*applied sciences*

Article

# End-to-End Neural Optical Music Recognition of Monophonic Scores

**Jorge Calvo-Zaragoza [1,2]* and David Rizo [3,4]**

[1] Schulich School of Music, McGill University, Montreal, QC H3A 1E3, Canada
[2] PRHLT Research Center, Universitat Politècnica de València, 46022 Valencia, Spain
[3] Instituto Superior de Enseñanzas Artísticas, 03690 Alicante, Spain; drizo@dlsi.ua.es
[4] Departamento de Lenguajes y Sistemas Informáticos, Universidad de Alicante, 03690 Alicante, Spain
* Correspondence: jcalvo@upv.es; Tel.: +34-965-909-900

Received: 28 February 2018; Accepted: 8 April 2018; Published: 11 April 2018

**Abstract:** Optical Music Recognition is a field of research that investigates how to computationally decode music notation from images. Despite the efforts made so far, there are hardly any complete solutions to the problem. In this work, we study the use of neural networks that work in an end-to-end manner. This is achieved by using a neural model that combines the capabilities of convolutional neural networks, which work on the input image, and recurrent neural networks, which deal with the sequential nature of the problem. Thanks to the use of the the so-called Connectionist Temporal Classification loss function, these models can be directly trained from input images accompanied by their corresponding transcripts into music symbol sequences. We also present the Printed Images of Music Staves (PrIMuS) dataset, containing more than 80,000 monodic single-staff real scores in common western notation, that is used to train and evaluate the neural approach. In our experiments, it is demonstrated that this formulation can be carried out successfully. Additionally, we study several considerations about the codification of the output musical sequences, the convergence and scalability of the neural models, as well as the ability of this approach to locate symbols in the input score.

**Keywords:** Optical Music Recognition; end-to-end recognition; Deep Learning; music score images

## 1. Introduction

During the past few years, the availability of huge collections of digital scores has facilitated both the music professional practice and the amateur access to printed sources that were difficult to obtain in the past. Some examples of these collections are the IMSLP (http://imslp.org) website with currently 425,000 classical music scores, or many different sites offering Real Book jazz lead sheets. Furthermore, many efforts are being done by private and public libraries to publish their collections online (see http://drm.ccarh.org/). However, in addition to this instant availability, the advantages of having the digitized image of a work over its printed material are restricted to the ease to copy and distribute, and the lack of wear that digital media intrinsically offers over any physical resource. The great possibilities that current music-based applications can offer are restricted to scores symbolically encoded. Notation software such as Finale (https://www.finalemusic.com), Sibelius (https://www.avid.com/en/sibelius), MuseScore (https://musescore.org), or Dorico (https://www.steinberg.net/en/products/dorico), computer-assisted composition applications such as OpenMusic (http://repmus.ircam.fr/openmusic/), digital musicology systems such as Music21 (http://web.mit.edu/music21/), or Humdrum (http://www.humdrum.org/), or content-based search tools [1], cannot deal with pixels contained in digitized images but with computationally-encoded symbols such as notes, bar-lines or key signatures.

Furthermore, the scientific musicological domain would dramatically benefit from the availability of digitally encoded music in symbolic formats such as MEI [2] or MusicXML [3]. Just to name an example, many of the systems presented in the Computational Music Analysis book edited by Meredith [4] cannot be scaled to real-world scenarios due to the lack of big enough symbolic music datasets.

Many different initiatives have been proposed to manually fill this gap between digitized music images and digitally encoded music content such as OpenScore (https://openscore.cc), KernScores (http://kern.ccarh.org), or RISM [5]—encoding just small excerpts (incipits). However, the manual transcription of music scores does not represent a scalable process, given that its cost is prohibitive both in time and resources. Therefore, to face this scenario with guarantees, it is necessary to resort to assisted or automatic transcription systems. The so-called Optical Music Recognition (OMR) is defined as the research about teaching computers how to read musical notation [6], with the ultimate goal of exporting their content to a desired format.

Despite the great advantages of its development, OMR is far from being totally reliable as a black box, as current optical character recognition [7] or speech recognition [8] technologies do. Commercial software is constantly being improved by fixing specific problems from version to version. In the scientific community, there are hardly any complete approach for its solution [9,10]. Traditionally, this has been motivated because of the small sub-tasks in which the workflow can be divided. Simpler tasks such as staff-line removal, symbol localization and classification, or music notation assembly, have so far represented major obstacles [11]. Nonetheless, recent advances in machine learning, and specifically in Deep Learning (DL) [12], not only allow solving these tasks with some ease, but also to propose new schemes with which to face the whole process in a more elegant and compact way, avoiding heuristics that make systems limited to the kind of input they are designed for. In fact, this new sort of approaches has broken most of the glass-ceiling problems in text and speech recognition systems [13,14].

This work attempts to be a seed work that studies the suitability of applying DL systems to solve the OMR task holistically, i.e., in an end-to-end manner, without the need of dividing the problem into smaller stages. For this aim, two contributions are introduced. First, a thorough analysis of a DL model for OMR, and the design and construction of a big enough quality dataset on which training and evaluating the system. Note that the most difficult obstacle that researchers usually find when trying to apply DL algorithms is the lack of appropriate ground-truth data, which leads to a deadlock situation; that is, learning systems need big amounts of labeled data to train, and the fastest way of getting such amounts of labeled data is the use of trained systems. We therefore aim at unblocking such scenario in our proposal.

Considering this as a starting point, we restrict ourselves in this work to the consideration of monodic short scores taken from real music works in Common Western Modern Notation (CWMN). This allows to encode the expected output as a sequence of symbols that the model must predict. Then, one can use the so-called Connectionist Temporal Classification (CTC) loss function [15], with which the neural network can be trained in an end-to-end fashion. It means that it is not necessary to provide information about the composition or location of the symbols in the image, but only pairs of input scores and their corresponding transcripts into music symbol sequences.

As mentioned previously, a typical drawback when developing this research is the lack of data. Therefore, to facilitate the development of our work, we also propose an appropriate dataset to train and evaluate the neural model. Its construction is adapted for the task of studying DL techniques for monodic OMR, so two considerations must be taken into account. On the one hand, the output formats devised here do not aim at substituting any of the traditional music encodings [16]. On the other hand, although some previous efforts have been done to build datasets for this purpose [17,18], none of them fits the requirements of size and nature required for our study.

It must be kept in mind that our approach has been preliminarily evaluated on synthetic music scores [19], and so we want to further study here its potential. More precisely, the contributions of this work are listed as follows:

- Consideration of different formulations with respect to the output representation. We will see that the way of representing the symbol sequence is not trivial, and that it influences the performance that the neural model is able to reach.
- A comprehensive dataset of images of monodic scores that are generated from real music scores.
- Thorough evaluation of the end-to-end neural approach for OMR, which includes transversal issues such as convergence, scalability, and the ability to locate symbols.

According to our experimental results, this approach proves to successfully solve the end-to-end task. Although it is true that we only deal with the case of relatively simple scores (printed and monodic), we believe that this work can be considered as a starting point to develop neural models that work in a holistic way on images of musical scores, which would be a breakthrough towards the development of generalizable and scalable OMR systems for all kind of printed and handwritten music scores.

The rest of the paper is structured as follows: we overview the background in Section 2; the dataset to be used is presented in Section 3; the neural approach is described in Section 4; the experiments that validate our proposal are reported in Section 5; finally, the conclusions are drawn in Section 6.

## 2. Background

We study in this work a holistic approach to the task of retrieving the music symbols that appear in score images. Traditionally, however, solutions to OMR have focused on a multi-stage approach [11].

First, an initial processing of the image is required. This involves various steps of document analysis, not always strictly related to the musical domain. Typical examples of this stage comprise the binarization of the image [20], the detection of the staves [21], the delimitation of the staves in terms of bars [22], or the separation between lyrics and music [23].

Special mention should be made to the staff-line removal stage. Although staff lines represent a very important element in music notation, their presence hinders the isolation of musical symbols by means of connected-component analysis. Therefore, much effort has been made to successfully solve this stage [24–26]. Recently, results have reached values closer to the optimum over standard benchmarks by using DL [27,28].

In the next stage, we find the classification of the symbols, for which a number of works can be found in the literature. For instance, Rebelo et al. [29] compared the performance of different classifiers such as *k*-Nearest Neighbors or Support Vector Machines for isolated music symbol classification. Calvo-Zaragoza et al. [30] proposed a novel feature extraction for the classification of handwritten music symbols. Pinheiro Pereira et al. [31] and Lee et al. [32] considered the use of DL for classifying handwritten music symbols. There results were further improved with the combination of DL and conventional classifiers [33]. Pacha and Eidenberger [34] considered an universal music symbol classifier, which was able to classify isolated symbols regardless of their specific music notation.

In addition, the last stage in which independently detected and classified components must be assembled to obtain real musical notation. After using a set of the aforementioned stages (binarization, staff-line removal, and symbol classification), Couasnon [35] considered a grammar to interpret the isolated components and give them musical sense. Following a similar scheme in terms of formulation, Szwoch [36] proposed the Guido system using a new context-free grammar. Rossant and Bloch [37], on the other hand, considered a rule-based systems combined with fuzzy modeling. A novel approach is proposed by Raphael and Wang [38], in which the recognition of composite symbols is done with a top-down modeling, while atomic objects are recognized by template matching. Unfortunately, in the cases discussed above, an exhaustive evaluation with respect to the complete OMR task is not shown, but rather partial results (typically concerning the recognition of musical symbols). Furthermore, all these works are based on heuristic strategies that hardly generalize out of the set of scores used for their evaluation. Moreover, a prominent example of full OMR is Audiveris [39], an open-source tool that performs the process through a comprehensive pipeline in which different types of symbols are processed independently. Unfortunately, no detailed evaluation is reported.

Full approaches are more common when the notation is less complex than usual, like scores written in mensural notation. Pugin [40] made use of hidden Markov models (HMM) to perform a holistic approximation to the recognition of printed mensural notation. Tardón et al. [41] proposed a full OMR system for this notation as well, but they followed a multi-stage approach with the typical processes discussed above. An extension to this work showed that the staff-line removal stage can be avoided for this type of notation [42]. Recently, Calvo-Zaragoza et al. [43] also considered HMM along with statistical language models for the transcription of handwritten mensural notation. Nevertheless, although these works also belong to the OMR field, their objective entails a very different challenge with respect to that of CWMN.

For the sake of clarification, Table 1 summarizes our review of previous work. Our criticism to this state of the art is that all these previous approaches on OMR either focus on specific stages of the process or consider a hand-crafted multi-stage workflow that only adapt to the experiments for which they have been developed. The scenario is different when working on a notational type different from CWMN, which could be considered as a different problem.

**Table 1.** Representative summary of previous works in OMR research.

| References | Task |
|---|---|
| [20–23] | Pre-processing of music score images |
| [24–28] | Staff-line removal |
| [29–34] | Symbol classification |
| [35–39] | Detection, classification, and interpretation |
| [40–43] | OMR in mensural notation |

We believe that the problem to progress in OMR for CWMN lies in the complexity involved in correctly modeling the composition of musical symbols. Unlike these hand-engineered multi-stage approaches, we propose a holistic strategy in which the musical notation is learned as a whole using machine learning strategies. However, to reduce the complexity to a feasible level, we do consider a first initial stage in which the image is pre-processed to find and separate the different staves of the score. Staves are good basic units to work on, analogously to similar text recognition where a single line of text is assumed as input unit. Note that this is not a strong assumption as there are successful algorithms for isolating staves, as mentioned above.

Then, the staff can be addressed as a single unit instead of considering it as a sequence of isolated elements that have to be detected and recognized independently. This also opens the possibility to boost the optical recognition by taking into account the musical context which, in spite of being extremely difficult to model entirely, can certainly help in the process. Thus, it seems interesting to tackle the OMR task over single staves in an holistic fashion, in which the expected output is directly the sequence of musical symbols present in the image.

We strongly believe that deep neural networks represent suitable models for this task. The idea is also encouraged by the good results obtained in related fields such as handwritten text recognition [7] or speech recognition [8], among others. Our work, therefore, aims at setting the basis towards the development of neural models that directly deal with a greater part of the OMR workflow in a single step. In this case, we restrict ourselves to the scenario in which the expected scores are monodic, which allows us to formulate the problem in terms of image-to-text models.

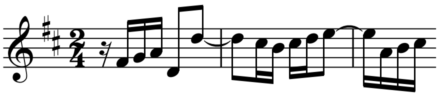
## 3. The PrIMuS Dataset

It is well known that machine learning-based systems require training sets of the highest quality and size. The "Printed Images of Music Staves" (PrIMuS) dataset has been devised to fulfill both requirements (the dataset is freely available at http://grfia.dlsi.ua.es/primus/). Thus, the objective pursued when creating this ground-truth data is not to represent the most complex musical notation corpus, but collect the highest possible number of scores ready to be represented in formats suitable for heterogeneous OMR experimentation and evaluation.

PrIMuS contains 87 678 real-music incipits (an incipit is a sequence of notes, typically the first ones, used for identifying a melody or musical work), each one represented by five files (see Figure 1): the Plaine and Easie code source [44], an image with the rendered score, the musical symbolic representation of the incipit both in Music Encoding Initiative format (MEI) [2] 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 on-purpose agnostic and semantic representations, that will be described below, are the ones used in our experiments.

```
%G-2@2/4$xFCü 6-{'FGA}8{D''D+}/{D6C'B}{''CD8E+}/{6E'AB''C}
```

a



b

```
<mdiv>
    <score>
        <score key.sig="2s" meter.count="2" meter.unit="4">
            <staffGrp>
                <staffDef clef.shape="G" clef.line="2" n="1" lines="5" />
            </staffGrp>
        </score>
        <section>
            <measure>
                <staff n="1">
                    <layer n="1">
                        <rest dur="16" />
                        <beam>
                            <note dur="16" oct="4" pname="f" />
                            <note dur="16" oct="4" pname="g" />
                            <note dur="16" oct="4" pname="a" />
                        </beam>
                        <beam>
                            <note dur="8" oct="4" pname="d" />
                            <note dur="8" oct="5" pname="d" tie="i" />
                        </beam>
```

c

```
clef-G2, keySignature-DM, timeSignature-2/4, rest-sixteenth, note-F#4_sixteenth,
note-G4_sixteenth, note-A4_sixteenth, note-D4_eighth, note-D5_eighth, tie, barline,
note-D5_eighth, note-C#5_sixteenth, note-B4_sixteenth, note-C#5_sixteenth,
note-D5_sixteenth, note-E5_eighth, tie, barline, note-E5_sixteenth,
note-A4_sixteenth, note-B4_sixteenth, note-C#5_sixteenth
```

d

```
clef.G-L2, accidental.sharp-L5, accidental.sharp-S3, digit.2-L4, digit.4-L2, rest.sixteenth-L3,
note.beamedRight2-S1, note.beamedBoth2-L2, note.beamedLeft2-S2, note.beamedRight1-S0,
note.beamedLeft1-L4, slur.start-L4, barline-L1, slur.end-L4, note.beamedRight1-L4,
note.beamedBoth2-S3, note.beamedLeft2-L3, note.beamedRight2-S3, note.beamedBoth2-L4,
note.beamedLeft1-S4, slur.start-S4, barline-L1, slur.end-S4, note.beamedRight2-S4,
note.beamedBoth2-S2, note.beamedBoth2-L3, note.beamedLeft2-S3
```

e

**Figure 1.** PrIMuS incipit contents example. Incipit RISM ID no. 000051759. *Inventions*. Heinrich Nikolaus Gerber. (**a**) Plaine and Easie Code source; (**b**) Verovio rendering; (**c**) Excerpt of MEI encoding; (**d**) Semantic encoding; (**e**) Agnostic encoding.

Currently, the biggest database of musical incipits available is RISM [5]. Created in 1952, the main aim of this organization is to catalog the location of musical sources. In order to identify musical works, they make use of the incipits of the contained pieces—as well as meta-data. To the date this article was written, the online version of RISM indexes more than 850,000 references, most of them monodic scores in CWMN. This content is freely available as an "Online public access catalog" (OPAC) (https://opac.rism.info). Due to the early origins of this repertoire, the musical encoding format used is Plaine and Easie Code (PAEC) [44].

PrIMuS has been generated using as source an export from the RISM database. Given as input the PAEC encoding of those incipits (Figure 1a), it is formatted in order to feed the musical engraver Verovio [45] that outputs both the musical score (Figure 1b) in SVG format—that is posteriorly converted into PNG format—and the MEI encoding containing the symbolic semantic representation of the score in XML format (Figure 1c). Verovio is able to render scores using three different fonts, namely: Leipzig, Bravura, and Gootville. This capability is used to randomly choose one of the three fonts used in the rendering of the different incipits, leading to a higher variability in the dataset. Eventually, the on-purpose semantic and agnostic representations have been obtained as a conversion from the MEI files.

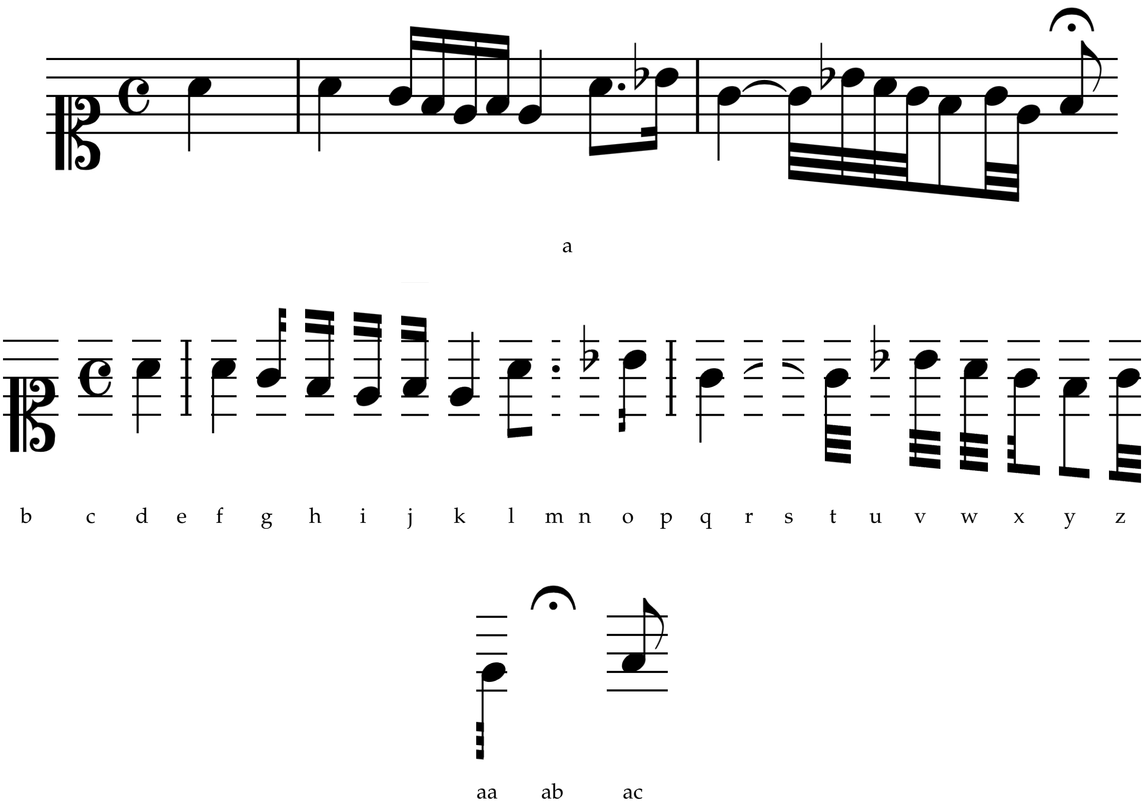
### *Semantic and Agnostic Representations*

As introduced above, two representations have been devised on-purpose for this study, namely the semantic and the agnostic ones. The former contains symbols with musical meaning, e.g., a D Major key signature; the latter consists of musical symbols without musical meaning that should be eventually interpreted in a final parsing stage. In the agnostic representation, a D Major key signature is represented as a sequence of two "sharp" symbols. Note that from a graphical point of view, a sharp symbol in a key signature is the same as a sharp accidental altering the pitch of a note. This way, the alphabet used for the agnostic representation is much smaller, which allows us to study the impact of the alphabet size and the number of examples shown to the network for its training. Both representations are used to encode single staves as one-dimensional sequences in order to make feasible their use by the neural network models. For avoiding later assumptions on the behavior of the network, every item in the sequence is self-contained, i.e., no contextual information is required to interpret it. 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* is a simple format containing the sequence of symbols in the score with their musical meaning (see Figure 1d). In spite of the myriad of monodic melody formats available in the literature [16], this on-purpose format has been introduced for making it easy to align it to the agnostic representation and grow-it in the future in the direction this research requires. As an example, the original Plaine and Easie code has not been directly used for avoiding its abbreviated writing that allows omitting part of the encoding by using previously encoded slices of the incipit. We want the neural network to receive a self-contained chunk of information for each musical element. Anyway, the original Plaine and Easie code and a full-fledged MEI file is maintained for each incipit that may be used to generate any other format. The grammar of the ground-truth files of the semantic representation is formalized in Appendix A (Tables A1 and A2).

The *agnostic representation* contains a list of graphical symbols in the score, each of them tagged given a catalog 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 inf Figure 1e). In order to represent beaming of notes, they have been vertically sliced generating non-musical pictograms (see Figures 2 and 3). 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. The grammar of the ground-truth files of the agnostic representation is formalized in Appendix A (Tables A3 and A4).

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



```
clef.C-L1, metersign.C-L3, note.quarter-S3, barline-L1, note.quarter-S3,
note.beamedRight2-L3, note.beamedBoth2-S2, note.beamedBoth2-L2, note.beamedLeft2-S2,
note.quarter-L2, note.beamedRight1-S3, dot-S3, accidental.flat-L4, note.beamedLeft2-L4,
barline-L1, note.quarter-L3, slur.start-L3, slur.end-L3, note.beamedRight3-L3,
accidental.flat-L4, note.beamedBoth3-L4, note.beamedBoth3-S3, note.beamedBoth3-L3,
note.beamedBoth1-S2, note.beamedBoth3-L3, note.beamedLeft3-L2
fermata.above-S6, note.eighth-S2
```

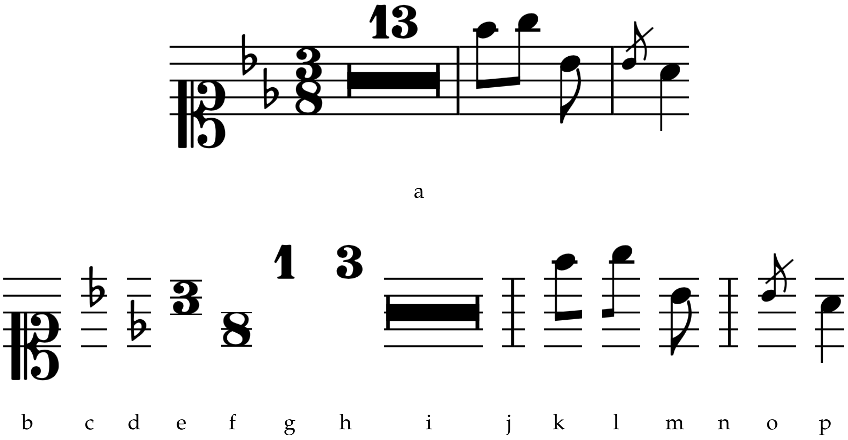
ad

**Figure 2.** Graphical symbol division example. Elements are ordered left-to-right, top-down. (a) Incipit. RISM ID no. 000051806. *Das alte Jahr vergangen ist*. Johann Sebastian Bach; (b)–(ac) Symbol division; (ad) Agnostic encoding.

Although both representations can be considered equivalent, each representation does need a different number of symbols to codify the same staff. This also affects the size of their specific vocabularies. To illustrate this issue, we show in Table 2 an overview of the composition of PrIMuS with respect to the considered representations.

**Table 2.** Composition of PrIMuS dataset in terms of number of samples (staves), size of the alphabet, and number of symbols with respect to the different representations.

| | **Agnostic** | **Semantic** |
|---|---|---|
| Number of staves | 87,678 | 87,678 |
| Alphabet size | 758 | 1781 |
| Music symbols | 2,397,824 | 2,095,836 |



```
clef.C-L1, accidental.flat-L4, accidental.flat-L2, digit.3-L4, digit.8-L2, digit.1-S5,
digit.3-S5, multirest-L3, barline-L1, note.beamedRight1-L6, note.beamedLeft1-S6,
note.eighth-L4, barline-L1, gracenote.eighth-L4, note.quarter-S3
```

q

**Figure 3.** Graphical symbol division example. (a) Beginning of incipit RISM ID no. 000108339. *Ormisda*. Giuseppe Maria Orlandini; (b)–(p) Symbol division; (q) Agnostic encoding.

## 4. Neural End-to-end Approach for Optical Music Recognition

We describe in this section the neural models that allow us to face the OMR task in an end-to-end manner. In this case, a monodic staff section is assumed to be the basic unit; that is, a single staff will be processed at each moment.

Formally, let $\mathcal{X} = \{(x_1, y_1), (x_2, y_2), ...\}$ 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. On the one hand, an image $x$ is considered to be a sequence of variable length, given by the number of columns. On the other hand, $y$ is a sequence of music symbols, each of which belongs to a fixed alphabet set $\Sigma$.

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

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

In this work, the statistical framework is formulated as regards Recurrent Neural Network (RNN), as they represent neural models that allows working with sequences [46]. Ultimately, therefore, the RNN will be responsible of producing the sequence of musical symbols that fulfills Equation (1). However, we first add a Convolutional Neural Network (CNN), that is in charge of learning how to process the input image [47]. In this way, the user is prevented from fixing a feature extraction process, given that the CNN is able to learn to extract adequate features for the task at issue.

Our work is conducted over a supervised learning scenario; that is, it is assumed that we can make use of a known subset of $\mathcal{X}$ with which to train the model. Since both types of networks represent feed-forward models, the training stage can be carried out jointly, which leads to a Convolutional

Recurrent Neural Network (CRNN). This can be implemented easily by connecting the output of the last layer of the CNN with the input of the first layer of the RNN, 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.

In principle, the traditional training mechanisms for a CRNN force to provide the expected output in each output frame. However, the restriction imposed above with respect to the end-to-end term refers to that, for each staff, the training set only provides its corresponding sequence of expected symbols, without any kind of explicit information about the location of the semantic or agnostic symbols in the image. This scenario can be nicely solved by means of the CTC loss function [15].

Basically, CTC provides a means to optimize the CRNN parameters so that it is likely to give the correct sequence $y$ given an input $x$. In other words, given the input $x$ and its corresponding transcript $y$, CTC directly optimizes $P(y|x)$. Although optimizing this likelihood exhaustively is computationally unfeasible, CTC performs a local optimization using an Expectation-Maximization algorithm similar to that used for training Hidden Markov Models [48].

The CTC loss function is used only for training. At the decoding stage, one has to take into account the output provided by the CRNN, which still predicts a symbol for each frame (column) of the convoluted image. However, the way in which the network is trained allows a straightforward decoding. To indicate a separation between symbols, or to handle those frames in which there is no symbol, CTC considers an additional symbol in the alphabet that indicates this situation (*blank* symbol).

Note that the model is not expected to provide information about the location of the symbols in the decoding stage because of the way it is trained. Anyway, from a musical perspective, it is not necessary to retrieve the exact position of each music symbol in the image but their context in order to correctly interpret it.

A graphical scheme of the framework is given in Figure 4. The details for its implementation is provided in the following sections.

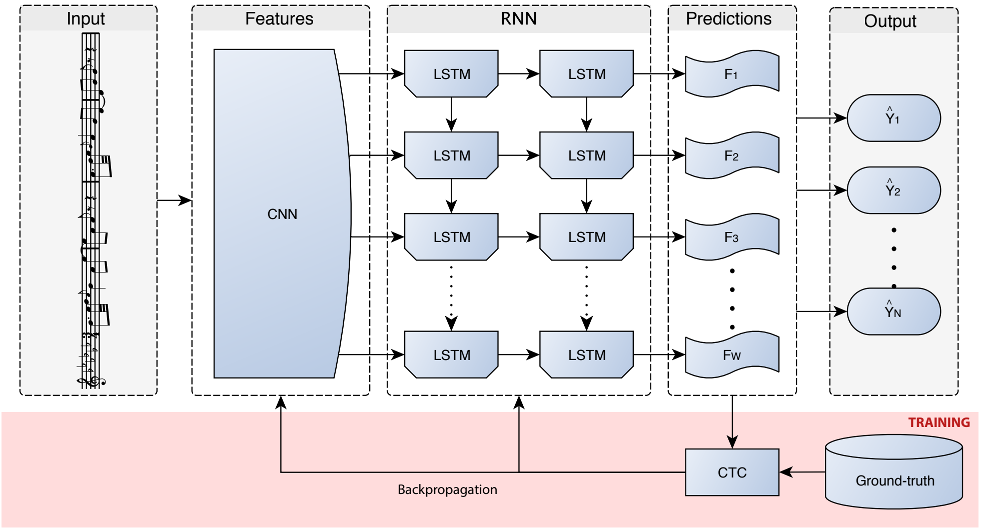


**Figure 4.** Graphical scheme of the end-to-end neural approach considered.

*Implementation Details*

The objective of this work is not to seek the best neural model for this task, but to study the feasibility of this framework. Thus, a single neural model is proposed that, by means of informal testing, has verified its goodness for the task.

The details concerning the configuration of the neural model are given in Table 3. As observed, input variable-width single-channel images (grayscale) 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 [49]: four convolutional layers with an incremental number of filters and kernel sizes of 3 × 3, followed by the max-pool 2 × 2 operator. In all cases, Batch Normalization [50] and Rectified Linear Unit activations [51] are considered.

At the output of this block, two recurrent bidirectional layers of 256 neurons, implemented as LSTM units [52], try to convert the resulting filtered image into a discrete sequence of musical symbols that takes into account both the input sequence and the modeling of the musical representation. Note that each frame performs a classification, modeled with a fully-connected layer with as many neurons as the size of the alphabet plus 1 (the *blank* symbol necessary for the CTC function). The activation of this neurons is given by a *softmax* function, which allows interpreting the output as a posterior probability over the alphabet of music symbols [53].

**Table 3.** Instantiation of the CRNN used in this work, consisting of 4 convolutional layers and 2 recurrent layers. Notation: Input$(h \times w \times c)$ means an input image of height $h$, width $w$ and $c$ channels; Conv$(n, h \times w)$ denotes a convolution operator of $n$ filters and kernel size of $h \times w$; 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.

| **Input (128 × *W* × 1)** |
|---|
| **Convolutional Block** |
| Conv$(32, 3 \times 3)$, MaxPooling$(2 \times 2)$ |
| Conv$(64, 3 \times 3)$, MaxPooling$(2 \times 2)$ |
| Conv$(128, 3 \times 3)$, MaxPooling$(2 \times 2)$ |
| Conv$(256, 3 \times 3)$, MaxPooling$(2 \times 2)$ |
| **Recurrent block** |
| BLSTM$(256)$ |
| BLSTM$(256)$ |
| Dense$(\|\Sigma\| + 1)$ |
| Softmax$()$ |

The learning process is carried out by means of stochastic gradient descent (SGD) [54], 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 Adadelta algorithm [55].

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 mechanism with the CTC loss function, the final decoding can be performed greedily: 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 and only only one symbol is concatenated to the final sequence. There are two ways to indicate a new symbol is predicted: either the predicted symbol is different from the previous frame 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 the input image, which in our case is highly unlikely to happen.

## 5. Experiments

### 5.1. Experimental Setup

Concerning evaluation metrics, there is an open debate on which metrics should be used in OMR [10]. This is especially arguable because of the different points of view that the use of its output has: it is not the same if the intention of the OMR is to reproduce the content or to archive it in order to build a digital library. Here we are only interested in the computational aspect itself, in which OMR is understood as a pattern recognition task. So, we shall consider metrics that, even assuming that they might not be optimal for the purpose of OMR, allow us to draw reasonable conclusions from the experimental results. Therefore, let us consider the following evaluation metrics:

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

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. Furthermore, 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).

Below we present the results achieved with respect to these metrics. In the first series of experiments we measure the performance that neural models can achieve as regards the representation used. First, they will be evaluated in an ideal scenario, in which a huge amount of data is available. Therefore, the idea is to measure the *glass ceiling* that each representation may reach. Next, the other issue to be analyzed is the complexity of the learning process as regards the convergence of the training process and the amount of data that is necessary to learn the task. Finally, we analyze the ability of the neural models to locate the musical symbols within the input staff, task for which it is not initially designed.

For the sake of reproducible research, source code and trained models are freely available [56].

### 5.2. Performance

We show in this section the results obtained when the networks are trained with all available data. This means that about 80,000 training samples are available, 10% of which are used for deciding when to stop training and prevent overfitting. The evaluation after a 10-fold cross validation scheme is reported in Table 4.

**Table 4.** Evaluation metrics with respect to the representation considered. Results reported represent averages from a 10-cross validation methodology.

| | **Representation** | |
|---|---|---|
| | **Agnostic** | **Semantic** |
| Sequence Error Rate (%) | 17.9 | 12.5 |
| Symbol Error Rate (%) | 1.0 | 0.8 |

Interestingly, the semantic representation leads to a higher performance than the agnostic representation. This is clearly observed in the sequence-level error (12.5% versus 17.9%), and somewhat to a lesser extent in the symbol-level error (0.8% versus 1.0%).

It is difficult to demonstrate why this might happen because of the way these neural models operate. However, it is intuitive to think that the difference lies in the ability to model the underlying musical language. At the image level, both representations are equivalent (and, in principle, the agnostic representation should have some advantage). On the contrary, the recurrent neural networks may find it easier to model the linguistic information of the musical notation from its semantic representation, which leads—when there is enough data, as in this experiment—to produce sequences that better fit the underlying language model.

In any case, regardless of the selected representation it is observed that the differences between the actual sequences and those predicted by the networks are minimal. While it cannot be guaranteed that the sequences are recognized with no error (only 12.5% at best), the results can be interpreted as that only around 1% of the symbols predicted need correction to get the correct transcriptions of the images. Therefore, the goodness of this complete approach is demonstrated, in which the task is formulated in an elegant way in terms of input and desired output.

Concerning computational cost we would like to emphasize that although the training of these models is expensive—in the order of several hours over high-performance Graphical Processing Units (GPUs)— the prediction stage allows fast processing. It takes around 1 second per score in a general-purpose computer like an Intel Core i5-2400 CPU at 3.10 GHz with 4 GB of RAM, and without speeding-up the computation with GPUs. We believe that this time is appropriate for allowing a friendly usability in an interactive application.

### 5.2.1. Error Analysis

In order to dig deeper into the previously presented results, we conducted an analysis of the typology of the errors produced. The most repeated errors for each representation are reported in Table 5.

**Table 5.** List of the 3 most common errors with respect to the representation considered. Percentages are relative to the total error rates from Table 4.

| **Rank** | **Representation** | | | |
|---|---|---|---|---|
| | **Agnostic** | | **Semantic** | |
| | **Symbol** | **Percentage** | **Symbol** | **Percentage** |
| # 1 | *barline-L1* | 45.5% | *barline* | 38.6% |
| # 2 | *gracenote.sixteenth-L4* | 1.8% | *tie* | 9.4% |
| # 3 | *accidental.natural-S3* | 1.4% | *gracenote.C5-sixteenth* | 1.5% |

In both cases, the most common error is the barline, with a notable difference with respect to the others. Although this may seem surprising at first, it has a simple explanation: the incipits often end without completing the bar. This, at the graphic level, hardly has visible consequences because the renderer almost always places the last barline at the end of the staff (most of the incipits contain complete measures). Thus, the responsibility of discerning whether there should be a barline or not lies almost exclusively in the capacity of the network to take into account "linguistic" information. The musical notation is a language that, in spite of being highly complex to model in its entirety, has certain regularities with which to exploit the performance of the system, as for instance the elements that lead to a complete measure. According to the results presented in the previous section, we can conclude that a semantic representation, in comparison with the agnostic one, makes it easier for the network to estimate such regularities. This phenomenon is quite intuitive, and may be the main cause of the differences between the representations' performance.

As an additional remark, note that both representations miss on grace notes, which clearly represent a greater complexity in the graphic aspect, and are worse estimated by the language model because of being less regular than conventional notes.

In the case of the semantic representation, another common mistake is the *tie*. Although we cannot demonstrate the reason behind these errors, it is interesting to note that the musical content generated without that symbol is still musically correct. Therefore, given the low number of *tie* symbols in the training set (less than 1%), the model may tend to push the recognition towards the most likely situation, in which the *tie* does not appear.

### 5.2.2. Comparison with Previous Approaches

As mentioned previously, the problem with existing scientific OMR approaches is that they either focus only on a sub-stage of the process (staff-line removal, symbol classification, etc.) or are heuristically developed to solve the task in a very specific set of scores. That is why we believe it would be quite unfair to compare these approaches with ours, the first one that covers a complete workflow exclusively using machine learning.

As an illustrative example of this matter, we include here a comparison of the performance of our approach with that of Audiveris (cf. Section 2) over PrIMuS, even assuming in advance that such comparison is not fair. As a representative of our approach, we consider the *semantic* representation, given that the output of Audiveris is also semantic (the semantic encoding format has been obtained from the Audiveris MusicXML batch-mode output).

Table 6 reports the accuracy with respect to the Symbol Error Rate attained, as a general performance metric, and the computational time measured as seconds per sample, from both our approach and Audiveris.

**Table 6.** Quantitative comparison with respect to accuracy (Symbol Error Rate in%) and processing time (avg. seconds per sample) between our CRNN-based approach and Audiveris. Values in bold highlight the best results in each metric.

| | **Symbol Error Rate (%)** | **Avg. s per Sample** |
|---|---|---|
| CRNN | **0.8** | **1** |
| Audiveris | 44.2 | 15 |

It can be observed that Audiveris, which surely works well in certain types of scores, is not able to offer a competitive accuracy in the corpus considered, as it obtains an SER above 40%. Additionally, the computation time is greater than ours under the same aforementioned hardware specifications, which validates the CRNN approach in this aspect as well.

## *5.3. Learning Complexity*

The vast amount of available data in the previous experiment prevents a more in-depth comparison of the representations considered. In most real cases, the amount of available data (or the complexity of it) is not so ideal. That is why in this section we want to analyze more thoroughly both representations in terms of the learning process of the neural model.

First, we want to see the convergence of the models learned in the previous section. That is, how many training epochs the models need for tuning their parameters appropriately. The curves obtained by each type of model are shown in Figure 5.

From these curves we can observe two interesting phenomena. On the one hand, both models converge relatively quick, as after 20 epochs the *elbow* point has already been produced. In fact, the convergence is so fast that the agnostic representation begins to overfit around the 40th epoch. On the other hand, analyzing the values in further detail, it can be seen that the convergence of the model that trains with the agnostic representation is more pronounced. This could indicate a greater facility to learn the task.

To confirm this phenomenon, the results obtained in an experiment in which the training set is incrementally increased are shown below. In particular, the performance of the models will be evaluated according to the size of training set sizes of 100, 1000, 10,000, and 20,000 samples. In addition, in order to favor the comparison, the results obtained in Section 5.2 will be drawn in the plots (around 80,000 training samples).

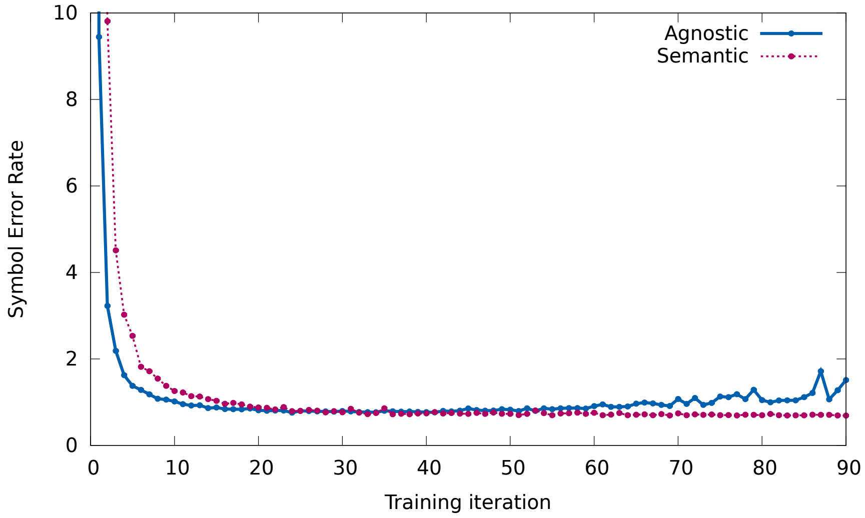


**Figure 5.** Symbol Error Rate over validation partition with respect to the training epoch.

The evolution of both Sequence and Symbol Error Rate are given in Figure 6a,b, respectively, for the agnostic and semantic representations.

These curves certify that learning with the agnostic model is simpler, because when the number of training samples is small, this representation achieves better results.

We have already shown that, in the long run, the semantic representation slightly outperforms its performance. However, these results may give a clue as to which representation to use when the scenario is not so ideal like the one presented here. For example, when either there is not so much training data available or the input documents depict a greater difficulty (document degradation, handwritten notation, etc.).

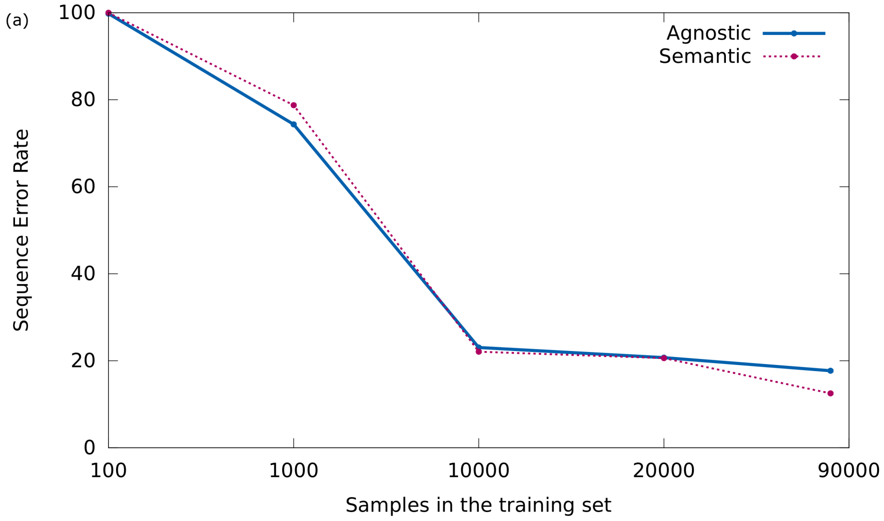


**Figure 6.** *Cont.*

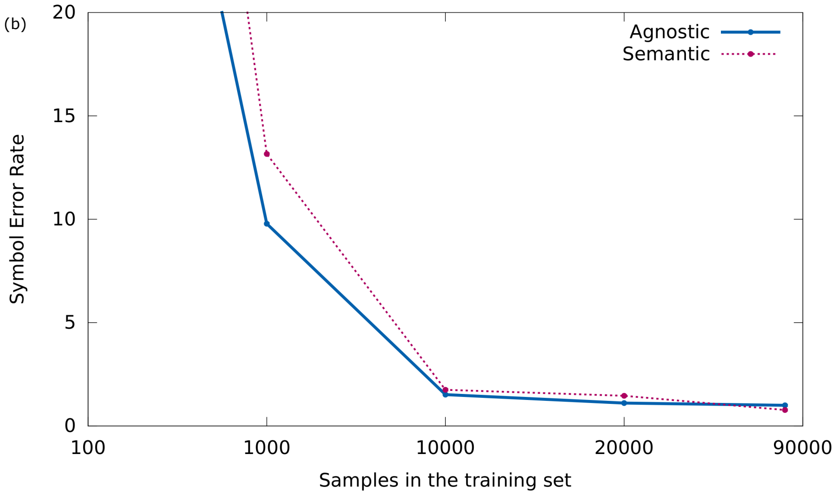


**Figure 6.** Comparison between *agnostic* and *semantic* representations in the evolution of the evaluation metrics with respect to the size of the training set. Note that the $x$-axis does not present a linear scale. (**a**) Sequence Error Rate; (**b**) Symbol Error Rate.

### 5.4. Localization

As already mentioned above, the CTC loss function has the advantage that it allows to train the model without needing an aligned ground-truth; that is, the model does not need to know the location of the symbol within the input image. In turn, this condition is a drawback when the model is expected to infer the positions of the symbols in the image during decoding. The CTC function only worries that the model converges towards the production of the expected sequences, and ignores the positions in which the symbols are produced.

We show in Figure 7 an image that is perfectly predicted by our approach, both in the agnostic (Figure 7a) and semantic (Figure 7b) case. We have highlighted (in gray) the zones in which the network predicts that there is a symbol, indicating with blue lines the boundaries of them. The non-highlighted areas are those in which the *blank* symbol is predicted.

In both cases it can be clearly observed that the neuronal model hardly manages to predict the exact location of the symbols. It could be interpreted that the semantic model has a slightly better notion—since it spans better the width of the image—but it does not seem that such information is useful in practice unless the approximate position is enough for the task using it.

This fact, however, is not an obstacle to correctly predict the sequence since the considered recurrent block is bidirectional; that is, it shares information in both directions of the $x$-dimension. Therefore, it is perfectly feasible to predict a symbol in a frame prior to those in which it is actually observed.

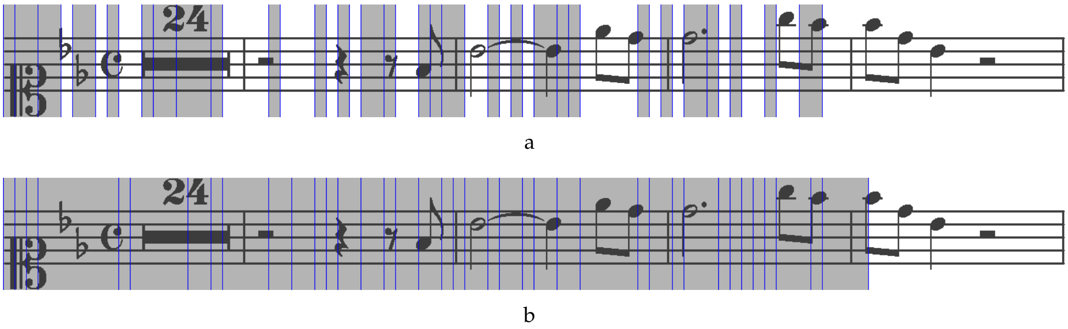


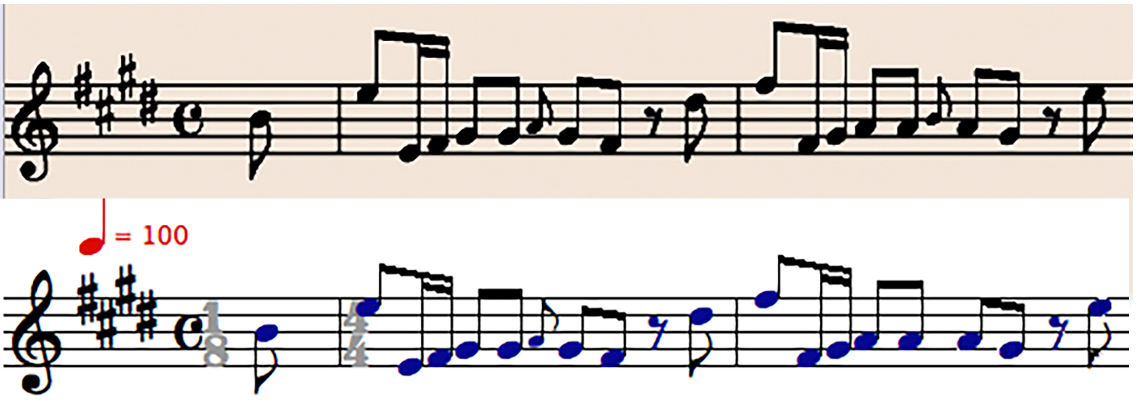
**Figure 7.** Example of symbol localization with respect to the different output representations. Gray areas correspond to detected symbols, while the blue lines indicates boundaries. Columns with no highlight are those in which the model predicts *blank*. (**a**) Agnostic representation; (**b**) Semantic representation.

### *5.5. Commercial OMR Tool Analysis*

There exist many commercial OMR tools with which a comparison can be carried out. For this analysis, we choose one of the best commercial tools available: Photoscore Ultimate (www.neuratron.com/photoscore.htm), version 8.8.2. It is publicized as having "*22 years*" of recognition experience and accuracies "*over 99.5%*". However, since Photoscore is conceived for its interactive use, it does not allow for batch processing. Therefore, the comparison of our approach with this tool will be conducted qualitatively by studying the behavior in some selected examples.

Images supplied to Photoscore have been converted into TIFF format with 8-bit depth, following the requirements of the tool. Below, we will show some snapshots of the output of the application, where the white spaces around the staves have been manually trimmed for saving space. Note that the output of Photoscore is not a list of recognized music glyphs, but the musical content itself. Thus, in its output, the tool superimposes some musical symbols that, despite being not present at the input image, indicate the musical context that the tool has inferred. Namely, these visual hints are time signatures drawn in gray, tempo marks in red, and musical figures preceded by a plus or a minus sign showing the difference between the sum of the figures actually present in the measure and the expected duration given the current time signature. From our side, we provide the sequence of music symbols predicted by our model so as to analyze the difference between both outputs.

For illustrating the qualitative comparison, we manually looked for samples that cover all possible scenarios. For instance, Figure 8b shows an incipit in which both systems fail, yet for different reasons: Photoscore misses the last *appoggiatura*, whereas our method predicts an ending *barline* that is not in the image (this error was already discussed in Section 5.2.1).



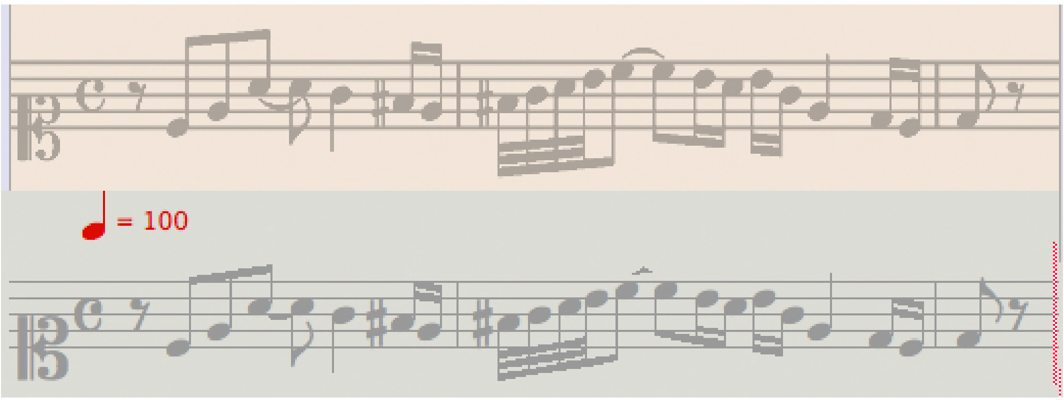
a

```
clef-G2, keySignature-EM, timeSignature-C, note-B4_eighth, barline,
note-E5_eighth, note-E4_sixteenth, note-F#4_sixteenth, note-G#4_eighth,
note-G#4_eighth, gracenote-A4_eighth, note-G#4_eighth, note-F#4_eighth,
rest-eighth, note-D#5_eighth, barline note-F#5_eighth, note-F#4_sixteenth,
note-G#4_sixteenth, note-A4_eighth, note-A4_eighth, gracenote-B4_eighth,
note-A4_eighth, note-G#4_eighth, rest-eighth, note-E5_eighth, barline
```

b

**Figure 8.** Incipit RISM ID no. 000100369. *Sonatas*. Christlieb Siegmund Binder. Both systems fail at recognizing the content within the image. (**a**) Photoscore output: The last *appoggiatura* is not detected; (**b**) Output of our system: It generates an ending barline not present in the original score.

For the sample depicted in Figure 9, the output of Photoscore is exact, but our system misses the first *tie*. The opposite situation is given by the sample of Figure 10, where Photoscore makes many mistakes. It is not able to correctly recognize the *tie* between the last two measures. In addition, the *acciaccaturas* are wrongly detected: they are identified as either *appoggiaturas* or totally different figures such as *eighth note* or *half note*. On the contrary, our system perfectly extracts the content from the score.

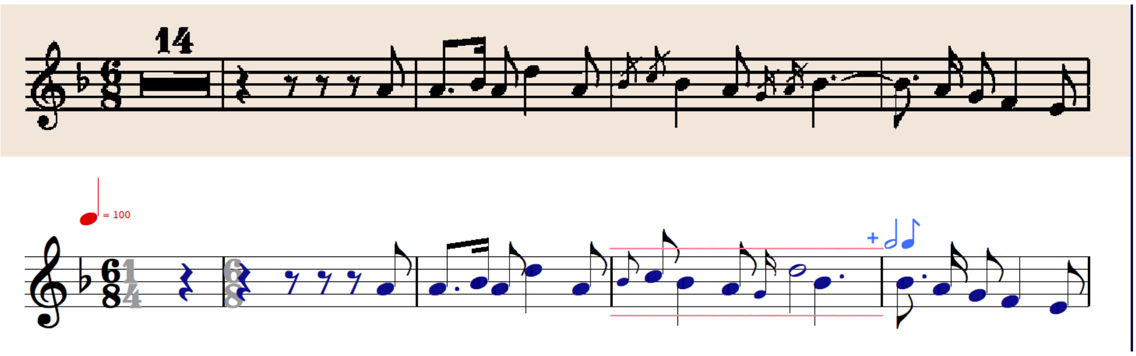


a

```
clef-C1, timeSignature-C, rest-eighth, note-C4_eighth, note-E4_eighth,
note-A4_eighth, note-A4_eighth, note-G4_quarter, note-F#4_sixteenth,
note-E4_sixteenth, barline, note-F#4_thirty_second,
note-G4_thirty_second, note-A4_thirty_second, note-B4_thirty_second,
note-C5_eighth, tie, note-C5_eighth, note-B4_sixteenth, note-A4_sixteenth,
note-B4_sixteenth, note-G4_sixteenth, note-E4_quarter, note-D4_sixteenth,
note-C4_sixteenth, barline, note-D4_eighth, rest-eighth
```

b

**Figure 9.** Incipit RISM ID no. 000051802. *Einige canonische Veränderungen. Excerpts*. Johann Sebastian Bach. Photoscore correctly recognizes the content, whereas our system fails. (**a**) Photoscore output: The sample is perfectly recognized; (**b**) Output of our system: The first *tie* is not detected.



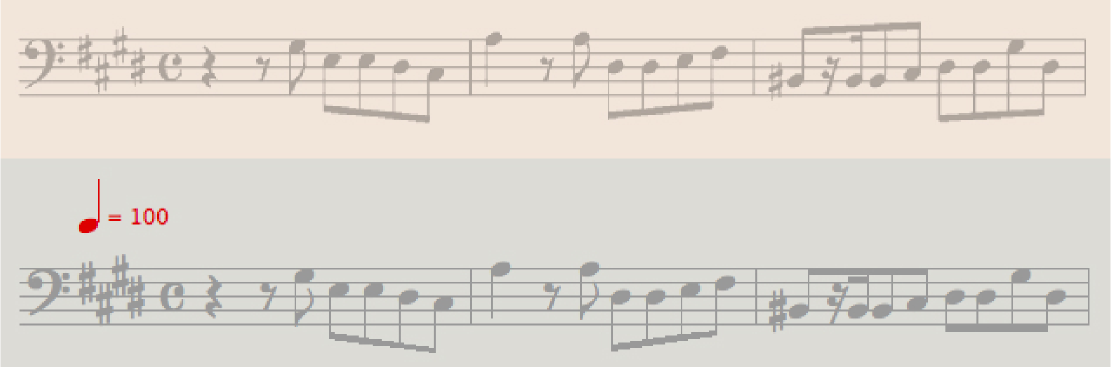
a

```
clef-G2 keySignature-FM, timeSignature-6/8, multirest-14, barline,
rest-quarter, rest-eighth, rest-eighth, rest-eighth, note-A4_eighth, barline,
note-A4_eighth., note-Bb4_sixteenth, note-A4_eighth, note-D5_quarter,
note-A4_eighth, barline, gracenote-Bb4_eighth, gracenote-C5_eighth,
note-Bb4_quarter, note-A4_eighth, gracenote-A4_sixteenth, gracenote-G4_sixteenth,
note-Bb4_quarter., tie, barline, note-Bb4_eighth., note-A4_sixteenth, note-G4_eighth,
note-F4_quarter, note-E4_eighth, barline
```

b

**Figure 10.** Incipit RISM ID no. 000101138. *Se al labbro mio non credi*. Giovanni Battista Pergolesi. Photoscore fails at recognizing the content, whereas the prediction of our system is exact. (**a**) Photoscore output: Many mistakes are made; (**b**) Output of our system: The sample is perfectly recognized.

Finally, the sample of Figure 11 is perfectly recognized by both Photoscore and our system.



a

```
clef-F4, keySignature-EM, timeSignature-C, rest-quarter, rest-eighth,
note-G#3_eighth, note-E3_eighth, note-E3_eighth, note-D#3_eighth,
note-C#3_eighth, barline, note-A3_quarter, rest-eighth, note-A3_eighth,
note-D#3_eighth, note-D#3_eighth, note-E3_eighth, note-F#3_eighth,
barline, note-B#2_eighth, rest-sixteenth, note-B2_sixteenth, note-B2_eighth,
note-C#3_eighth, note-D#3_eighth, note-D#3_eighth, note-G#3_eighth,
note-D#3_eighth, barline
```

b

**Figure 11.** Incipit RISM ID no. 000100016. *Es ist das Heil uns kommen her*. Johann Sebastian Bach. Both Photoscore and our system perfectly recognize the music content within the image. (**a**) Photoscore output: The sample is perfectly recognized; (**b**) Output of our system: The sample is perfectly recognized.

The examples given above show some of the most representative errors found. During the search of these examples, however, it was difficult to find samples where both system failed. In turn, it was easy to find examples where Photoscore failed and our system did not. Obviously, we do not mean that our system behaves better than Photoscore, but rather that our approach is competitive with respect to it.

## 6. Conclusions

In this work, we have studied the suitability of the use of the neural network approach to solve the OMR task in an end-to-end fashion through a controlled scenario of printed single-staff monodic scores from a real world dataset.

The neural network used makes use of both convolutions and recurrent blocks, which are responsible of dealing with the graphic and sequential parts of the problem, respectively. This is combined with the use of the so-called CTC loss function that allows us to train the model in a less demanding way: only pairs of images and their corresponding transcripts, without any geometric information about the position of the symbols or their composition from simple primitives.

In addition to this approach, we also present the Printed Images of Music Staves (PrIMuS) dataset for use in experiments. Specifically, PrIMuS is a collection of 87 678 incipits extracted from the RISM repository and rendered with various fonts using the Verovio tool.

The main contribution of the present work consisted of analyzing the possible codifications that can be considered for representing the expected output. In this paper we have proposed and studied two options: an agnostic representation, in which only the graphical point of view is taken into account, and a semantic representation, which codifies the symbols according to their musical meaning.

Our experiments have reported several interesting conclusions:

- The task can be successfully solved using the considered neural end-to-end approach.

- The semantic representation that includes musical meaning symbols has a superior glass ceiling of performance, visibly improving the results obtained using the agnostic representation.
- In general, errors occur in those symbols with less representation in the training set.
- This approach allows a performance comparable to that of commercial systems.
- The learning process with the agnostic representation made up of just graphic symbols is simpler, since the neural model converges faster and the learning curve is more pronounced than those with the semantic representation.
- Regardless of the representation, the neural model is not able to locate the symbols in the image—which could be expected because of the way the CTC loss function operates.

As future work, this work opens many possibilities for further research. For instance, it would be interesting to study the neural approach in a more general scenario in which the scores are not perfectly segmented into staves or in non-ideal conditions at the document level (irregular lighting, geometric distortions, bleed-through, etc.). However, it is undoubted that the most promising avenue is to extend the neural approach so that it is capable of dealing with a comprehensive set of notation symbols, including articulation and dynamic marks, as well as with multiple-voice polyphonic staves. We have seen in PrIMuS that there are several symbols that may appear simultaneously (like the numbers of a time signature), and the neural model is able to deal with them. However, it is clear that polyphony, both at the single staff level (eg., chords) or at the system level, represents the main challenge to advance in the OMR field. 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 these representations, it is feasible to think of establishing a synergy that ends up with better results in all senses.

**Acknowledgments:** This work was supported by the Social Sciences and Humanities Research Council of Canada, and the Spanish Ministerio de Economía y Competitividad through Project HISPAMUS Ref. No. TIN2017-86576-R (supported by UE FEDER funds).

**Author Contributions:** J.C.-Z. and D.R. conceived and designed the experiments; D.R. generated the ground-truth data; J.C.-Z. performed the experiments; J.C.-Z. and D.R. analyzed the results; J.C.-Z. and D.R. wrote the paper.

**Conflicts of Interest:** The authors declare no conflict of interest. The founding sponsors had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, and in the decision to publish the results.

## Abbreviations

The following abbreviations are used in this manuscript:

| | |
|---|---|
| OMR | Optical Music Recognition |
| DL | Deep Learning |
| HMM | Hidden Markov Models |
| CTC | Connectionist Temporal Classification |
| RNN | Recurrent Neural Network |
| CNN | Convolutional Neural Network |
| CRNN | Convolutional Recurrent Neural Network |
| CWMN | Common Western Music Notation |
| SGD | Stochastic Gradient Descent |
| PrIMuS | Printed Images of Music Staves |
| GPU | Graphical Processing Units |

## Appendix A. Grammars of Agnostic and Semantic Representations

The grammars describing both the agnostic and semantic encodings introduced in Section 3 are detailed below in EBNF notation.

**Table A1.** Lexical rules for the semantic grammar.

| TOKEN | DEFINITION |
|---:|:---|
| **digit** | `(‘0’..‘9’)` |
| **integer** | `(‘0’..‘9’)+` |
| **slash** | `‘.’` |
| **clefnote** | `{‘C’ \| ‘G’ \| ‘F’}` |
| **linenumber** | `{‘1’ \| ‘2’ \| ‘3’ \| ‘4’ \| ‘5’}` |
| **accidentals** | `{‘bb’ \| ‘b’ \| ‘n’ \| ‘#’ \| ‘x’}` |
| **metersigns** | `{“C” \| “C/”}` |
| **trill** | `“trill”` |
| **fermata** | `“fermata”` |
| **clef** | `“clef”` |
| **note** | `“note”` |
| **gracenote** | `“gracenote”` |
| **rest** | `“rest”` |
| **multirest** | `“multirest”` |
| **barline** | `“barline”` |
| **thickbarline** | `“thickbarline”` |
| **figure** | `{’’quadruple_whole” \| “double_whole” \| “whole” \| “half” \| “quarter” \| “eighth” \| “sixteenth” \| “thirty_second” \| “sixty_fourth” \| “hundred_twenty_eighth” \| “two_hundred_fifty_six”}` |
| **dot** | `‘.’` |
| **tie** | `“tie”` |
| **diatonicpitch** | `{‘A’ \| ‘B’ \| ‘C’ \| ‘D’ \| ‘E’ \| ‘F’ \| ‘G’}` |
| **keysignature** | `“keySignature”` |
| **timesignature** | `“timeSignature”` |
| **minor** | `‘m’` |
| **major** | `‘M’` |
| **sep** | `TAB` |
| **sepsymbol** | `’-’` |
| **sepvalues** | `‘_’` |

**Table A2.** Semantic file grammar.

sequence → (symbol **sep** symbol)*

symbol → **clef sepsymbol clefnote linenumber**

| **timesignature sepsymbol** (**metersigns** | (**integer slash integer**))

| **keysignature sepsymbol diatonicpitch accidentals**? (**major** | **minor**)?

| (**note** | **gracenote**) **sepsymbol** pitch **sepvalues figure** dots? (**sepvalues fermata**)? (**sepvalues trill**)?

| **tie**

| **barline**

| **rest sepsymbol figure** dots? (**sepvalues fermata**)?

| **multirest sepsymbol integer**

pitch → **diatonicpitch accidentals**? octave

octave → **integer**

dots → **dot**+

**Table A3.** Lexical rules for the agnostic grammar.

| TOKEN | DEFINITION |
|---:|:---|
| **digit** | `(‘0’..‘9’)` |
| **integer** | `(‘0’..‘9’)+` |
| **clefnote** | `{‘C’ \| ‘G’ \| ‘F’}` |
| **accidentals** | `{“double_flat” \| “flat” \| “natural” \| “sharp” \| “double_sharp”}` |
| **metersigns** | `{“C” \| “C/”}` |
| **startend** | `{“start” \| “end”}` |
| **position** | `{“above” \| “below”}` |
| **trill** | `“trill”` |
| **fermata** | `“fermata”` |
| **clef** | `“clef”` |
| **note** | `“note”` |
| **gracenote** | `“gracenote”` |
| **rest** | `“rest”` |
| **accidental** | `“accidental”` |
| **barline** | `“barline”` |
| **thickbarline** | `“thickbarline”` |
| **metersign** | `“metersign”` |
| **digit** | `“digit”` |
| **slur** | `“slur”` |
| **multirest** | `“multirest”` |
| **beams** | `{“beamLeft” \| “beamBoth” \| “beamRight”}` |
| **figures** | `{’’quadruple_whole” \| “double_whole” \| “whole” \| “half” \| “quarter” \| “eighth” \| “sixteenth” \| “thirty_second” \| “sixty_fourth” \| “hundred_twenty_eighth” \| “two_hundred_fifty_six”}` |
| **sep** | `TAB` |
| **sepsymbol** | `‘.’` |
| **sepverticalposition** | `‘-’` |
| **linespace** | `{‘L’ \| ’S’}` |

**Table A4.** Agnostic file grammar.

sequence → (symbol **sep** symbol)*

symbol → specificSymbol **sepverticalpos** verticalPos

verticalPos → **linespace integer**

specificSymbol → **clef sepsymbol clefnote**
| **note sepsymbol** (figure | beam)
| **rest sepsymbol** figure
| **accidental sepsymbol accidentals**
| **barline** | **thickbarline**
| **metersign sepsymbol metersigns**
| **digit sepsymbol integer**
| **slur sepsymbol startend**
| **fermata sepsymbol position**
| **trill**
| **multirest**
| **gracenote sepsymbol** (figure | beam)

figure → **figures**

beam → **beams digits**

## References

1. Casey, M.A.; Veltkamp, R.; Goto, M.; Leman, M.; Rhodes, C.; Slaney, M. Content-Based Music Information Retrieval: Current Directions and Future Challenges. *Proc. IEEE* **2008**, *96*, 668–696.
2. Roland, P. The music encoding initiative (MEI). In Proceedings of the First International Conference on Musical Applications Using XML, Milan, Italy, 19–20 September 2002; pp. 55–59.
3. Good, M.; Actor, G. Using MusicXML for File Interchange. In Proceedings of the International Conference on Web Delivering of Music (WEDELMUSIC), Leeds, UK, 15–17 September 2003; p. 153.
4. Meredith, D. *Computational Music Analysis*, 1st ed.; Springer: New York, NY, USA, 2015.
5. Keil, K.; Ward, J.A. Applications of RISM data in digital libraries and digital musicology. *Int. J. Digit. Libr.* **2017**, *50*, 199.
6. Bainbridge, D.; Bell, T. The Challenge of Optical Music Recognition. *Comput. Humanit.* **2001**, *35*, 95–121.
7. Liwicki, M.; Graves, A.; Bunke, H.; Schmidhuber, J. A novel approach to on-line handwriting recognition based on bidirectional long short-term memory networks. In Proceedings of the 9th International Conference on Document Analysis and Recognition, Curitiba, Brazil, 23–26 September 2007.
8. Graves, A.; Mohamed, A.R.; Hinton, G. Speech recognition with deep recurrent neural networks. In Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, Vancouver, BC, Canada, 26–31 May 2013; pp. 6645–6649.
9. Ng, K.; McLean, A.; Marsden, A. Big Data Optical Music Recognition with Multi Images and Multi Recognisers. In Proceedings of the Electronic Visualisation and the Arts, London, UK, 8–10 July 2014; doi:10.14236/ewic/eva2014.26.
10. Byrd, D.; Simonsen, J.G. Towards a Standard Testbed for Optical Music Recognition: Definitions, Metrics, and Page Images. *J. New Music Res.* **2015**, *44*, 169–195.
11. Rebelo, A.; Fujinaga, I.; Paszkiewicz, F.; Marçal, A.R.S.; Guedes, C.; Cardoso, J.S. Optical music recognition: State-of-the-art and open issues. *Int. J. Multimed. Inf. Retr.* **2012**, *1*, 173–190.
12. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444.
13. Amodei, D.; Anubhai, R.; Battenberg, E.; Case, C.; Casper, J.; Catanzaro, B.; Chen, J.; Chrzanowski, M.; Coates, A.; Diamos, G.; et al. Deep Speech 2 : End-to-End Speech Recognition in English and Mandarin. In Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, 19–24 June 2016; pp. 173–182.
14. Voigtlaender, P.; Doetsch, P.; Ney, H. Handwriting Recognition with Large Multidimensional Long Short-Term Memory Recurrent Neural Networks. In Proceedings of the 15th International Conference on Frontiers in Handwriting Recognition ICFHR 2016, Shenzhen, China, 23–26 October 2016; pp. 228–233.

15. Graves, A.; Fernández, S.; Gomez, F.; Schmidhuber, J. Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks. In Proceedings of the 23rd International Conference on Machine Learning—ICML '06, Pittsburg, PA, USA, 25–29 June 2006; ACM: New York, NY, USA, 2006; pp. 369–376, doi:10.1145/1143844.1143891.
16. Selfridge-Field, E. *Beyond MIDI: The Handbook of Musical Codes*; MIT Press: Cambridge, MA, USA, 1997.
17. Fornés, A.; Dutta, A.; Gordo, A.; Lladós, J. CVC-MUSCIMA: A ground truth of handwritten music score images for writer identification and staff removal. *Int. J. Doc. Anal. Recognit. (IJDAR)* **2011**, *15*, 243–251.
18. Hajic, J., Jr.; Novotný, J.; Pecina, P.; Pokorný, J. Further Steps Towards a Standard Testbed for Optical Music Recognition. In Proceedings of the 17th International Society for Music Information Retrieval Conference, New York City, NY, USA, 7–11 August 2016; pp. 157–163.
19. Calvo-Zaragoza, J.; Valero-Mas, J.J.; Pertusa, A. End-to-End Optical Music Recognition Using Neural Networks. In Proceedings of the 18th International Society for Music Information Retrieval Conference, Suzhou, China, 23–27 October 2017; pp. 472–477.
20. Pinto, T.; Rebelo, A.; Giraldi, G.A.; Cardoso, J.S. Music Score Binarization Based on Domain Knowledge. In Proceedings of the 5th Iberian Conference—Pattern Recognition and Image Analysis, IbPRIA 2011, Las Palmas de Gran Canaria, Spain, 8–10 June 2011; pp. 700–708.
21. Campos, V.B.; Calvo-Zaragoza, J.; Toselli, A.H.; Vidal-Ruiz, E. Sheet Music Statistical Layout Analysis. In Proceedings of the 15th International Conference on Frontiers in Handwriting Recognition ICFHR 2016, Shenzhen, China, 23–26 October 2016; pp. 313–318.
22. Vigliensoni, G.; Burlet, G.; Fujinaga, I. Optical Measure Recognition in Common Music Notation. In Proceedings of the 14th International Society for Music Information Retrieval Conference, Curitiba, Brazil, 4–8 November 2013; pp. 125–130.
23. Burgoyne, J.A.; Ouyang, Y.; Himmelman, T.; Devaney, J.; Pugin, L.; Fujinaga, I. Lyric extraction and recognition on digital images of early music sources. In Proceedings of the 10th International Society for Music Information Retrieval Conference, Kobe, Japan, 26–30 November 2009; pp. 723–727.
24. Dalitz, C.; Droettboom, M.; Pranzas, B.; Fujinaga, I. A Comparative Study of Staff Removal Algorithms. *IEEE Trans. Pattern Anal. Mach. Intell.* **2008**, *30*, 753–766.
25. Dos Santos Cardoso, J.; Capela, A.; Rebelo, A.; Guedes, C.; Pinto da Costa, J. Staff Detection with Stable Paths. *IEEE Trans. Pattern Anal. Mach. Intell.* **2009**, *31*, 1134–1139.
26. Géraud, T. A morphological method for music score staff removal. In Proceedings of the 21st International Conference on Image Processing (ICIP), Paris, France, 19–20 September 2014; pp. 2599–2603.
27. Calvo-Zaragoza, J.; Pertusa, A.; Oncina, J. Staff-line detection and removal using a convolutional neural network. *Mach. Vis. Appl.* **2017**, *28*, 665–674.
28. Gallego, A.; Calvo-Zaragoza, J. Staff-line removal with selectional auto-encoders. *Expert Syst. Appl.* **2017**, *89*, 138–148.
29. Rebelo, A.; Capela, G.; Cardoso, J.S. Optical recognition of music symbols: A comparative study. *Int. J. Doc. Anal. Recognit.* **2010**, *13*, 19–31.
30. Calvo-Zaragoza, J.; Valero-Mas, J.J.; Rico-Juan, J.R. Recognition of Handwritten Music Symbols using Meta-features Obtained from Weak Classifiers based on Nearest Neighbor. In Proceedings of the 6th International Conference on Pattern Recognition Applications and Methods ICPRAM, Porto, Portugal, 24–26 February 2017; pp. 96–104.
31. Pinheiro Pereira, R.M.; Matos, C.E.; Braz Junior, G.; de Almeida, J.a.D.; de Paiva, A.C. A Deep Approach for Handwritten Musical Symbols Recognition. In Proceedings of the 22nd Brazilian Symposium on Multimedia and the Web 2016—Webmedia '16, Teresina, Brazil, 8–11 November 2016; pp. 191–194.
32. Lee, S.; Son, S.J.; Oh, J.; Kwak, N. Handwritten Music Symbol Classification Using Deep Convolutional Neural Networks. In Proceedings of the 3rd International Conference on Information Science and Security, Beijing, China, 8–10 July 2016.
33. Calvo-Zaragoza, J.; Sánchez, A.J.G.; Pertusa, A. Recognition of Handwritten Music Symbols with Convolutional Neural Codes. In Proceedings of the 14th IAPR International Conference on Document Analysis and Recognition, ICDAR 2017, Kyoto, Japan, 9–15 November 2017; pp. 691–696.
34. Pacha, A.; Eidenberger, H. Towards a Universal Music Symbol Classifier. In Proceedings of the 12th International Workshop on Graphics Recognition, 14th IAPR International Conference on Document Analysis and Recognition, GREC@ICDAR 2017, Kyoto, Japan, 9–15 November 2017; pp. 35–36.

35. Couasnon, B. Dmos: A generic document recognition method, application to an automatic generator of musical scores, mathematical formulae and table structures recognition systems. In Proceedings of the Sixth International Conference on Document Analysis and Recognition, Bangalore, India, 13 September 2001; pp. 215–220.
36. Szwoch, M. Guido: A Musical Score Recognition System. In Proceedings of the 9th International Conference on Document Analysis and Recognition (ICDAR 2007), Curitiba, Brazil, 23–26 September 2007; pp. 809–813.
37. Rossant, F.; Bloch, I. Robust and adaptive OMR system including fuzzy modeling, fusion of musical rules, and possible error detection. *EURASIP J. Adv. Signal Process.* **2006**, *2007*, 081541.
38. Raphael, C.; Wang, J. New Approaches to Optical Music Recognition. In Proceedings of the 12th International Society for Music Information Retrieval Conference ISMIR 2011, Miami, FL, USA, 24–28 October 2011; pp. 305–310.
39. Bitteur, H. Audiveris. Available online: https://github.com/Audiveris/audiveris (accessed on 21 March 2018).
40. Pugin, L. Optical Music Recognition of Early Typographic Prints using Hidden Markov Models. In Proceedings of the 7th International Conference on Music Information Retrieval, Victoria, BC, Canada, 8–12 October 2006; pp. 53–56.
41. Tardón, L.J.; Sammartino, S.; Barbancho, I.; Gómez, V.; Oliver, A. Optical Music Recognition for Scores Written in White Mensural Notation. *EURASIP J. Image Video Process.* **2009**, *2009*, doi:10.1155/2009/843401.
42. Calvo-Zaragoza, J.; Barbancho, I.; Tardón, L.J.; Barbancho, A.M. Avoiding staff removal stage in optical music recognition: Application to scores written in white mensural notation. *Pattern Anal. Appl.* **2015**, *18*, 933–943.
43. Calvo-Zaragoza, J.; Toselli, A.H.; Vidal, E. Early Handwritten Music Recognition with Hidden Markov Models. In Proceedings of the 15th International Conference on Frontiers in Handwriting Recognition ICFHR 2016, Shenzhen, China, 23–26 October 2016; pp. 319–324.
44. Brook, B. The Simplified 'Plaine and Easie Code System' for Notating Music: A Proposal for International Adoption. *Fontes Artis Musicae* **1965**, *12*, 156–160.
45. Pugin, L.; Zitellini, R.; Roland, P. Verovio—A library for Engraving MEI Music Notation into SVG. In Proceedings of the 15th International Conferencefor Music Information Retrieval Conference, Taipei, Taiwan, 27–31 October 2014.
46. Graves, A. Supervised Sequence Labelling with Recurrent Neural Networks. Ph.D. Thesis, Technical University of Munich, Munich, Germany, 2008.
47. Zeiler, M.D.; Fergus, R. Visualizing and Understanding Convolutional Networks. In Proceedings of the 13th European Conference on Computer Vision—ECCV 2014, Zurich, Switzerland, 6–12 September 2014, Part I; pp. 818–833.
48. Rabiner, L.; Juang, B.H. *Fundamentals of Speech Recognition*; Prentice Hall, Inc.: Upper Saddle River, NJ, USA, 1993.
49. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, preprint arXiv:1409.1556.
50. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Proceedings of the 32nd International Conference on Machine Learning—ICML 2015, Lille, France, 6–11 July 2015; pp. 448–456.
51. Glorot, X.; Bordes, A.; Bengio, Y. Deep sparse rectifier neural networks. In Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, FL, USA, 11–13 April 2011; pp. 315–323.
52. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780.
53. Bourlard, H.; Wellekens, C. Links Between Markov Models and Multilayer Perceptrons. *IEEE Trans. Pattern Anal. Mach. Intell.* **1990**, *12*, 1167–1178.
54. Bottou, L. Large-scale machine learning with stochastic gradient descent. In Proceedings of the COMPSTAT' 2010, Paris, France, 22–27 August 2010; Springer: Berlin, Germany, 2010; pp. 177–186.
55. Zeiler, M.D. ADADELTA: An adaptive learning rate method. *arXiv* **2012**, preprint arXiv:1212.5701.
56. Calvo-Zaragoza, J. TensorFlow Code to Perform End-to-End Optical Music Recognition on Monophonic Scores Through Convolutional Recurrent Neural Networks And CTC-Based Training. Available online: http://github.com/calvozaragoza/tf-deep-omr (accessed on 9 April 2018).