Optical Music Recognition by Long Short-Term Memory Recurrent Neural Networks

Arnau Baró-Mas

Abstract

Optical Music Recognition is the task of transcribing a music score into a machine readable format. Music scores today are often in a sheet format and society can not enjoy it. Many music scores are written in a single staff, and therefore, they could be treated as a sequence. Thus, this work explores the use of Long Short-Term Memory (LSTM) Recurrent Neural Networks for reading the music score sequentially, where the LSTM helps in keeping the context. Moreover, Bidirectional LSTM has been tested in order to improve the recognition task insomuch as the two direction reduces the ambiguity in some predictions. For training, a synthetic dataset of more than 40K images of incipts from RISM database has been used to validate the proposed approaches. It is labeled at primitive level. Finally, this knowledge acquired by the BLSTM has been transferred to a handwritten dataset that has been created from music scores belonging CVC-MUSCIMA database to validate the proposed approach in a real scenario.

Index Terms

Optical Music Recognition; Recurrent Neural Network; Long Short-Term Memory; Bidirectional Long Short-Term Memory.

I. Introduction

MUSIC scores are the main medium for transmitting music. In the past, the scores started being handwritten, later they became printed and thesedays, they are usually written in digital format. A musical score can be written (edited) or read (transcribed).

Music scores are normally in a sheet format. The transcription into some machine readable format can be carried out manually. However, the complexity of music notation inevitably leads to burdensome software for music score editing, which makes the whole process very time-consuming and prone to errors. Consequently, automatic transcription systems for musical documents represent interesting tools. The field devoted to address this task is known as Optical Music Recognition (OMR) [1]–[3]. Typically, an OMR system takes an image of a music score and automatically export its content into some symbolic structure such as MEI or MusicXML. Some of the existing tools are PhotoScore [1] or SharpEye [2]. OMR have other many applications such as writer identification, renewal old music scores, generate audio files and find differences between the same play but different authors.

The process of recognizing the content of a music score is complex, and therefore the workflow of an OMR system is very extensive [1]. An OMR has to deal with difficulties as the 2-dimensionalities that a music score has. A music score is read from left to right, and in each staff, symbols appear with specific rhythm (duration) and a pitch (melody).

Nowadays, there are still much more computer systems for editing new music scores rather than systems for reading them (OMR). This work focuses on recognizing the content appearing on a single staff section (e.g. scores for violin, flute, etc.), much in the same way as most text recognition research focuses on recognizing words appearing in a given line image [4]. There are existing algorithms that achieve good performance to both isolate staff sections and separate music and lyrics (accompanying text) [5]. For this reason, one can assume that the staves are already segmented, and therefore, can be processed as a sequence. To address this specific task, the proposed architecture is based on Recurrent Neural Networks (RNN), since they have been applied with great success to many sequential recognition tasks such as speech [6] or handwriting [4] recognition. Specifically, to avoid the vanishing gradient problem, a Long Short-Term Memory (LSTM) is used. Moreover, a Bidirectional LSTM is used to benefit from context information. The method has been tested with handwritten music scores. In order to cope with the lack of handwritten labeled data, data augmentation has been used.

The rest of the dissertation is organized as follows. Section II details the terminology and the different symbols that can appear in a music score. Section III overviews the relevant methods in the literature. Section IV explains the general idea of a RNN and explains how a LSTM works. Section V describes how LSTMs have been adapted to recognize music score. Section VI discusses the results. Finally, conclusions and future work are drawn in Section VII.

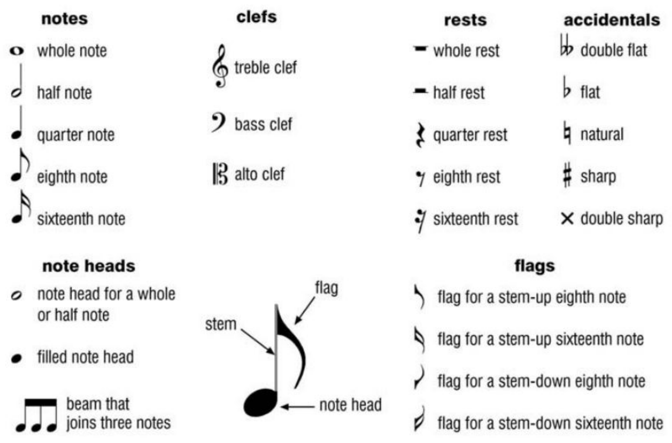
Author: Arnau Baró-Mas, abaro@cvc.uab.es
Advisor 1: Alicia Fornés-Bisquerra, Departament de Ciències de la computació, Universitat Autònoma de Barcelona, Barcelona, Spain
Advisor 2: Jorge Calvo-Zaragoza, Schulich School of Music, McGill University, Montreal, Canada
Thesis dissertation submitted: September 2017

[1]http://www.neuratron.com/photoscore.htm
[2]http://www.visiv.co.uk/

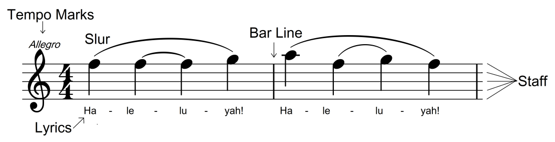
II. Terminology Music Notation

In a music score there are different symbols as symbol notes, clefs rests, etc (see Figure 1). The most common terminology is the following:

- Staff: Set of five horizontal lines and four spaces that each represent a different musical pitch.
- Clef: Music Symbol used to indicate the pitch of written notes.
- Bar lines: Vertical lines which separate every bar unit or measure.
- Notes: Is the pitch and duration of a sound. Composed of note heads, beams, stems, flags and accidentals.
- Accidental: A sign that alters one or several notes by raising or lowering the pitch.
- Rest: Is an interval of silence in a piece of music, marked by a symbol indicating the length of the pause.
- Slurs: Curve that indicates that several notes have to be played without separation.
- Dynamic and Tempo Markings: Indicates the speed of the rhythm and indicates how low/soft the music should be played.
- Lyrics: Set of words that the singers have to sing.



(a)



(b)

Figure 1. Common Symbols in a Music Score. Image (a) extracted from [7].

III. State of the art

This section describes the key references of Optical Music Recognition and overviews the Deep Learning architectures that are relevant to the present work.

A. *Optical Music Recognition*

The OMR algorithm has to be able to recognize each element located in the music score. Figure 2 illustrates the usual pipeline from a scanned music score to a machine-readable format. The steps are the following. First, preprocessing the image. The aim of this step to the musical document is to reduce problems in segmentation. Normally, before segmenting the musical symbols and/or primitives, the staff lines are removed. Hence, the segmentation task is simplified. Afterwards, the primitives are merged to form symbols. Some methodologies use rules or grammars in order to be able to validate and solve some ambiguities from the previous step. In the last step, a format of musical description is created with the information of the previous steps. These steps will be described in detail next.

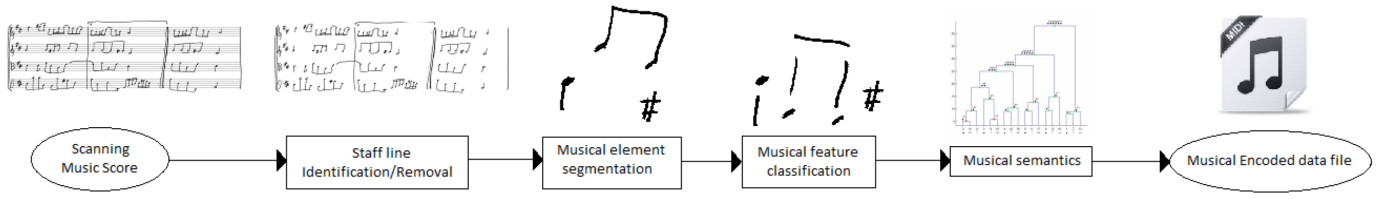

Scanning Music Score
Staff line Identification/Removal
Musical element segmentation
Musical feature classification
Musical semantics
Musical Encoded data file
MIDI

Figure 2. General pipeline for OMR

1) Preprocessing and layout analysis: The most common techniques to improve the results of the segmentation are binarization, noise removal and blur correction. However, other techniques as enhancement, skew correction or deskewing, among others have also been proposed. In music scores documents it is important to segment the document into regions. Authors in [8] propose a new algorithm to segment the regions that include text and regions containing music scores. This segmentation is based on bag of visual words and random block voting algorithms. By posterior probability each block is classified as text or music score. Staff lines are one of the more important parts of a music score. They provide information about the pitches looking the vertical coordinate and they also provide a horizontal direction for the temporal coordinate system. OMR usually removes the staff lines [9] making the recognition task easier. Normally, the staff removal algorithm are based on projections and run-length analysis, contour-line tracking, or graphs. Moreover, these algorithms have to deal with overlapping symbols and the distorted lines if they have been handwritten as Figure 3 shows.

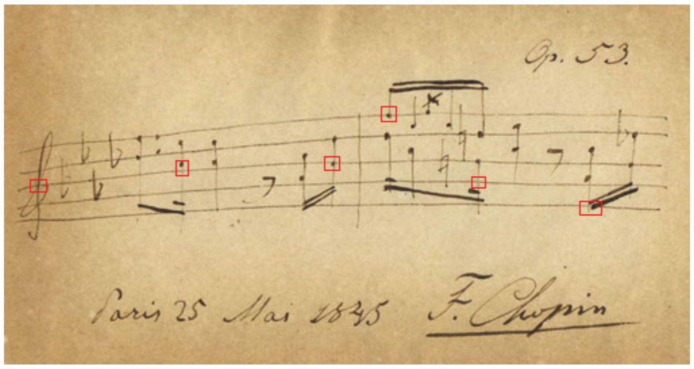
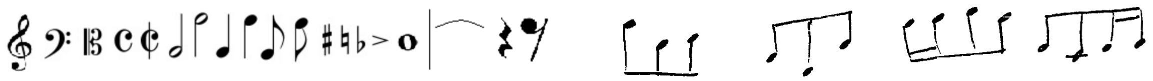


Figure 3. Vintage music score written by Chopin. The staff lines are handwritten, thus, they are not equidistant. The red rectangles show some crossing symbols over the staff lines.

2) Symbol recognition: The recognition of music symbols consists in the recognition of isolated and compound music symbols. This classification is done because the techniques are different, because compound music symbols cannot be recognized as a whole. It is impossible to have examples of all possible combination of this type of symbols, only a part of the population can be obtained. Figure 4 shows isolated and compound music symbols. Therefore, the classification is the following:

- *Isolated Music Symbols* Isolated music symbols are defined as these symbols that have [0,1] note-head (Figure 4 (a) shows some isolated music symbols). The most popular techniques to detect this kind of symbols are: Segmentation methods, Grammar/Rules, Sequence, Graphs and Deep Neural Networks. Isolated symbol are the easiest symbols to recognize, they can be recognized using a shape descriptor and a single-class classifier.
- *Compound Music Symbols* Compound music symbols are defined as these symbols that have [2,∞) note-heads. Usually, they are recognized using primitive-based techniques. The most popular methods to deal with this kind of symbols are: Grammar/Rules, Graphs and Deep Neural Networks. Grammars or rules are used in order to validate the detected compound music symbols. Some techniques as segmentation can not be used because they are more difficult to be recognized and have infinite combinations between them. Figure 4 (b) shows the huge variability of compound music symbols.



(a) Some Isolated Music Symbols. (b) Some Compound Music Symbols.

Figure 4. Examples of music symbols.

The main symbol recognition techniques have been classified into different groups:

- *Segmentation:* Segmentation-based methods are the simplest ones. Usually, each symbol is segmented. Afterwards, template matching techniques are used. Some papers propose to use some algorithms that modify the shape of the symbol blurring it in order to make the matching easier. Finally, the authors use classification algorithms such as Support Vector Machine and K-Nearest Neighbors. In [10] the authors describe a method based on the Dynamic Time Warping algorithm to recognize symbols. This method is invariant to rotation and scale. Authors of [11] introduce a symbol shape descriptor. This method is invariant to rotation and reflection. Furthermore, they present the Blurred Shape Model descriptor. BSM encodes the spatial probability of the shape. In [12] authors compare several methods classifying symbols. They performed their experiments by Neural Networks, Nearest Neighbour, Support vector machines and Hidden Markov models. And in [13] they propose to learn the best distance for the k-nearest neighbour classifier and the performed of the method is compared with the support vector machine classifier.

- *Grammars / Rules:* This method propose to define a grammar or rule based on combinations of primitives (*i.e.* note-heads, steams, beam, etc). Grammars/Rules make use of morphological operations to detect circles (note-heads), vertical lines (steams, bars) and horizontal lines (beams). After the primitive identification process, the pre-defined rules are applied in order to find the more probable combination of primitives to obtain the musical symbol. In [14] the authors use some rules in order to join the primitives previously detected. They construct a dendrogram and they validate each level of this dendrogram by this set of rules. Authors in [15] propose a grammar to help the detection of most errors on note duration.
- *Sequence:* Models using Hidden Markov Models (HMM) in optical music recognition tasks have produced remarkable results in monophonic music scores. The main reason of their good performance is that music scores generally can be seen as a lineal sequence. However, in more complex documents with more than one voice (polyphonic), HMM do not have a good performance. In addition, HMM are able to segment and recognize without a previous preprocessing task. In [16] the author presents an OMR using Hidden Markov Models without staff line removal. And in [17] they present an approach using maximum posteriori adaptation in order to improve an OMR based on Hidden Markov Models.
- *Graph:* Graph-based techniques try to increase the classical appearance-based approaches providing structural information. Graphs can define relations between previously detected graphical primitives or just the image skeleton codifying shape information. Nodes correspond to key-points or primitives whereas the edges codify their relations. This method avoids the problem about the compound components but increases the complexity of the algorithm. Authors in [18] use graph-like classification applied to ancient music optical recognition. This method combines a number of different classifiers to simplify the type of features and classifiers used in each classification step.

3) Validation: This step is related with the previous one. Some grammars or rules are defined by the authors in order to make more robust the recognition step in front of ambiguities. Some works [14], [15], [19] proposes the use of grammars to correct some mistakes as repeating or missing symbols. Another aspect that could be verified with these grammars is if the number of beats match with the time signature.

4) Final representation: Most optical music recognition systems provides an output representing the input's music score at the end of the process. The most common output files are MIDI[3], MusicXML[4] or MEI[5]. MIDI (Musical Instrument Digital interface) is a communication technical standard used in electronic music devices. MusicXML is an open musical notation format based on Extensible Markup Language (XML). And MEI as MusicXML is an open-source effort to define a system for encoding musical documents in a machine-readable structure.

5) Summary: Table I shows advantages and disadvantages of previous methods. First column shows techniques described previously, second column advantages of each technique and third column disadvantages of each technique. One might conclude that segmentation-based techniques are only suitable for isolated symbols. Contrary, grammars and graphs are able to deal with compound symbols although many isolated symbols are difficult to be represented by graphic primitives (e.g. clefs and rests).

Table I
ADVANTAGES AND DISADVANTAGES OF DIFFERENT TECHNIQUES USED IN OMR

	Advantages	Disadvantages
Segmentation	Simplest method. Machine learning based.	It needs a segmentation preprocess. Difficulties in recognizing compound music symbols
Grammars/Rules	Able to recognize compund music notes and correct errors.	It needs a segmentation preprocess.
Sequence	Good performance in monophonic music scores. It does not need a segmentation preprocess.	Cannot recognize polyphonic music scores.
Graph	Able to recognize compund music notes.	Complexity of the algorithm.

B. Recognition of Handwritten Music Scores

Concerning handwritten scores, although it is remarkable the work in Early musical notation [17], [18] the recognition of handwritten Western Musical Notation still remains a challenge. The main two reasons are the following. First, the high variability in the handwriting style increases the difficulties in the recognition of music symbols. Second, the music notation rules for creating compound music notes (groups of music notes) allow a high variability in appearance that require special attention. In order to cope with the handwriting style variability when recognizing individual music symbols (e.g. clefs, accidentals, isolated notes), the community has used specific symbol recognition methods [10], [11] and learning-based techniques such as Sector Vector Machine's, Hidden Markov Model's or Neural Network's [12]. As stated in [20], in the case of the recognition of compound music notes, one must deal not only with the compositional music rules, but also with the ambiguities in the detection and classification of graphical primitives (e.g. headnotes, beams, stems, flags, etc.). It is true that temporal information is undoubtedly helpful in on-line music recognition, as it has been shown in [21], [22]. Nowadays, a musician can find several

[3]https://www.midi.org/
[4]http://www.musicxml.com/
[5]http://music-encoding.org/

applications for mobile devices, such as StaffPad [6], MyScript Music [7] or NotateMe [8]. Concerning the off-line recognition of handwritten groups of music scores, much more research is still needed. PhotoScore seems to be the only software able to recognize off-line handwritten music scores, and its performance when recognizing groups of notes is still far from satisfactory. One of the main problems is probably the lack of sufficient training data for learning the high variability in the creation of groups of notes.

C. Deep Learning

Deep Learning may seem very novel technique. However, as Goodfellow, Bengio and Courvill claim in [23] that it appeared between 1940s-1960s. Deep learning in those years was called cybernetics. Later, between 1980s-1990s it was called connectionism and the current resurgence in 2006 finally has been called deep learning. The cybernetics appears by hand of biological learning [24], [25] and the first model of perceptron [26]. Afterwards, connectionism appears with the back-propagation [27] and finally deep learning appears in 2006 by Hinton, Bengio and Ranzato [28]–[30]. Next, the most popular techniques are brievly explained.

1) Convolutional Neural networks: Convolutional (CNN) networks consist of several convolutional layers and optionally followed by fully connected layers. CNN are easy to train, and it takes advantage of the 2D structure of an input image. These networks have demonstrated to obtain excellent results in classification tasks even though these networks would need a symbol segmentation preprocessing. Some examples are VGG [31] or AlexNet [32].

2) Multilayer perceptron: Multilayer perceptron (MLP) is a feedforward artificial neural network. By training on a dataset it learns a function by a set of features. MLP is similar as a logistic regression classifier but MLP between the input and the output can have one or more non-linear layers, called hidden layers. This network consists of multiple layers of nodes in a directed graph.

3) Fully Convolutional Networks: Fully Convolutional Networks (FCN) is similar learnable as a fully connected CNN. FCN have not any fully-connected layers or MLP. FCN learns local spatial information in order to make decisions. In [33] the authors introduce the dense captioning. The FCN proposed processes an image without external regions proposals, and trained end-to-end. Whereas [34] propose position-sensitive score maps to solve the problem created by translation-invariance in image classification and translation-variance in object. They perform firstly the region proposal, and then the region classification detection.

4) Recurrent Neural Networks: Recurrent Neural Networks (RNN) process a sequence structured input. Usually the inputs of classical neural networks are independent of each other. However, in some tasks the network must be able to deal with documents that follow a sequence. It is called recurrent because the information is passed from one ste the same process but having into account the previous time steps computations. The authors by [35] introduce the deep RNN by a novel framework based on neural operators. In order to cope with the vanishing gradient problem, Long Short-Term Memory (LSTM) [36] networks appear. The architecture is very similar but LSTM decides which information has to keep an which has to remove. In [37] the author shows that Long Short-term Memory recurrent neural networks can be used to generate complex sequences.

5) Attention models: The attention mechanism is based on the animals vision. Instead of using all the information that we have, we only use the relevant information. In [38] they perform a CNN with attentive context which incorporates global and local contextual information into the region-based providing better object detection performance. The authors in [39] propose a model that is able to extract information from an image by adaptively selecting regions or locations and only processing the selected regions. And in [40] they introduce an attention based model that automatically learns to describe the content of image.

From the above techniques, Recurrent Neural Networks seem the more appropriate for the recognition of music scores.

D. Deep Learning in Music

Some deep learning techniques have been applied to audio processing. The most well-known are:

- CNNs have been applied to process the audio files to detect and recognize the sound of certain objects and scenes in video. In [41] they use CNNs to compensate differences between video and audio sampling rates. Whereas, the authors in [42] use CNN in order to process sounds.
- RNNs have been applied in MIDI generation. The authors of [43], [44] use RNNs, specifically LSTM in order to produce new music files taking advantage of sequence model of the music input files. Whereas, in [35] they use RNNs in polyphonic music in order to make predictions.

As far as we know, there are very few works that have used deep learning techniques in Optical Music Recognition. For example, a Multilayer perceptron network has been used for classifying isolated music symbols [12]. A very recent OMR work has been published [45], which uses a Convolutional Sequence-to-Sequence network for recognizing printed scores.

[6] http://www.staffpad.net/
[7] https://www.myscript.com/technology/
[8] http://www.neuratron.com/notateme.html

IV. Background

In this section we will described the background that we will use in this work.

A. Recurrent Neural Networks

This subsection describes how a Recurrent Neural Networks works. Recurrent Neural Networks (RNN) [46] are networks which keep information in order to use that in future predictions. Ideally solves sequence and list problems thanks to have a chain architecture. Similarly humans do the same, when a person is reading a text he understands the current word based on the understating of previous words. Thus, RNN are networks with loops to keep this information. A RNN can be seen as a network that passes a message to the same network (replicated) over time. Figure 5 shows a RNN unrolled. Firstly, in the left side, it shows a network (A) which have a input (x) in each time step (x_t) and for each input it have and output (h_t), moreover the network (A) sends information to itself. In the right it shows the same process unrolled.

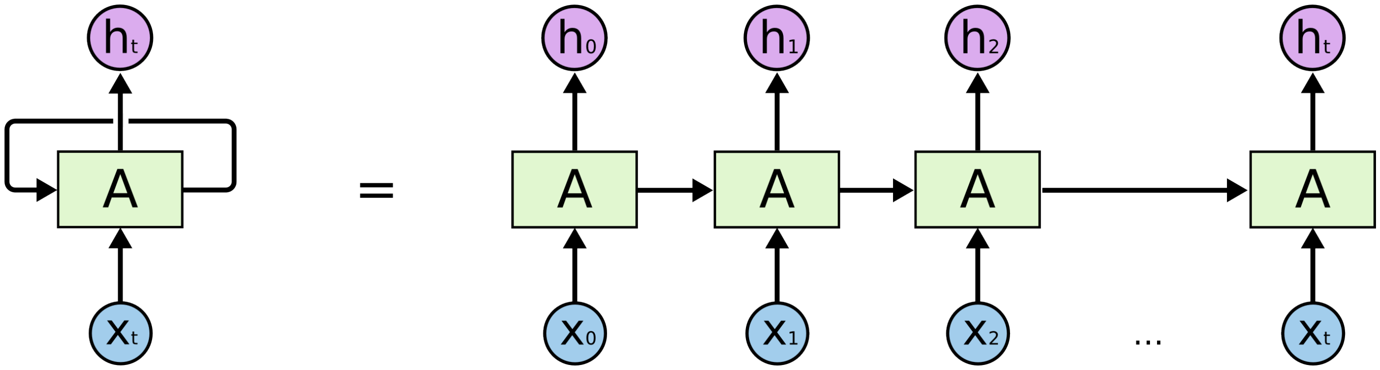
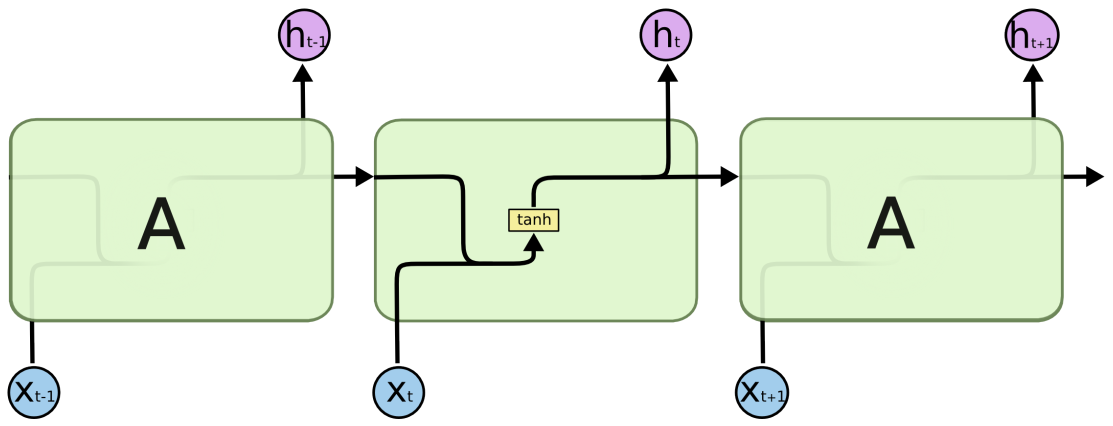


Figure 5. Recurrent Neural Network with loop and unrolled RNN. Image extracted from [46]

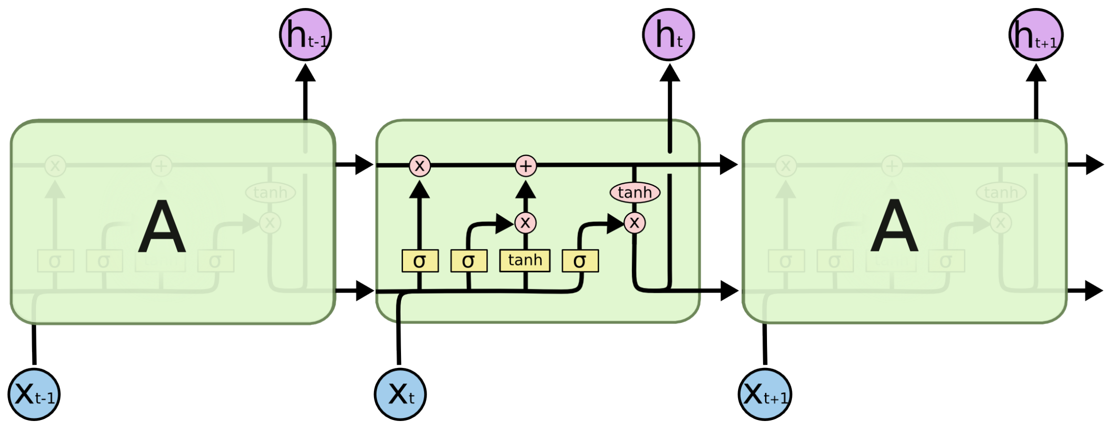
One of the advantages of RNN is that they can use previous information to the current task. However, in some problems it is only necessary to look at recent information to perform the present, but there are other problems that they need more context information. The Reccurrent Neural networks are able to learn the past information. but is possible that if the gap between the relevant information is very large, RNNs are not able to learn to connect the information. Long Short-Term Memory Networks described in the next section to solve this problem are needed to use.

B. Long Short-Term Memory Network

This subsection describes how a Long Short-Term Memory Network works. Long Short-Term Memory Networks (LSTMs) are a type of RNN. LSTM are able to learn long-term dependencies and can avoid the vanishing gradient problem. They can keep information for long time. As subsection IV-A describes, RNNs have a chain architecture and LSTMs are not an exception. The principal difference between their structure is that RNNs have a single $tanh$ layer(Figure 6 (a)) while LSTMs have four interactive neural network layers (Figure 6 (b)): $sigmoid$, $sigmoid$, $tanh$, $sigmoid$.



(a) RNN contains a single layer



(b) LSTM contains four interacting layer

Figure 6. Difference strucutre between RNN and LSTM. Image extracted from [46]

The most important part of LSTMs is the cell state. The information that flows by the cell state in each step of the loop only is changed by some minor linear interactions. LSTMs are able to remove or add information to the cell state using gates.

The first gate is the forget gate. This gate decides which information will be removed and which information will be kept by the cell state. The inputs of the current forget gate are the current input of the network and previous output. As output, after the sigmoid, it produces a number between 0 and 1 for each number of the cell state. 0 means that information have to be removed and 1 represents that that information have to be kept.

The second gate that intervene in this process is the input gate. The input gate participates in the first part of this second step. In the first part the input gate through a sigmoid layer decides which values have to update. Then, a tanh layer construct a vector with the new candidate values $\widetilde{C}_t$. For last, these will be combined to create a state update.

Next, C_{t-1} has to be updated to the new C_t. To create the new C_t, C_{t-1} is multiplied by f_t calculated in the first step and $i_t \cdot \widetilde{C}_t$ is added. The result are the new candidates, weighted by how update each value.

Finally, the output is calculated by the new cell state, the input of the network and the previous output h_{t-1}. First the sigmoid of the input of the network and the previous output h_{t-1} are calculated . The new cell state is put through tanh layer to push the values to be between -1 and 1. The result is multiplied by the result of the sigmoid layer and only output the relevant parts.

Thanks to these gates, the information will be saved or discarded depending on the gradients values.

V. Optical Music Recognition -Long Short-Term Memory Networks

This section describes the contribution of this work. Firstly, the problem statement is presented, describing the similarities with text recognition and the main difficulties. Secondly the proposed architecture is detailed. Then the Loss function is explained. The last subsection describes how the method is adapted to recognize handwritten music scores.

A. Problem statement

Music scores are a particular kind of graphical document that include text and graphics. The graphical information corresponds to staffs, notes, rests, clefs, accidentals, etc., whereas textual information corresponds to dynamics, tempo markings, lyrics, etc. Concerning the recognition of the graphical information, Optical Music Recognition (OMR) has many similarities with Optical Character Recognition (OCR), as follows:

- Isolated symbols. In case of recognizing isolated music symbols (e.g. clefs, accidentals, rests, isolated music notes), the task is similar to the recognition of handwritten characters or digits (see the first row of Figure 8). The recognizer, in this case has to be able to recognize symbols that are the same but apparently the size, the visual appearance and/or the shape are different.
- Compound music notes. The recognition of compound music notes (i.e. groups of notes joined using beams) could be seen as the task of recognizing handwritten words (see the second row of Figure 8). Here, the system must deal with touching elements (characters or music notes).
- Bar units. Sentences could be compared to bar units. Both text sentences and bar units must follow syntactical rules (see the third row of Figure 8). In the case of text, there are grammatical rules such as the order subject + verb + objects, the agreement based on grammatical person/gender, singular/plural, verb inflection, etc. Similarly, music scores must follow music notation theory. For example, the number of beats in a bar unit must sum up the time signature.

1) Difficulties of OMR: It must be noted that the difficulties in OMR are higher than in OCR because OMR requires the understanding of two-dimensional relationships. Indeed, music scores use a particular diagrammatic notation that follow the 2D structural rules defined by music theory. First of all, music notes follow a 2-dimensional notation (they have two components: rhythm and pitch). In addition, some considerations must be taken into account in specific cases such as:

- Compound music notes. Music notation allows a huge freedom when connecting music notes, which increases the difficulties in the recognition and interpretation of compound notes. For example, music notes can connect horizontally (with beams), and vertically (chords), and the position and appearance highly depends on the pitch (melody), rhythm and the musical effects that the composer has in mind. Figure 7 shows several examples of compound music groups that are equivalent in rhythm.
- Bar unit. Many music scores are polyphonic (more than one voice). These scores have the particularity of they have more notes and they do not sum up the time signature (see the third row of Figure 8).
- Ornaments. There are elements as music tuplets or ornaments notes that when you are summing up the beats they scape from the restriction. See the fourth row of Figure 8, where the red symbol it is an ornament. When counting the beats this is not to be counted.

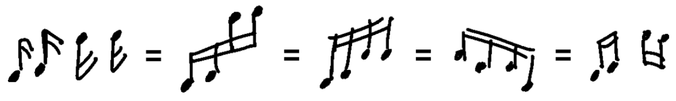


Figure 7. Equivalent (in rhythm) compound Sixteenth notes.

2) Language Models: In order to improve the results, language models can be used as in OCR. In music recognition there are different techniques, for example, the time signature is used to count if the sum of predicted rhythm units (beats) between the bar lines is correct. Thus, ambiguities in rhythm could be corrected by using syntactical rules and grammars. However, it is very difficult to apply grammars or syntactical rules in many music scores. As it is said in *Difficulties of OMR* section there are lots of music scores with more than one voice. Consequently, many notes are played in parallel, and therefore they do not count as beats. Ornaments must be treated independently, and they do not count either. Finally, a musicologist could define the harmonic rules that should be applied in order to reduce the ambiguities in polyphonic scores and semantics could also be defined using knowledge modeling techniques. However, these harmonic rules depend on the composer and the time period. Therefore, the incorporation of this knowledge seems unfeasible in this OMR stage.

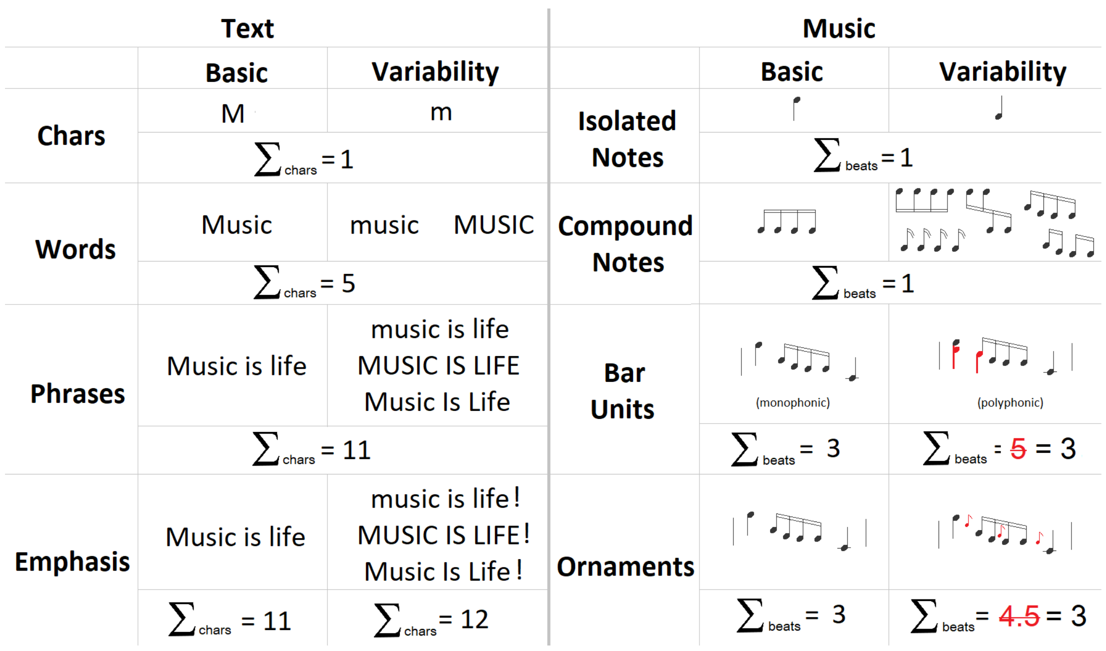


Figure 8. Comparison text vs music.

B. Proposed architecture

Following the comparison with handwritten text recognition, many music scores are written in a single staff and can be read following a sequence. For this reason, in order to make a similar lecture by a computer a RNN is appropriate to perform this task. It must to be said that although the input music scores will be single staff in these scores could appear multiple melodic voices as two voices in the same staff or chords. In other words, in a same time instance could appear more than one music note or symbol (e.g. time signature). In this work a Long Short-Term Memory (LSTM) RNN has been used. The LSTM architecture decides which information has to keep as context and which one has to remove (i.e. forget). Likewise, text is read sequentially and as it is mentioned in Section I, there are works that demonstrate that RNN performs very well in sequential recognition tasks such as speech [6] or handwriting [4] recognition.

Figure 9 shows the pipeline stages: When a batch of music scores enters into the system, first of all it is preprocessed (Section V-B1). Next, each symbol will be recognized by the LSTM network (Section V-B2) and the output of the LSTM is passed by two fully connected layer (Section V-B3). The final output is obtained (Section V-B4) with the recognized symbols.

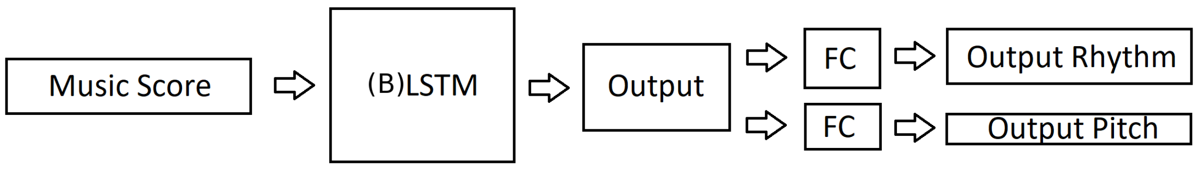


Figure 9. Architecture of the network

These steps are described next:

1) Input: To train the network a synthetic Dataset has been used, that is detailed in Section VI-A1. The LSTM is trained by batches of images that are resized to a height of 50 pixels in order to feed pixel columns into the proposed model. The maximum width can be variable depending on the widest image in the batch. Images with a width shorter than the widest one will have padding. It must to be said that the staff lines have not been removed because if they are not perfectly removed, then the symbols could be distorted or broken. In addition, if a perfectly staff removal had been achieved, the staff lines would have been needed anyway to recognize the pitch. Also, it has to be said that no feature extraction has been used, in order to maintain the spatial information and the spatial order as much as possible.

2) Long Short-Term Memory: The recurrent neural network consists of a Long Short-Term Memory (LSTM). Concretelly, a bidirectional LSTM increases the performance by reducing some ambiguities because the context information from both sides (forward and backward directions) is taken into account. Thus, it processes the input in both directions getting information of the whole symbol, and therefore, it is more accurate. For example, if one direction recognizes a note-head, the other direction can discard that the vertical line that it is reading next is not a bar line, but instead a note stem (both stems and bar lines are straight vertical lines). For this project the LSTM of PyTorch [9] has been used. The parameters of the network are:

- The LSTM has 3 layers.
- Hidden size 128.
- 100 epochs.
- Size of batch 128 per batch in LSTM and 64 in BLSTM.
- Learning rate of 10^{-4}.

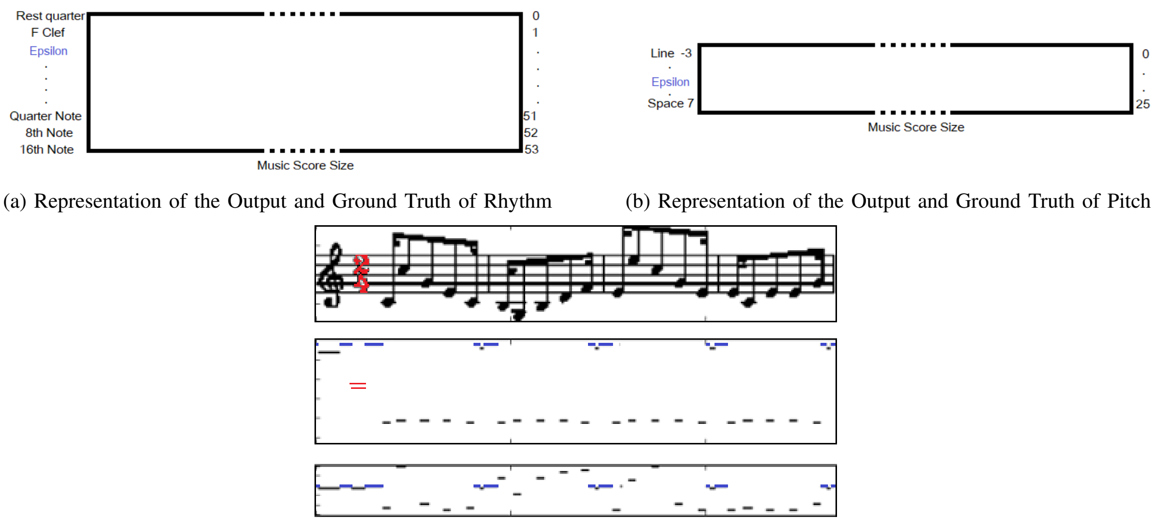
These values have been experimentally found. It must to be said that the net is trained column by column so you get one output per column. In other words, the output will end up being as long as the input image.

3) Fully Connected Layers: At the end of the network, after the LSTM's output, two fully connected (FC) layers allow us to separate the rhythm and the melody in two different outputs. The reason to split the output in two parts is that the combination between melody and rhythm is almost infinite. All possible combinations of rhythm-melody would have to be created manually as possible classes. Then a single activation per instant of time would have been obtained and a softmax layer could have been used. However, this would have required a huge number of classes, and moreover, one could have forgotten some classes. The other reason is that by separating the output in two parts, more examples are provided to learn a symbol. In other words, to learn the symbols in the rhythm part, training samples can be reused. For example, when learning quarter notes, no matter the melody (pitch), the activation of the quarter note is the same, and therefore, all quarter notes (independently of their pitch) can be used for training.

4) Output: After the FC layers, the next step is to calculate the loss and backpropagate. In the validation and test process after calculating the loss, a threshold of 0,5 (found experimentally) is applied to both outputs (Rhythm and Pitch) in order to increase the performance. The outputs as the ground truth of each music score of each dataset is represented by two binary matrices, one for the rhythm and another for the melody (the pitch). Each part in the horizontal axis is as long as the input image. And the vertical axis is 54 for the melody and 26 for the rhythm. 54 and 26 indicate the number of different possible symbols in the dataset, and the number of different possible pitches (i.e. locations in the staff) of the notes in the music score (See Figure 10). Some symbols have been manually added to make easier the recognition task.

- Epsilon(ε) is used to know where each symbol starts and ends, as it is used in text recognition. This symbol can be seen as a separator. Wherever this symbol is activated, it means that it is not possible to have any other symbol activated as well (see Figure 10c blue marks). This symbol appears in both the rhythm and pitch groundtruthes.
- *No note* is used to indicate that a symbol has not any pitch. This symbol only appears in the pitch groundtruth.

Then these outputs are converted to an array. One with the detections of the rhythm, another for the pitch and the last one with the combination of rhythm and pitch. These arrays will be used to evaluate the method.



(a) Representation of the Output and Ground Truth of Rhythm

(b) Representation of the Output and Ground Truth of Pitch

(c) Real example of Music Score and the corresponding GT in Binary Matrix. First row the music score, Second row the Rhythm Groundtruth and Third row the Pitch Groundtruth

Figure 10. Example of groundtruth representation.

[9]http://pytorch.org/

C. Loss Function

In music, in one instance of time we can find one or more symbols, for example, the case of chords or time signature (see Figure 10c red marks). And the loss function has to be carefully chosen because a multilabel loss function is needed. In other words,the loss function must allow to have more than one activation per time step. Therefore, the softmax cannot be used because it is thought for classification problems with single-label activation. Two different loss functions have been used: $SmoothL1Loss$ (Equation 1) and $MultiLabelSoftMarginLoss$ (Equation 2). The loss is calculated independently. One loss for the rhythm and other for the melody (pitch). Once both losses are calculated, they are summed and backpropagated.

$$SmoothL1Loss(x,y) = \frac{1}{n}\sum \begin{cases} 0.5(x_i - y_i)^2, & \text{if } |x_i - y_i| < 1 \\ |x_i - y_i| - 0.5, & \text{otherwise} \end{cases} \tag{1}$$

$$MultiLabelSoftMarginLoss(x,y) = -\sum_i y[i] \cdot \log\left(\frac{1}{1+e^{-x[i]}}\right) + (1-y[i])\log\left(\frac{e^{-x[i]}}{1+e^{-x[i]}}\right) \tag{2}$$

D. From Synthetic Training to Handwritten Recognition - Data Augmentation

The recognition of handwritten music scores is difficult due to the huge variability in the handwriting styles. Nowadays, there are lots of handwritten music scores in archives that have not been transcribed, and therefore, their music remains locked and hidden to the society. An additional problem to the recognition of handwritten music scores is that there is no labeled data to train. For this reason, two techniques have been used to try to solve this problem. The first is data augmentation and the second is to use printed data as a pretraining. This second option is based on the idea of transfer learning. Transfer Learning is a problem where some knowledge is stored from some problem and this knowledge is applied to other related problem. For example, in [47] the authors train a CNN with synthetic data. Then, they use the network structure and weights to initialize another network and do fine-tunning with a handwritten dataset.

1) Data Augmentation: As far as I know there are no labeled datasets of handwritten scores. For this reason I have manually labelled some handwritten scores and have used data augmentation, by applying some distortions. In this way some different representations of the same music scores can be obtained. Different distortions as dilation, erosion, blur or Gaussian noise (see Figure 11) have been used. Moreover, each staff has been cropped in measures (bar units) and, then, several bar units have been shuffled with others (see Figure 12). Consequently, the number of labeled staffs has been increased.

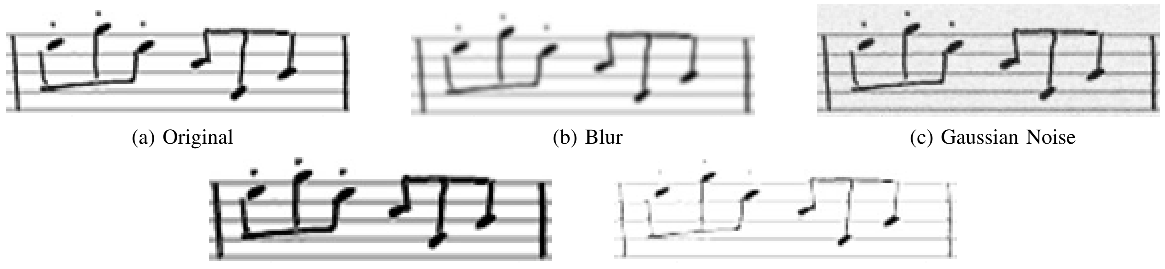

(a) Original (b) Blur (c) Gaussian Noise
(d) Dilation (e) Erode

Figure 11. Some distortions applied in the Handwritten Dataset

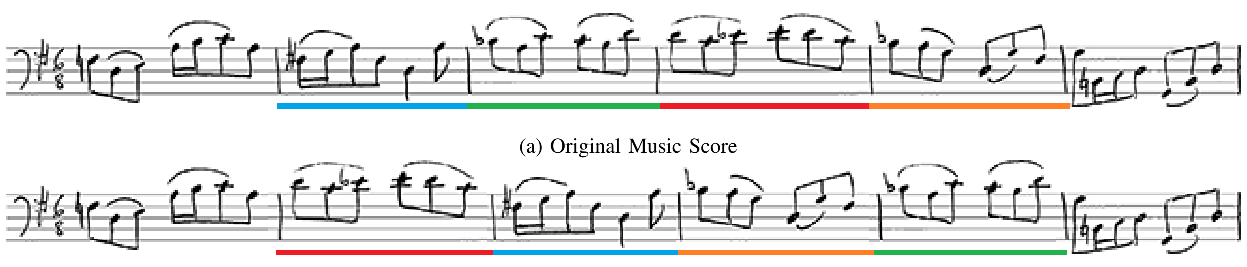

(a) Original Music Score
(b) Shuffled Music Score

Figure 12. Example of handwritten music score with shuffled measures.

2) From Synthetic Training to Handwritten Recognition: The proposed methodology has been first evaluated with synthetic data to demonstrate the suitability of BLTSM to the recognition of music scores. After training the network with the synthetic dataset, the method is evaluated in a real scenario with handwritten music scores. The CVC-MUSCIMA [48] described in section VI-A2 has been used. The main idea is to first train with a synthetic dataset and save the best configuration. Then, the training of the handwritten dataset starts from this baseline and a refinement is done.

VI. Experiments and Results

This Section is devoted to validate the performance of the developed approach and discuss the results. Two datasets are used illustrating the different scenarios.

A. Datasets

1) Printed Dataset: This Synthetic dataset is composed of almost 50000 music scores with 3 different typographies. The dataset corresponds to incipts from the RISM dataset [10]. It is labeled in primitive level. The staffs are divided in 60% (29815) for training, 20% (9939) for validation and 20% (9939) for test.

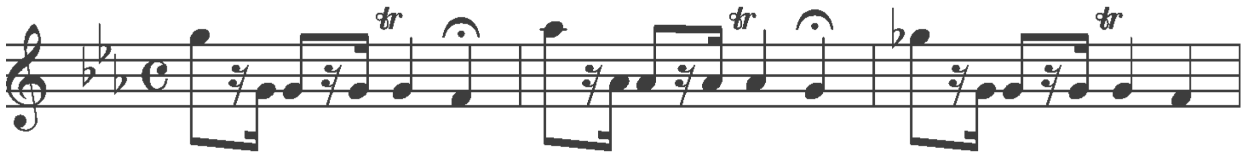


Figure 13. Example of the synthetic dataset.

2) Handwritten Dataset: In order to evaluate how the method works in a handwritten scenario, one page of the CVC-MUSICMA dataset has been used. It is a handwritten dataset that contains 1000 music sheets written by 50 different musicians. This dataset is not labeled. One page with 6 staves has been manually labeled. Each staff has six bar units, so the first and the last have remained fixed, and the remaining 4 bar units have been mixed together. As a result, for each staff, 256 (4^4) different staves have been created. In this way, the number of labelled staves increases from 6 to 1536 labeled staves. These staves are divided in four no shuffled staves for training (1024 shuffled), one for validation (256 shuffled) and one for test (256 shuffled) (see Figure 14). Please note that the bar units are only shuffled within the same staff. This means that the bar units in the test staff have never appeared in the training nor validation sets.

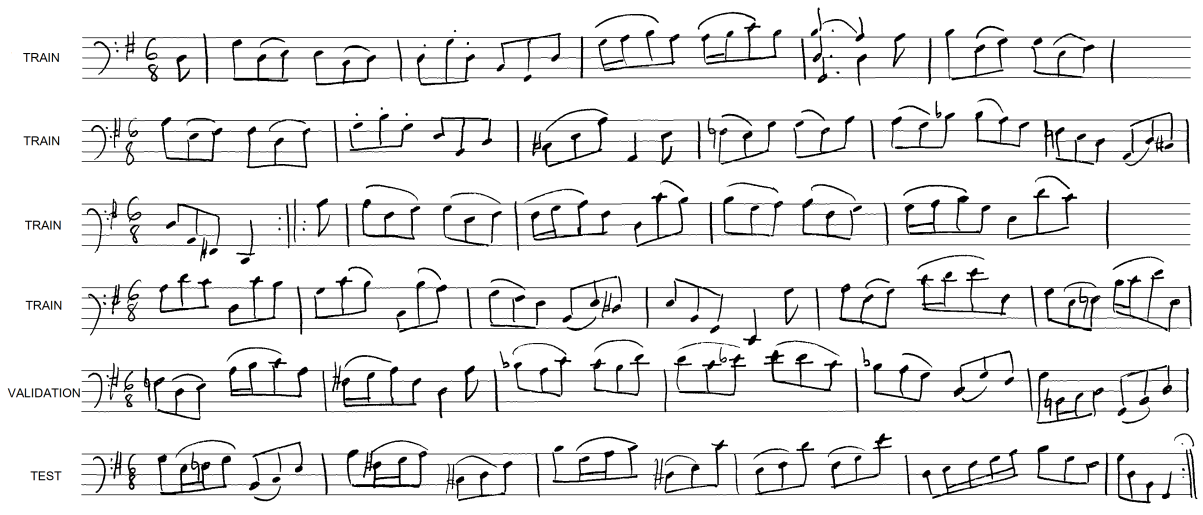


Figure 14. Music score page used in the handwritten experiments (extracted from the CVC-MUSCIMA dataset).

B. Evalutation

The evaluation of OMR is not very well defined. In [45] they evaluate the pitch and the rhythm separating and then evaluating it together, so they are evaluating their method at symbol level. In this work, the three parts will be similarly evaluated. In order to evaluate each of these parts, the outputs produced by the FC layers have been evaluated, a threshold is applied (see more details in the following subsection) and converted into arrays to be evaluated by the metric Symbol Error Rate.

[10]http://www.rism.info/

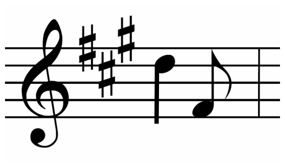


Figure 15. Example of music score

An example of the format of the three output arrays, corresponding to Figure 15, is the following:

- Rhythm: [gClef, accidental sharp, accidental sharp, accidental sharp, quarter note, eight note, bar line]
- Pitch: [noNote, L5, S3, S5, L4, S1, noNote]
- Rhythm+Pitch: [[gClef, noNote], [accidental sharp, L5], [accidental sharp, S3], [accidental sharp, S5], [quarter note, L4], [eight note, S1], [bar line, noNote]]

1) Symbol Error Rate (SER): The metric that has been used to evaluate the method, Symbol Error Rate, is based on the well-known Word Error Rate (WER) metric [49] used in both speech recognition and machine translation. This metric is based on the Levenshtein distance, the difference between them is that the Levenshtein distance calculates the differences at the characters level and the WER does it at the word level. In the case of music scores, given a prediction and a reference array, the SER is defined as the minimum number of insertions, substitutions and deletions to convert one array into the other.

$$SER = \frac{S + D + I}{N} \tag{3}$$

where S is the number of substitutions, D is the number of deletions, I is the number of insertions and N is the number of symbols in reference. Dynamic programming is used to find the minimum value.

$$SER(i, j) = min \begin{cases} SER(i-1, j) + 1 \\ SER(i, j-1) + 1 \\ SER(i-1, j-1) + \Delta(i, j) \end{cases} \tag{4}$$

where $\Delta(i, j)$ is 0 if the symbols $predicted_i$ and $reference_j$ are equals and 1 if the symbols are different.

2) Output's Threshold Evaluation: As explained in previous sections to obtain better results on the output of FC layers, a threshold is applied. This threshold has been experimentally found. The network has been trained with the synthetic data and with different thresholds. The results are given by the test set. Table II shows in the first column the thresholds that have been tested and from the second to the fourth column the results in terms of Symbol Error Rate the rhythm, pitch and both, respectively. As we can see, the best threshold is 0,5 even though 0,4 also obtains very similar results. Figure 16 shows the results of Table II in a more visual way.

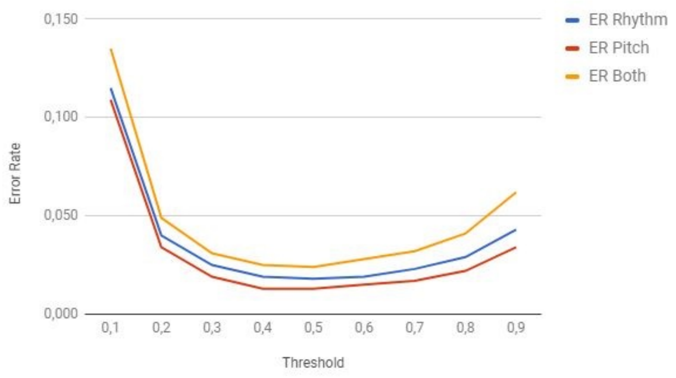


Figure 16. Evaluation of the best threshold in terms of Error Rate: Rhythm, Pitch and Rhythm+Pitch.

Table II
SHOWS THE EXPERIMENTALLY RESULTS ON SEARCHING THE BEST THRESHOLD. THIS TABLE REFERS TO THE PLOT SHOWN IN FIGURE 16

Threshold	**Rhythm - Symbol Error Rate**	**Pitch - Symbol Error Rate**	**Rhythm+Pitch - Symbol Error Rate**
0.1	0.115	0.109	0.135
0.2	0.040	0.034	0.049
0.3	0.025	0.019	0.031
0.4	0.019	0.013	0.025
0.5	**0.018**	**0.013**	**0.024**
0.6	0.019	0.015	0.028
0.7	0.023	0.017	0.032
0.8	0.029	0.022	0.041
0.9	0.043	0.034	0.062

C. Printed Dataset

The printed dataset has been used for different experiments, with different thresholds, different loss functions. It has been used to trained the LSTM and the BLSTM. Next, Table III shows the average and standard deviation applying LSTM and BLSTM. The first column shows the loss function and the network that has been used, the second column the results in term of error rate of the rhythm, the third column the results in term of error rate of the pitch and the last column the results in term of error rate of the rhythm and pitch jointly.

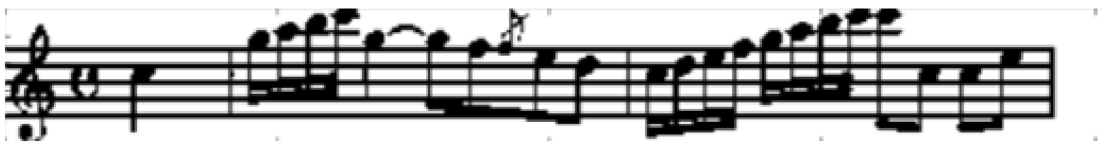
Table III
RESULTS USING LSTM AND BLSTM. ALL RESULTS ARE BETWEEN [0-1]. THE FIRST NUMBER IS THE MEAN OF THE FIVE EXECUTIONS AND THE NUMBER BETWEEN PARENTHESIS IS THE STANDARD DEVIATION

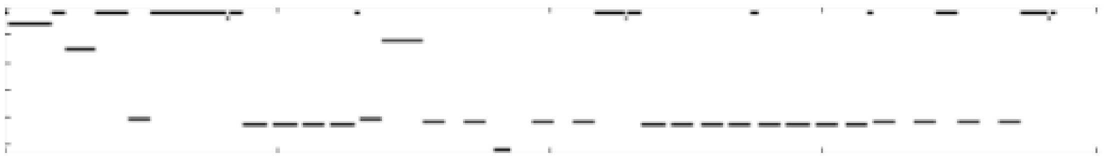
	Rhythm - Symbol Error rate	Pitch - Symbol Error Rate	Rhythm+Pitch - Symbol Error Rate
LSTM - Smooth L1 Loss	0.326 (± 0.007)	0.293 (± 0.008)	0.426 (± 0.009)
BLSTM - Smooth L1 Loss	**0.020** (± 0.001)	**0.015** (± 0.001)	**0.028** (± 0.002)
LSTM - Multi Label Soft Margin Loss	0.431 (± 0.017)	0.567 (± 0.051)	0.747 (± 0.063)
BLSTM - Multi Label Soft Margin Loss	0.027 (± 0.002)	0.023 (± 0.002)	0.036 (± 0.003)

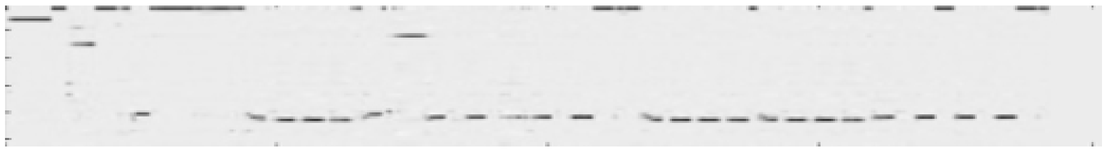
Table III shows that the BLSTM produces better results. As expected, the main reason is that the network really benefits from the context in both directions. One can conclude that for printed music scores written in a single staff the BLSTM works very well. Concretely, the Smooth L1 function only has 1.5% of SER when recognizing the pitch, 2% of SER when recognizing the rhythm and 2.8% when recognizing the pitch an rhythm both jointly.

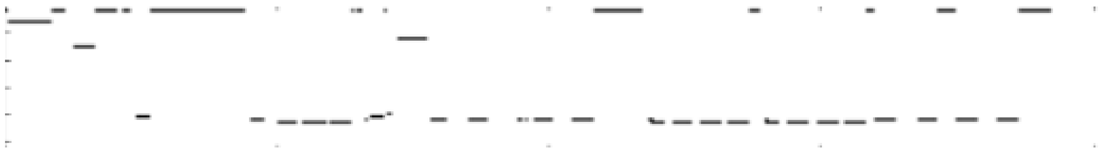
In Figures 17 and 18 we can see some qualitative results. In the first subfigure we can see the input of the LSTM or BLSTM respectively. The second subfigure is the Rhythm Ground Truth and the third subfigure is the LSTM's/BLSTM's output. Fourth subfigure is the same output but only activating the positions with confidence higher than 50%. In the fifth subfigure we can see the Melody Ground Truth. The sixth subfigure is the LSTM/BLSTM's output and the seventh subfigure the same output but only activating the positions with confidence higher than 50%.

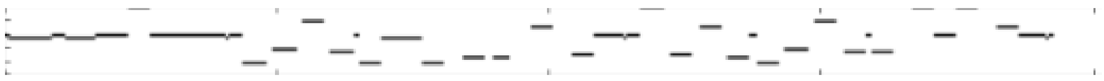
In many cases the network confuses very similar symbols, such as the eighth note with the sixteenth note. Or, in the case of the melody, the network confuses very close pitches, such as correlative notes.

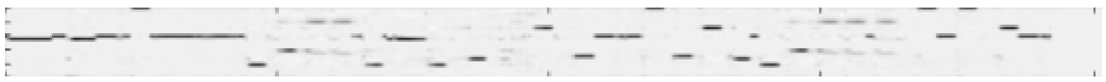

(a) Input image


(b) Rhythm ground truth


(c) Rhythm Output


(d) Rhythm Output with threshold higher than 50%


(e) Melody ground truth


(f) Melody Output

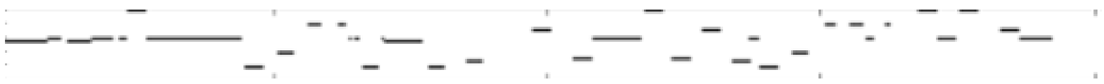
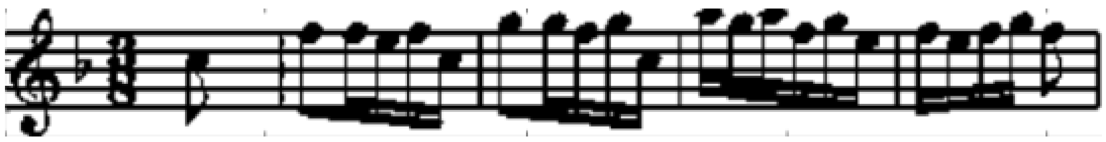
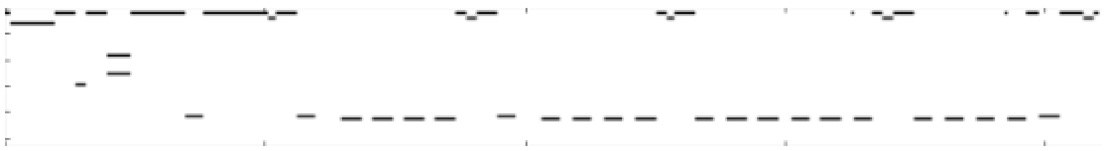
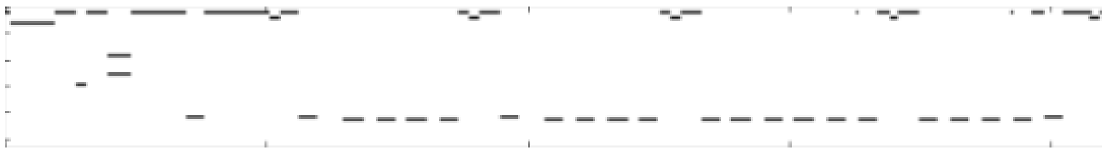

(g) Melody Output with threshold higher than 50%

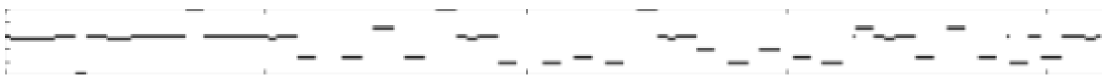
Figure 17. Qualitative Results Example using LSTM.


(a) Input image


(b) Rhythm ground truth

(c) Rhythm Output


(d) Rhythm Output with threshold higher than 50%


(e) Melody ground truth

(f) Melody Output

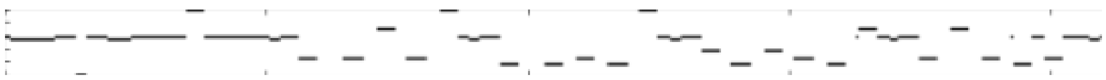

(g) Melody Output with threshold higher than 50%

Figure 18. Qualitative Results Example using BLSTM.

D. Handwritten Dataset

This work aims to go beyond the synthetic printed scores. For this reason, some handwritten scores were labeled to try to transfer the good performance of the synthetic printed scores to the handwritten ones. Table IV shows the average and standard deviation applying the handwritten dataset as test. The first column shows which dataset has been used for training and whether some distortion has been applied to the dataset (i.e. data augmentation). The second column shows the results in term of the error rate of the rhythm, the third column shows the error rate of the pitch and the last column the results in term of the error rate of the rhythm and pitch jointly.

Table IV
RESULTS USING THE HANDWRITTEN DATASET AS TEST. ALL RESULTS ARE BETWEEN [0-1]. THE FIRST NUMBER IS THE MEAN OF THE FIVE EXECUTIONS AND THE NUMBER BETWEEN PARENTHESIS IS THE STANDARD DEVIATION.

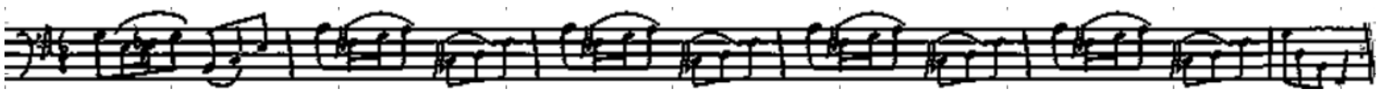
Train	Rhythm - Symbol Error Rate	Pitch - Symbol Error Rate	Rhythm+Pitch - Symbol Error Rate
Printed	0.832 (± 0.002)	0.713 (± 0.001)	0.952 (± 0.001)
Handwritten	1 (± 0)	1 (± 0)	1 (± 0)
Printed+Handwritten	0.446 (± 0.037)	0.512 (± 0.034)	0.693 (± 0.058)
Printed (data Aug)	0.612 (± 0.001)	0.644 (± 0.001)	0.813 (± 0.002)
Handwritten (data Aug)	1 (± 0)	1 (± 0)	1 (± 0)
Printed (data Aug)+Handwritten	**0.431** (± 0.028)	**0.473** (± 0.039)	**0.653** (± 0.025)
Printed+Handwritten (data Aug)	0.483 (± 0.013)	0.542 (± 0.040)	0.735 (± 0.037)
Printed (data Aug)+Handwritten (data Aug)	0.453 (± 0.028)	0.477 (± 0.050)	0.684 (± 0.050)

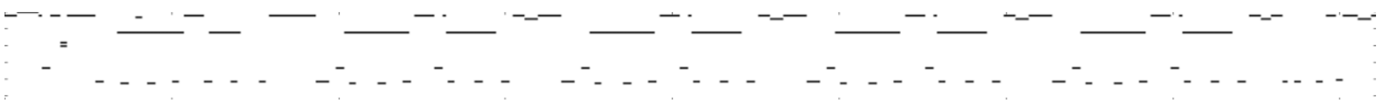
From the table, one can observe that the performance when training with only printed or handwritten music scores is very low (the error rate is 95% and 100% respectively). This clearly means that, on one hand, the printed scores are not representative of the appearance of handwritten symbols; and on the other hand, the handwritten music scores that have been labeled are insufficient. Secondly, it can be observed that when printed scores are used as pretraining, and then the handwritten ones are used for refining, the overall performance increases (see the third row, with an error rate close 70%). Similarly, the use of data augmentation also improves the overall results. The best configuration is the use of printed scores with data augmentation and handwritten scores for fine-tuning (the SER of the rhythm, pitch and joint rhythm+pitch is 43%, 47% and 65% respectively).

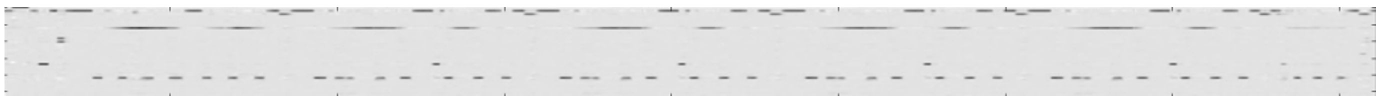
It must to be said that, even though the final results are not impressive, the symbol error rate has decreased from 100% to 65% using printed scores with data augmentation as training and refining with few handwritten scores. It seems that the data augmentation is more effective in synthetic musical scores rather than in handwritten ones. However, it seems at first sight that in Figure 11, the data augmentation in handwritten scores seems fairly realistic. One conclusion is that better data augmentation techniques must be explored so that the resulting scores are really useful in the training step.

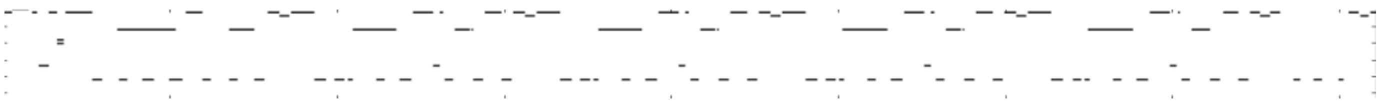
In any case, it should be remarked that, with only 6 labelled handwritten staves, the symbol error has significantly decreased. For example, if we compare with the printed training (first row), the SER has decreased 30 points (from 95% to 65%), when considering the rhythm and melody as a whole. If we consider them separately, the rhythm has dropped 40 points (from 83% to 43%) and the melody 24 points (from 71% to 47%). This significant decrease is encouraging. This trend indicates that, with more labeled handwritten scores, the results would be much better. Indeed, the loss in train was decreasing and the test loss was maintained or raising, which is a sign of overfitting (and one of the causes of overfitting is a lack of data).

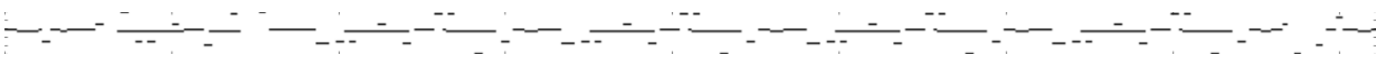
In Figure 19 we can see some qualitative results. In Figure 19a we can see the handwritten input of the BLSTM. Figure 19b is the Rhythm Ground Truth and Figure 19c is the BLSTM's output. Figure 19d is the same output but only activating the positions with confidence higher than 50%. In Figure 19e we can see the Melody Ground Truth. Figure 19f is the BLSTM's output and Figure 19g the same output but only activating the positions with confidence higher than 50%.

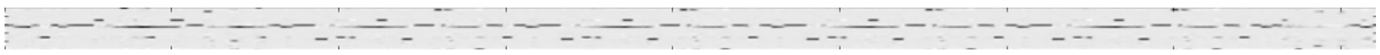

(a) Input image


(b) Rhythm ground truth


(c) Rhythm Output


(d) Rhythm Output with threshold higher than 50%


(e) Melody ground truth


(f) Melody Output

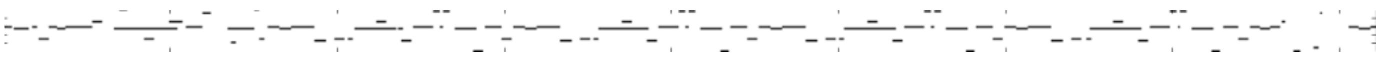
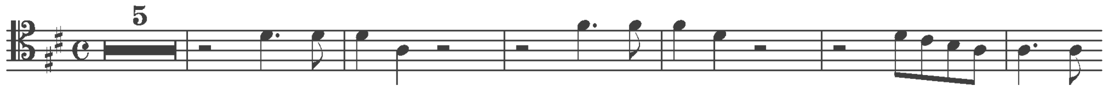

(g) Melody Output with threshold higher than 50%

Figure 19. Qualitative Results Example of BLSTM with a Handwritten Music Score.

E. Comparison with a commercial OMR software

The proposed method has been compared with PhotoScore, a commercial OMR software able to recognize printed and also handwritten music scores. Figures 20 and 21, show qualitative results from both approaches (printed and handwritten). As it can be noticed, PhotoScore performs very well in easy parts, whereas its performance considerably decreases in case of complex compound music notes. In this aspect, this approach is much more stable. In any case, it must be said that this comparison is not completely fair. PhotoScore has some features to improve its performance that are not considered in this method. First, PhotoScore is a complete OMR system that recognizes the whole score, which probably uses huge training data to deal with the variability in the handwriting style. Given that it recognizes all music symbols (including clefs, accidentals and rests), it probably uses syntactic rules for validation. For instance, the system can recognize the time signature and then validate the amount of music notes at each bar unit (which is used to solve ambiguities). Contrary, in this work, no syntactic rules have been applied.


(a) Original Image

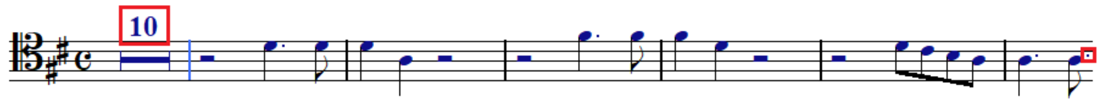
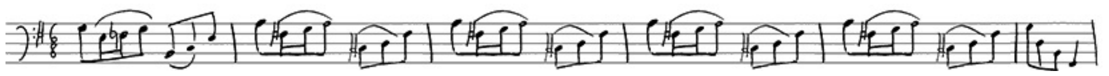

(b) Visual Result of PhotoScore

Figure 20. Recognition of a Printed image using the PhotoScore Commercial OMR software.

Figure 20 shows that, even with a very simple music score, PhotoScore has produced two errors. First, it has confused the time of silence (5-10), and second it has added a duration dot at the end. It must be said that the method proposed in this work has correctly recognized all the music symbols.


(a) Original Image

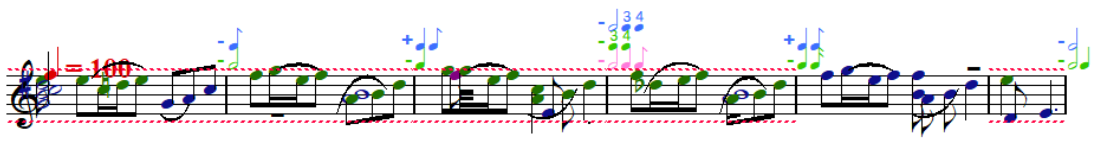

(b) Visual Result of PhotoScore

Figure 21. Recognition of a Handwritten image of PhotoScore Commercial OMR

Figure 21 shows the result of recognizing a handwritten music score. It can be observed that the performance of PhotoScore is very low. It must be highlighted that, from the second to the fifth bar units, the music symbols are exactly the same (the same measure is repeated). However, each bar unit is recognized differently. In the case of our BLSTM-based method, we obtain a 44.78% of SER in rhythm, a 37.31% of SER in melody and a 50.75% in both together. Figure 22 shows a visual output of the network, which has been manually drawn. The output array of the transcription is shown next.

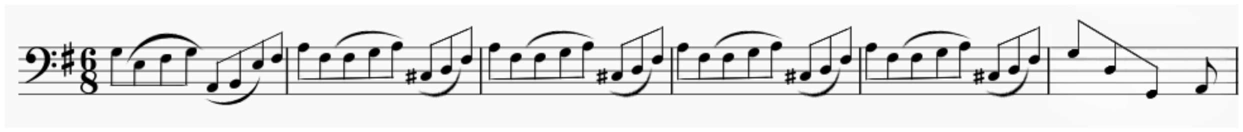


Figure 22. My visual output of Figure 21 a

Transcription of Figure 22 divided by bar units:

1) [['fClef', 'noNote'], ['accidentalSharp', 'L4'], ['timeSig8', 'noNote'], ['timeSig6', 'noNote'], ['8thNote', 'S4'], ['slurOrTie', 'noNote'], ['8thNote', 'S3'], ['8thNote', 'L4'], ['8thNote', 'S4'], ['8thNote', 'S1'], ['slurOrTie', 'noNote'], ['8thNote', 'L2'], ['8thNote', 'S3'], ['8thNote', 'L4'],
2) ['barlineSingle', 'noNote'], ['8thNote', 'L5'], ['8thNote', 'L4'], ['8thNote', 'L4'], ['slurOrTie', 'noNote'], ['8thNote', 'S4'], ['8thNote', 'L5'], ['accidentalSharp', 'S2'], ['8thNote', 'S2'], ['slurOrTie', 'noNote'], ['8thNote', 'L3'], ['8thNote', 'L4'],
3) ['barlineSingle', 'noNote'], ['8thNote', 'L5'], ['8thNote', 'L4'], ['8thNote', 'L4'], ['slurOrTie', 'noNote'], ['8thNote', 'S4'], ['8thNote', 'L5'], ['accidentalSharp', 'S2'], ['8thNote', 'S2'], ['slurOrTie', 'noNote'], ['8thNote', 'L3'], ['8thNote', 'L4'],

4) ['barlineSingle', 'noNote'], ['8thNote', 'L5'], ['8thNote', 'L4'], ['8thNote', 'L4'], ['slurOrTie', 'noNote'], ['8thNote', 'S4'], ['8thNote', 'L5'], ['accidentalSharp', 'S2'], ['8thNote', 'S2'], ['slurOrTie', 'noNote'], ['8thNote', 'L3'], ['8thNote', 'L4'],
5) ['barlineSingle', 'noNote'], ['8thNote', 'L5'], ['8thNote', 'L4'], ['8thNote', 'L4'], ['slurOrTie', 'noNote'], ['8thNote', 'S4'], ['8thNote', 'L5'], ['accidentalSharp', 'S2'], ['8thNote', 'S2'], ['slurOrTie', 'noNote'], ['8thNote', 'L3'], ['8thNote', 'L4'],
6) ['barlineSingle', 'noNote'], ['8thNote', 'S4'], ['8thNote', 'L3'], ['8thNote', 'L1'], ['8thNote', 'S1'], ['barlineSingle', 'noNote']]

It must be said that the BLSTM approach does not use grammars or rules to check that the sum of beats must be equal to the time signature (in the bar unit we have one extra note). However, it must to be said that this method, unlike PhotoScore, is consistent when recognizing the same measures (from 2 to 5 measures). Concretely, the melody is closer to the groundtruth than the rhythm. One reason could be the fact that the number of eight notes and natural accidentals is extremely limited in the training set (see Figure 14).

VII. Conclusions and Future Work

In this work, an optical music recognition method has been proposed. This method is based on following a sequence path with LSTM and BLSTM Recurrent Neural Networks.

The first conclusion is that a sequential methodology is suitable for recognizing single staff music scores, given that the high performance in printed scores supports this hypothesis. However, it must to be noted that when dealing with more complex music scores, for example polyphonic scores (where the music is written in more than one staff), a sequential method could not be appropriate.

Secondly, when the handwritten scores have been trained, results are moderate. However, there were very few labeled music scores (only 5 staves for train and validation). For this reason, some data augmentation techniques have been applied to increase the volume of data, showing that the performance improves. Nevertheless, the inclusion of more labelled handwritten scores (instead of so many printed ones), would obtain much better results.

Future work will be focused on investigating more realistic data augmentation methods for handwritten scores, as well as transfer learning methods (for example, freezing weights in LSTMs's to recognize handwritten music scores). Finally, I would also like to apply music rules or semantics in order to solve ambiguities, and also, investigate more suitable techniques for recognizing complex polyphonic music scores. For example, CNNs could be used as feature extractor, or attention models to extract information only to the relevant areas. In addition, graphs could be a suitable technique in music because all the symbols in a page can be represented as a graph where the nodes are the graphical primitives. In a more technical aspect, I would like to convert the output of the architecture into a MIDI file, able to be listened, or to convert it into a sheet format as PhotoScore does.

Acknowledgment

Firstly, I would like to express my sincere gratitude to my advisors Dr. Alicia Fornés and Dr. Jorge Calvo for the continuous support of my Msc thesis, for their patience and motivation. My sincere thanks also goes to Pau Riba for helping me whenever I have needed him. Additionally, I would like to thank Albert Berenguel and Pau Rodríguez for them several suggestions during the project. Last but not the least, I would like to thank my family, friends and CVC's colleagues for supporting me spiritually throughout doing this project and my life in general.

References

[1] A. Rebelo, I. Fujinaga, F. Paszkiewicz, A. R. S. Marçal, C. Guedes, and J. S. Cardoso, "Optical music recognition: state-of-the-art and open issues.," *IJMIR*, vol. 1, no. 3, pp. 173–190, 2012.
[2] D. Bainbridge and T. Bell, "The challenge of optical music recognition," *Computers and the Humanities*, vol. 35, no. 2, pp. 95 – 121, 2001.
[3] A. Fornés and G. Sánchez, "Analysis and recognition of music scores," in *Handbook of Document Image Processing and Recognition*, pp. 749–774, Springer-Verlag London, 2014.
[4] A. Graves and J. Schmidhuber, "Offline handwriting recognition with multidimensional recurrent neural networks," in *Advances in neural information processing systems*, pp. 545–552, 2009.
[5] J. A. Burgoyne, Y. Ouyang, T. Himmelman, J. Devaney, L. Pugin, and I. Fujinaga, "Lyric extraction and recognition on digital images of early music sources," in *Proceedings of the 10th International Society for Music Information Retrieval Conference*, pp. 723–727, 2009.
[6] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 6645–6649, IEEE, 2013.
[7] A. Fornés, *Writer Identification by a Combination of Graphical Features in the Framework of Old Handwritten Music Scores*. PhD dissertation, Universitat Autònoma de Barcelona, 2009.
[8] F. Pedersoli and G. Tzanetakis, "Document segmentation and classification into musical scores and text," *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 19, no. 4, pp. 289–304, 2016.
[9] A. Fornés, V. C. Kieu, M. Visani, N. Journet, e. B. Dutta, Anjan", and J.-M. Ogier, *The ICDAR/GREC 2013 Music Scores Competition: Staff Removal*, pp. 207–220. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014.
[10] A. Fornés, J. Lladós, G. Sánchez, and D. Karatzas, "Rotation invariant hand drawn symbol recognition based on a dynamic time warping model," *IJDAR*, vol. 13, no. 3, pp. 229–241, 2010.

[11] S. Escalera, A. Fornés, O. Pujol, P. Radeva, G. Sánchez, and J. Lladós, “Blurred Shape Model for binary and grey-level symbol recognition,” *Pattern Recognition Letters*, vol. 30, no. 15, pp. 1424–1433, 2009.
[12] A. Rebelo, G. Capela, and J. S. Cardoso, “Optical recognition of music symbols: A comparative study,” *IJDAR*, vol. 13, no. 1, pp. 19–31, 2010.
[13] A. Rebelo, J. Tkaczuk, R. Sousa, and J. S. Cardoso, “Metric learning for music symbol recognition,” in *2011 10th International Conference on Machine Learning and Applications and Workshops*, vol. 2, pp. 106–111, Dec 2011.
[14] A. Baró, P. Riba, and A. Fornés, “Towards the recognition of compound music notes in handwritten music scores,” in *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pp. 465–470, Oct 2016.
[15] B. Coüasnon and B. Rétif, “Using a grammar for a reliable full score recognition system,” in *ICMC*, 1995.
[16] L. Pugin, “Optical music recognitoin of early typographic prints using hidden markov models,” in *ISMIR*, 2006.
[17] L. Pugin, J. A. Burgoyne, and I. Fujinaga, “Map adaptation to improve optical music recognition of early music documents using hidden markov models,” in *ISMIR*, 2007.
[18] J. C. Pinto, P. Vieira, and J. M. Sousa, “A new graph-like classification method applied to ancient handwritten musical symbols,” *Document Analysis and Recognition*, vol. 6, no. 1, pp. 10–22, 2003.
[19] T. Matsushima, S. Ohteru, and S. Hashimoto, “An integrated music information processing system: Psb-er,” *Proceedings of the international computer music conference*, pp. 191–198, 1989.
[20] K. Ng, *Music Manuscript Tracing*, pp. 330–342. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002.
[21] H. Miyao and M. Maruyama, “An online handwritten music symbol recognition system,” *International Journal of Document Analysis and Recognition (IJDAR)*, vol. 9, pp. 49–58, Mar 2007.
[22] J. Calvo-Zaragoza, J.; Oncina, “Recognition of pen-based music notation with probabilistic machines,” in *Proceedings of the 7th International Workshop on Machine Learning and Music*, (Barcelona, Spain), 2014.
[23] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.
[24] W. S. McCulloch and W. Pitts, “A logical calculus of the ideas immanent in nervous activity,” *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.
[25] D. O. Hebb, *The organization of behavior: A neuropsychological theory*. New York: Wiley, June 1949.
[26] F. Rosenblatt, “The perceptron: A probabilistic model for information storage and organization in the brain,” *Psychological Review*, vol. 65, no. 6, pp. 386–408, 1958.
[27] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, “Learning representations by back-propagating errors,” *Nature*, vol. 323, pp. 533–536, Oct. 1986.
[28] G. E. Hinton, S. Osindero, and Y.-W. Teh, “A fast learning algorithm for deep belief nets,” *Neural Comput.*, vol. 18, pp. 1527–1554, July 2006.
[29] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, “Greedy layer-wise training of deep networks,” in *Advances in Neural Information Processing Systems 19* (P. B. Schölkopf, J. C. Platt, and T. Hoffman, eds.), pp. 153–160, MIT Press, 2007.
[30] M. aurelio Ranzato, C. Poultney, S. Chopra, and Y. L. Cun, “Efficient learning of sparse representations with an energy-based model,” in *Advances in Neural Information Processing Systems 19* (P. B. Schölkopf, J. C. Platt, and T. Hoffman, eds.), pp. 1137–1144, MIT Press, 2007.
[31] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *CoRR*, vol. abs/1409.1556, 2014.
[32] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems 25* (F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, eds.), pp. 1097–1105, Curran Associates, Inc., 2012.
[33] J. Johnson, A. Karpathy, and F. Li, “Densecap: Fully convolutional localization networks for dense captioning,” *CoRR*, vol. abs/1511.07571, 2015.
[34] J. Dai, Y. Li, K. He, and J. Sun, “R-FCN: object detection via region-based fully convolutional networks,” *CoRR*, vol. abs/1605.06409, 2016.
[35] R. Pascanu, Ç. Gülçehre, K. Cho, and Y. Bengio, “How to construct deep recurrent neural networks,” *CoRR*, vol. abs/1312.6026, 2013.
[36] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Comput.*, vol. 9, pp. 1735–1780, Nov. 1997.
[37] A. Graves, “Generating sequences with recurrent neural networks,” *CoRR*, vol. abs/1308.0850, 2013.
[38] J. Li, Y. Wei, X. Liang, J. Dong, T. Xu, J. Feng, and S. Yan, “Attentive contexts for object detection,” *CoRR*, vol. abs/1603.07415, 2016.
[39] V. Mnih, N. Heess, A. Graves, and K. Kavukcuoglu, “Recurrent models of visual attention,” *CoRR*, vol. abs/1406.6247, 2014.
[40] K. Xu, J. Ba, R. Kiros, K. Cho, A. C. Courville, R. Salakhutdinov, R. S. Zemel, and Y. Bengio, “Show, attend and tell: Neural image caption generation with visual attention,” *CoRR*, vol. abs/1502.03044, 2015.
[41] A. Owens, P. Isola, J. H. McDermott, A. Torralba, E. H. Adelson, and W. T. Freeman, “Visually indicated sounds,” *CoRR*, vol. abs/1512.08512, 2015.
[42] Y. Aytar, C. Vondrick, and A. Torralba, “Soundnet: Learning sound representations from unlabeled video,” *CoRR*, vol. abs/1610.09001, 2016.
[43] Y. C. Sübakan and P. Smaragdis, “Diagonal rnns in symbolic music modeling,” *CoRR*, vol. abs/1704.05420, 2017.
[44] V. Kalingeri and S. Grandhe, “Music generation with deep learning,” *CoRR*, vol. abs/1612.04928, 2016.
[45] E. van der Wel and K. Ullrich, “Optical music recognition with convolutional sequence-to-sequence models,” *CoRR*, vol. abs/1707.04877, 2017.
[46] C. Olah, “Understanding lstm networks.” http://colah.github.io/posts/2015-08-Understanding-LSTMs/. Accessed: 2017-08-25.
[47] Y. Tang, L. Peng, Q. Xu, Y. Wang, and A. Furuhata, “Cnn based transfer learning for historical chinese character recognition,” in *2016 12th IAPR Workshop on Document Analysis Systems (DAS)*, pp. 25–29, April 2016.
[48] A. Fornés, A. Dutta, A. Gordo, and J. Lladós, “Cvc-muscima: A ground truth of handwritten music score images for writer identification and staff removal,” *International Journal on Document Analysis and Recognition - IJDAR*, vol. 15, pp. 1–9, 09 2011.
[49] V. Frinken and H. Bunke, “Continuous handwritten script recognition,” in *Handbook of Document Image Processing and Recognition*, pp. 391–425, Springer, 2014.