent block	
Layer 1	Layer 2	Layer 3	Layer 4	Layer 5	Layer 6
Conv(64, 5 × 5)	Conv(64, 5 × 5)	Conv(128, 3 × 3)	Conv(128, 3 × 3)		
BatchNorm	BatchNorm	BatchNorm	BatchNorm	BLSTM(256)	BLSTM(256)
LeakyReLU(0.20)	LeakyReLU(0.20)	LeakyReLU(0.20)	LeakyReLU(0.20)	Dropout(0.50)	Dropout(0.50)
MaxPool(2 × 2)	MaxPool(1 × 2)	MaxPool(1 × 2)	MaxPool(1 × 2)		

The selected RISM set amounts to a total 63 011 music symbols, representing 341 different classes. From the total number of symbols, 15 699 belong to a vertical distribution—two or more symbols that share the same horizontal position. There are 7 546 vertical distributions.

3.3 Neural network configuration

As mentioned in Sect. 2, the neural model considered in this work is based on the architecture by Calvo-Zaragoza et al. [11]. In this sense, while the configuration is broadly described in Sect. 2, the actual composition of each layer is depicted in Table 1.

The model is trained following the backpropagation method provided by CTC for 300 epochs using the ADAM optimizer [23] with a fixed learning rate 0.001 and a batch size of 16 elements.

3.4 Evaluation protocol

Concerning evaluation metrics, there is an open debate on how to evaluate the capabilities of OMR systems [8, 25]. In this work, OMR is simply understood as a pattern recognition task, so we shall consider metrics that allow us to draw reasonable conclusions from the experimental results. Due to that, the performance of the recognition schemes presented is assessed by considering the symbol error rate (SER, %) as utilized in previous works addressing end-to-end transcription tasks [12]. This figure of merit is computed as the average number of elementary editing operations (insertions, deletions, or substitutions) necessary to match the sequence predicted by the model with the ground truth sequence, normalized by the length of the latter.

The number of separators used in the transcription to deal with simultaneities in the horizontal dimension (the "time" line) is different for the four proposed encodings. This fact might have implications for learning. Hence, we also report the results restricted to the symbols that are not separators: We refer to the SER metric as *NonSep-SER*. To obtain more insights into the system's performance on simultaneities as opposed to monophonic segments, we also show the proportion of NonSep-SER caused by the editing operations that happen in regions (i) that belong to a simultaneity in the ground truth, defined as *Sim-SER*, and (ii) that are monophonic in the ground truth, denoted as *NonSim-SER*.

4 Results

This work aims to provide insights into (i) which serialized ways of encoding the music content presented in Sect. 2.1 are more suitable for recognizing real homophonic music scores and (ii) how the use of synthetic training data affects the transcription of the previously mentioned data. For that, we specifically consider two evaluation cases: a first one, denoted as *Best Encoding Experiment*, devoted to finding the most suitable code out of the four proposed in Sect. 2.1 for the OMR output; and a second one, named *Best Algorithmic Composition Method Experiment*, that studies the most appropriate composition method for the OMR learning task.

It must be noted that for all the considered scenarios the evaluation set refers to the real homophonic scores collected from RISM. Hence, the synthetic data generated are created in a way that it contains a similar number of measures per staff, symbols per staff, and vertical distributions per staff—all of them on average—as the selected RISM corpus. Moreover, the generated data are distorted in the same way as the RISM data are.

4.1 Best encoding experiment

The experiment aims to determine which of the four serialization proposals works best as the output for the OMR task in a homophonic music scenario. To do so, a corpus of 1 500 labeled scores, each consisting of a single staff, is generated using the system for automatic generation of labeled data explained in the previous section. Each sample is a pair composed of the image with a rendered staff and its corresponding representation with the format imposed by one of the four musical encodings proposed, like in the example

Table 2 Results obtained for each encoding when models are tested on the 1 954 selected RISM incipits. *Remain* stands for *remain-at-position character* coding, *Advance* stands for *advance-position character* coding, *Parenthesized* stands for *parenthesized* coding, and *Verbose* stands for *verbose* coding. Best results are highlighted in bold type

	Remain	Advance	Parenthesized	Verbose
SER (%)	26.8	**17.1**	26.3	19.1
NonSep-SER (%)	26.8	**26.0**	26.9	28.3
Sim-SER (%)	09.5	**09.2**	09.2	09.9
NonSim-SER (%)	17.3	**16.8**	17.7	18.4

shown in Fig. 7. As explained above, the operation of the three composition methods of the automatic generation system makes them produce heterogeneous music scores. That is why the three of them are equally used when generating this corpus so that it is not biased in favor of any particular style. We refer to this procedure as the *mix* composition method.

We derive two non-overlapping partitions—train and validation—corresponding to 60% and 40% of the data, respectively, following a fivefold cross-validation scheme. Each fold is tested on the selected RISM corpus described in Sect. 3.2. The results obtained in terms of the SER metric—the figures provided represent the average values for the test partition in which the validation data achieve its best performance for each of the considered cases—are presented in Table 2.

An initial remark is that the results depicted in Table 2 indicate that the neural network is indeed learning from synthetic data but, as seen in previous efforts [2], the encoding of the output for this OMR task plays an important role in the training, and consequently, in the recognition performance. The advance-position character coding achieves the best results for the NonSep-SER. The remain-at-position character and the parenthesized codes follow closely, both with similar results, while the verbose code places the last. This order is altered when the results are observed from the SER metric side: Wordy encodings are favored, leading to deceiving insights. In other words, a more wordy code, e.g., the verbose encoding, could predict all the separators symbols correctly while missing the remaining symbols—the ones that would really matter to the user—and still achieve a lower error rate than another less verbose encoding, e.g., the remain encoding.

We believe that the fact that the advance encoding performs better is due to how the CTC loss function works: The system reads vertical slices and outputs the symbols present on them. In the case of contiguous input frames containing only staff lines, the system could output either nothing or an "empty-output" symbol, such a symbol being the "+" advance separator. On the opposite side, we find that the remain and parenthesized codes overload the learning process as we are forcing the system to output more symbols in the same situation. This idea is reinforced by the verbose encoding's results. Therefore, neither the remain-at-position character code nor the parenthesized code nor the verbose code is a suitable choice for transcribing the content of homophonic scores with the CTC objective function.

Table 3 Statistical significance analysis of the different presented encoding schemes considering the Wilcoxon signed-rank test with a significance value of $p < 0.05$ for the symbol error rate metric when the corresponding separators symbols are excluded from the computation, i.e., NonSep-SER. Symbols $<$, $>$, and $=$ represent that the error of the method in the row is significantly lower than, greater than, or no different to that in the column, respectively

	Remain	Advance	Parenthesized	Verbose
Remain	–	$>$	$=$	$<$
Advance	$<$	–	$<$	$<$
Parenthesized	$=$	$>$	–	$<$
Verbose	$>$	$>$	$>$	–

As the last remark, we would like to point out that the results obtained suggest that simultaneities do not present a recognition problem by themselves. When decomposing the NonSep-SER into its two component fractions, Sim-SER and NonSim-SER, Table 2 reports that the highest proportion of errors occurs in monophonic zones.

To support the relevance of those statements, we shall now assess the results in terms of statistical significance. For that, we resort to the nonparametric Wilcoxon signed-rank test [14]. This analysis considers that each result obtained for each fold constitutes a sample of the distributions to compare. Considering this assessment scheme, the results obtained are reported in Table 3.

The results obtained with a significance value of $p < 0.05$ show that the advance-position character coding has significant differences with respect to the other representations and therefore it will be used in all further experiments.

Table 4 Results obtained for each composition method when models are tested on the 1 954 selected RISM incipits. *Normal* stands for *normal distribution* method, *Random* stands for *random walk* method, *Logistic* stands for *sonification of the logistic equation* method, and *Mix* stands for *mix* method (it refers to the equal use of the three previous methods in the data set). Best results are highlighted in bold type

	Normal	Random	Logistic	Mix
SER (%)	21.1	**16.8**	39.8	17.1
NonSep-SER (%)	32.7	**25.2**	67.2	26.0
Sim-SER (%)	10.3	**09.3**	14.6	09.2
NonSim-SER (%)	22.4	**15.9**	52.6	16.8

Table 5 Statistical significance analysis of the different presented composition methods considering the Wilcoxon signed-rank test with a significance value of $p < 0.05$ for the symbol error rate metric. Symbols $<$, $>$, and $=$ represent that the error of the method in the row is significantly lower than, greater than, or no different to that in the column, respectively

	Normal	Random	Logistic	Mix
Normal	–	$>$	$<$	$>$
Random	$<$	–	$<$	$=$
Logistic	$>$	$>$	–	$>$
Mix	$<$	$=$	$<$	–

4.2 Best algorithmic composition method experiment

This experiment is addressed to identify which algorithm(s) for the synthetic generation of data works better for real data. For this purpose, we also generate a corpus of 1 500 labeled scores for the three remainder composition methods, given that the one for the mix algorithm is already generated. The results obtained in terms of the different considered metrics are presented in Table 4. Note that the figures provided represent the average values for the test partition, in which the validation data achieve its best performance for each of the considered cases.

The results reveal the following conclusions: (1) The sonification of the logistic equation method by itself does not work well since about 40% of the symbols predicted are wrong; (2) the normal distribution method halves the SER compared to the previous method; and (3) the random walk method and mixing data from all methods (mix method) further enhance that improvement, with both methods depicting similar values. Such a trend is visible in all the figures of merit considered. To support the relevance of those statements, we also assess the results in terms of statistical significance following the nonparametric Wilcoxon signed-rank test introduced in Sect. 4.1. The analysis is reported in Table 5.

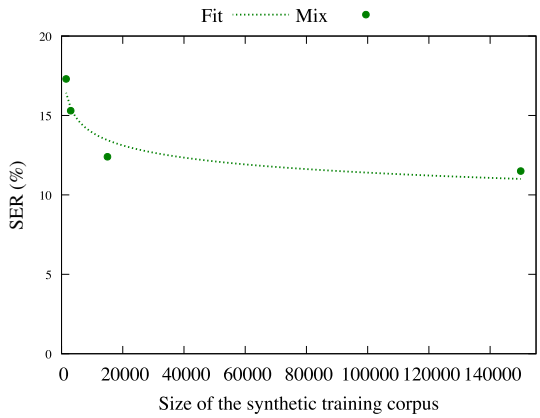


Fig. 14 Average results obtained on the RISM data for different sizes of the synthetic training corpus. The synthetic procedure follows the mix method since it has achieved, on average, lower error rates (see Table 6). Nevertheless, the conclusions can be extrapolated to the random walk method. Nonlinear least squares are used to fit the data

The results obtained with a significance value of $p < 0.05$ show that random and mix methods significantly outperform the other composition strategies while showing no significant differences between them. This implies that the two composition algorithms are most suitable for generating synthetic training data.

We would like to reduce the error figures on the selected RISM corpus by exploiting the fact that we have an "infinite" generator; that is, thanks to the automatic generation system we will always be able to generate new data that the neural network has not probably seen before. To gain insights into this issue first, we compute the greatest achievable performance with real training data—the lower bound that we want to surpass. For that, we derive three non-overlapping partitions—train, validation, and test—corresponding to 60%, 20%, and 20% of the 1 954 selected RISM incipits, respectively, following a fivefold cross-validation scheme. We train a model using those partitions. We compare it with those trained with 1 500, 3 000, 15 000, and 150 000 samples generated using the random walk method and the mix method, respectively, when evaluated over the same RISM test partition. Table 6 reports the results obtained.

The results reveal an exponential decay in the various figures of merit considered: while going from 1 500 to 15 000 improves SER by 5 points, multiplying the size of the data by 10 for the second time achieves an improvement of less than 1 point. It suggests that in the second zone, we reach the plateau of the curve (see Fig. 14). In other words, there exists a *glass ceiling* when recognizing real scores with synthetic-trained models. The nonparametric Wilcoxon signed-rank test introduced in Sect. 4.1 reinforces the finding by stating that the error rates are not significantly different.

The lower error bound found, of around 12%, might be due to the underlying (musical) language model of the composi-

Table 6 Results obtained for each composition method when models are trained using different set sizes and tested on the same RISM samples as the real-only model. Best results are highlighted in bold type

	Size of the synthetic training corpus								RISM
	1 500		3 000		15 000		150 000		
	Random	Mix	Random	Mix	Random	Mix	Random	Mix	
SER (%)	16.9	17.3	15.3	15.3	12.7	12.4	12.1	11.5	**3.3**
NonSep-SER (%)	25.3	26.4	22.9	23.3	18.9	18.3	17.7	16.8	**4.8**
Sim-SER (%)	09.7	09.7	08.7	08.9	07.6	07.3	07.4	07.2	**2.5**
NonSim-SER (%)	15.6	16.7	14.2	14.4	11.3	11.0	10.3	09.6	**2.3**

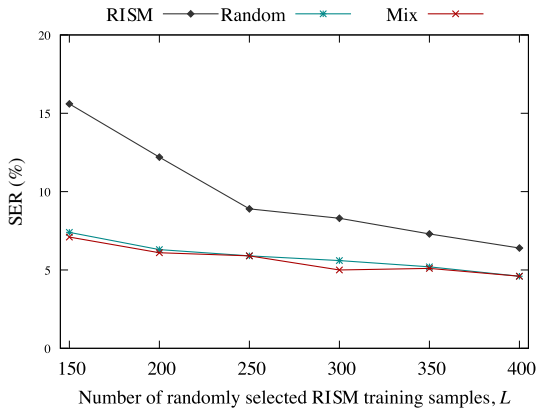


Fig. 15 Average symbol error rate (%) attained for each scenario with respect to the number of randomly selected RISM training staves, L

tion algorithms. The generated synthetic corpora are created in a way that contains a similar number of measures per staff, symbols per staff, and vertical distributions per staff—all of them on average—as the selected RISM corpus as well as the same graphical appearance. However, it might not be enough as the synthetic data distribution fails to capture all the characteristics of the real data. Table 6 shows that training with real homophonic scores yields an error rate of 3.3%. This is less than one-third of the errors made by the best model trained on synthetic data only.

We would like to stress again how simultaneities do not represent a problem. When the training data correctly capture the distribution of the test data, the error rate in the simultaneities and in the monophonic segments is roughly the same (see RISM column in Table 6). At the same time, this reinforces our idea about the glass ceiling. For the rest of the models, the error rate is bigger in monophonic regions since the synthetic distribution is not properly modeling that of RISM data. The underlying problem is one of out-of-distribution learning, for which no satisfactory solution is known at this time, at least for the CRNN-CTC framework.

To validate our intuition about the cause of the glass ceiling, we start from the premise that one possible solution would be to add scores from the test distribution to the training set. For that, we derive three non-overlapping partitions—train, validation, and test—corresponding to L, L, and $1\,954-2L$ of the 1 954 selected RISM incipits, respectively, where L is the number of randomly selected samples. We add the train and validation sets to the corresponding synthetic train and validation partitions of the random and mix corpora of size 150 00, respectively, and use the test set to evaluate such synthetic-with-real models. We compare those models with a model trained only with the aforementioned RISM partitions. We want to see (i) whether the glass ceiling can be broken by using real samples and (ii) how many of them are needed if (i) proves to be the case. It must be noted that a fivefold cross-validation scheme is followed for this experiment to ensure that the results are not conditioned to the randomly selected samples. The results obtained in terms of the SER metric for the contemplated scenarios are presented in Fig. 15 and Table 7.

First, it is necessary to state that the glass ceiling caused by the synthesis process can be broken by incorporating real scores into the synthetic train partition. This outcome supports the initial premise that the lower error bound attained with synthetic-only models was due to the synthesis process. We only need 50 real scores to decrease the 17% error rate to 10% and 100 to halve it—when combined with synthetic data. If used on their own, the real-only model is not able to solve the transcription task. Such a model starts to do so after 150 samples, and even so, using either only synthetic data or combining it with real data still yields better results.

Table 7 Results obtained in terms of the symbol error rate for each scenario with respect to the number of randomly selected RISM training staves, L. Best results are highlighted in bold type

	Number of randomly selected RISM training staves, L								
	0	50	100	150	200	250	300	350	400
RISM	–	90.6	74.1	15.6	12.2	8.9	8.3	7.3	6.4
Random	**16.8**	10.6	08.4	07.4	06.3	**5.9**	5.6	5.2	**4.6**
Mix	17.1	**10.2**	**07.9**	**07.1**	**06.1**	**5.9**	**5.0**	**5.1**	**4.6**

This implies that manually labeling some samples is compensated as combining synthetic and real data bring out synergies that help reduce the baseline error rate set by the only-real model. Moreover, as analyzed in [3], the posterior correction of errors of the synthetic-and-real models is more than offset by the time saved when compared with the real model, as the latter needs more manually transcribed training samples.

Regarding the synthesis method, it can be seen in both Fig. 15 and Table 7 that even though their performance is quite similar for all cases, the mix method tends to achieve slightly lower error rates.

5 Conclusions

In this work, we have studied the suitability of the state-of-the-art end-to-end neural approach to recognize real homophonic music scores by presenting and analyzing four different encodings for the OMR output. Throughout the research, we have trained the neural network with synthetic scores, created by the system of automatic generation of labeled data. This makes the use of an infinite music data generator helpful in dramatically reducing the costs of acquiring scores for training OMR systems.

As reported in the first part of the experiments, our serialized ways of encoding the music content prove to be appropriate for our DL-based OMR, as the learning process is successful, and low SER figures are eventually attained. In addition, it is shown that the choice of the encoding has some impact on the lower bound of the error rates that can be achieved: the advance-character position code is the one that most benefits the learning process in the recognition of vertical structures found in a homophonic music environment. These facts reinforce our initial claim that the encoding of the output for OMR deserves further consideration within the end-to-end DL paradigm.

It has also been possible to demonstrate that the algorithmic composition method used in the creation of synthetic music has a strong influence on the recognition results, being the random walk method and the mix method the most suitable algorithmic composition techniques. However, although the learning process was successful, there exists a *glass ceiling* when recognizing real scores: the Sym-ER never decreased below 11%, regardless of largely increasing the size of the training set. To break the *glass ceiling*, the use of real sheet music is necessary. It indicates that there is a part of the learning process that is not related to the graphical aspects of the scores but to the underlying (musical) language model. We believe this opens up new avenues for research. For example, modeling more intelligent systems of automatic generation of labeled data. It might be convenient to first learn some characteristics of the language model of the music that will be recognized at a later time in order to generate music scores that follow such a particular style. Then, the *glass ceiling* could be broken and the advantage of having an infinite data generator could be exploited to the fullest.

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature. This paper is part of the I+D+i PID 2020-118447RA-I00 (MultiScore) project, funded by MCIN/AEI/10.13039/501100011033. The first author is supported by grant FPU19/04957 from the Spanish Ministerio de Universidades.

Data availability The datasets generated during and/or analyzed during the current study are available from the corresponding author on reasonable request.

Declarations

Conflict of interests The authors declare that they have no conflicts of interest.

References

1. Alfaro Contreras M (2018) Construcción de un corpus de referencia para investigación en reconocimiento automático de partituras musicales. Technical report, Universidad de Alicante. (In Spanish)
2. Alfaro-Contreras M, Calvo-Zaragoza J, Iñesta JM (2019) Approaching end-to-end optical music recognition for homophonic scores. In: Iberian conference on pattern recognition and image analysis, pp 147–158. Springer
3. Alfaro-Contreras M, Rizo D, Iñesta JM, Calvo-Zaragoza J (2021) OMR-assisted transcription: a case study with early prints. In: Proceedings of the 22nd international society for music information retrieval conference, pp 35–41, Online. ISMIR
4. Bainbridge D, Bell T (2001) The challenge of optical music recognition. Comput Humanit 35(2):95–121
5. Baró A, Badal C, Fornés A (2020) Handwritten historical music recognition by sequence-to-sequence with attention mechanism. In: 17th International conference on frontiers in handwriting recognition, ICFHR 2020, Dortmund, Germany, 2020, pp 205–210
6. Baró A, Riba P, Fornés A (2018) A starting point for handwritten music recognition. In: 1st International workshop on reading music systems. France, Paris, pp 5–6
7. Burgoyne JA, Pugin L, Eustace G, Fujinaga I (2007) A comparative survey of image binarisation algorithms for optical recognition on degraded musical sources. In: Proceedings of the 8th international conference on music information retrieval, ISMIR 2007, Vienna, Austria, 2007, pp 509–512

8. Byrd D, Simonsen JG (2015) Towards a standard testbed for optical music recognition: Definitions, metrics, and page images. J New Music Res 44(3):169–195
9. Calvo-Zaragoza J, Jr JH, Pacha A (2020) Understanding optical music recognition. ACM Comput Surv, 53(4): 1–77
10. Calvo-Zaragoza J, Micó L, Oncina J (2016) Music staff removal with supervised pixel classification. Int J Doc Anal Recognit 19(3):211–219
11. Calvo-Zaragoza J, Rizo D (2018) Camera-PrIMuS: neural end-to-end optical music recognition on realistic monophonic scores. In: Proceedings of the 19th international society for music information retrieval conference, ISMIR 2018, Paris, France, 2018, pp 248–255
12. Calvo-Zaragoza J, Rizo D (2018) Camera-PrIMuS: neural end-to-end optical music recognition on realistic monophonic scores. In: Proceedings of the 19th international society for music information retrieval conference, pp 248–255, Paris, France
13. Calvo-Zaragoza J, Toselli AH, Vidal E (2019) Handwritten music recognition for mensural notation with convolutional recurrent neural networks. Pattern Recognit Lett 128:115–121
14. Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7:1–30
15. Dutta A, Pal U, Fornés A, Lladós J (2010) An efficient staff removal approach from printed musical documents. In: 20th International conference on pattern recognition, ICPR 2010, Istanbul, Turkey, 2010, pp 1965–1968
16. Fornés A, Sánchez G (2014) Analysis and recognition of music scores. In: Handbook of document image processing and recognition, pp 749–774
17. Gallego A-J, Calvo-Zaragoza J (2017) Staff-line removal with selectional auto-encoders. Expert Syst Appl 89:138–148
18. Good M et al (2001) MusicXML: an internet-friendly format for sheet music. In: XML conference and expo, pp 03–04
19. Graves A (2008) Supervised sequence labelling with recurrent neural networks. PhD thesis, Technical University Munich
20. Graves A, Fernández S, Gomez F, Schmidhuber J (2006) Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In: Proceedings of the 23rd International Conference on Machine Learning, ICML '06, pp 369–376, New York, NY, USA. ACM
21. Graves A, Schmidhuber J (2009) Offline handwriting recognition with multidimensional recurrent neural networks. In: Advances in neural information processing systems, pp 545–552
22. Hankinson A, Roland P, Fujinaga I (2011) The music encoding initiative as a document-encoding framework. In: Proceedings of the 12th international society for music information retrieval conference, pp 293–298
23. Kingma DP, Ba J (2015) Adam: a method for stochastic optimization. In: Bengio, Y, LeCun, Y., (eds) In: 3rd International conference on learning representations, ICLR 2015, San Diego, CA, USA, 7-9, 2015, Conference Track Proceedings
24. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436–444
25. Mengarelli L, Kostiuk B, Vitorio JG, Tibola MA, Wolff W, Silla CN (2020) OMR metrics and evaluation: a systematic review. Multimed Tools Appl 79(9):6383–6408
26. Miranda E (2001) Composing music with computers. Focal Press, New York
27. Pacha A, Calvo-Zaragoza J, Jr JH (2019) Learning notation graph construction for full-pipeline optical music recognition. In: Flexer A, Peeters G, Urbano J, Volk A, (eds) In: Proceedings of the 20th international society for music information retrieval conference, ISMIR 2019, Delft, The Netherlands, 2019, pp 75–82
28. Pacha A, Eidenberger H (2017) Towards a universal music symbol classifier. In: 2017 14th IAPR International conference on document analysis and recognition (ICDAR), 2, pp 35–36. IEEE
29. Pedersoli F, Tzanetakis G (2016) Document segmentation and classification into musical scores and text. Int J Doc Anal Recognit 19(4):289–304
30. Raphael C, Wang J (2011) New approaches to optical music recognition. In: Klapuri A, Leider C, editors, In: Proceedings of the 12th international society for music information retrieval conference, ISMIR 2011, Miami, Florida, USA, October 24-28, 2011, pp 305–310
31. Rebelo A, Capela G, Cardoso JdS (2010) Optical recognition of music symbols. Int J Doc Anal Recognit 13(1):19–31
32. Rebelo A, Fujinaga I, Paszkiewicz F, Marçal A, Guedes C, Cardoso J (2012) Optical music recognition: state-of-the-art and open issues. Int J Multimed Inf Retr 1:173–190
33. Shi B, Bai X, Yao C (2017) An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. IEEE Trans Pattern Anal Mach Intell 39(11):2298–2304
34. Williams RJ, Zipser D (1995) Gradient-based learning algorithms for recurrent networks and their computational complexity. In: Chauvin Y, Rumelhart DE, (eds.) Back-propagation: Theory, architectures and applications, 13: 433–486