Universitat Autònoma de Barcelona

Reading Music Systems: From Deep Optical Music Recognition to Contextual Methods

Arnau Baró Mas

Reading Music Systems:
From Deep Optical Music Recognition to Contextual Methods

A dissertation submitted by **Arnau Baró Mas** at Universitat Autònoma de Barcelona to fulfil the degree of **Doctor of Philosophy**.

Bellaterra, September 2022

Director	**Dra. Alicia Fornés** Universitat Autònoma de Barcelona Centre de Visió per Computador
Thesis Committee	**Dr. José Manuel Iñesta** Universitat d'Alacant Llenguatges i sistemes informàtics
	Dr. Oriol Ramos Universitat Autònoma de Barcelona Centre de Visió per Computador
	Dr. Marçal Rossinyol AllRead Machine Learning Technologies Deep Learning

This document was typeset by the author using LaTeX 2$_\varepsilon$.

The research described in this book was carried out at the Computer Vision Center, Universitat Autònoma de Barcelona.

ISBN Pending

Printed by Ediciones Gráficas Rey, S.L.

Knowledge speaks, but wisdom listens.

– Jimi Hendrix

Lo importante es el camino y en él
caer, levantarse, insistir, aprender.

– Mägo de Oz

Cremant etapes com qui té pressa per fer.

– La Pegatina

A la familia, amics, i a la Sara ...

Agraïments

I get by with a little help from my friends.

– John Lennon

A Eivissa en les últimes hores del dia, contemplant com el sol es va ponent, arribo a l'ocàs d'aquesta tesi.

Durant els sis anys i escaig que he passat al Centre de Visió per Computador, des dels meus inicis en el món de la recerca, he creuat el meu camí amb diverses persones a les quals hauria d'agrair quelcom per ajudar-me a estar on soc ara. Tant als companys que hi són com els que ja han marxat del centre, des del personal de recepció, màrqueting, fins a serveis informàtics, entre altres. Als companys del grup d'anàlisi de documents, en especial al Jialuo Chen, al Lei Kang i al Pau Torras. M'agradaria fer una especial esmena a aquells amb els que he compartit tants cafès en els quals han sorgit diverses idees o simplement hem fet petar la xerrada, l'Asma, l'Albert, la Carola, el Pep, el Pau Rodríguez, l'Edgar, el Xavier. I sobretot, el meu més sincer agraïment, al Pau Riba, que des del primer dia en què vaig arribar al CVC ha estat disposat a ajudar-me en tot el que he necessitat, fins i tot en marxar.

També vull agrair a l'Alicia Fornés, la directora d'aquesta tesi, haver confiat en mi quan tan sols era un estudiant de grau d'enginyeria informàtica per realitzar en aquell moment el que era el treball final de grau. Que seguidament passarà a ser un treball final de màster i finalment una tesi de doctorat. Sense la teva passió per l'ensenyament, la recerca en visió per computador i la música, aquesta tesi no hauria estat possible.

No vull acabar aquestes línies sense recordar-me dels amics de sempre. Aquells que són imprescindibles, els que estan quan més fan falta, els que t'ajuden a desconnectar quan els experiments no funcionen o els resultats no són els esperats. Gràcies, Cristina, Esther, Francesca, Meritxell i Patricia.

A la meva dona, la Sara, no tinc paraules per agrair-li tot el que m'agradaria. Els nostres camins s'han creuat en el millor moment. Segurament és la persona que més ha sofert aquesta tesi, sobretot aquesta última etapa. Gràcies per estar

sempre al meu costat, gràcies per la teva paciència i empatia. T'estimo.

Finalment, però no menys important vull agrair als meus pares, al Pere i a la Silvia, i al meu germà, Pau, per haver-me recolzat tots aquests anys en totes les decisions que he pres fins a finalitzar aquesta etapa.

Portinatx, 18 d'agost de 2022

Abstract

The transcription of sheet music 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.

Document analysis is the subject that deals with the extraction and processing of documents through image and pattern recognition. It is a branch of computer vision. Taking music scores as source, the field devoted to address this task is known as Optical Music Recognition (OMR). Typically, an OMR system takes an image of a music score and automatically extracts its content into some symbolic structure such as MEI or MusicXML.

In this dissertation, we have investigated different methods for recognizing 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. These methods are based in two different methodologies. On the one hand, we present two methods based on Recurrent Neural Networks, in particular, the Long Short-Term Memory Neural Network. On the other hand, a method based on Sequence to Sequence models is detailed.

Music context is needed to improve the OMR results, just like language models and dictionaries help in handwriting recognition. For example, syntactical rules and grammars could be easily defined to cope with the ambiguities in the rhythm. In music theory, for example, the time signature defines the amount of beats per bar unit. Thus, in the second part of this dissertation, different methodologies have been investigated to improve the OMR recognition. We have explored three different methods: (a) a graphic tree-structure representation, Dendrograms, that joins, at each level, its primitives following a set of rules, (b) the incorporation of Language Models to model the probability of a sequence of tokens, and (c) graph neural networks to analyze the music scores to avoid meaningless relationships between music primitives.

Finally, to train all these methodologies, and given the method-specificity of the datasets in the literature, we have created four different music datasets. Two

of them are synthetic with a modern or old handwritten appearance, whereas the other two are real handwritten scores, being one of them modern and the other old.

Keywords – Computer Vision, Pattern Recognition, Document Image Analysis and Recognition, Deep Neural Networks, Optical Music Recognition, Handwritten Music Recognition, Music Context.

Resum

La transcripció de partitures a algun format llegible per un ordinador pot realitzar-se manualment. No obstant això, la complexitat de la notació musical condueix inevitablement a un enutjós programari d'edició de partitures, la qual cosa fa que tot el procés sigui molt lent i propens a errors. Per això, els sistemes de transcripció automàtica de documents musicals són eines interessants.

L'anàlisi de documents és el camp que tracta l'extracció i el processament de documents mitjançant el reconeixement d'imatges i patrons. Aquest és una branca de la visió per computador. Sent les partitures musicals el document a analitzar, el camp dedicat a abordar aquesta tasca es coneix com a reconeixement òptic de música (OMR). Normalment, un sistema OMR pren una imatge d'una partitura i extreu automàticament el seu contingut a alguna estructura simbòlica com MEI o MusicXML.

En aquesta tesi, hem investigat diferents mètodes per reconèixer símbols musicals d'una sola línia de pentagrama (partitures de violí, flauta, etc.), de la mateixa manera que la majoria de les recerques sobre el reconeixement de text se centren en el reconeixement de les paraules que apareixen en una imatge d'una línia de text. Aquests mètodes es basen en dues metodologies diferents. D'una banda, presentem dos mètodes basats en Xarxes Neuronals Recurrents, en particular la xarxa neuronal; *Long Short-Term Memory*. D'altra banda, es detalla un mètode basat en *Sequence to Sequence*.

El context musical és necessari per a millorar els resultats d'OMR, igual que els models lingüístics i els diccionaris ajuden en el reconeixement de text. Per exemple, es podrien definir fàcilment regles sintàctiques i gramàtiques per a fer front a les ambigüitats del ritme. En teoria musical, el compàs defineix la quantitat de temps per unitat de compàs. Així, en la segona part d'aquesta dissertació s'han investigat diferents metodologies per a millorar el reconeixement dels mètodes d'OMR. Hem explorat tres mètodes diferents: (a) una representació gràfica en forma d'arbre en la qual cada nivell uneix les primitives seguint un conjunt de regles, és el que es denomina Dendrogrames, (b) la incorporació de Models de Llenguatge per modelar la probabilitat d'una seqüència de tokens i (c) xarxes neuronals basades en grafs per analitzar les partitures per a evitar relacions sense sentit entre les primitives

musicals.

Finalment, per a entrenar totes aquestes metodologies i donada l'especificitat de les bases de dades segons els mètodes a usar a la literatura, hem creat quatre conjunts de dades musicals diferents. Dos d'ells són sintètics amb aparença moderna o manuscrita antiga i els altres dos són manuscrits reals, un d'ells modern i l'altre antic.

Paraules Clau – Visió per Computador, Reconeixement de Patrons, Anàlisis i Reconeixement d'Imatges en Documents, Xarxes Neuronals Profundes, Reconeixement Òptic de música, Reconeixement de Música Manuscrita, Context Musical.

Resumen

La transcripción de partituras a algún formato legible por un ordenador puede realizarse manualmente. Sin embargo, la complejidad de la notación musical conduce inevitablemente a un engorroso software de edición de partituras, lo que hace que todo el proceso sea muy lento y propenso a errores. Por ello, los sistemas de transcripción automática de documentos musicales son herramientas interesantes.

El análisis de documentos es el campo que trata la extracción y el procesamiento de documentos mediante el reconocimiento de imágenes y patrones. Este es una rama de la visión por computador. Siendo las partituras musicales el documento a analizar, el campo dedicado a abordar esta tarea se conoce como reconocimiento óptico de música (OMR). Normalmente, un sistema OMR toma una imagen de una partitura y extrae automáticamente su contenido a alguna estructura simbólica como MEI o MusicXML.

En esta tesis, hemos investigado diferentes métodos para reconocer los símbolos musicales de una sola línea de pentagrama (partituras de violín, flauta, etc.), de la misma manera que la mayoría de las investigaciones sobre el reconocimiento de texto se centran en el reconocimiento de las palabras que aparecen en una imagen de una línea de texto. Estos métodos se basan en dos metodologías diferentes. Por un lado, presentamos dos métodos basados en Redes Neuronales Recurrentes, en particular la red neuronal; *Long Short-Term Memory*. Por otro lado, se detalla un método basado en *Sequence to Sequence*.

El contexto musical es necesario para mejorar los resultados de OMR, de la misma forma que los modelos lingüísticos y los diccionarios ayudan al reconocimiento de texto. Por ejemplo, se podrían definir fácilmente reglas sintácticas y gramáticas para hacer frente a las ambigüedades del ritmo. En teoría musical, el compás define la cantidad de tiempos por unidad de compás. Así, en la segunda parte de esta disertación se han investigado diferentes metodologías para mejorar el reconocimiento de los OMR. Hemos explorado tres métodos diferentes: (a) una representación gráfica en forma de árbol en la que cada nivel une las primitivas siguiendo un conjunto de reglas, es lo que se denomina Dendrogramas, (b) la incorporación de Modelos de Lenguaje para modelar la probabilidad de una secuencia de tokens y (c) redes neuronales de grafos para analizar las partituras para evitar

relaciones sin sentido entre las primitivas musicales.

Finalmente, para entrenar todas estas metodologías y dada la especificidad de los métodos en la literatura, hemos creado cuatro conjuntos de datos musicales diferentes. Dos de ellos son sintéticos con apariencia moderna o manuscrita antigua y los otros dos son manuscritos reales, siendo uno de ellos moderno y el otro antiguo.

Palabras Clave – Visión por computador, Reconocimiento de Patrones, Análisis y Reconocimiento de Imágenes en Documentos, Redes Neuronales Profundas, Reconocimiento Óptico de Música, Reconocimiento de Música Manuscrita, Contexto Musical.

Contents

List of Tables

List of Figures

List of Algorithms

1 | Introduction

Music has no effect on research work,but both are born of the same source and complement each other through the satisfaction they bestow.

– Albert Einstein

This chapter details the context of this thesis concerning Optical Music Recognition. It contains, the music notation, and the differences with handwritten text recognition. In addition, the objectives and contributions of this work, are explained. Finally, the structure of the thesis is summarized.

1.1 Setting the Context

A music score is composed of a set of music symbols written in a printed or handwritten form, following musical notation rules to indicate, among others, the rhythm and melody of a song. A music score normally is transmitted on paper, but if we take a look back its medium usually was a parchment and oral transmission at the beginning.

The first music scores dates back to around 2,000 BC. It was not until the 1980s that one of the first software, the *Music Construction Set*, was created to manually compose and transcribe a score into a computer format. The importance of creating software to speed up the automatic transcription of music scores should be emphasised to preserve the music heritage. After 4000 years of composing music scores, the manual transcription of them becomes an impossible task.

Automatic transcription systems for musical documents represent interesting tools. The field devoted to address this task is known as Optical Music Recognition (OMR) [6, 21, 116]. Typically, an OMR system takes an image of a music score and automatically export its content into some symbolic structure such as MEI, MusicXML, MIDI, etc. Some of the existing tools are PhotoScore[1] or SharpEye[2]. The music research field has also many other applications such as writer

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

identification, renewal old music scores, the generation of audio files, and finding differences between the same composition but written by different musicians.

1.2 Music Notation

For a long period of time, music written on paper had no standardisation. Figure 1.1 shows different types of music scores. Figure 1.1.a shows a music score used in the byzantine rites [11]. Later, appears neumes (see Figure 1.1.b) a signs above the text which shows how to sing (tempo, dynamics and accentuation). Gregorian's music scores (see Figure 1.1.c) as the last music notations is a pneumatic notation which indicates to the singer how to sing in a mnemonic way. Also, there exists another type of music scores that is specific for an instrument, such as the tablature (see Figure 1.1.d). Finally, a standard notation was created, nowadays known as Western Music Notation (see Figure 1.1.e), which will be the type of notation that we are going to work with in this thesis.

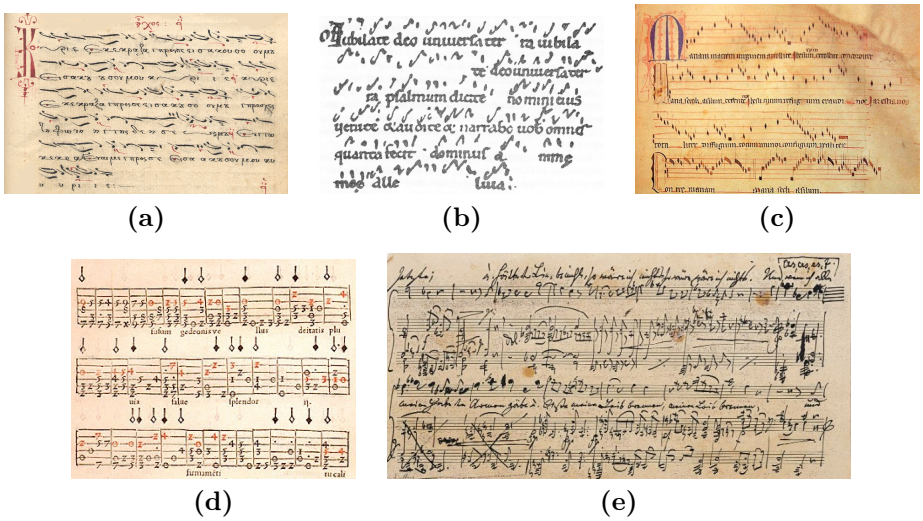


Figure 1.1: Examples of different music scores.

From the 1980s on wards, through advances in technology, the presence of music in computers has increased. Nowadays a computer is able to write or edit a song or a music score, and even to play a synthesizer or virtual instrument. Anyway, some musical notation knowledge is needed to read a music score. Music notation varies notably in music scores depending on the style or genre of music [61, 125]. Classical scores always have a clef, normally treble clef, bass clef or alto clef, the time signature and sometimes the key signature. The time signature shows how many beats has to sum up each measure [47, 73]. The first number

indicates the number of beats and the second number the unit (*e.g.* 2 for half notes, 4 for quarter notes, 8 for eighth notes). The melody is notated on the staff lines using noteheads. Lyrics could appear very close to notes. Finally, some tempo and dynamics may appear in music scores, since the classical period *ca.* 1750, using Italian expressions such as *Grave* (slow), *Allegro* (fast), *pianissimo* (very soft) or *forte* (strong). Before this period, in the Baroque *ca.* 1600-1750 dynamics and tempo were not indicated in the score, since it was assumed that the experience of musicians and singers was enough to know how to play or sing each song. In a music score there are different symbols as symbol notes, clefs rests, etc (see Figure 1.2). The most common terminology is the following:

- Staff: Set of five horizontal lines and four spaces, in which the positions are used to represent the musical pitch.
- Clef: Music Symbol used to indicate the pitch of written music notes.
- Bar lines: Vertical lines which separate every bar unit or measure.
- Notes: Define the pitch and duration of a sound. Composed of note heads, beams, stems, flags and accidentals.
- Accidental: A sign that alters notes by raising or lowering their pitch.
- Rest: 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 must be played without separation.
- Dynamic and Tempo Markings: They indicate the speed and the volume of the music to be played.
- Lyrics: Set of words that the singers have to sing.

1.3 Motivation

OMR has lately reached a very good performance on scanned printed music scores, especially for monophonic scores with low density of symbols. The recognition of handwritten scores is still a challenge due to the variabilities in the handwriting styles. Besides, the availability of labelled datasets of old handwritten music scores is scarce, which hinder the training of deep-learning based architectures. Although there are several projects [3],[4],[5],[6],[7] on ancient manuscript music cataloging, digitization and/or transcription, the immense amount and variability of existing music

[3]SIMSSA: `https://simssa.ca/`
[4]Hispamus: `https://grfia.dlsi.ua.es/hispamus/`
[5]IFMuC: `http://pagines.uab.cat/ifmuc/es`
[6]RISM: `http://www.rism.info/home.html`
[7]MDC: `http://mdc.csuc.cat/cdm/search/collection/musicatedra!partiturBC`

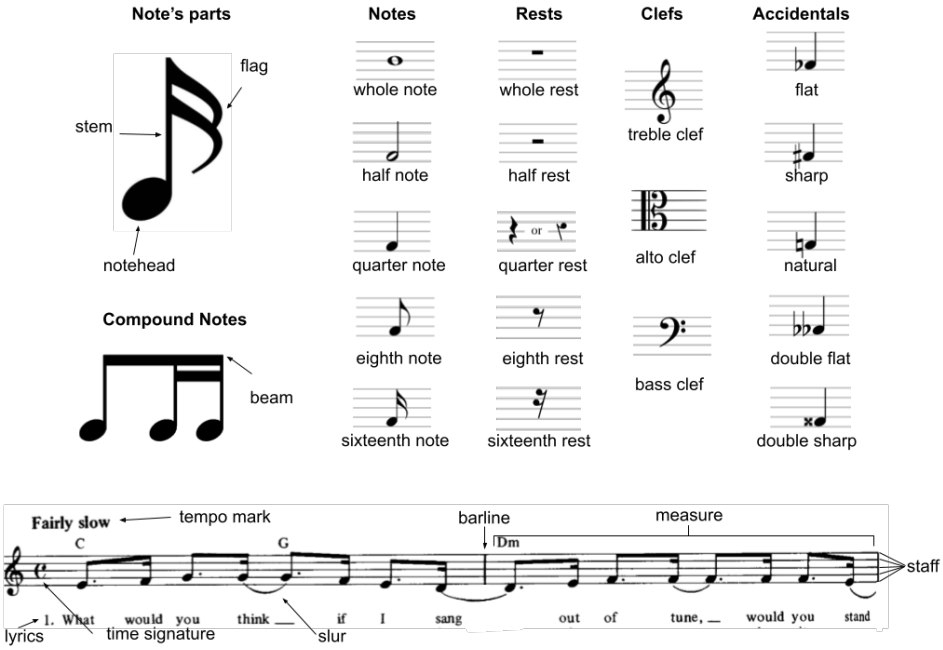


Figure 1.2: Different symbols found in a music score.

works in archives makes it impossible in practice to manually transcribe and disseminate the entire source and forces musicology to, most of the times, carry out strictly qualitative research. This problem, obvious at a global level, is equally self evident at the local one.

In summary, music scores have been written in paper format for centuries. Due to paper degradation, its digitization and conversion into a machine-readable format is desired. For this second task, it is important to develop systems able to transcribe music scores, either printed or handwritten in a sheet of paper. Given the resurgence of Deep Learning, researchers have taken advantage of this technology, enhancing the OMR performance. Concretely, the research is often inspired in powerful Deep Learning techniques applied to Computer Vision which have shown good performance in text recognition [1, 118, 137, 138].

For above reasons, the motivation of this thesis is to explore the power of deep learning for proposing reading music systems.

1.4 Difficulties of Optical Music Recognition

As commented before, music scores are usually written in paper sheet. Historical handwritten scores are particularly interesting to transcribe because there are many of them stored in archives, churches and libraries throughout. Most of them have never been transcribed nor listened, which makes it important to devote efforts towards their conservation, transcription, study and dissemination. These scores are much harder to recognise than regular typeset ones because of hundreds of years worth of paper degradation, the evolution of music notation conventions and the irregular nature of handwriting, which leads to many ambiguities and hard-to-read passages even for trained humans. Moreover, in the case of old and historical scores, the difficulties increase due to paper degradation, the frequent appearance of touching elements (*e.g.* lyrics and music symbols often overlap), or even the lack of standard notation, where one can find music scores that do not follow current music notation rules, as it can be seen in Figure 1.3.

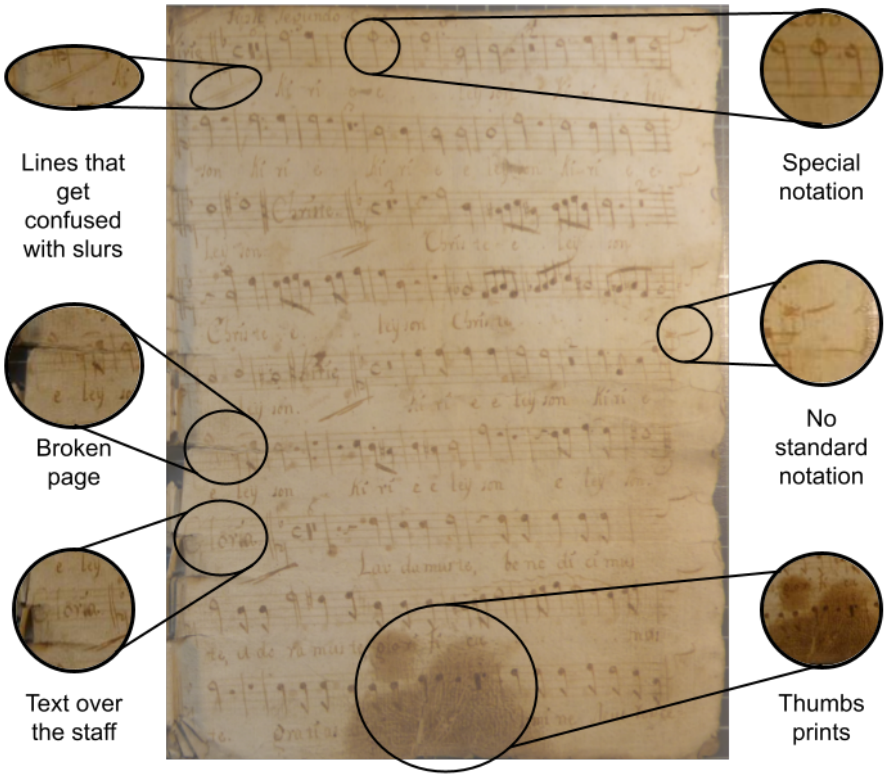


Figure 1.3: Difficulties recognizing music scores.

In the case of an European church music, for example, in a significant parish or

cathedral, which will probably have more than 500 years of history of handwritten music behind, one can find more than 2000 records of different works by various composers. This can easily mean tens of thousands of handwritten score pages in one single church. To give an example of the magnitude of this problem, we should consider that in a city like Barcelona there are four such musical chapels. Although there are few exceptions, their documentation is almost completely preserved in their archives, so the amount of music to be transcribed and analyzed is overwhelming for a manual process. This task is very time consuming and requires a huge amount of human resources (for example, a music transcriber can devote 1-3 hours to transcribing one music page, depending on the density of music symbols). 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.

1.4.1 Music vs Text Recognition

Music symbols have to be read from left to right, and in each staff, symbols appear with a specific rhythm (duration) and pitch (melody). In other words, music is harder to read than text due to its bi-dimensional nature. Figure 1.4 shows some similarities and differences between text and music. An Optical Music Recognition system has some similarities with an Optical Character Recognition system [16, 17, 53, 77, 94, 110, 112, 128]. The recognition of characters in text is similar to isolated notes in music. The recognition of words is similar to the recognition of compound notes. Phrases can be seen as a bar units (measures). In this stage syntactical rules appear. For example, grammatical rules as Subject+Verb+Object are used in text, whereas in music notation, the number of beats have to sum up the time signature. And emphasis can be seen as ornaments. However, music has some other difficulties. Compound music notes have a huge freedom when connecting music notes. Polyphonic scores do not sum up the time signature or ornaments escape from the restriction of summing up the beats.

1.5 Objectives and Contributions of this Thesis

In this section we describe the objectives and contributions of this thesis.

1.5.1 Objectives

The main objective of this work is to develop novel techniques based on Deep Learning for music recognition. During this dissertation three different scenarios have been exploited according to how the scores have been written (*e.g.* printed or handwritten) and the time period (*e.g.* modern or ancient). Thus, the main objective could be decomposed in two sub-objectives.

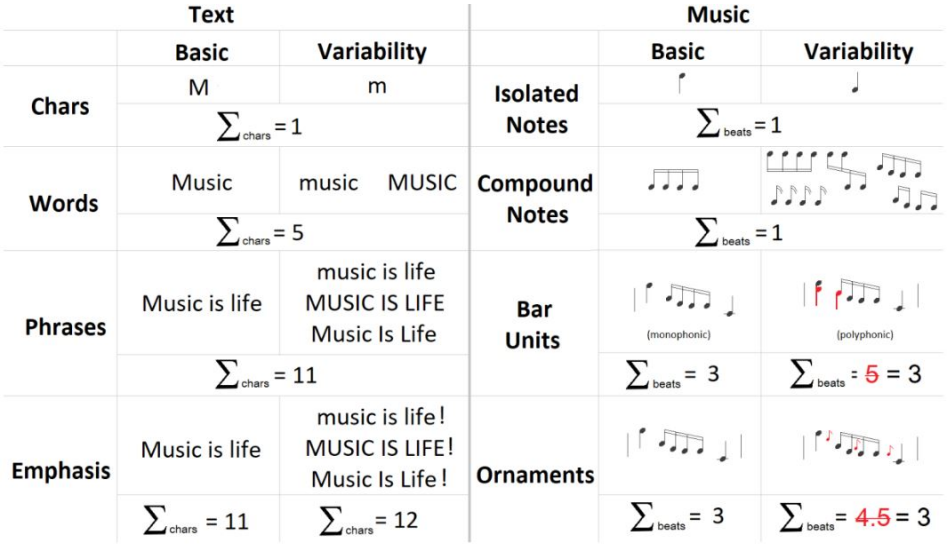


Figure 1.4: Similarities and differences between music and text.

- To adapt Deep Learning methods to create full Optical Music Recognition Systems. The methods will be evaluated on modern, old, printed and handwritten scores.

- To propose some novel or existing OMR methods that incorporate music context to improve the performance.

1.5.2 Contributions

The contributions of this thesis are described next.

Datasets

In order to use Deep Learning to recognize music scores, a huge quantity of data is needed. Four different datasets have been created. Each dataset is different from the rest (modern/old notation, synthetic/handwritten, etc).

OMR based on Bidirectional Long Short-Term Memory Neural Network

This method directly recognizes the music content appearing on an image following the sequence. We address OMR by separating the two components of the musical symbols: duration and pitch. This separation is done both in terms of training and evaluation. This provides the RNN with greater robustness, as it can focus on the specific aspects that concern each component.

OMR based on Convolutional Recurrent Neural Network for Handwritten Scores

We propose a full staff-wise Handwritten Music Recognition (HMR) system. It also disentangles the output of the network in the two main components of music notation: rhythm and pitch. First, we add Convolutional Neural Networks as feature extractor. Secondly, since the existing amount of annotated handwritten music scores is scarce, we propose a novel data augmentation technique, and incorporate transfer learning from printed scores.

OMR based on Sequence-to-Sequence for Old Scores

This work focuses on the adaptation of a sequence-to-sequence model with attention for historical music recognition. The adaptation of sequence-to-sequence to the particularities of old music scores is crucial to accelerate the process from its discovery to its digital transcription, enabling researchers to analyze, publicize and divulge unknown composers and compositions that traditional methods are forced to neglect.

Sequence to Sequence with language models

Even the recent Seq2Seq architectures fail when recognizing historical manuscripts. Language Model (LM) can tackle many of these ambiguities in the recognition. The integration of Language models into a Seq2Seq architecture to minimise the ambiguities when recognising historical handwritten scores. Concretely, we integrate a LM through three different techniques: Shallow, Deep [65] and Candidate Fusion [78].

Convolutional Graph Neural Networks

In order to allow an OMR system to know which primitives form each particular symbol, a structure that relates them is needed. Music has a large variability of symbols, combining rhythm and melody. But, despite this variability, it shares a common set of primitives (*e.g.* stem, notehead, beam, flag, etc) that are representable. Since graphs are suitable for music representation because of its flexibility and n-dimensional representation power, we propose an OMR method based on Graph Neural Networks (GNNs).

1.6 Organization

The rest of the dissertation is organized in ten chapters and two global chapters. First, the methods that have been traditionally used for Optical Music Recognition and the datasets created during this years are described.

- Chapter 2 presents the state of the art. This chapter details the evolution of Optical Music Recognition from the firsts classical methods to the most recent ones using Deep Learning.

- Chapter 3 details the datasets that have been created. We provide both real and synthetic datasets that emulate real scores to deal with the lack of data.

PART I: Neural Network-based OMR methods

The first part of the thesis details some methods adapted to music recognition.

- Chapter 4 explores the use of Recurrent Neural Networks. Specifically, the Long Short-Term Memory (LSTM). A staff line is read from left to right through the LSTM network.
- Chapter 5 is quite related to the previous one. In this chapter, the staff line is first convolved and the features are the input of a Bidirectional Long Short-Term Memory Neural Network.
- In Chapter 6 a sequence-to-sequence model with attention mechanism is proposed to recognize historical music scores.

PART II: Contextual OMR methods

In the second part of this thesis, methods that use context music information are described. Some methodologies are used to improve a previously described OMR method, whereas some other methods are proposed in which the context is implicit.

- In Chapter 7, dendrograms are used to represent the unions of music primitives and validate them as compound music notes.
- Chapter 8 focuses on enhancing the sequence-to-sequence models by the incorporation of language models to improve transcriptions in which the aforementioned unclear writing produces statistically unsound mistakes.
- Chapter 9 takes advantage of the bi-dimensional structure of music notation through the proposal of Graph Neural Networks to recognize music scores.

Finally, in Chapter 10 the conclusions are drawn. In addition, next steps in this field are discussed for a future research lines.

2 State of the art: From classical OMR to Deep Learning

Things are always at their best in their beginning.

– Blaise Pascal

This section describes the key references of Optical Music Recognition that are relevant to the present thesis. First, the classic machine learning techniques used in OMR are described. Usually, classic works focus on a part of the OMR's process, so we will detail each step of the pipeline. Finally, the most relevant methods using Deep Learning are explained.

The Optical Music Recognition has to be able to recognize each element located in the music score, independently of the input format, either synthetic, real paper, or in digital format such as a tablet [22, 55, 56]. The classification of OMR methods may vary depending on the criteria. OMR methods can be classified according to the type of methodology (classical or deep learning methods), or according to the task it deal with (full or some steps of the OMR pipeline). In this thesis we will classify the methods according to whether they have been developed with classical machine learning techniques or by deep learning. We will also make an special mention to those works for handwritten scores.

2.1 Classical Machine Learning

In its early days, when machine learning was the most widely used methodology, the recognition of music scores was divided in stages, following a pipeline. The methods focused on each particular stage. Figure 2.1 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 is to reduce problems in segmentation. Normally, before segmenting the musical symbols and/or primitives, the staff lines are removed. Hence, the classification 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.

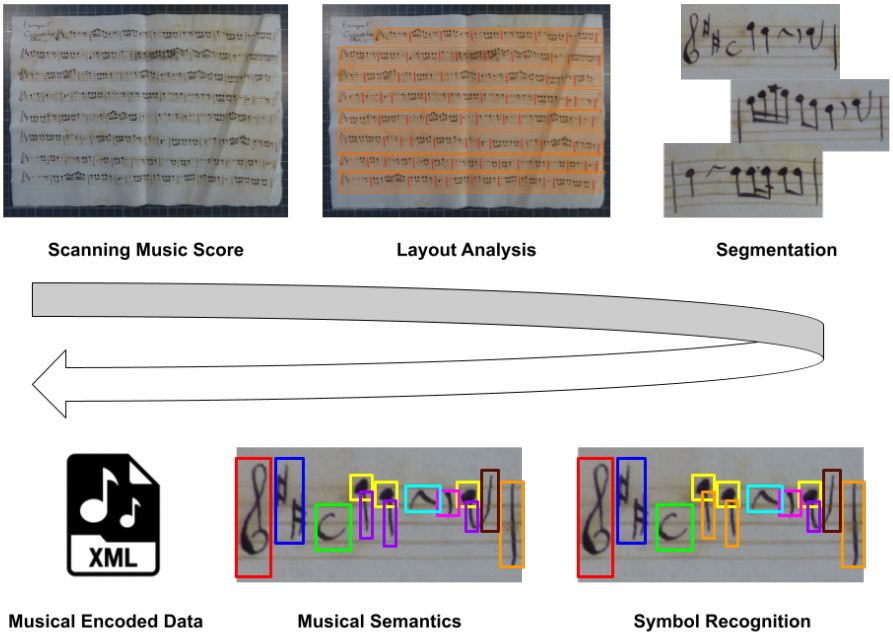


Figure 2.1: Pipeline of a classical OMR system.

- **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 [105] 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 [49] making the recognition task easier. Normally, the staff removal algorithm [44, 91] 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. Figure 2.2 shows an example of staff removal.

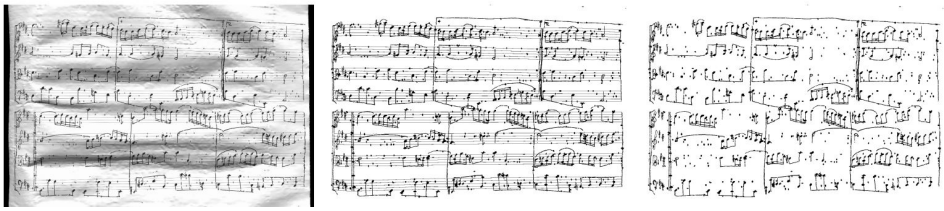
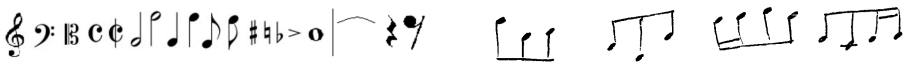


Figure 2.2: A music page, its binary image and the page without the staff lines. Reprinted from [49].

- **Symbol recognition** The recognition of music symbols consists in the recognition of isolated and compound music symbols. This classification is done because compound music symbols cannot be recognized as a whole, so the techniques are different. 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 2.3 shows isolated and compound music symbols. Therefore, the classification is the following:
  - *Isolated Music Symbols:* Isolated music symbols are defined as symbols that have [0,1] note-head. The most popular techniques to detect this kind of symbols are: classification 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[33, 92, 115].
  - *Compound Music Symbols:* Compound music symbols are defined as symbols that have [2,∞) note-heads. Usually, they are recognized using primitive-based techniques [9]. 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 classification can not be used because they are more difficult to be recognized and have infinite combinations between them.



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

Figure 2.3: Examples of music symbols.

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

- *Classification:* Detection-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 [50] the authors describe a method based on the Dynamic Time Warping (DTW) algorithm to recognize symbols. Figure 2.4 shows some symbols and the features vectors that the DTW algorithm uses to calculate the distance. This method is invariant to rotation and scale. Authors of [45] introduce the Blurred Shape Model descriptor, a symbol descriptor that is invariant to rotation and reflection. The BSM encodes the spatial probability of the symbol shape. In [115] authors compare several methods for classifying symbols. They performed their experiments by Neural Networks, Nearest Neighbour, Support vector machines and Hidden Markov models. And in [117] they propose to learn the best distance for the k-nearest neighbour classifier and the performance of the method is compared with the support vector machine classifier.

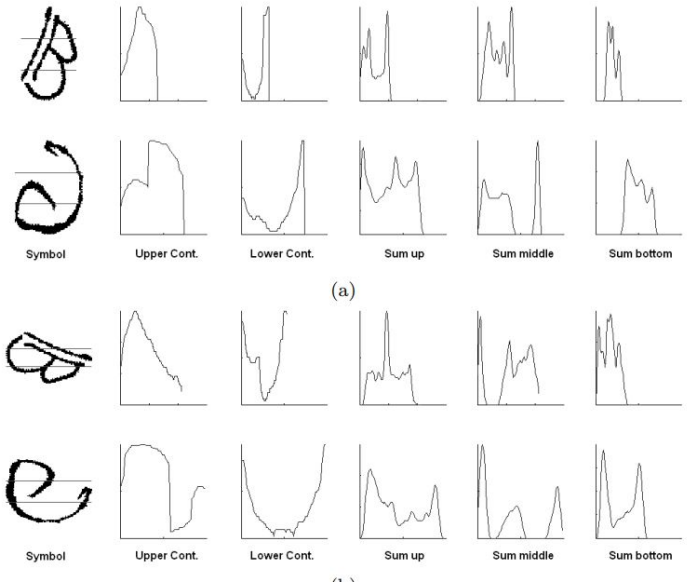


Figure 2.4: Symbol invariant to rotation and scale. Reprinted from [50].

– *Grammars / Rules:* These methods propose to define a grammar or rules 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. Authors in [37] propose a grammar to help the detection of most errors on note duration. Figure 2.5 shows an example of correcting errors. The first measure in the fist staff a beat is missing.

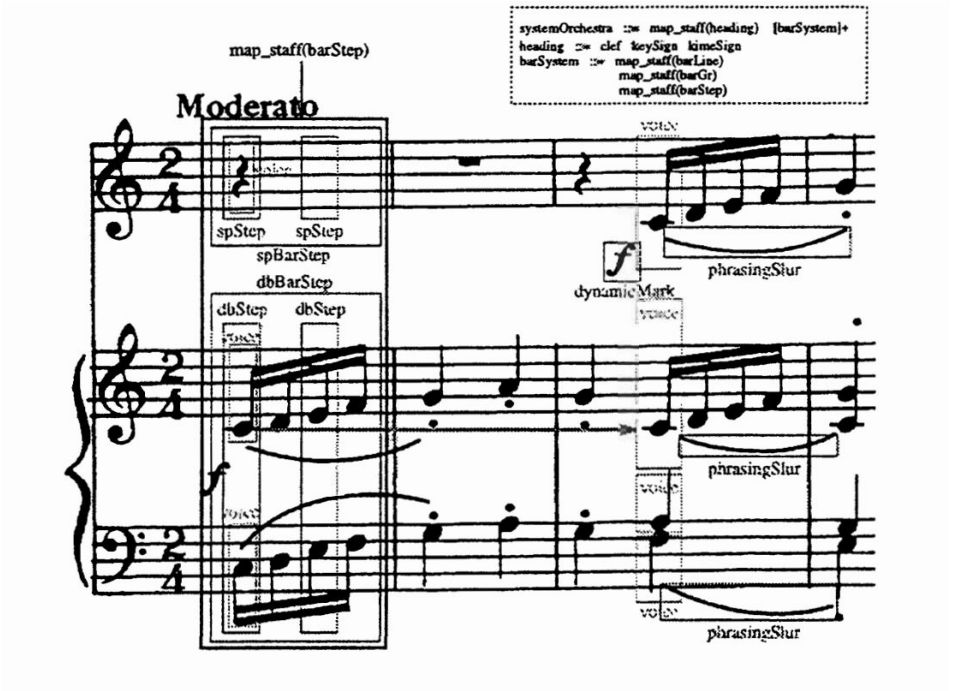


Figure 2.5: Grammars helps to reduce detection errors. Reprinted from [37].

- *Sequence:* OMR models using Hidden Markov Models (HMMs)[113] 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), HMMs do not have a good performance. In addition, HMMs are able to segment and recognize without any previous preprocessing task. For example, in [109] the author presents an OMR using Hidden Markov Models without staff line removal. And in [111] 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 with 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 [107] 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.

- **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 [37, 87] propose 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 matches the time signature.

- **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[1], MusicXML[2] or MEI[3]. 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.

Summary Table 2.1 shows advantages and disadvantages of the previously described methods. The first column shows techniques described previously, the second column shows the advantages of each technique whereas the third column shows their disadvantages. One might conclude that detection-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).

2.1.1 Recognition of Handwritten Music Scores

Concerning handwritten scores, although it is remarkable the work in Early musical notation [107, 111] 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

[1] `https://www.midi.org/`
[2] `http://www.musicxml.com/`
[3] `http://music-encoding.org/`

Table 2.1: Advantages and Disadvantages of different techniques used in OMR

Method	Advantages	Disadvantages
Classification	Simplest method. Machine learning based.	It needs a segmentation preprocess. Difficulties in recognizing compound music symbols
Grammars/Rules	Able to recognize compound 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	High complexity of the algorithm.

community has used specific symbol recognition methods [46, 50] and learning-based techniques such as Sector Vector Machine's, Hidden Markov Model's or Neural Network's [115]. Figure 2.6 shows some results using the Blurred Shape Model descriptor.

As stated in [93], 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 [23, 90]. Figure shows some results on online recognition. Nowadays, a musician can find several applications for mobile devices, such as StaffPad [4], MyScript Music [5] or NotateMe [6]. 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.

2.2 Deep learning

Deep Learning may seem very novel technique. However, as Goodfellow, Bengio and Courvill claim in [59] 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 [71, 88] and the

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

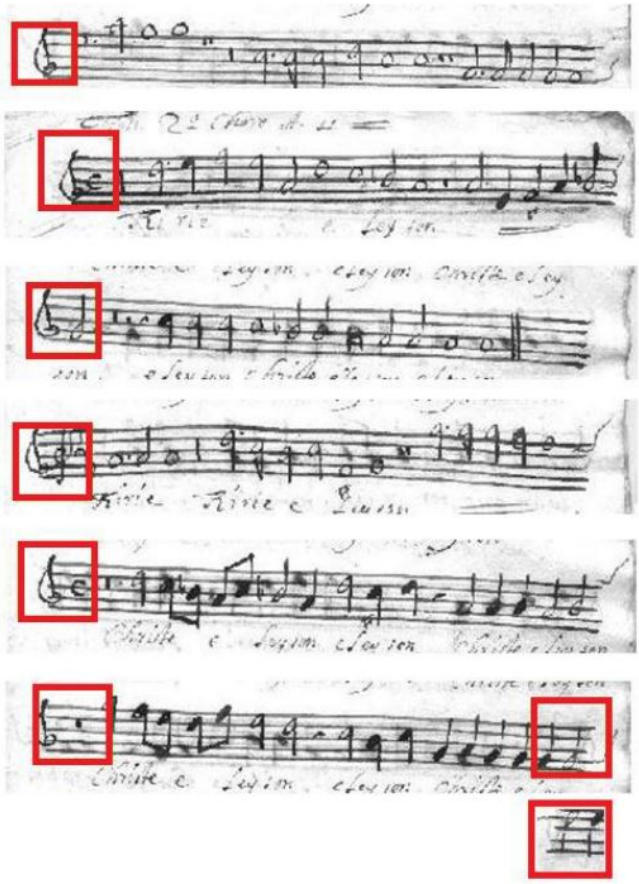


Figure 2.6: Clef detection in old handwritten music score images. Reprinted from [46].

first model of perceptron [120]. Afterwards, connectionism appears with the back-propagation [121] and finally deep learning appears in 2006 by Hinton, Bengio and Ranzato [12, 74, 114].

2.2.1 Deep learning in Optical Music Recognition

In the last decade, several deep learning techniques have been applied to music or audio processing. For example, the authors of [133] uses Convolutional Neural Netwoks (CNN) for detecting symbols or primitives. Figure 2.8 shows two different datasets with the object detector output. The authors suggest to predict dense

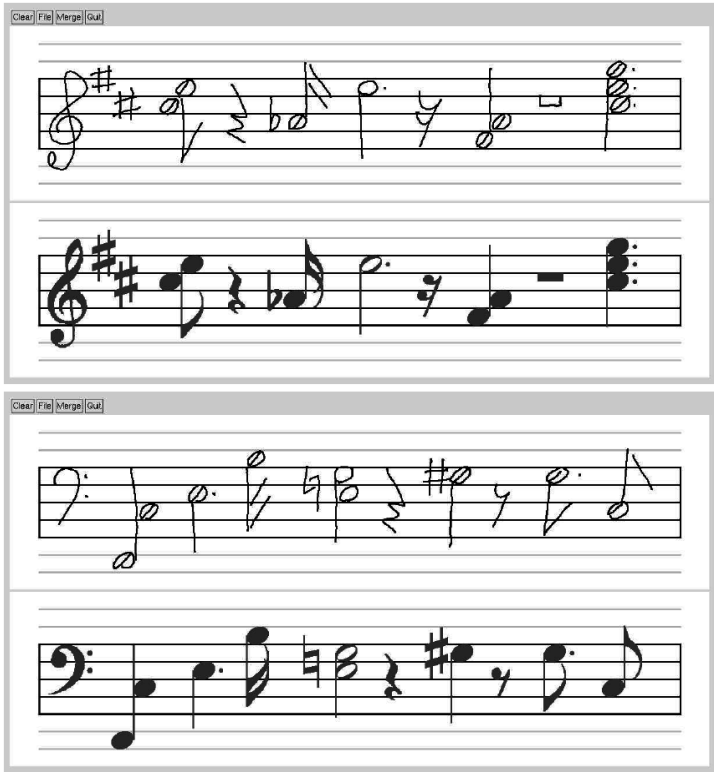
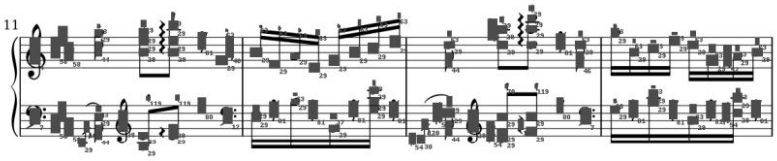


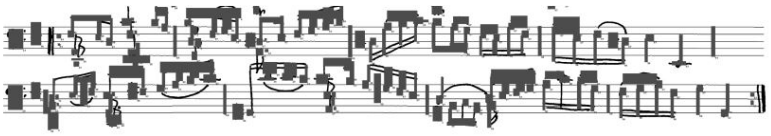
Figure 2.7: Some results in online recognition. Reprinted from [90].

energy maps using ResNets [70] and watershed transform that will be used to predict where is a symbol, what class does it belong to, and to define the bounding box. Wen *et al.* [136] use Convolutional Neural Networks in order to recognize music symbols. First of all, they binarize the music score and remove the staff lines. Then, they detect the symbols with connected components, and finally, to recognize each music symbol, they use the CNN as a classifier.

Recurrent Neural Networks also have been used in OMR, the authors of [26] use long short-term memory recurrent neural networks (LSTMs) [75]. They use a Convolutional Neural Network followed by a Recurrent Neural Network. The first network deals with feature extraction problem, and then, the RNN deals with the sequential nature of the problem. To obtain the correct sequence given a music score, the authors use the Connectionist Temporal Classification(CTC) loss[62] commonly used in text recognition. Figure 2.9 shows some qualitative results comparing the Photoscore's (comercial software) results and their results. In [30] the authors of this paper suggest to use Recurrent Convolutional Networks in order to create one of first complete end-to-end optical music recognition for monophonic music scores. They use the CTC loss function to avoid the alignment



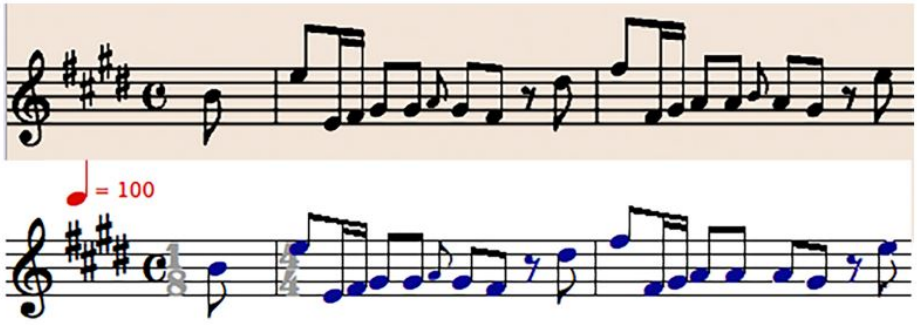
(a) Example result from DeepScores



(b) Example result from MUSCIMA++

Figure 2.8: Objected detection method. Images shows qualitative results of primitive detecion. Reprinted from [133].

between the image and the groundtruth. The results are evaluated obtaining good results using a synthetic dataset created though lilypond software. In the same direction, in [135] the application of sequence-to-sequence models shows very good results. The authors propose a convolutional sequence-to-sequence model. To train their model, they use monophonic scores removing dynamics, expressions and chord symbols.



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 2.9: Results using a CRNN. First, the original image, a) is the PhotoScore's result and b) their results. Reprinted from [26].

2.2.2 Deep learning in Handwritten Music Recognition

To a lesser extent, deep learning has also been applied to handwritten music recognition. In this section we will highlight some of the most recent work. On [97] the authors deal with the recognition of handwritten music scores by proposing an end-to-end trainable object detector for music symbols. They use the COCO dataset [85] to train the network and finetunnig with MUSCIMA-CVC[68]. Figure 2.10 shows qualitative results on the object detection task using a handwritten dataset, specifically MUSCIMA++. Calvo-Zaragoza *et al.* uses CNNs for staff line detection[24]. The authors of [106] propose a basic methodology using Convolutional Neural Networks. Specifically they compare how well LeNet[84], AlexNet[83] and GoogleNet[130] works, recognizing symbols of the HOMUS Dataset[22] classified in 32 classes.

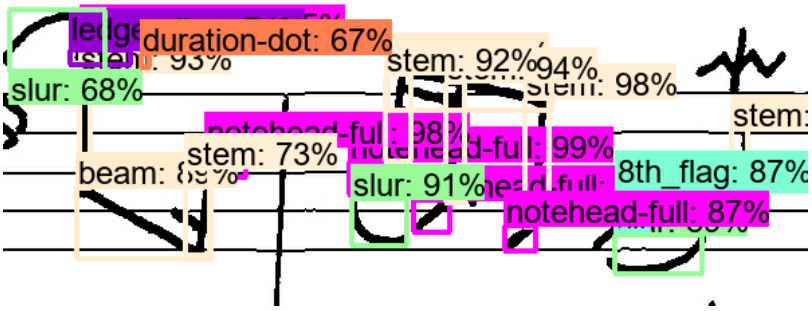


Figure 2.10: Object detector in handwritten music recognition. Reprinted from [97].

Other works have used Recurrent Neural Networks for handwritten music scores. On [3] the authors review the state of the art methods to decrease the error rate combining different premises. On the one hand, they use as a baseline the method presented by Alfaro-Contreras *et al.* in [4] but using as input of the CRNN the features extracted and concatenated (after the convolutional, after the first recurrent layer or after the recurrent block). On the other hand, they use the encoding presented in [122] separating the rhythm and pitch. Castellanos *et al.* [32] combine the page layout using Selectional Auto-Encoders to extract the staff (Figure 2.11 shows the staff region prediction) and a CRNN to detect the music in each staff.

2.3 Summary

In this chapter we have seen different methods for optical music recognition, classified into methods based on machine learning and methods based on deep learning. Moreover, we have also mentioned some of the papers that deal with handwritten

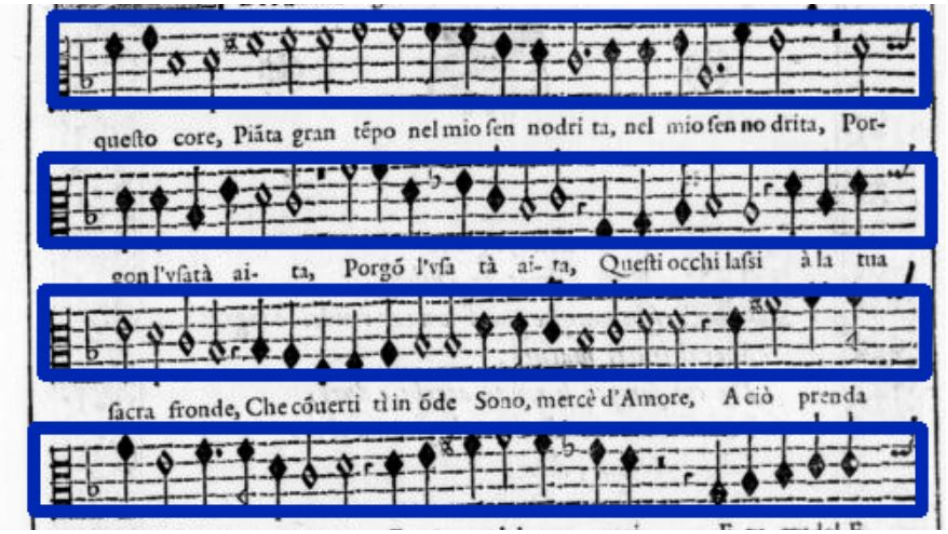


Figure 2.11: Staves detected in a page layout process. Reprinted from [32].

music scores. From the literature review, we observe that most classical methods tend to focus on one part of the recognition process, while deep learning ones start to have a tendency towards a full OMR pipeline. Even though with this trend, the problem is not solved and more research is needed.

3 | Created Datasets

Information is not knowledge. Knowledge is not wisdom. Wisdom is not truth. Truth is not beauty. Beauty is not love. Love is not music. Music is THE BEST.

– Frank Zappa

Optical Music Recognition research compared with other research fields is relatively novel. Therefore, only a few set of datasets exists for this task. This chapter will mention the existing datasets. Then, the datasets that have been created throughout this thesis will be detailed, explaining their particularities.

3.1 Introduction

Optical Music Recognition datasets normally are made for specific targets, methods or steps of the OMR pipeline (*i.e.* object detection, writer identification, *etc*). Depending on the task the datasets will be encoded in one way or another. A dataset intended for object detection will contain bounding boxes with the coordinates and the corresponding class, while a dataset intended for a recurrent network having the classes sequentially will be enough. Table 3.1 shows some OMR datasets. On the first column appears the name, in the second column if it is handwritten or typeset(printed), in the third column the size, in the fourth column the format of the information and finally in the last column the task it is intended for. These include: HOMUS[22], CVC-MUSCIMA[48] and its extension MUSCIMA++[68], Deep-Scores[133], PrIMuS[26], the Universal Music Symbol Collection[98] and DoReMi[126].

Table 3.1: Optical Music Recognition datasets. Table extracted from [95].

Name	Engraving	Size	Format	Typical usages
Handwritten Online Musical Symbols (HOMUS)	Handwritten	15200 symbols	Text-File	Symbol Classification (online + offline)
Universal Music Symbol Collection	Typeset + Handwritten	∼90000 symbols	Images	Symbol Classification (offline)
CVC-MUSCIMA	Handwritten	1000 score images	Images	Staff line removal, writer identification
MUSCIMA++	Handwritten	>90000 annotatations	Images, Measure Annotations, MuNG	Symbol Classification, Object Detection, End-To-End, Measure Recognition
DeepScores	Typeset	300000 images	Images, XML	Symbol Classification, Object Detection, Semantic Segmentation
PrIMuS	Typeset	87678 incipits	Images, MEI, Simplified encoding, agnostic encoding	End-to-End Recognition
Multimodal Sheet Music Dataset[43]	Typeset	497 songs	Images, MIDI, Lilypond, MuNG (noteheads)	End-to-End Recognition, Multimodal Retrieval, Score Following
Sheet Midi Retrieval Dataset[132]	Typeset	200 songs	Images (Jpg and PDF), MIDI, CSV	Multimodal Retrieval, Score Following
AudioLabs v1[140]	Typeset	940 score images; 24,329 bounding boxes	Images	Box Annotation Detection
AudioLabs v2	Typeset	940 score images; 85,980 bounding boxes	Images	Box Annotation Detection
MuseScore	Typeset	>340000 files	MuseScore, PDF, MusicXML	Various
MuseScore Monophonic MusicXML Dataset[135]	Typeset	17000 IDs	IDs for MuseScore files	Various
Capitan collection[27]	Handwritten	10230 symbols	Images, Text-File	Symbol Classification
SEILS Dataset [102, 103, 104]	Typeset	30 madrigals, 150 original images, 930 symbolic files	Images (PDF), .ly, .mid, .xml, .musx, .krn, .mei, .mns, .agnostic, .semantic	Various
Rebelo Dataset	Typeset	15000 symbols	Images	Symbol Classification
Fornes Dataset	Handwritten	4100 symbols	Images	Symbol Classification
Choi Accidentals Dataset	Typeset	2955 images	Images with special filename	Symbol Classification

Name	Engraving	Size	Format	Typical usages
Audiveris OMR	Typeset	800 annotations	Images, XML	Symbol Classification, Object Detection
Printed Music Symbols Dataset	Typeset	200 symbols	Images	Symbol Classification
Music Score Classification Dataset[99]	Typeset	1000 score images	Images	Sheet Classification
OpenOMR Dataset [42]	Typeset	706 symbols	Images	Symbol Classification
Gamera MusicStaves [39]	Typeset	32 score images	Images	Staff line removal
Early Typographic Prints[111]	Typeset	240 score images		
Silva Online Handwritten Symbols[38]	Handwritten	12600 symbols		
IMSLP	Typeset	>420000 score images	PDF	Various
Byrd Dataset[19]	Typeset	34 score images	Images	Various
DoReMi	Typeset	911771 symbols	ML, metadata, images, MIDI, MEI, MUSICXML	Object Detection, Reconstruction and Encoding, End-to-end

The HOMUS (Handwritten Online Musical Symbols) dataset is an online dataset which incorporates the image of the note and the sequence of strokes. This work also incorporates a method to test it, based on Nearest Neighbor an Hidden Markov Models (HMM). CVC-MUSCIMA was created to deal with the writer identification task and staff removal for this reason this dataset has 20 music scores written by 50 writers. Then, a part of the CVC-MUSCIMA was labeled and named MUSCIMA++ to train models for object detection. For the same task, but in typeset format, DeepScores includes detailed annotation and oriented bounding boxes. Printed Images of Music Staves (PrIMuS) is a dataset thought to end-to-end object recognition task. The authors provides as groundtruth the MEI, MIDI, the image and the sequence of symbols (rhythm) and positions (pitch). Exist another version with some distortions called Camera-PrIMuS [25]. The Universal Music Symbol Collection is a dataset which includes symbols from 7 different datasets as MUSCIMA++, HOMUS, Audiveris OMR dataset, Printed Music Symbols dataset,OpenOMR dataset, Rebelo's Datasets [115] and Fornés's dataset [51] to train a classifier. Finally, DoReMi dataset is the largest dataset created specifically for Deep learning and to deal with several steps on the OMR pipeline. It includes bounding boxes, images in different format, MusicXML files. Figure 3.1 shows some examples of the most popular datasets.

3.1.1 Summary

As time went by, during the execution of this thesis, and as the Table 3.1 shows, there is more and more data to train music recognition systems. However, most of these are focused on only one part of the pipeline. For this reason we have been forced to create new datasets to train a full OMR system.

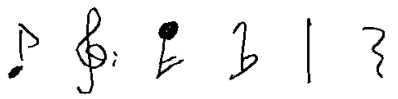
Then, the rest of the chapter (section 3.2) is organized as follows. Section 3.2.1 will describe a new version of CVC-MUSCIMA. In section 3.2.2 we describe a synthetic dataset which emulates old music scores. Section 3.2.3 explains a dataset based on old music scores. Finally, section 3.2.4 details a dataset to train object detection and graph methods.

3.2 Our created datasets

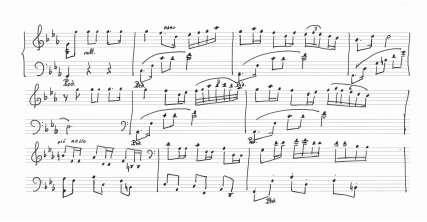
This section describes the datasets created in this thesis.

3.2.1 Symbol level music scores, new version from MUSCIMA dataset

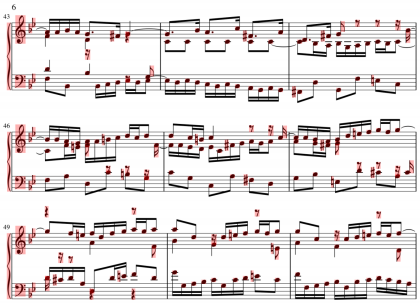
The MUSCIMA++ dataset [68] is a selection of 140 pages from the CVC-MUSCIMA dataset [48], annotated at primitive level. Although these primitives are related



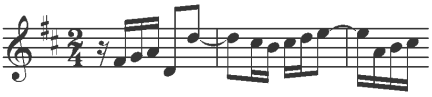
(a) HOMUS



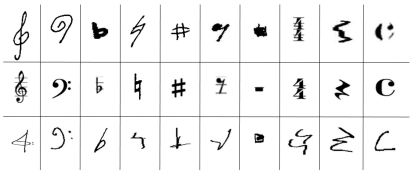
(b) MUSCIMA



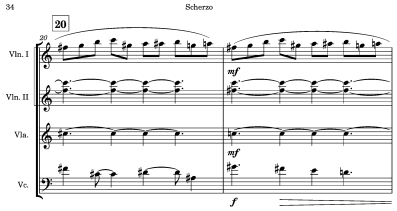
(c) DeepScores



(d) PrIMus



(e) Universal Music Symbol Collection



(f) DoReMi

Figure 3.1: Most popular datasets.

each other using a graph, they cannot be directly used for OMR evaluation. For this reason, having into account the graph relations and keeping the noteheads as the main node of notes, we have manually labelled 20 music pages at symbol level (including slurs, dynamic marks, etc.) in order to evaluate a full OMR system. In any case, we should take into account that the original CVC-MUSCIMA dataset was created for staff removal and writer identification (for this reason, it contains the same 20 different musical compositions, rewritten by 50 different writers). This fact leads us to some limitations when splitting the sets *i.e.* into train, validation and test. Our method must never see the same musical composition at test and train or it may be biased towards the recognition of a specific melody. For this reason, we have selected these 20 pages (musical compositions), from different writers (see Table 3.2). Figure 3.2 shows two staves of the new dataset.

Table 3.2: Selected Muscima++ pages for train, validation and test sets. We indicate the number of staves per page, and if the page is polyphonic.

	Train												Validation				Test			
Page	2	4	5	6	7	8	9	13	15	16	19	20	11	12	14	17	1	3	10	18
Writer	20	12	21	16	31	35	32	23	43	34	1	18	49	18	29	44	17	13	15	10
Polyphonic					✓		✓	✓			✓	✓			✓	✓			✓	✓
# Staves	6	5	7	6	4	6	4	5	5	8	3	8	6	8	6	6	5	7	6	8

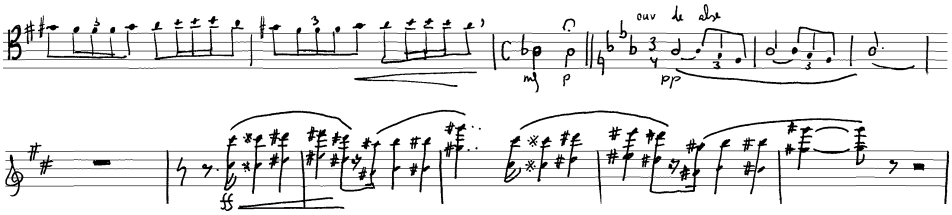


Figure 3.2: Two music staves from different writer and page.

3.2.2 Synthetic Old Music Scores

Since the amount of historical labelled data is scarce, we must look for alternatives. Therefore, we have generated two synthetic datasets using Lilypond [1]. Each one contains about 30,000 bar images, and are divided into 60% train, 20% validation and 20% test. These two datasets are complementary: one simulates the particularities of historical scores, whereas the other provides examples of a large diversity of symbols, including polyphony. These datasets are described next:

- Old synthetic (monophonic): This dataset tries to imitate the texture and degradation of the paper of historical scores adding a background. Also, the type and diversity of symbols is limited, similar to the historical scores used in the experiments. Figure 3.3 shows a measure from the old synthetic dataset.
- Modern synthetic (polyphonic): This dataset contains polyphonic symbols written in one staff, i.e. stacks of notes meant to be played all at once, such as chords. Figure 3.4 shows a measure from this dataset. This data will allow our model to generalize to any kind of historical music score, either monophonic or poliphonic.

3.2.3 Old Music Scores by Pau Llinàs

We have created a historical dataset,a motet composed by Pau Llinàs, a catalan musician who worked as chapel master in Santa Maria del Pi of Barcelona between

[1] `https://lilypond.org/`

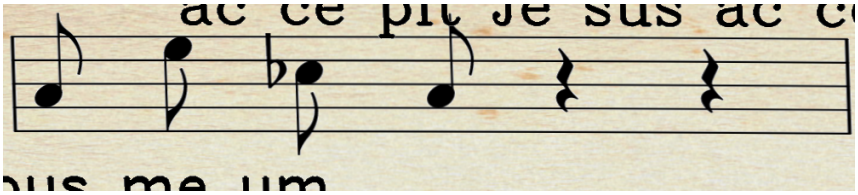


Figure 3.3: Example image from the old synthetic dataset.

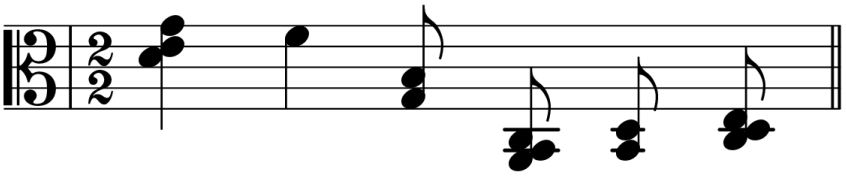


Figure 3.4: Example image from the modern (polyphonic) dataset.

1709 and 1749. Most probably, the work was written around that time since Llinàs spent most of his life as a chapel master and, thus, composing professionally in this chapel [5]. This religious motet belongs to psalm number 148, –Laudate Domine (Praise the Lord)– of the Book of Psalms and, though religious, it's not considered a proper liturgical work as it does not belong to the liturgical mass texts –Kyrie, Gloria, Credo, Sanctus, Benedictus, Agnus. Like all religious motets, it could be sung in a variety of extra-liturgical contexts. In ours, it was usually before the Eucharist, during its preparation by the priest [36]. It is preserved in 12 separated parts, instead of a full score (most common in this time period). It consists of 32 pages, written in a choral form, with 10 sung parts arranged in two choirs, one with four voices and another with six ones. At the instrumental side, we find a bass part (written for a low-pitched monodic instrument like a bassoon, cello or double bass) and a thoroughbass part (written for an organ, harp or, ultimately, any harmonic instrument). This motet actually belongs to the *Fons Musical de la Catedral de Barcelona* and ha.s been incorporated at the *Biblioteca Nacional de Catalunya* (BNC) catalogue [40].

For our experimental validation, we have manually labeled 40 music staves, containing 245 measure images. These are divided into 147 measures for training, 49 for validation and 49 for test. Figure 3.5 shows differents measures and Figure 3.6 shows a page from this historical dataset, illustrating their main difficulties. On the first staff we can observe that the lyrics are touching the staff and in some cases even the symbols. Also, at the end of the third staff or at the beginning of the fifth staff the are ink stains.

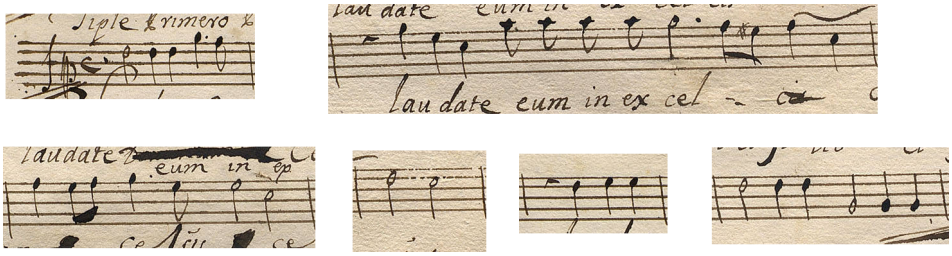


Figure 3.5: Different examples of measures.

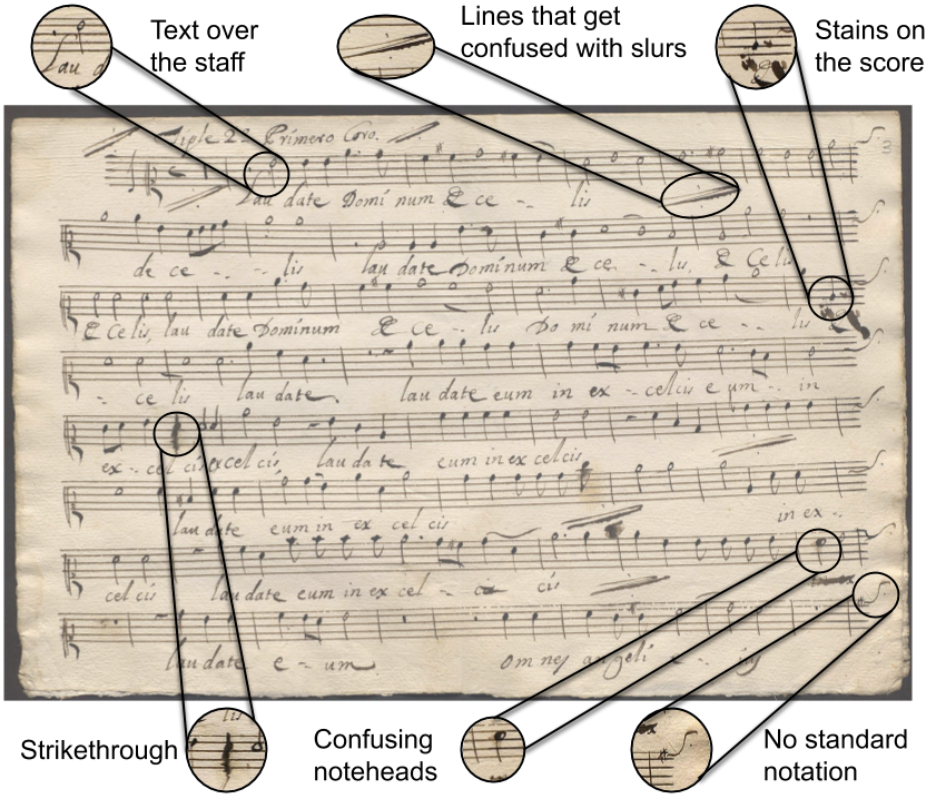


Figure 3.6: Page example from the historical dataset.

3.2.4 Printed measures for Object detection and Graph recognition

As stated before, to recognize music scores one must deal with the two dimensional nature of music. It is read from left to right, but also from top to bottom. So,

to properly train deep learning models, we need a considerable amount of labelled music scores. Indeed, the problem of current datasets is that they are not suited for training GNNs. First, because the few existing datasets labelled at graph level are very small (such as MUSCIMA++ [68]), and second, because most of them (e.g. DeepScores [133, 134]) do not have an accurate labelling of symbols and primitives in the score and the relationship between them.

For this reason, we have created a new dataset in order to train our GNN model. Concretely, we have created a new printed-synthetic dataset using Lilypond [2]. This dataset contains more than 18K measures. We have splitted the dataset between train, validation and test in the following way: 60% Train, 20% Validations and 20%Test. Table 3.3 shows the number of primitves per each symbol that appears in the dataset. And Table 3.4 shows the different relations that the dataset has and how many we can find in the dataset. Finally, we provide the images and the groundtruth in the following formats:

1. At primitive level in COCO format[85] for object detection task.
2. At primitive level including the position in the score in case the symbol has pitch (e.g notehead line 4).
3. At primitive level in XML format including the position in the score in case the symbol has pitch and the relations between primitives, so that one can build the music symbols (e.g. compound notes). Figure 3.7 shows an example of a music measure groundtruth. The arrows state the relationship, taking the notehead as the centre of the union of a musical symbol.

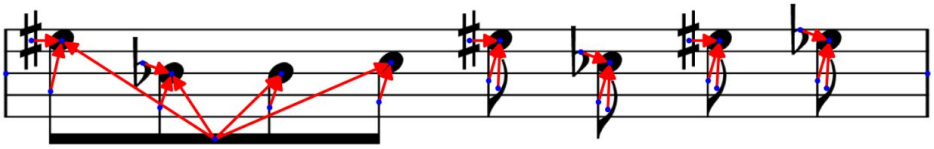


Figure 3.7: Groundtruth of a music measure.

[2]https://lilypond.org

Table 3.3: Atomic symbols and number of appearances in the dataset.

Symbol	Number of appearances
f-clef	2462
thin_barline	37505
timeSig_2-2	2837
stem	98417
notehead-full	91338
8th_flag	12560
sharp	23962
natural	10957
quarter_rest	4790
half_rest	4484
flat	24272
16th_flag	20084
c-clef	2970
beam	16068
notehead-empty	7309
timeSig_cut	2595
8th_rest	3358
16th_rest	4436

Table 3.4: Number of relations between primitives.

Source	Target	Number of relations
notehead-full	stem	91338
notehead-full	8th_flag	12560
notehead-full	sharp	22155
notehead-full	natural	10083
notehead-full	flat	22472
notehead-full	16th_flag	20110
notehead-full	beam	64218
notehead-empty	flat	1800
notehead-empty	stem	7079
notehead-empty	sharp	1807
notehead-empty	natural	1

Part I

Neural Network-based Optical Music Recognition Methods

With the emergence of Deep Learning, Neural Networks have performed very good results recognizing text. In this part, we explore the adaptability of text recognition models to deal with the sequentiality of a music staff. In particular, methods based on Long Short-Term Memory neural Networks and Sequence to Sequence models will be explored.

4 Optical Music Recognition based on Long Short-Term Memory Neural Network

Rhythm and harmony find their way into the inward places of the soul.

– Plato

Many music scores are written in a single staff, and therefore, they could be treated as a sequence. Therefore, this chapter 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. For training, we have used a synthetic dataset of more than 40,000 images, labeled at primitive level. The experimental results are promising, showing the benefits of our approach.

4.1 Introduction

Traditionally, OMR has been approached considering multi-stage systems [116]. The different stages comprise several small sub-tasks such as image binarization [108], staff-line removal [54], or music symbol classification [98]. This chapter, however, focuses on directly recognizing the music content appearing on an image.

We do assume that the image depicts 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 [64]. Note that this is not a strong assumption, as there exist algorithms that achieve good performance for both isolating staff sections [31] and separating music and lyrics (accompanying text) [18]. Similarly, one can assume that 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 [63] or handwriting [64] recognition. The Recurrent 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. To avoid the vanishing gradient

problem, Long Short-Term Memory (LSTM) units are considered. LSTM are able to learn long-term dependencies and can avoid the vanishing gradient problem. They can keep information for long time. Moreover, Bidirectional LSTMs are used to benefit from context information.

The rest of the chapter is organized as follows. Section 4.2 describes the method, whereas Section 4.3 analyzes the results. Finally, the conclusions are drawn in Section 4.4

4.2 Long Short-Term Memory Networks

Single staff sheet music can be seen as a sequence. In this way, a music score is read from left to right. In order to automatically process the music score and take into account the sequence of music symbols, a *Recurrent Neural Network* (RNN) seems an appropriate tool. In this work, we propose to make use of *Long Short-Term Memory* (LSTM) [75] networks. LSTMs have the ability to decide which information has to be kept as context and which information has to be removed, *i.e.* forgotten.

Figure 4.1 shows the different stages of our proposed pipeline. Firstly, the input music scores are preprocessed (Subsection 4.2.1). Afterwards, each column of the image is processed by an LSTM network (Subsection 4.2.2). The output of the LSTM is passed by two fully connected layers in order to distinguish between rhythm and pitch (Subsection 4.2.3). Finally, the output of the system is the recognition of symbols, including rhythm and pitch (Subsection 4.2.4). The different steps of this pipeline are explained in the following sections.

4.2.1 Input

The proposed architecture is trained by batches of images that are resized to a fixed height of 50 pixels. Then, these images are fed into the proposed model using pixel-wise columns. The maximum width can be variable depending on the widest image in the batch. Therefore, images with a shorter width are padded with 0's to the maximum width of the batch. In this work, the staff lines have not been removed in order to avoid noise and distortions in the musical symbols. In addition, staff lines provide useful information in terms of the pitch. Note that features are not extracted from the image in order to maintain the spatial information and the spatial order as much as possible.

4.2.2 Long Short-Term Memory

A LSTM network has been used in order to recognize the elements of the sheet music. In this work, we use a bidirectional network to increase the performance and

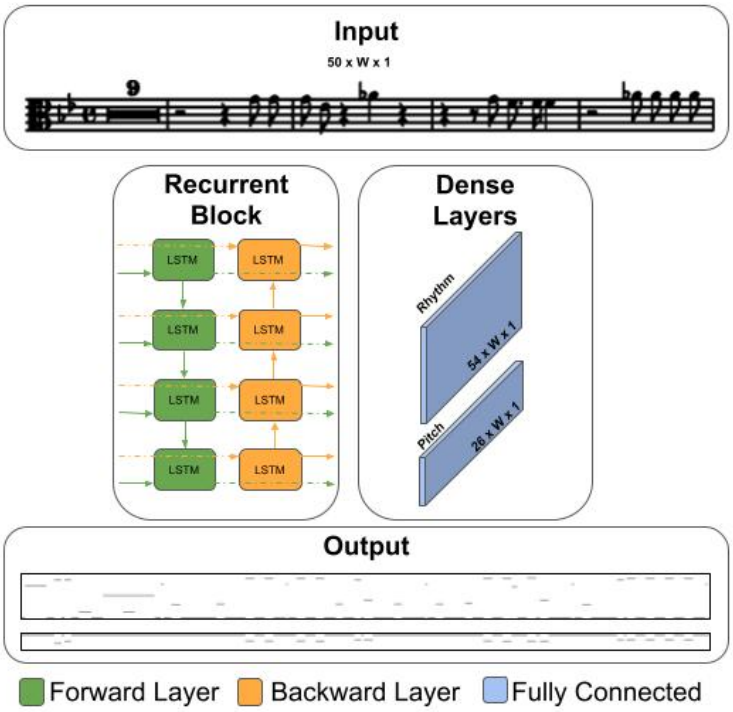


Figure 4.1: Architecture of the network

reduce ambiguities when recognizing some symbols. The combination of forward and backward pass allows to recognize symbols that may be confused if only one direction is used.

The proposed LSTM network is composed by 3 recurrent layers with a hidden state size of 128. We have trained our architecture for 100 epochs with a batch size of 128 or 64 for LSTM and BLSTM respectively. These values have been experimentally set. It must to be said that the network is trained column by column so it predicts one output per column. In other words, the output will end up being as long as the input image.

4.2.3 Fully Connected Layers

At the end of the LSTM network, we propose to use two heads in order to separately predict rhythm and melody. Therefore, after the LSTM output, two fully connected layers (FC) are used to obtain two different outputs. The reason to split the output in two parts is that the number of combinations between melody and rhythm is very high. In case of using a single output, all possible combinations of

rhythm-melody must be created as possible classes. Therefore, we propose to exploit the idea that rhythm and pitch can be independent. Thus, rhythm is decided by the symbol whereas the pitch depends on the position with respect the staff lines. Following this idea, many more examples are available to train our system.

4.2.4 Output

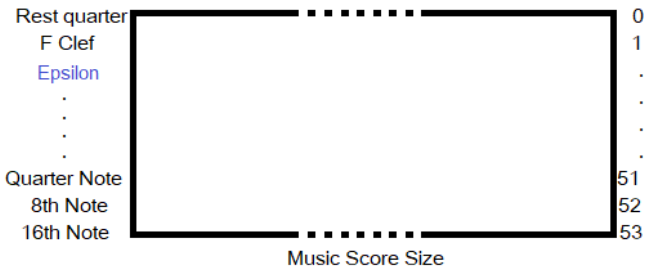
After the FC layers, the next step is to calculate the loss and backpropagate the errors. Even though two heads are used to separate between rhythm and pitch, the output of our system should be able to deal with multiple classes per time step. Therefore, in validation and test, a threshold is applied to both outputs (Rhythm and Pitch) in order to obtain the corresponding classes. The outputs and the ground-truth of each music score is represented by two binary matrices, one for the rhythm and another for the melody or pitch. Horizontally, it corresponds to the width of the input image whereas the vertical axis tells the different classes for symbols and pitch, 54 and 26 respectively. Pitch corresponds to locations in the staff. Figure 4.2 shows the structure of the matrix for both melody 4.2a and rhythm 4.2b and Fig. 4.3 shows a real example with its corresponding groundtruth. The corresponding pixels where the symbol is located in the music score will be activated in both matrices indicating which symbols are activated in each time step *i.e.* pixels. The following symbols have been manually added to ease 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 4.3 blue marks). This symbol appears in both the rhythm and pitch ground-truths.
- *No note* is used to indicate that a symbol has not any pitch. This symbol only appears in the pitch ground-truth.

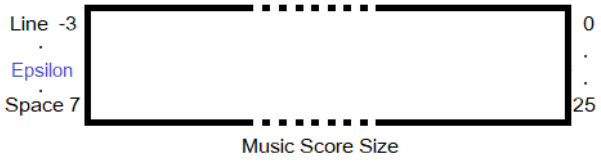
Finally, these outputs are converted into an array. One with the detection of the rhythm, another for the pitch and the last one with the combination of rhythm and pitch. These arrays will be used to evaluate the method.

4.2.5 Loss Function

In music, we can find one or more symbols in one instance of time, for example, chords or time signature (see Figure 4.3 red marks). Therefore, a multilabel loss function has to be chosen to deal with the before-mentioned problem. In other words, the loss function must allow more than one activation per time step. Thus, the softmax activation function cannot be used because it is thought for single-label classification problems. In this work, two different loss functions have been



(a)



(b)

Figure 4.2: Output representation for Rythm (a) and Pitch (b).

used: On the one hand, SmoothL1Loss creates a criterion that uses a squared term if the absolute element-wise error falls below 1 and an L1 term otherwise (Equation 4.1). On the other hand, MultiLabelSoftMarginLoss creates a criterion that optimizes a multi-label one-versus-all loss based on max-entropy (Equation 4.2). The loss is calculated independently for rhythm and melody. Once both losses are calculated, they are summed and backpropagated.

$$\text{SmoothL1Loss}(x, y) = \frac{1}{n} \sum \begin{cases} 0.5(x_i - y_i)^2, & \text{if } |x_i - y_i| < 1 \\ |x_i - y_i| - 0.5, & \text{otherwise} \end{cases} \tag{4.1}$$

$$\begin{aligned} \text{MultiLabelSoftMarginLoss}(x, y) = &- \sum_i y[i] \cdot \log\left(\frac{1}{1 + e^{-x[i]}}\right) \\ &+ (1 - y[i]) \log\left(\frac{e^{-x[i]}}{1 + e^{-x[i]}}\right) \end{aligned} \tag{4.2}$$

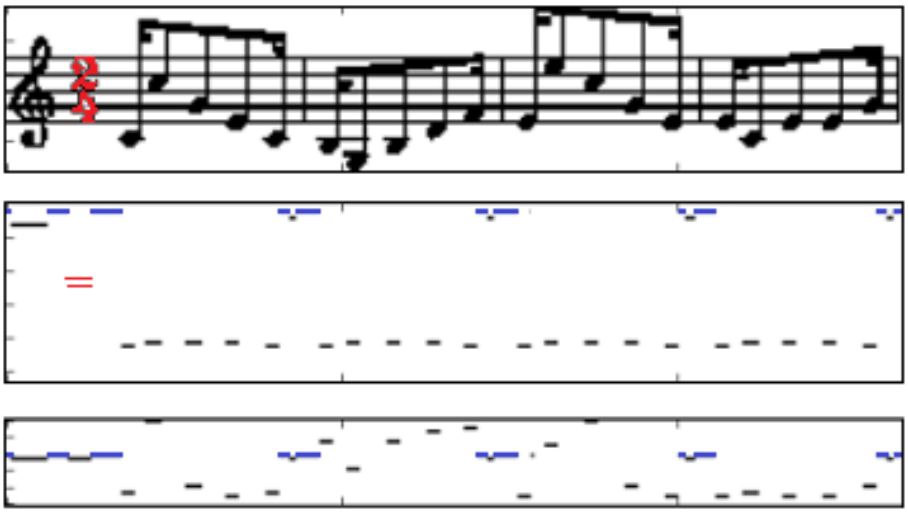


Figure 4.3: Example of Music Score and the corresponding Ground-truth in a Binary Matrix. The first row is the music score. The second row is the Rhythm Ground-truth. The third row is the Pitch Ground-truth.

4.3 Experimental Validation

This Section presents and discusses the experimental results.

4.3.1 Dataset

A Synthetic dataset has been used to train the network. This collection is composed of more than 50000 music scores with 3 different typographies. The dataset corresponds to incipts from the RISM catalog [1]. It is composed of almost 50000 music scores with 3 different typographies. The staffs are divided in 60% (29815) for training, 20% (9939) for validation and 20% (9939) for test.

4.3.2 Evaluation

The evaluation of a complete OMR system is not well defined in the literature. Thus, we propose to follow the evaluation described in [135]. The authors proposed to evaluate three aspects of the framework; pitch, rhythm and their combination. Note that the combination of pitch and rhythm corresponds to the performance of the whole system. The chosen evaluation metric is the *Symbol Error Rate* (SER)

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

applied to an array produced by the system. Note that a threshold is applied to convert the output of the FC layers to an array of symbols.

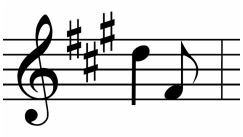


Figure 4.4: Example of music score.

An example of the format of the three output arrays, corresponding to Figure 4.4, 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][2].
- Rhythm+Pitch: [[gClef, noNote], [accidental sharp, L5], [accidental sharp, S3], [accidental sharp, S5], [quarter note, L4], [eight note, S1], [bar line, noNote]].

Symbol Error Rate (SER)

This metric is based on the well-known Word Error Rate (WER) metric [52] used in speech and text recognition. SER also uses the Levenshtein distance. The main difference between them is that the Levenshtein distance computes the differences at character level, WER does it at the word level and SER does it at symbol level. In the case of music scores, given a prediction and a reference ground-truth, the SER is defined as the minimum number of edit operation *i.e.* insertions, substitutions and deletions, to convert one array into the other.

$$SER = \frac{S + D + I}{N} \tag{4.3}$$

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

$$SER(i,j) = min \begin{cases} SER(i-1,j) + 1 \\ SER(i,j-1) + 1 \\ SER(i-1,j-1) + \Delta(i,j) \end{cases} \tag{4.4}$$

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

[2] L = Line; S=Space; L1 is the bottom line on the staff and S1 is the space between line 1 and 2

Output's Threshold Evaluation

A threshold is applied to decide which symbols are activated at each time step. Note that this threshold is needed because we have no knowledge about the number of symbols appearing at each time step. This threshold has been experimentally set using a grid search from 0 to 1 and step of 0.1. We have selected the combination of rhythm and pitch Error Rate as a metric to choose the best threshold. Figure 4.5 shows the evolution of the error rate depending on the threshold. As we can see, the best threshold is 0.5 even though 0.4 achieves similar results.

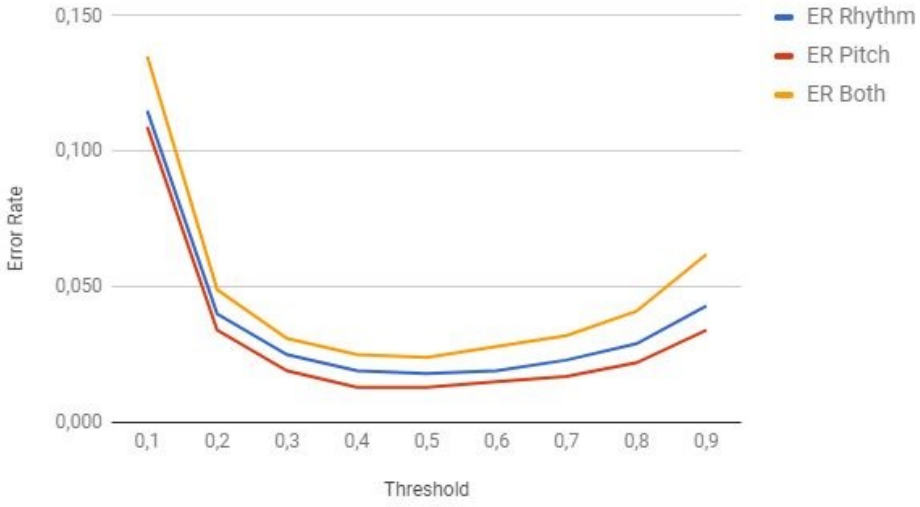


Figure 4.5: Evaluation of the best threshold in terms of Error Rate: Rhythm, Pitch and Rhythm+Pitch.

4.3.3 Results

All the results that are shown in this section are obtained using *Adam* as optimizer with a learning rate of 10^{-4}. In this work, the PyTorch [3] library has been used in order to build the proposed framework.

Table 4.1 shows an error rate comparison in terms of the average and standard deviation among 5 runs. In this comparison, single directional and bidirectional LSTM are analyzed with the two described loss functions. The first column shows the loss function and the network that has been used, the second one shows the error rate of the rhythm, the third one the results concerning the pitch; and the last column shows the results when considering the rhythm and pitch jointly. Note that the BLSTM produces better results. Moreover, regarding the loss function,

[3] http://pytorch.org/

the Smooth L1 function obtains better results with 1.5% SER when recognizing the pitch, 2% SER when recognizing the rhythm and 2.8% SER when recognizing the pitch an rhythm jointly. Using a bidirectional network, the input is processed in both directions. Thus, it obtains information of the whole symbol, and becomes more accurate. For example, if one direction recognizes a note-head, the other direction can discard that the vertical line that it is reading is a bar line, but instead a note stem (both stems and bar lines are straight vertical lines).

Table 4.1: Results using LSTM and BLSTM. All results are between [0-1] given in error rate (ER). The first number is the mean of the five executions and the number between parenthesis is the standard deviation

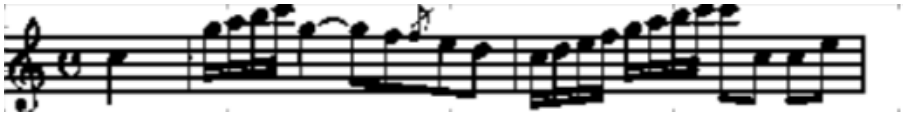
	Rhythm (R) Symbol ER	Pitch (P) Symbol ER	R + P Symbol ER
LSTM Smooth L1	0.326 (± 0.007)	0.293 (± 0.008)	0.426 (± 0.009)
BLSTM Smooth L1	**0.020** (± 0.001)	**0.015** (± 0.001)	**0.028** (± 0.002)
LSTM Multi Label Soft Margin	0.431 (± 0.017)	0.567 (± 0.051)	0.747 (± 0.063)
BLSTM Multi Label Soft Margin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

In Figures 4.6 and 4.7 we can see some qualitative results. First subfigure shows the input of the system. The second, third and fourth subfigures correspond to the Rhythm ground-truth, output and thresholded output respectively. In the fifth, sizth and seventh subfigures we can see the Melody ground-truth, output and thresholded output respectively.

4.3.4 Comparison with a commercial OMR software

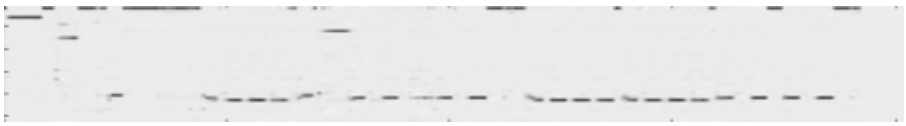
The proposed method has been compared with PhotoScore [4], a commercial OMR software able to recognize printed and also handwritten music scores. Figure 4.8 show qualitative results. Note that this comparison might not be completely fair. PhotoScore has some features to improve its performance that are not considered in our method. PhotoScore probably uses syntactic rules for validation. For instance, the commercial 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 our work, no syntactic rules have been applied. This is an important

[4] http://www.neuratron.com/photoscore.htm



(a) Input image

(b) Rhythm ground truth



(c) Rhythm Output

(d) Rhythm Output with threshold higher than 50%

(e) Melody ground truth

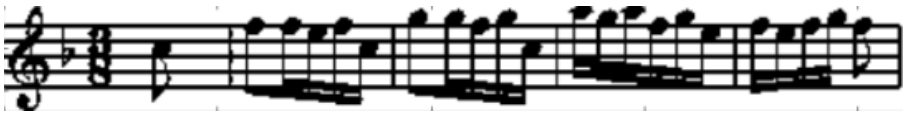
(f) Melody Output

(g) Melody Output with threshold higher than 50%

Figure 4.6: Qualitative Results Example using LSTM.

difference because we do not correct any miss-classification using music notation rules.

Figure 4.8 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.



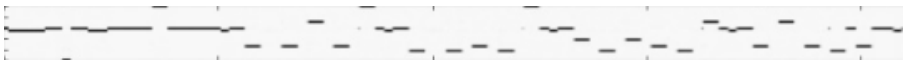
(a) Input image

(b) Rhythm ground truth

(c) Rhythm Output

(d) Rhythm Output with threshold higher than 50%

(e) Melody ground truth



(f) Melody Output

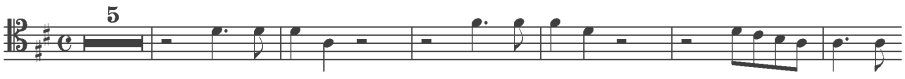
(g) Melody Output with threshold higher than 50%

Figure 4.7: Qualitative Results Example using BLSTM.

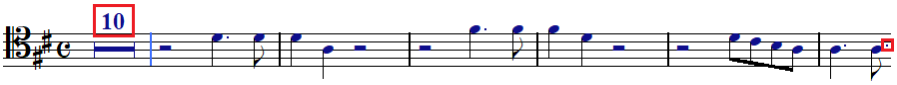
4.4 Conclusions

In this chapter, we have proposed an optical music recognition method that deals with single staff sheet music as a sequence making use of (B)LSTM networks.

The obtained results show that single staff music scores could be recognized by means of RNN. We have also shown the benefits of using a BLSTM instead of an LSTM applied to musical images. However, the recognition of scores as sequences has some limitations. For instance, more complex music scores (e.g. scores with



(a) Original Image



(b) Visual Result of PhotoScore

Figure 4.8: Recognition of a music score using the PhotoScore Commercial OMR software. The errors are shown in red color.

multiple voices) require further research.

In the next chapter we will improve the current method in order to tackle the handwritten music scores.

5 Optical Music Recognition based on Convolutional Recurrent Neural Network for Handwritten Scores

I know nothing with any certainty but the sight of the stars makes me dream.

– Vincent Van Gogh

Optical Music Recognition systems achieve good performance under relatively good conditions. However, their accuracy dramatically decreases when dealing with handwritten scores, mainly because of the high variability in the handwriting style. In this chapter, we propose a full Handwritten Music Recognition (HMR) system based on Convolutional Recurrent Neural Networks, data augmentation and transfer learning, that can serve as a baseline for the research community.

5.1 Introduction

Although the interest in OMR has reawakened with the appearance of deep learning, as far as we know, the few existing methods that attempt to recognize handwritten scores are mostly focused on solving a particular stage of OMR, such as layout analysis [20] or detection and classification of graphic primitives [67] or music symbols [97, 133].

Therefore, in this chapter we propose a full staff-wise Handwritten Music Recognition (HMR) system, which can serve as a baseline for future improvements in this research field. Our architecture is based on Convolutional and Recurrent Neural Networks. This work is based on the method explained in the chapter 4, where we addressed OMR for printed scores as a sequential recognition task, disentangling the output of the network in the two main components of music notation: rhythm and pitch. In the present chapter, we improve this architecture to deal with handwritten scores, and we show its viability both in printed and handwritten scenarios. We have observed that there are not complete OMR systems for handwritten scores on Western notation yet. There only exist successful approaches for sub-stages of the process. Nevertheless, these methods are based on the detection of music symbols, instead of the full OMR pipeline. Moreover, the reported re-

sults might not be really convincing because the MUSCIMA++ dataset is a subset of the CVC-MUSCIMA dataset [48], which was created for writer identification. Since the above mentioned works randomly split the pages into train, validation and test partitions, using writer-independent partitions only, the same music work could appear in the training and test sets at the same time, with the only difference of being written by different persons. Consequently, the system could be biased towards the recognition of these specific sequences of melodies and rhythms.

Then, we have improved our method in the following way: First, we add Convolutional Neural Networks as feature extractor. Secondly, since the existing amount of annotated handwritten music scores is scarce, we propose a novel data augmentation technique, and incorporate transfer learning from printed scores. Finally, we also share the handwritten data that we have created (section 3.2.1) that has been manually labelled for the experimental evaluation.

The remainder of this chapter is organized as follows. Section 5.2 describes our architecture. Section 5.3 explains how we deal with few handwritten data. Section 5.4 discusses the results, and conclusions are drawn in Section 5.5.

5.2 Convolutional Recurrent Neural Network

Many music scores, including polyphonic ones, are written using a single staff. Therefore, we propose to read each staff as a sequence, similar to text recognition [52], by using Long Short-Term Memory (LSTM) Recurrent Neural Networks (RNN). Although they can extract information directly from image pixels, we incorporate Convolutional Neural Networks (CNN) as image feature extractor. Figure 5.1 shows an schema of our architecture. The different stages are described next.

Input: In this chapter, we assume that the music scores pages have been previously segmented into staves. The segmented staves correspond to binary images resized to a height of 100 pixels in order to feed pixel columns of the same size to the network. The aspect ratio will be kept, therefore the width will change for each batch. The images of the same batch are padded according to the longest staff in the batch.

Convolutional Block: The convolutional block is composed by three convolutional layers increasing the depth and kernel size of 3x3, followed by Batch Normalization [76] and Rectified Linear Unit activation [58]. Finally a max-pool 2x1 operator is used to reduce the vertical dimension while keeping the same image width. In other words, the output of the Convolutional Block will have the same width as the input image.

Recurrent Block: This block uses Bi-directional LSTM networks (BLSTM) [75] to benefit from context when recognizing each symbol. Compared to RNNs, LSTMs are able to learn long-term dependencies, avoiding the vanishing gradient

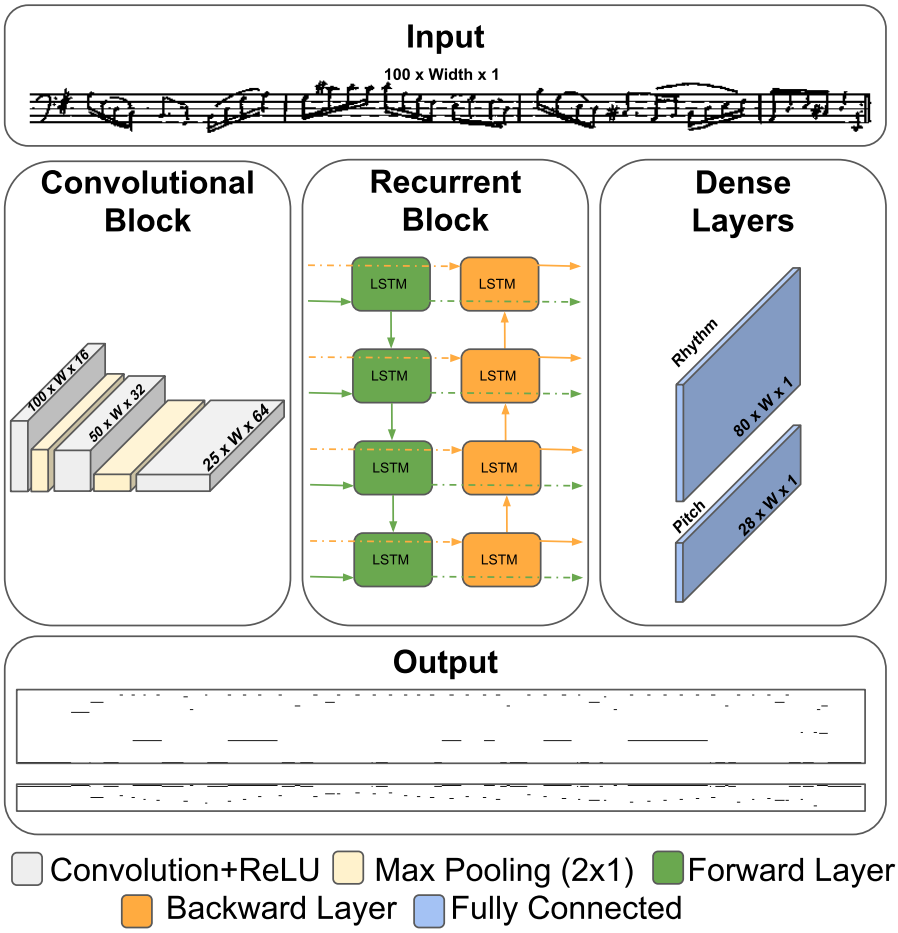


Figure 5.1: Architecture of our method. Each staff is the input of the convolutional block to extract features, and then, it passes the recurrent block. Finally, two fully connected layers separate the rhythm and melody.

problem, and keeping information for longer time. Moreover, the bi-directionallity provides an extra information that reduces ambiguities, because it takes into account the forward and backward directions. For example, if one direction is reading a vertical line and the other direction is seeing a notehead, the network can correctly predict a quarter note. Figure 5.2 shows an example of the ambiguity reduction provided by both directions. In our architecture, we use four BLSTM layers of 512 neurons each.

Dense Layers: After the recurrent block, we incorporate two fully connected (FC) layers. In this way, we will obtain two outputs: one for the rhythm and one for the pitch. If we had one single output, we should consider any combination of

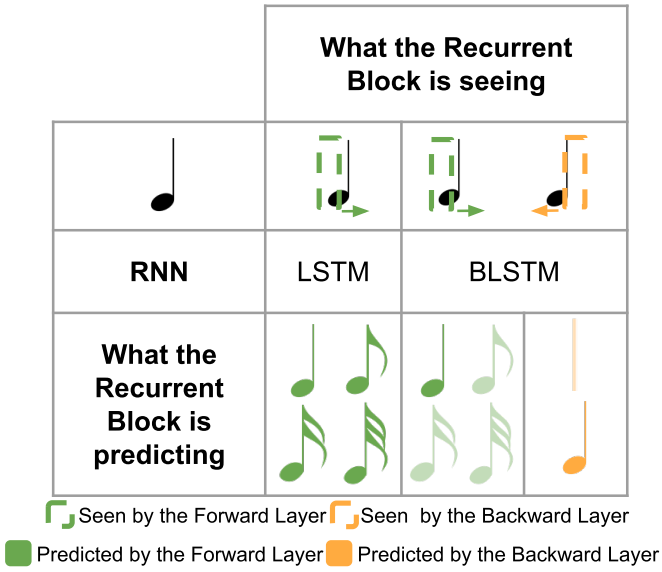


Figure 5.2: BLSTM predictions. The backward direction helps to reduce the ambiguities when predicting a symbol.

pitch-rhythm as a different class, which would become in a very large number of classes. Another reason to separate pitch-rhythm and consider them independent, is that we can obtain many more examples of each class to train. For instance, the system learns the shape of a 16th note, no matter its pitch. Please note that here we define the pitch as the location of the note in the staff (e.g. the note is located on the third staff line), instead of the real pitch (e.g. C4 note), because it depends on the clef. Also, in this way, we can represent all pitches with few classes.

Output: Finally, the output of each dense layer is a matrix, whose columns are symbol and pitch probabilities per pixel column in the original image. Each matrix has the same width as the original image and has a height of 80 classes for the rhythm and 28 classes for the pitch. By thresholding these matrices, we can decide which symbols appear in the music scores. In the last chapter, based on our previous work [7], we performed an exhaustive analysis where we evaluated several thresholds. The one which provided the best performance was 0.5. In other words, the network has to be at least 50% confident when recognizing each symbol. Note that more than one symbol may appear at the same time step (column). Two symbols have been manually added to ease the recognition:

- Epsilon (ε) is used to know where a symbol starts and ends. If ε is activated, none of the other symbols can be activated. This symbols works as a separator.
- *No note* is a symbol only found in the pitch matrix. When this symbol is

activated it means that the symbol activated in the rhythm matrix (at the same instance of time) has not pitch (e.g. symbols without pitch, such as rests).

Finally, these outputs are converted into an array, combining the rhythm and pitch. These arrays will be used to evaluate the method at rhythm and pitch level and also to evaluate the complete system, where both parts should be predicted correctly.

As it has been stated, in OMR several symbols can appear at the same time stamp (e.g. chords, time signature, etc.). Hence, several labels can be predicted at the same output step. For this reason, we choose the Smooth L_1-loss function. Concretely, our architecture has been trained using the *Stochastic Gradient Descent* (SGD) optimizer with Momentum and weight decay *i.e.* L_2 regularization. The Smooth L_1-loss has been used as objective function defined as

$$\mathcal{L}(x, y) = \frac{1}{n} \sum \begin{cases} 0.5(x_i - y_i)^2, & \text{if } |x_i - y_i| < 1 \\ |x_i - y_i| - 0.5, & \text{otherwise,} \end{cases} \tag{5.1}$$

where x is the output of the proposed architecture and y is the target we want to achieve. The proposed loss function can deal with multi-label problems being less sensitive than L_2-loss with respect to outliers.

5.3 Data Augmentation and Transfer Learning

This section describes the training strategies that have been used to exploit our architecture. As stated before, there is very few labelled handwritten data. Since little groundtruth data for training leads to overfitting problems, we propose two different strategies. First, we propose to apply transfer learning by fine-tuning a printed model with handwritten data. Second, we propose a data augmentation technique for music scores.

Transfer learning. Training our system with printed scores give insights of the suitability of the proposed approach for OMR. However, a model for printed scores may fail when recognizing handwritten scores due to the elastic deformations in handwriting styles. To overcome this issue, we propose to pre-train our model with printed scores, and then, fine-tune it with the few available handwritten data.

Data augmentation. To increase the amount and variability of training data, some distortions have been applied to both the printed and handwritten training sets. First, we have applied dilation, erosion and blurring distortions. Note that this data augmentation has been randomly applied for each music score. Beside the morphological operations, the number of handwritten music scores in the training set has been increased by shuffling the bar units. For this purpose, we crop each measure (bar unit) and shuffle among the different measures of the staff, with

the exception of the first and the last bar unit. These two measures are fixed because the first one contains the clef and the time signature whereas the last one can contain a final barline. Figure 5.3 shows the different data augmentation techniques applied to each dataset. Note that this shuffling also prevents the model to learn a specific melody and rhythm.

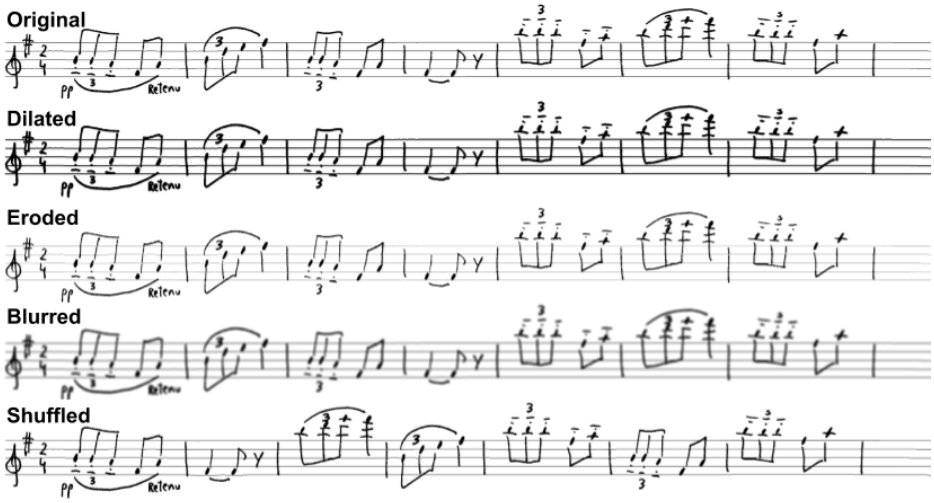


Figure 5.3: Different techniques of data augmentation. Dilating, eroding and blurring have been applied to both datasets, printed and handwritten ones. Shuffling is only applied to the handwritten dataset.

5.4 Experimental Validation

This section experimentally validates the performance of our architecture. As it has already mentioned, we propose to firstly train a model able to recognize printed musical scores and later transfer this learning to handwritten data. Hence, two datasets have been used.

5.4.1 Datasets

Printed dataset: we use a subset of PrIMuS dataset [26], which consists of rendered incipts from the RISM[1]. It is annotated at primitive level *i.e.* the symbols are labelled as noteheads, steams and flags, among others instances instead of quarter notes, 8th notes, 16th notes and such on. This dataset is latter converted

[1] http://rism.info

into symbol level. Our set contains almost 50,000 music scores rendered with 3 different typographies.

Handwritten dataset: 20 pages of MUSCIMA++ dataset [68] have been selected al labeled manually to test this approach. For more information see section 3.2.1.

5.4.2 Evaluation

We use the Symbol Error Rate (SER) [26, 30, 135] metric. Similarly to Word Error Rate (WER) [52], commonly used in text recognition community, SER is computed as the Levenshtein distance: the sum of edit operations that are needed to convert the output of our method into the groundtruth in terms of symbol insertions (I), substitutions (S) and deletions (D). Formally,

$$SER = \frac{S + D + I}{N}, \tag{5.2}$$

where N is the number of symbols in the ground truth. The lower this value, the better.

To perform the evaluation at different levels, we propose to evaluate Rhythm and Pitch separately. Therefore, we will provide the SER for both outputs of the proposed architecture. Finally, both outputs are merged and the SER for pairs Rhythm and Pitch (considered as one symbol) is provided.

5.4.3 Results on Printed Documents

We first evaluate our model in the printed scenario. Thus, we can test the suitability of our architecture in a controlled scenario. An ablation study has been performed to test several architecture details. Table 5.1 presents this study in order to evaluate the importance of the BLSTM recurrent block, CNN features and Data augmentation. Moreover, we compare the current work with our previous work [7].

As expected, the best configuration uses a CNN to extract image features containing richer information than merely using pixel columns. Moreover, the BLSTM provides more context information and improves the previous approaches. Finally, data augmentation slightly improves the performance whereas making it more robust to the initialization. The first row shows our previous work, while the last row shows the best configuration of the current work. The main difference is that here we propose to incorporate a convolutional block before the recurrent layers, and we have increased the number of neurons from 128 to 512 and layers from 3 to 4. In this way, we obtained a better performance (the SER decreases from 0.028 to 0.003 when we consider the rhythm and pitch together).

Table 5.1: Results on printed documents. 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. The first row corresponds to our previous work, the others are results of the current architecture.

RNN	CNN	Data Augm.	Rhythm SER	Pitch SER	Rhy.+Pit. SER
BLSTM [7]	-	✓	0.020 (± 0.001)	0.015 (± 0.001)	0.028 (± 0.002)
LSTM	-	-	0.168 (± 0.014)	0.144 (± 0.011)	0.174 (± 0.012)
LSTM	-	✓	0.163 (± 0.009)	0.139 (± 0.013)	0.169 (± 0.008)
BLSTM	-	-	0.005 (± 0.002)	0.003 (± 0.001)	0.006 (± 0.002)
BLSTM	-	✓	0.005 (± 0.002)	0.002 (± 0.000)	0.005 (± 0.001)
BLSTM	✓	-	0.003 (± 0.001)	0.002 (± 0.001)	0.003 (± 0.001)
BLSTM	✓	✓	0.002 (± 0.001)	0.001 (± 0.000)	0.003 (± 0.001)

Table 5.2: Results on handwritten documents. 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.

Pre-train Printed	D. Augm. Printed	BLSTM	CNN	D. Augm. Handwritten		Rhythm SER	Pitch SER	Rhythm+Pitch SER
				Shuffle	Morph.			
-	-	-	-	-	-	0.826 (± 0.009)	0.709 (± 0.012)	0.899 (± 0.007)
✓	-	-	-	-	-	0.771 (± 0.021)	0.668 (± 0.021)	0.872 (± 0.016)
✓	✓	-	-	-	-	0.762 (± 0.019)	0.690 (± 0.004)	0.854 (± 0.019)
✓	✓	✓	-	-	-	0.523 (± 0.018)	0.464 (± 0.020)	0.610 (± 0.016)
✓	✓	✓	✓	-	-	0.493 (± 0.015)	0.396 (± 0.012)	0.559 (± 0.015)
✓	✓	✓	✓	✓	-	**0.476** (± 0.009)	**0.387** (± 0.008)	**0.545** (± 0.009)
✓	✓	✓	✓	✓	✓	0.490 (± 0.005)	0.393 (± 0.004)	0.554 (± 0.007)

5.4.4 Results on Handwritten Documents

As stated before, we aim to create a full staff-wise HMR system for handwritten music scores that can serve as starting point for future improvements in this field. Table 5.2 shows the results of our method using the selected pages of the MUSCIMA++ dataset. Note that each line introduces an improvement to the previous one. In the first row, we do not use any of the proposed improvements (no pre-training, CNNs, etc.). Observe that pre-training with printed data decreases the error (second row). Data augmentation on printed data helps a little bit. However, in the fourth row, we can see that the BLSTM is the key modification to reduce the error rates by 0.2 points. This is because of its ability to use context to minimize ambiguities. Then, the feature extraction based on CNN also helps to recognize the handwritten music scores (fifth row). By shuffling the measures (the sixth row) we obtain the best approach. Finally, in the last row, we observe that morphological operations for data augmentation only introduce noise and increases the error rates. The main reason for this behaviour could be that morphological techniques may make printed scores look closer to handwritten, but when these techniques are used in handwritten scores, the result may look unrealistic.

Table 5.3: Results on the handwritten documents, shown per page. 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.

	Polyph.	Rhythm SER	Pitch SER	Rhy.+Pit. SER
W. 17 - P. 1	-	0.528 (± 0.019)	0.349 (± 0.019)	0.594 (± 0.014)
W. 13 - P. 3	-	0.226 (± 0.018)	0.175 (± 0.008)	0.270 (± 0.016)
W. 15 - P. 10	✓	0.716 (± 0.017)	0.620 (± 0.010)	0.796 (± 0.018)
W. 10 - P. 18	✓	0.483 (± 0.018)	0.422 (± 0.008)	0.565 (± 0.013)

Using the best configuration in Table 5.2, we provide the results from each one test page in Table 5.3 (two of them are polyphonic). Note that each row corresponds to a different writer and different page.

5.4.5 Comparison with commercial OMR software

Since we could not find any complete OMR for handwritten scores in the literature, we could not make a quantitative comparison. However, we could find a commercial software for qualitative evaluation. Photoscore is a commercial software able to recognize handwritten and printed music scores. It must to be said that we do not know whether Photoscore uses any post-processing or grammar rules (detecting the time signatures might be counting the number of beats in each measure and validating the recognition) in the recognition, so the comparison could not be completely fair.

Figures 5.4-5.8 show some qualitative results comparing the Photoscore results with our method. We have used different colors to highlight the common mistakes of our method. The blue color is used when different symbols appear in the same column, and our method is not capable to relate each symbol with the correspondent pitch. Orange boxes show that some symbols, as accents, could confuse our system. For example, sometimes the method predicts that an accent is a notehead, thus it detects the notehead located higher up (see Fig. 5.8), whereas other times it can predict that accent is another notehead and detects a chord (see Fig. 5.7). In red we show when the system confuses some symbols because of shape, for example a text dynamics is confused by a quarter note. Finally, Fig. 5.6 shows in green the difficulties to detect all noteheads in a chord. In these images, please note that when we draw the output of our network, the compound music symbols have been manually joined for better visualization.

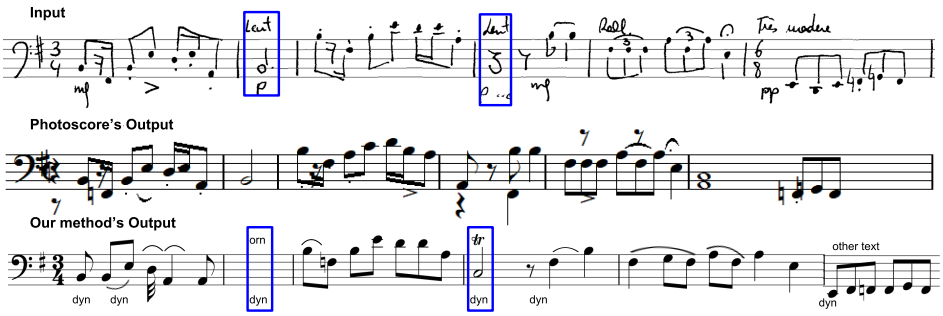


Figure 5.4: Qualitative comparison with Photoscore. Example of one staff of page 1. The blue box shows that our method is not able to recognize the symbols when several of them appear in the same column.

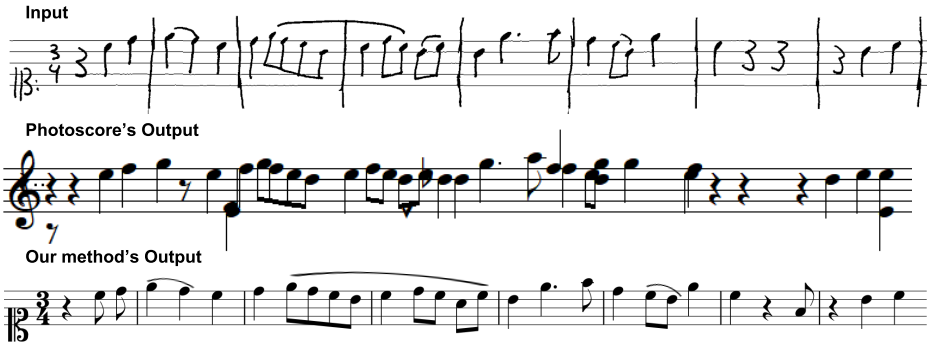


Figure 5.5: Qualitative comparison with Photoscore. Example of one staff of page 3. Contrary to Photoscore, note that our method could detect all the slurs.

5.4.6 Discussion

From these results, we could conclude that our methodology is valid and has shown to be able to recognize simple staves. From the qualitative point of view, bearing in mind that the Photoscore software might be using music rules for validation, our method obtains pretty good results. In fact, in many cases, our method outperforms Photoscore.

Concerning the quantitative results, although we are aware that the overall SER is close to 50%, these results are promising. First, we have used very few handwritten data, and secondly, we have not applied any grammar or rule to validate each bar unit.

Nevertheless, there are several limitations, most of them related to the way of labelling the data, which are described next.

Polyphonic music scores: The ground-truth is not able to relate which pitch

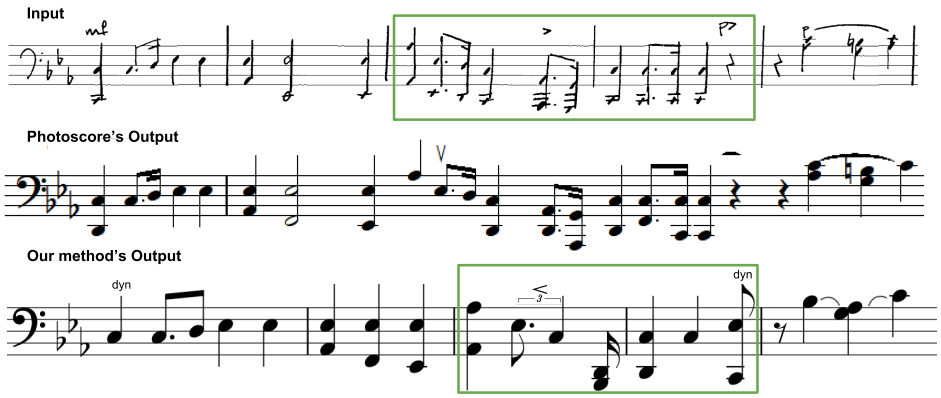


Figure 5.6: Qualitative comparison with Photoscore. Example of one staff of page 10. The green box shows that our method is not able to recognize all the noteheads in polyphonic music scores.

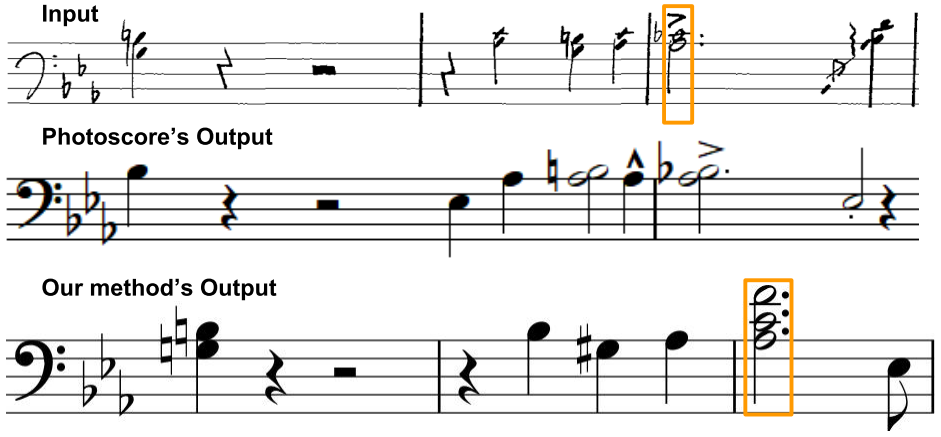


Figure 5.7: Qualitative comparison with Photoscore. Example of one staff of page 10. The orange box shows that our method could confuse some symbols by others by the position *i.e.* accents by noteheads.

corresponds to each notehead in the case that the rhythm within a chord (or polyphonic voices) is different (see Fig. 5.9 red symbols). However, it is able to recognize polyphony correctly if the rhythm is the same for all the symbols (see Fig. 5.9 green symbols).

Repeated symbols: If a symbol without pitch appears more than one time at the same time step, the method will only detect one (see Fig. 5.9 the blue slur will not be recognized).

Compound Music Symbols: The compound music symbols such as 8th notes,

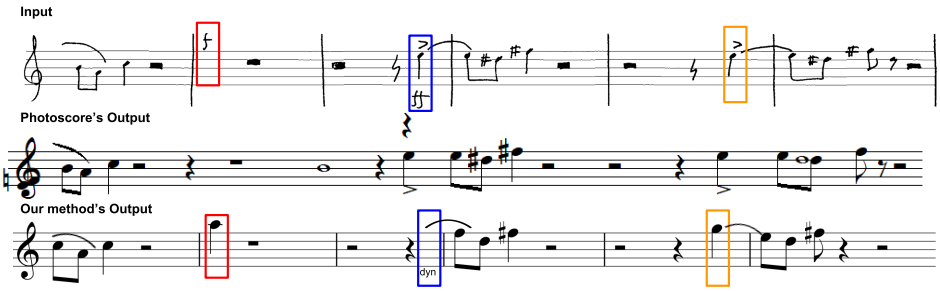


Figure 5.8: Qualitative comparison with Photoscore. Example of one staff of page 18. The red box shows that our method could confuse some symbols by others by the shape. The blue box shows that our method is not able to recognize the symbols when there are many in the same column. The orange box shows that our method could confuse some symbols by others by the position.

16th notes, 32th notes and so on, joined by a beam, will be separately recognized because there is no symbol for notating this *i.e* each notehead will have its steam and its flag, will not be joined by a beam.

Clef position on the stave: The ground truth does not provide the position of the clef on the stave. This means that a bass clef on the third or forth staff lines are predicted as the same.

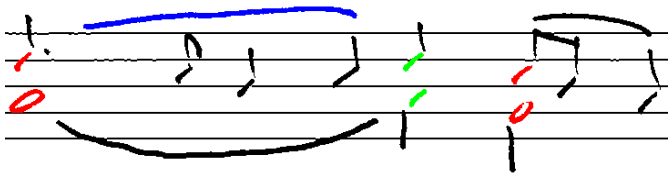


Figure 5.9: Method's limitations. In red polyphonic notes that could not be correctly recognized because they have different duration at the same time step. In blue the slur that will not be detected because there is another slur at the same time. In green, symbols that will be correctly detected because they have the same duration.

5.5 Conclusions

In this chapter, we have proposed a complete Handwritten Music Recognition (HMR) system based on CNNs and RNNs, data augmentation and transfer learning from printed scores. The experimental results have demonstrated the viability of this approach, showing that staves can be recognized as a sequence using BLSTMs, and also, that the convolutional block acts as an effective feature extractor. We have first demonstrated that our architecture is valid through the

evaluation over printed scores. Secondly, we have showed that our methodology greatly benefits from data augmentation from handwritten scores as well as transfer learning from printed scores.

Taking into account that we have used only 20 pages of the MUSCIMA++ database in the experiments, the results are promising. We hope that these results, together with our labelled data, can serve as a baseline for the community, fostering the research towards full OMR systems.

Of course, the incorporation of more handwritten data labelled at symbol level would help to obtain better results. For this reason, in the next chapter we will treat with an old music scores dataset, slightly bigger than the current dataset. As the dataset is more complicated, we will compare the current method with a proposed Sequence to Sequence model based OMR approach.

6 Optical Music Recognition based on Sequence-to-Sequence with Attention Mechanism for Old Scores

Old music used to mean something.
There is none of that today.

– Ziggy Marley

The recognition of old handwritten music scores remains a challenge because of the variabilities in the handwriting styles, paper degradation, lack of standard notation, etc. Therefore, the research in OMR systems adapted to the particularities of old manuscripts is crucial to accelerate the conversion of music scores existing in archives into digital libraries, fostering the dissemination and preservation of our music heritage. In this chapter we explore the adaptation of sequence-to-sequence models with attention mechanism (used in translation and handwritten text recognition) and the generation of specific synthetic data for recognizing old music scores. The experimental validation demonstrates that our approach is promising, especially when compared with long short-term memory neural networks.

6.1 Introduction

The research in OMR systems adapted to the particularities of old music scores is crucial to accelerate the process from its discovery to its digital transcription, enabling researchers to analyze, publicize and divulge unknown composers and compositions that traditional methods are forced to neglect. This paradigm shift from traditional musicological research -which is usually focused on the aesthetic assessment and compositional characteristics of a certain number of composers- far from opposing it, would become a fundamental tool to complement these studies. This would provide a much more accurate overview of the local characteristics of each music and its relation to other geographical contexts (transmission, influences, circuits, etc.).

Recognizing historical documents implies dealing with few labelled data. One solution is to train with synthetic data and refine with real handwritten one, as in [8]. Another solution is to explore unsupervised domain adaptation techniques, as proposed in [79] for handwritten text recognition. Although Adversarial Networks

for domain adaptation have been explored in *et al.* for recognizing music symbols, their application to music score recognition is still an open problem.

Concerning old scores, the are some works for mensural notation. For example, Calvo-Zaragoza *et al.* propose to use hidden Markov models and N-gram language models [29], whereas in [28] they opt for convolutional neural network with a recurrent neural network and language models. Pacha *et al.* [96] use a R-CNN with Inception ResNet V2 for music object detection and mensural scores. However, mensural notation is rather simple compared to western notation, so the research in OMR for historical documents dated from 17th-18th is still necessary. Moreover, the adaptation of the groundtruth to train an object detection method is very tedious. Methods like [96], not only need an expert to transcribe the music score, but also to annotate in the image score the position of each symbol. Note that such a detailed annotation is not needed for training recurrent neural networks, including sequence-to-sequence methods.

For the above reasons, in this chapter we propose an OMR system for old handwritten scores. Our method is based on sequence-to-sequence models with an attention mechanism, which have been successfully applied to translation and handwritten text recognition. As far as we know, this is the first OMR method based on sequence-to-sequence (seq2seq) model with attention mechanism adapted for historical music score recognition. Also, and since the lack of available transcribed scores for training deep learning systems pose a challenge, we also generate specific synthetic data that emulates the particularities of old scores, for example lyrics touch the stave or even the musical symbols (see section 3.2.2). The experiments demonstrate the suitability of our approach, especially when compared to Long Short-Term Memory Recurrent Neural Networks. Finally, the method has been tested in a real dataset that we have labeled (see section 3.2.3).

The rest of the chapter is organized as follows. Section 6.2 explains the architecture that we have used as baseline. Section 6.3 describes our proposed architecture. Section 6.4 relates how to deal with the lack of historical data. Section 6.5 discusses the results, and section 6.6 draws the conclusions.

6.2 Baseline: Long Short-Term Memory Neural Network

Before describing our Seq2Seq OMR architecture, we first describe our baseline, based on Long Short-Term Memory Recurrent Neural Networks [8, 26]. Note that our baseline is based on recurrent models because of the sequentiality of monopohonic music staves.

Although long short-term memory networks are capable of directly treating the raw image, the performance improves when adding a Convolutional Neural network (CNN) as a feature extractor. Thus, our baseline is composed of a Con-

volutional Neural Network and a bidirectional Long Short-Term Memory neural network (BLSTM) with Connectionist Temporal Classification (CTC) loss. Figure 6.1 shows the model architecture. The modules are described next.

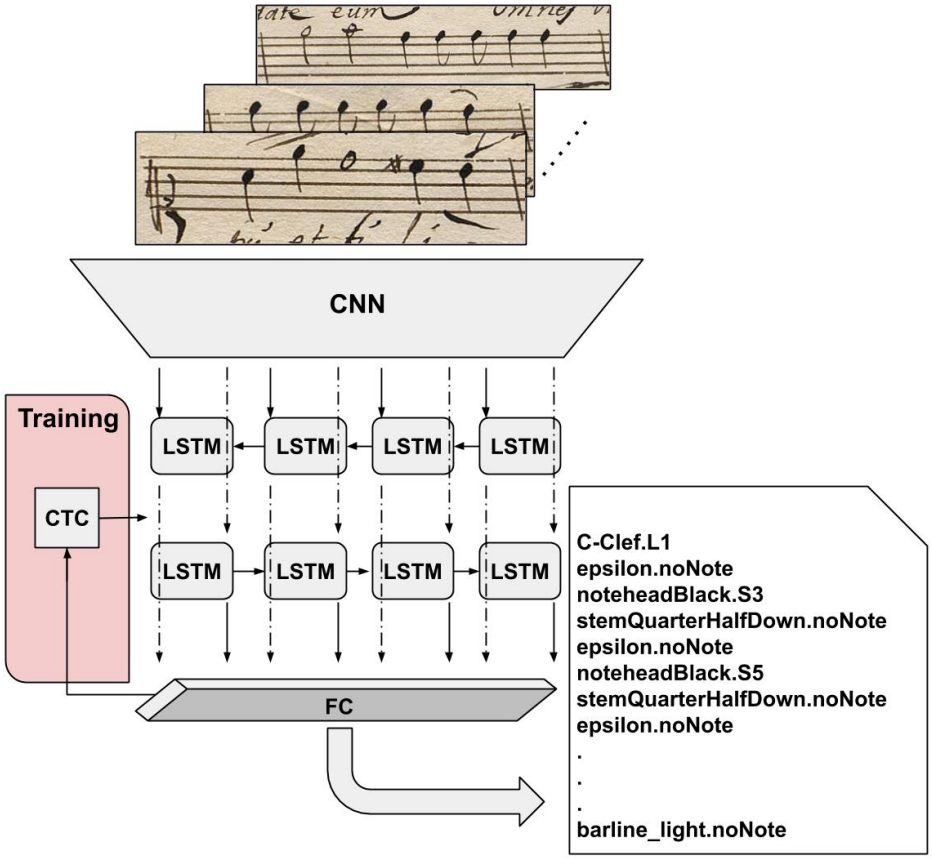


Figure 6.1: Convolutional Neural Network and Bidirectional Long Short-Term Memory model.

- **Convolutional Network:** In this step we extract the features that will be used in the next steps. The convolutional network is composed of the first three layers of the ResNet18 [70], consisting of convolution, batch normalization and rectified linear unit activation.

- **Bidirectional LSTM:** The BLSTM gets as input the features from the CNN. We use a LSTM to reduce the vanish gradient problem since LSTMs can remember information for longer time. We use bi-directonal LSTMs to

increase context information (from left and right sides in the image) and reduce the number of ambiguities.

- **Fully connected layers:** The results obtained by the BLSTM network are passed to a fully connected layer to return the final result.

- **Connectionist Temporal Classification:** This step helps to evaluate the output and check that the predictions are correct. As a loss function we use the Connectionist Temporal Classification (CTC) [62], which is trained using Stochastic Gradient Descent (SGD) optimizer with Momentum.

6.3 Sequence to Sequence

As explained before, music scores are written on staves following a sequence, so our approach is also based on recurrent models. Concretely, our method is based on the sequence-to-sequence (seq2seq) text recognition method [80], and adapted to music scores.

6.3.1 Sequence-to-Sequence model with attention mechanism

This methodology makes use of an attention-based encoder-decoder framework. Thus, our model consists of 3 components, the encoder, the attention mechanism and the decoder. Figure 6.2 depicts our proposed architecture for optical music recognition.

- **Encoder.** Given an input image, the encoder extracts high-level features encoding the contents of the image. These features will be later used to obtain the contents of the image in a machine readable format. In this work, the proposed encoder is implemented with a VGG-19-BN network [127] with pre-trained weights from ImageNet. Moreover, the last max-pooling is removed. Finally, the VGG features are reshaped into a two-dimensional feature map that will be further used as the input to a multi-layered Bidirectional Gated Recurrent Unit (BGRU) which provides extra positional information.

- **Attention Mechanism.** As an attention module, we use a location-based attention as proposed by Chorowski *et al.* [35]. This takes into account the location information explicitly for a better alignment. Otherwise, content-based attention expects the location information to be coded in the extracted features in order to differentiate the different representations of the features of the same symbol in different positions. The attention mechanism is in charge of aligning our feature representations with our decoding steps.

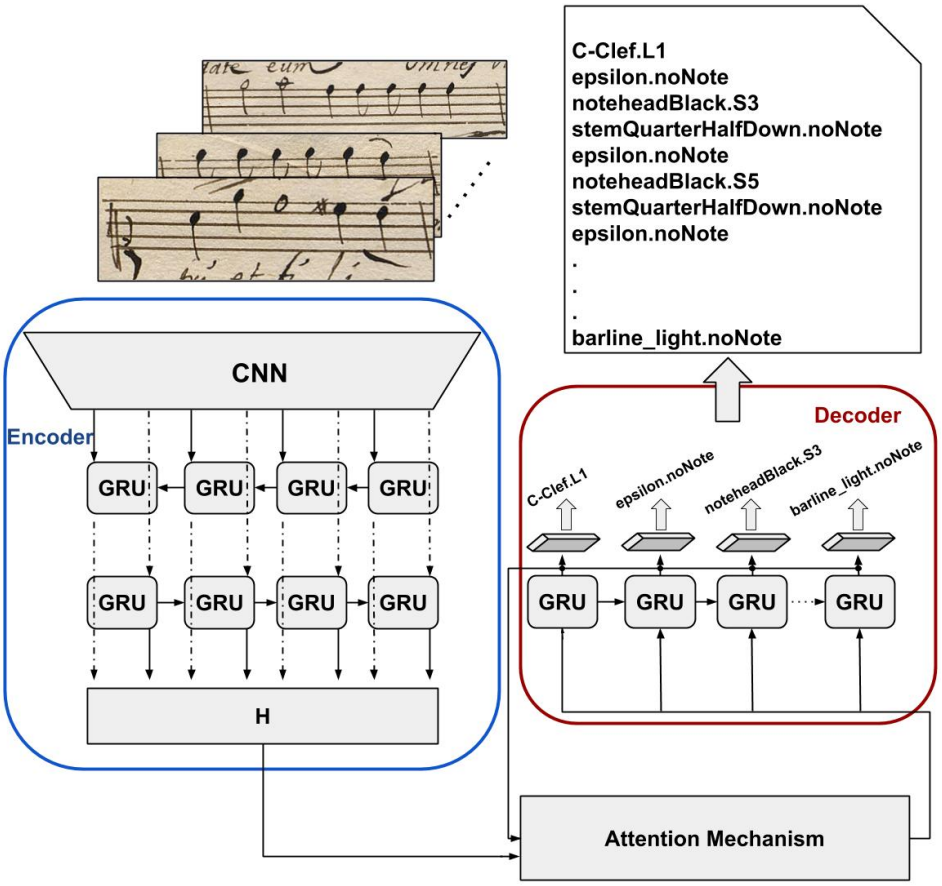


Figure 6.2: Sequence-to-sequence model with attention mechanism.

- **Decoder.** Finally, the decoder module is formed by a one-directional multi-layered GRU. The decoder provides the recognized symbols in several steps following a sequential order. At each time step the decoder GRU receives the concatenation of its previous embedding vector (in step $i-1$) and the current context vector (defined by the encoded features and our attention mechanism) in order to predict a new symbol. Moreover, to enhance the decoder we have used, on the one hand a multi-nominal which takes into account several decoding paths to obtain the final prediction and, on the other hand, label smoothing that allows a better generalization preventing over-confident predictions.

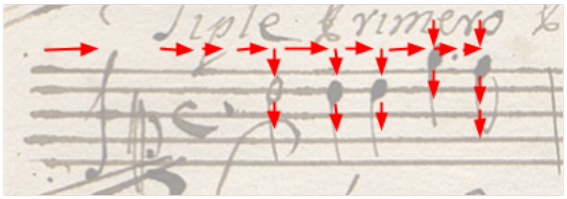
6.3.2 Adaptation to music scores

Recurrent methods, including sequence to sequence (seq2seq) models, have shown good performance when applied to handwritten text recognition (HTR). But the recognition of music is much more complex than text. The main reason is that the nature of text is one-dimensional: a sequence of characters. In music, however, we have to deal with two-dimensional sequences. First, music notes are composed of rhythm and melody. Second, some elements, such as ornaments or articulations, usually appear above or below notes. In addition, groups of notes (e.g. chords) or even other symbols (e.g. slurs or ties group notes, providing musicality to the work) appear at the same instant of time. Furthermore, music notation allows notes (e.g. 8th, 16th, etc.) to be written isolated or together (grouped using beams). Besides, an 8th note can have a stem looking up or down. Although isolated or compound symbols could seem the same to a non-expert user, a musician or musicologist makes differences (especially during the interpretation).

For the above reasons, we need to adapt the seq2seq model designed for text recognition to the particularities of music scores. It is true that, since music elements are located in a 2D space on the staff, these elements could be represented using a graph, such as in [68]. Thus, one possible solution is to treat the problem as a graph serialization task, which can be defined as the conversion of a 2D graph into a 1D string. In our case, music scores have been annotated at primitive level (i.e., note heads, stems, beams, flags, rests, etc.), so the output of our architecture will be a sequence of 1D music primitives. Therefore, we can solve the problem by defining a reading order, from left to right and from top to bottom, as illustrated in Figure 6.3. In a horizontal lecture, when we move one step in the staff (the position of the horizontal arrow), we use the symbol epsilon (ϵ) as a separator. Contrary, if the vertical primitives belong together (e.g. same symbol), they appear at the same time step, as denoted using vertical arrows.

6.4 Dealing with the lack of data

Deep learning methods are data hungry, i.e. they need a lot of labelled data to train. We have created two datasets detailed in section 3.2.2. These synthetic datasets are used to pretrain our system. We train our model using curriculum learning [13] for improving the performance. At the beginning, we train with few real historical measures and lots of synthetic ones. After n epochs, we increment the number of historical measures and decrease the synthetic ones. At the end, the training data is 100% historical.



C-Clef.L1~epsilon~timeSig_common~epsiln~halfRest~epsilon
~noteheadHalf.L4~stemDown~epsilon~noteheadBlack.L4~
stemDown~epsilon~noteheadBlack.L4~stemDown~epsilon~
noteheadBlack.S5~stemDown~epsilon~dot~epsilon~
noteheadBlack.L5~flag8thDown~epsilon~barline_light

Figure 6.3: Example of the labelling of the groundtruth, creating a 1D sequence. The transcription is written reading each measure from left to right, and from top to bottom if the symbol is divisible into primitives.

6.5 Experimental Validation

This section experimentally validates our approach. To test it we have used the Pau Llinàs dataset detailed in section 3.2.3.

6.5.1 Results on historical music scores

We have used the Symbol Error rate (SER) metric to evaluate our approach. This has already been used in several music recognition publications as a substitute for the well-known Character/Word Error Rate in text recognition. The SER is defined by

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

where S denotes the substitutions, D the deletions and I the insertions and N the number of symbols in the groundtruth. As it is a metric that evaluates the error, the lower the better. Next, we evaluate our sequence-to-sequence architecture, and compare with the baseline described in Section 6.2.

Quantitative Results

Table 6.1 shows the comparison between our Seq2Seq model and the baseline model using Convolutional Neural Network and Bidirectional Long Short Term Memory Neural Networks with Connectionist time classification (CNN+BLSTM) [8]. The first column indicates the method used, the second column indicates which dataset has been used for training and the third column indicates the percentage of Symbol Error Rate (SER). From the Table 6.1, we can observe that, in all

Table 6.1: Quantitative results comparing the CNN+BLSTM model and our sequence-to-sequence (Seq2Seq) model.

Architecture	Dataset Train	Test SER (%)
CNN+ BLSTM	Historical	56.20
	Modern Synthetic	96.20
	Old Synthetic	75.20
	Modern + Old Synthetic	74.40
Seq2Seq	Historical	40.39
	Modern Synthetic	83.80
	Old Synthetic	61.89
	Modern + Old Synthetic	60.69

setups, the Seq2Seq outperforms the BLSTMs by a large margin. As expected, the best result is obtained when training with real historical data, even though the amount of real labelled data is very low. We also observe that training with the modern synthetic dataset leads to a very low performance. However, if we train with the old synthetic dataset, we can reduce the SER by 20 points. Finally, if we combine both synthetic datasets (50% modern and 50% old), there is more varied data during training, so the methods obtain a slightly better SER.

Given that the best results are obtained using our proposed Seq2Seq approach and combining all synthetic data (modern+old), we have performed a second experiment considering the scenario where both real and synthetic data are available for training. As explained in Section 6.4, we use curriculum learning to train with easy examples first, and gradually incorporate more difficult ones. Table 6.2 shows how we have modified the percentage of historical and synthetic data at training time. The first four columns of the table shows the percentage of measures used for training and validation for each dataset, whereas the last column shows the SER on the real historical test set. We start the first epochs (see the first row) with few historical data and a high percentage of synthetic data. Every 10 epochs we augment the percentage of real data, while decreasing the amount of synthetic one. To minimize the overfitting problem, and given that the amount of synthetic scores are much higher than the historical ones, in the validation set, we do exactly the opposite: we have started with a high percentage of historical data, which is progressively decreased during training. At the end of the training phase, the training set has mainly historical data whereas the validation set has mainly synthetic one.

From the results reported in Table 6.2, we can conclude that training with real and synthetic data highly benefits the overall system performance. Indeed, the obtained SER of 31.79% is significantly lower than the SER of 40.39% that was

Table 6.2: Results using our Seq2Seq model with Curriculum learning. We show the amount of data of each kind used during training.

Percentage in Training (%)		Percentage in Validation (%)		Test SER (%)
Historical	**Modern+Old Syn.**	**Historical**	**Modern+Old Syn.**	
10	90	100	0	60.03
40	60	70	30	66.20
60	40	50	50	43.38
80	20	30	70	37.86
90	10	20	80	34.56
100	0	10	90	**31.79**

obtained when training with historical data only, as shown in Table 6.1.

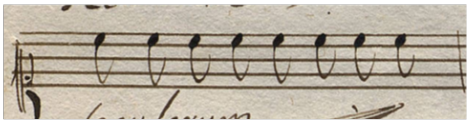
Qualitative Results

Figure 6.4 shows some qualitative results from the sequence-to-sequence model. We have highlighted in red some common mistakes. In the first example, we see that the lyric is often confused by slurs. Some times the shape between the stem and the flag is also confused. The position of a notehead can be frequently displaced *i.e.* a note in the space 3(S3) could be wrongly predicted as to be in line 3(L3) or line 4(L4).

6.6 Conclusions

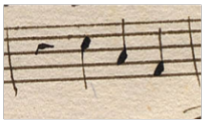
In this chapter we have proposed a sequence-to-sequence architecture with attention mechanism for recognizing historical handwritten music scores. We have experimentally demonstrated that our model obtains promising results, especially compared to Bidirectional Long Short-Term Memory networks. We have also shown that the generation of specific synthetic data that simulates old scores is beneficial. In this sense, we have demonstrated that curriculum learning can gain leverage from the combination of real and synthetic data, improving the overall performance.

Nevertheless, the difficulties of historical scores in terms of paper degradation, touching lyrics and music symbols as well as the lack of annotated data still pose a challenge for optical music recognition. Concerning this last issue, the addition of music context in the recognizer can help to improve the performance. Thus, the next chapters will be dedicated to incorporate information from the musical notation theory to the recognition methods. Moreover, we believe that the research community can benefit from our three labelled datasets, which will be publicly available.



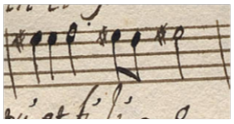
SER: 7.40%

Clef.L1~epsilon~startSlur.noNote~epsilon~noteheadBlack.S4~flag8thDown.noNote~epsilon~noteheadBlack.S4~flag8thDown.noNote~epsilon~noteheadBlack.S4~flag8thDown.noNote~epsilon~noteheadBlack.S4~flag8thDown.noNote~epsilon~noteheadBlack.S4~flag8thDown.noNote~epsilon~noteheadBlack.S4~flag8thDown.noNote~epsilon~noteheadBlack.S4~flag8thDown.noNote~epsilon~noteheadBlack.S4~flag8thDown.noNote~epsilon~barline_light.noNote



SER: 7.14%

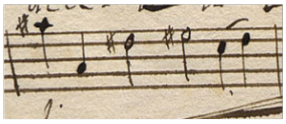
barline_light.noNote~epsilon~quarterRest.noNote~epsilon~noteheadBlack.S3~steamQuarterHalfDown.noNote~epsilon~steamQuarterHalfUp.noNote~noteheadBlack.S2~epsilon~steamQuarterHalfUp.noNote~noteheadBlack.L2~epsilon~barline_light.noNote



SER: 57.57%

barline_light.noNote~epsilon~startSlur.noNote~epsilon~sharp.S4~epsilon~epsilon~sharp.S4~epsilon~sharp.S4~epsilon~noteheadBlack.S4~steamQuarterHalfDown.noNote~epsilon~sharp.S4~steamQuarterHalfDown.noNote~epsilon~steamQuarterHalfDown.noNote~epsilon~endSlur.noNote~steamQuarterHalfDown.noNote~epsilon~steamQuarterHalfDown.noNote~epsilon~barline_light.noNote

(difficult to determine correct and incorrect predictions)



SER: 32.25%

barline_light.noNote~epsilon~startSlur.noNote~epsilon~noteheadBlack.S5~flag8thDown.noNote~epsilon~steamQuarterHalfUp.noNote~noteheadBlack.L2~epsilon~dot.noNote~epsilon~noteheadBlack.L4~flag8thDown.noNote~epsilon~sharp.S4~epsilon~noteheadHalf.S4~steamQuarterHalfDown.noNote~epsilon~noteheadBlack.L4~epsilon~epsilon~steamQuarterHalfDown.noNote~epsilon~noteheadBlack.S3~steamQuarterHalfDown.noNote~epsilon~endSlur.noNote~epsilon~barline_light.noNote

Figure 6.4: Qualitative results. Mistakes shown in red color. From left to right and from up to down. First image: a slur is predicted instead of lyrics. Second image: the pitch of one notehead is confused. Third and fourth images: multiple mistakes because lyrics are too close to music symbols.

Part II

Contextual OMR methods

Music context have been narrowly studied in the Optical Music Recognition literature. Music context helps Optical Music Recognition methods to decrease the number of mistakes or errors done in the recognition. The methods have information from musical notation to deal with the complexity of the problem. Moreover, we will improve models used in the first part of this thesis with the advantages that the music context provides.

7 | Dendrograms

Research is formalized curiosity. It is poking and prying with a purpose. It is a seeking that he who wishes may know the cosmic secrets of the world and they that dwell therein.

– Zora Neale Hurston

---

*The existing Optical Music Recognition approaches can only deal with very simple handwritten scores mainly because of the variability in the handwriting style and the variability in the composition of groups of music notes (*i.e. *compound music notes). In this chapter, we focus on this second problem and propose a method based on perceptual grouping for the recognition of compound music notes. Our method has been tested using several handwritten music scores of the CVC-MUSCIMA database and compared with a commercial Optical Music Recognition (OMR) software. Given that our method is learning-free, the obtained results are promising.*

7.1 Introduction

The music notation rules for creating compound music notes (*i.e.* groups of music notes) allow a high variability in appearance that require special attention. It is one of the points that makes the music recognition more harder to recognize than text. As seen in Chapter 1 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 7.1 shows several examples of compound music groups that are equivalent in rhythm.

Paying attention in 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. note-heads, beams, stems, flags, etc.). It is true that temporal information is undoubtedly helpful in on-

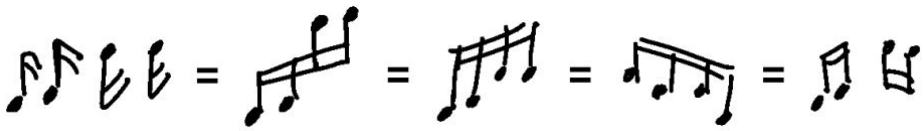


Figure 7.1: Equivalent (in rhythm) compound Sixteenth notes.

line music recognition, as it has been shown in [23, 90]. Concerning the off-line recognition of handwritten groups of music notes, much more research is still needed. As far as we know, PhotoScore is 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.

In this chapter we focus on the off-line recognition of handwritten music scores, putting special attention to compound music notes. For this task, we avoid the need of training data and propose a learning-free hierarchical method inspired in perceptual grouping techniques that have been applied to text detection [66] and object recognition [2]. The idea is to hierarchically represent the graphical primitives according to perceptual grouping rules, and then, validate the groupings using music rules.

The rest of the chapter is organized as follows. First, section 7.2 describes the preprocessing and the detection of the graphics primitives. Section 7.3 explains the hierarchical representation to combine the graphics primitives into more complex elements, and the validation of each group hypothesis. Section 7.4 discusses the experimental results. Finally, conclusions are drawn in Section 7.5.

7.2 Preprocessing and Detection of Primitives

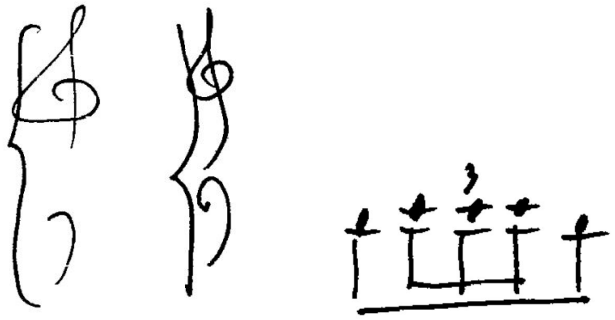
In the preprocessing, we remove music braces and ties. In this step we assume that the input image is binary and the staff lines are already removed by using any of the staff removal methods in the literature [49]. Then we detect the graphics primitives: note-heads, vertical lines and beams.

1. Preprocessing

    a Brace removal: In polyphonic scores, braces indicate the staffs that are played together, such as scores for different instruments. Given that braces appear at the beginning, we analyze the connected components at the beginning of the staffs. Following the musical notation theory, a brace must cross consecutive staffs. Thus, these elements are approximated to a straight line, and if the estimated line crosses several staffs,

it is classified as a brace. Afterwards, braces are removed using the straight line estimation in order to avoid the deletion of other elements such as clefs. The removal of braces will ease the posterior recognition of music symbols, such as clefs and key-signatures. Figure 7.2a shows some examples of braces where some of them are overlapping the clefs. For more details, see [119].

b Tie removal: Long ties are used for adding expressibility in music performance. However, they can be easily misclassified as beams due to the handwriting style of the musician. 7.2b shows a problematic case, where the beam is disconnected from the stems. Therefore, we propose to detect and remove the long ties by analyzing the aspect ratio of the horizontally long connected components.



(a) **(b)**

Figure 7.2: a) Examples of braces that are gathered with clefs. b) Beam easily confused as a tie.

2. Detection of Graphics Primitives The starting point to construct the proposed hierarchical representation is the detection of basic primitives that defines the musical vocabulary or compound notes. These basic primitives are created by means of simple detectors.

   a Vertical lines detection: Vertical lines are key elements that are mainly used to represent stems and bar lines. Since music notes are mainly composed by note-heads, stems, beams and flags (see Fig.3), we must identify the bar lines so that we can keep the rest of vertical lines as stem candidates. For this task, we first detect all the vertical lines using a median filter, and then, we analyze them to identify the bar lines. The bar line identification consists of two steps:

      - Properties checking: The vertical line is kept as a bar line candidate if it (almost) crosses all the staff and it has no blobs (note-heads) at it extrema points.

- Consistency checking: The bar lines in the same page must have similar length and must cross the same staffs. Therefore, the consistency is analyzed as follows. First, we vertically sort the bar line candidates using their centroid. Here, one candidate is an outlier if its length is very different from the candidates in the same line. These outliers are analyzed just in case they have not been correctly detected so they must be joined with other vertical lines. Otherwise, they are rejected.

b Note-head detection: Note-heads play a key-role in music notes, since they provide the melody. Moreover, its the only common component in all type of music notes. Hence, detecting correctly a note-head is of key importance for the correct symbol construction. Figure 7.3 shows in red the different types of note heads that must be detected. Filled-

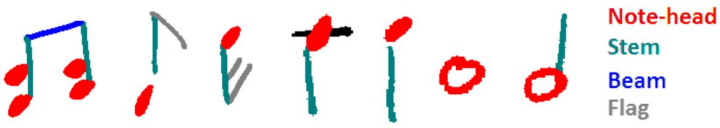


Figure 7.3: Graphics Primitives.

in note-heads are detected using mathematical morphology. First, two elliptic structural elements are defined using different angles (30^{o} and -30^{o}). Then a morphological closing is performed using both structural elements. Finally, blobs closer to a vertical line are considered filled-in noteheads. For the detection of white note-heads, the filled-in noteheads are first removed from the image. Then, the holes are filled so that we can find white note-heads using the same strategy. In both cases, too large blobs are rejected.

c Beam detection: The beam's appearance highly depends on the melody. Consequently, a descriptor based on densities, profiles or gradients (e.g. SIFT, HOG) will be unstable. For this reason, we propose the detection of beams by adapting a pseudo-structural descriptor [86] for handwritten word spotting. The feature vector is created from the information from every key-point in the word. For each key-point, the characteristic Loci Features encode the frequency of intersection counts following a certain direction path. Thus, the shape of the strokes is not taken into account. For the detection of beams, we propose to modify the pseudo-structural descriptor as follows. For each pair of consecutive detected note-heads (and stems), we take the region in between, and divide it into two parts (left and right). Then, we compute the characteristic Loci Features in the vertical direction (i.e. the number of transitions). Finally, we take the statistical mode (the most frequent value), which indicates the amount of beams that link each pair of notes (see Fig.

7.4). In this way, the descriptor is invariant to the beam's appearance and orientation.

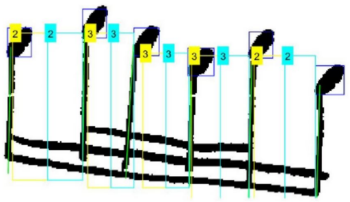


Figure 7.4: Detected primitives in a compound note. The numbers indicate the number of detected beams per region.

7.3 Perceptual Grouping

Once we have detected the graphics primitives, the next step consists in grouping them to recognize the compound music notes. First, we create a hierarchical representation of primitives (see Fig.7.5), and then we validate the different grouping hypothesis using syntactical rules.

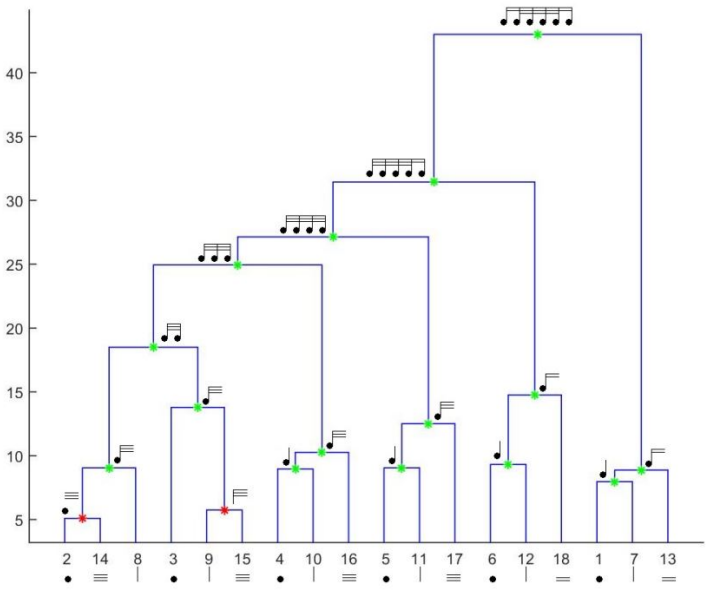


Figure 7.5: Validation hypothesis (dendrogram) of the compound note shown in Figure 7.4

1. Hierarchical representation

   Inspired in the perceptual grouping techniques for text detection [66] and object recognition [2], we propose to build a dendrogram to hierarchically represent the graphics primitives. In our case the criteria for grouping is the proximity of the graphics primitives, which means that the coordinates of the primitives' centers are used as features to create the hierarchical clustering. Since compound music notes must contain at least one note-head, we use the detected note-head candidates as seeds to start the grouping in a bottom-up manner. Thus, we avoid the creation of many non-meaningful grouping regions. Notice that the different grouping hypothesis can overlap. For instance, a chord is composed of several note-heads that share the same stem (e.g. see the first note in Fig. 7.3). Thus, this stem belongs to more than one group hypothesis.

2. Validation of grouping hypothesis

   The next step is the validation of the groupings. In case of text detection, the grouping validation could be performed by recognizing the text. For example, a grouping hypothesis could be accepted whenever an OCR can recognize the word. In our case, the recognition of the compound notes as a whole is not possible because the creation of a dictionary of music notes is unfeasible: there are almost-infinite combinations of compound notes. Moreover, we would need an huge amount of samples to train a shape recognizer. Therefore, we propose to validate each one of the grouping hypothesis through the following music notation rules:

   - Whole note = {[white-note-head]+ }.
   - Half note = {[white-note-head]+, stem}.
   - Quarter note = {[filled-in-note-head]+, stem}.
   - 8th note = {[filled-in-note-head]+, stem, beam}.
   - 16th note = {[filled-in-note-head]+, stem, beam, beam}.
   - 32th note = {[filled-in-note-head]+, stem, beam, beam, beam}.
   - 64th note = {[filled-in-note-head]+, stem, beam, beam, beam, beam}.

   The symbol + indicates that minimum one appearance of this primitive is required. In summary, only the grouping hypothesis that can be validated using these rules will remain. All the other hypothesis will be rejected.

7.4 Experimental Validation

For the experiments, we have selected a subset of the CVC-MUSCIMA dataset [48]. Concretely, we have manually created the ground-truth of 10 music pages, which contain a total of 1932 music notes. The music scores are from 4 different

writers, mostly polyphonic music (containing several voices and chords). As stated in the introduction, since we focus on the recognition of compound music notes, we leave out of our experiments the recognition of isolated symbols (e.g. clefs, accidentals), which could be faced with symbol recognition methods, as shown in [50]. Table 7.1 shows the experimental results. The first column indicates the music page that has been used *e.g.* 'w5-p02' means page 2 from writer 5). The second column indicates whether the score is polyphonic or monophonic. The third and forth columns show the detection of note-heads, whereas the last two columns show the detection of music notes (e.g. half, quarter, 8th note, etc.). The metrics used are the Precision (number of correctly detected elements divided by the number of detected elements), and Recall (number of correctly detected elements divided by the number of elements in the dataset).

Score	Polyphonic	Noteheads		Notes	
		P	R	P	R
w5-002		0.6	0.62	0.49	0.5
w5-010	✓	0.63	0.62	0.36	0.35
w5-011		0.58	0.6	0.48	0.5
w5-012	✓	0.72	0.73	0.64	0.65
w10-002		0.61	0.67	0.47	0.52
w10-010	✓	0.62	0.61	0.4	0.39
w10-011		0.64	0.54	0.55	0.47
w10-012	✓	0.59	0.55	0.49	0.45
w17-012	✓	0.64	0.73	0.6	0.68
w38-012	✓	0.76	0.82	0.72	0.78
Mean	-	0.64	0.65	0.52	0.53

Table 7.1: Results. The detection of note-heads and music notes are shown in terms of Precision (P) and Recall (R). All results are between [0-1].

We observe that the mean Precision and Recall of music notes is around 52%. The main reason is that the detection of note-heads (which are used as seeds in the grouping) is sensitive to the handwriting style. For example, in scores from writer 10, the head-note detector misses almost half of the note-heads. Consequently, the detection of music notes is always lower that this value. In some other cases, such as scores from writers 17 and 38, the note-head detector works much better, which in turn allows the music notes detector to be much higher (recall is 68% and 78%, respectively). Our method has been compared with PhotoScore[1], a commercial OMR software able to recognize handwritten music scores. Figures 7.6 and 7.7 show qualitative results from both approaches. As it can be noticed, PhotoScore performs very well in easy parts, whereas its performance decreases considerably in case of complex compound music notes. In this aspect, our approach is much

[1] `https://www.neuratron.com/photoscore.htm`

more stable.

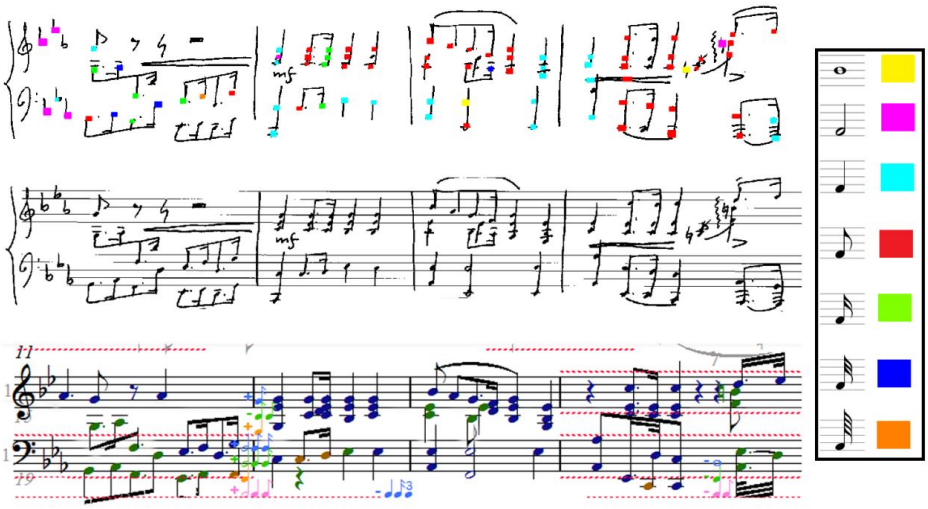


Figure 7.6: Results on 'w10-p10'. First row: our method. Second row: original image. Third row: PhotoScore results

Table 7.2 shows the quantitative results. As it can be seen, our method outperforms the recognition of compound music notes 'w38-012'. Contrary, the differences in the recognition of the 'w10-010' score are very high. There are two main reasons: first, our limitation of correctly detecting note-heads (the recall is around 60% in this score); and secondly, the accidentals (e.g. sharps or naturals) that appear inside the compound music symbols (see Fig.7.8) create confusion in the dendrogram. In addition, flats are similar to half notes, and they are frequently confused. 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 our method. First, PhotoScore is a complete OMR system that recognizes the whole score, which probably uses training data to deal with the variability in the handwriting style. Since it recognizes all music symbols (including clefs, accidentals and rests), it can use 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).

7.5 Conclusions

In this chapter we have proposed a learning-free method for recognizing compound groups of music notes in handwritten music scores. Our method is composed of a hierarchical representation of graphics primitives, perceptual grouping rules and a validation strategy based on music notation. Since our method does not use any

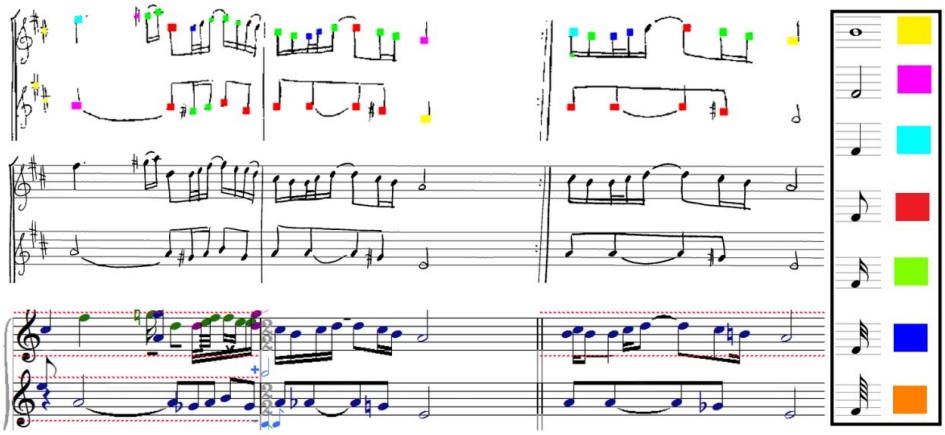


Figure 7.7: Results on'w38-p012'. First row: our method. Second row: original image. Third row: PhotoScore results.

Score	PhotoScore		Our method	
	P	R	P	R
w10-010	0.63	0.61	0.4	0.39
w38-012	0.69	0.74	0.72	0.78

Table 7.2: Comparison with the commercial PhotoScore OMR software. Detection of music notes in terms of Precision (P) and Recall (R). All results are between [0-1]..

Figure 7.8: Compound notes with accidentals.

training data, the experimental results are encouraging, especially when compared with a commercial OMR software.

In the next chapter, we will go a step further by proposing a learning-based method with language models in order to improve the recognition of old music scores. We take as a baseline the Sequence to Sequence model explained in the chapter 6.

8 | Language Models into Seq-2-Seq Models

An error doesn't become a mistake until you refuse to correct it.

– Orlando Aloysius Battista

In this chapter we explore the incorporation of language models into a Seq2Seq-based architecture to try to improve transcriptions where the aforementioned unclear writing produces statistically unsound mistakes, which as far as we know, has never been attempted for this field of research on this architecture. After studying various Language Model integration techniques, the experimental evaluation on historical handwritten music scores shows a significant improvement over the state of the art, showing that this is a promising research direction for dealing with such difficult manuscripts.

8.1 Introduction

Even though the recent Sequence to Sequence (Seq2Seq) architectures have demonstrated its capacity to reliably recognise handwritten text, their performance is still far from satisfactory when applied to historical handwritten scores. Indeed, the ambiguous nature of handwriting, the non-standard musical notation employed by composers of the time and the decaying state of old paper make these scores remarkably difficult to read, sometimes even by trained humans. Nevertheless, in the handwritten text recognition literature, we have found that the incorporation of a Language Model (LM) can tackle most of these ambiguities. This technique consists on the application of a statistical LM trained to identify probable sequences of tokens, which can then be used to assess the likelihood of the recognised sequence and perform due corrections in the case of an unreasonably unexpected output [65, 78]. As in n-grams, it regulates what sequences are considered most likely.

Inspired by this idea, in this chapter we explore the integration of LMs into a Seq2Seq architecture to minimise the ambiguities when recognising historical handwritten scores. Concretely, we integrate a LM through three different techniques: Shallow, Deep [65] and Candidate Fusion [78]. From the exhaustive evaluation of their performance on historical manuscripts, we discuss the advantages and disadvantages of these models, concluding that they are capable to significantly boost

the performance in the aforementioned domain.

The structure for this chapter is the following. Section 8.2 is devoted to describing the architecture. Section 8.3 describes the adaptation of the input data for music score recognition, and the datasets employed to train the LMs. Section 8.4 summarises experiments performed to evaluate the performance of the various models including a discussion of the results. Finally, section 8.5 addresses the conclusions and closing words.

8.2 On the Integration of Language Models

This section describes the core Seq2Seq system for OMR, the three LM models and their integration into the architecture.

As stated before, our architecture is inspired in the Seq2Seq OMR model described in [10, 135]. The whole architecture is depicted in Figure 8.1, with a reference to the LM integration step (see the dashed lines). Next, we describe its properties and its inference process.

8.2.1 Sequence to Sequence model

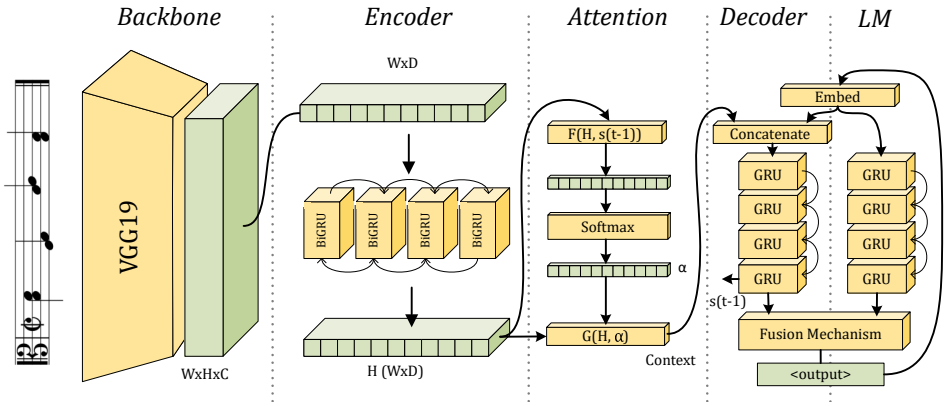


Figure 8.1: Summary of the Seq2Seq model used in this work.

Seq2Seq models [129] are architectures capable of converting arbitrary-length input sequences into arbitrary-length output sequences. They are an Encoder-Decoder architecture: the input sequence is transformed by the Encoder into an intermediate representation that the Decoder will use to generate the output sequence.

A score image, which is treated as a sequence of column vectors, is fed into a Convolutional Neural Network based on a VGG19 [127] with its last max pooling

layer removed. Then, the Encoder, a bidirectional stack of Gated Recurrent Units (GRU) [34], generates an intermediate representation comprised of as many feature vectors as the convolutional output. The idea behind this bidirectionality is that, by processing the input image from both ends of the sequence, the model has the information of the full image for all inference steps and is therefore much more context-aware.

When the Encoder has processed the input image completely, the Decoder iteratively receives the generated hidden state alongside the last predicted token in the sequence, which produces the next output token until a special "end" token is produced. In order to assess the relevance of each of the hidden state's vectors, a location attention mechanism (Chorowsky *et al.* [35]) weights each vector in the hidden state, with the idea of making the model capable of "focusing" on specific regions of the input image.

8.2.2 Language Model Integration

LMs are systems that model the probability distribution of possible tokens at time step t conditioned by predictions at time steps 1 to $t-1$. Many language modelling techniques exist throughout such as n-grams [28], but RNNs are known to be a superior choice overall [89], thus this work focuses on a single LM architecture consisting on four stacked GRUs.

LM integration with Seq2Seq models has been explored through various approaches aiming at improving recognition performance. Three of such approaches have been explored in this work: Shallow, Deep [65], which are among the most used methods, and Candidate Fusion [78], which showed good performance on handwritten text recognition.

Figure 8.2 shows a depiction of these methods and the following paragraphs are devoted to describing them in detail.

Shallow Fusion (Gulcehre *et al.*)

This technique was devised in the context of neural machine translation. It is a very intuitive system in which the final output is obtained by summing log probabilities from the LM and the Seq2Seq model. Let P, P_{CL} and P_{LM} be the probability distribution of tokens predicted by the full model, the Seq2Seq component and the LM respectively, and let λ be an arbitrary hyperparameter set on training, Shallow Fusion is implemented as

$$\log P\left(y_t|y_1 \dots y_{t-1}\right) = \log P_{CL}\left(y_t|y_1 \dots y_{t-1}\right) + \lambda \log P_{LM}\left(y_t|y_1 \dots y_{t-1}\right). \quad (8.1)$$

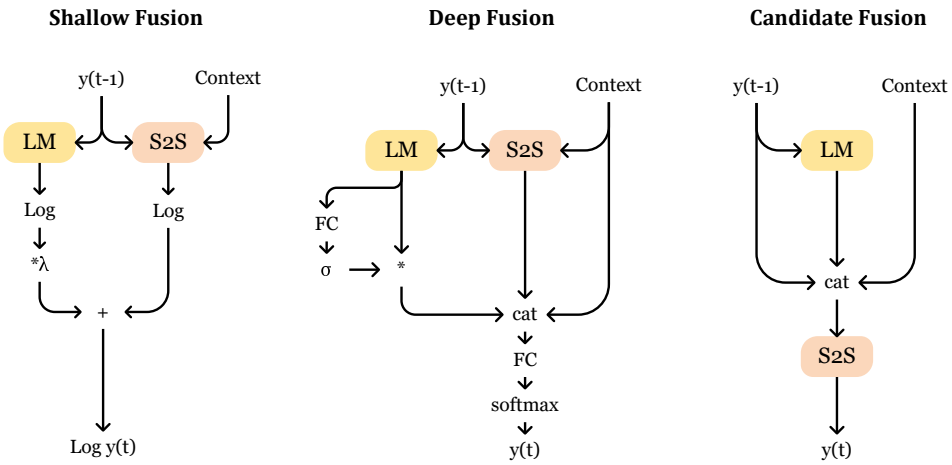


Figure 8.2: Dataflow graph depicting every integration method that was implemented

Deep Fusion (Gulcehre *et al.*)

This method comes from the same context as Shallow Fusion and builds further on its idea by merging both LM and Seq2Seq's outputs in a more fine-grained manner. Essentially, the λ parameter is substituted by a coarse gating mechanism and the final output is obtained using more information from across the model. Let σ be the sigmoid activation function and W_{DF} and b_{DF} be learnable parameters, Deep Fusion is implemented as

$$P(y_t|y_1 \dots y_{t-1}) = \text{softmax}(W_{DF} h_t^{DF} + b_{DF}). \tag{8.2}$$

The Deep Fusion hidden state h_t^{DF} is obtained concatenating the Seq2Seq context vector c_t, the Classifier's hidden state h_t^{CL} and a gated version of the LM's hidden state, as seen in

$$h_t^{DF} = \left[c_t; h_t^{CL}; g_t h_t^{LM}\right]. \tag{8.3}$$

The coarse gate mechanism g_t is in its turn computed as

$$g_t = \sigma(v_g^T h_t^{LM} + b_g) \tag{8.4}$$

where v_g and b_g are learnable parameter vectors. We use the implementation seen in [131], which does not feed the previously inferred character in equation 8.3.

Candidate Fusion (Kang *et al.*)

This method was shown to be more suitable than Deep and Shallow fusion in the context of Handwritten Text Recognition. The core idea behind it is to reinforce the decision process of the Seq2Seq Decoder at each output time step by feeding

it the output of the LM, so that both pipelines can be leveraged accordingly. It can be defined as

$$h_t = \text{Decoder}(\left[c_t, y_{t-1}, p_{t-1}^{lm}\right], h_{t-1}) \tag{8.5}$$

where c_t is the current context vector, y_{t-1} is the previous prediction and p_{t-1}^{lm} is the probability distribution obtained by the LM with the output at the previous time step.

Some comparisons can be drawn among all three methods both from their literature and their architectures. The main selling point for Shallow Fusion is that it adds very little complexity into the model, which is compensated by the fact that it requires hyperparameter tuning for its λ value and the impossibility to modify said value depending on the LM output. Deep Fusion poses as a more flexible model that can learn to weight the importance of the output of the LM, at the cost of incorporating further layers into the model. Finally, Candidate fusion boosts the communication between the LM and the Seq2Seq component and produces an output obtained not by linearly combining both outputs at the final inference step, but rather by letting the Seq2Seq combine the criteria of visual features and Language Probabilities. However, this might involve more training for the model to become acquainted with the output of the LM.

All these methods require both the classifier and the LM to be properly pretrained for successful integration. More detail is provided in section 8.4.

8.3 Dataset

This section describes the adaptation of the data for music score recognition using the Seq2Seq architecture.

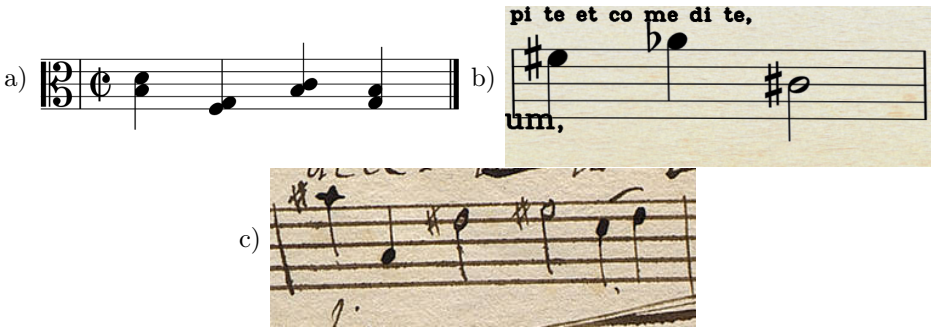


Figure 8.3: Sample measures from the SM, SO and HW datasets respectively.

8.3.1 Serialising Input Data

Input music measures are annotated at a *musical primitive* level. This means that notes are not full tokens by themselves, but are instead divided into their core elements: noteheads with their pitch and type (black or white), stems with their orientation, flags, beams and so on. There are also some tokens which are atomic, such as time signatures, dots, accidentals and rests, and some twin tokens that require opening and closing, such as beginning and end segments of a slur or a beam.

The `epsilon` token is a special one used to separate groups of primitives belonging to different symbols placed in adjacent columns. Thanks to this, 2D music notation can be serialised into a flat one-dimensional array of tokens that Seq2Seq can work with. An example of this format is given in Figure 8.4.

```
barline_light.noNote, epsilon,
sharp.S4, epsilon, noteheadBlack.S4,
steamQuarterHalfDown.noNote,
epsilon, dot.noNote, epsilon,
noteheadBlack.S4, flag8thDown.noNote,
epsilon, noteheadBlack.S3,
steamQuarterHalfDown.noNote
epsilon, noteheadBlack.S3,
steamQuarterHalfDown.noNote, epsilon,
barline_light.noNote
```

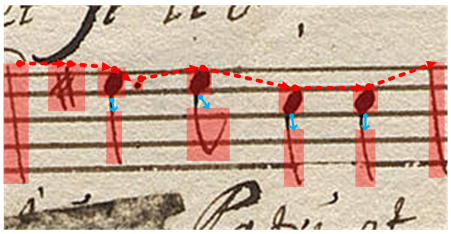


Figure 8.4: Sample measure from the HW dataset with its ground truth annotation. Bounding boxes indicate the boundaries of what each "atomic" token is, dotted arrows indicate epsilons in the transcription and small vertical arrows indicate symbols that are placed together between epsilons (or rather, primitives belonging to the same symbol).

8.3.2 Training Datasets

Various datasets of differing characteristics were used to train the models, each of them for a specific task (more detail on section 8.4). Their description is shown below along with some examples (See Figure 8.3). Note also that, when referring to synthetic datasets, we imply the musical content of these scores is randomly generated (thus we assume that these datasets are, except for some trivial examples, disjoint).

Synthetic Modern (SM): Dataset comprised of polyphonic measures of synthetic typeset scores. Most usual music symbols can be found: G, C and F clefs, accidentals, note components, time signatures and barlines, to name a few. See more information in section 3.2.2. An example is shown in Figure 8.3a.

Synthetic Old (SO): A synthetic dataset with monophonic measures distorted with typical paper degradation effects. Similar to SM in terms of the range of tokens present. See more information in section 3.2.2. An example is shown in Figure 8.3b.

Handwritten (HW): A compilation of measures of real handwritten scores from a church in Barcelona called Santa María del Pi. They were composed by its Kapellmeister Pau Llinàs back in the 18th century for choral interpretation during liturgical events. See more information in section 3.2.3. An example is shown in Figure 8.3c.

Adjusted Synthetic Modern (ASM): A reduced version of the SM dataset (see section 8.4).

8.4 Experimental Validation

The evaluation of all proposed LM integration methods was performed under two training strategies, characterised by the dataset which was used to pretrain the LM. Regardless of the LM dataset, training parameters and strategies were the same altogether. For the sake of reproducibility, Table 8.1 summarises these hyperparameters and characterises the various datasets employed throughout.

Table 8.1: Reproducibility table. The first segment is devoted to training hyperparameters. The second one to showing relevant information about the various datasets that have been employed.

Parameters	All Training	Data	SM	SO	HW
Optimiser	Adam	**Train Samples**	18,900	17,872	147
Learning Rate (LR)	$3 \cdot 10^{-4}$	**Valid Samples**	6,300	5,957	49
LR Checkpoints	@ 20, 40, 60, 80, 100 epoch	**Test Samples**	6,300	5,957	49
LR Sigma	0.5	**Avg. Line Length**	22	15	17
Loss Function	Cross-Entropy	**Classes**	109	123	62

Since the goal for our work is to improve results on handwritten scores, a training strategy was conceived to gain the benefits of extra data from synthetic scores while preventing optimisation towards them. All integration methods tested hereby require both the LM and the recogniser to be properly pretrained. Thus, we first trained a LM with an unmodified version of the SM dataset. Since we were aware that this dataset had many tokens that were not present in the HW one, we created a version of the SM dataset comprised of the 66% of samples which contained a higher ratio of tokens also present in the HW one, which we will refer to as ASM, and we trained another LM with it. The idea was trying to "de-noise" the output of the LM in HW scores so that its predictions had a higher level of confidence.

In both cases, we trained the Seq2Seq classifier with the unmodified SM dataset until the model did not improve for 30 epochs. We then joined both models and

trained them using a Curriculum Learning strategy: initially, 90% of samples in the training mix were from the SO dataset and the remaining 10% from the HW dataset. Every 10 epochs the proportion of SO scores decreased by 10% over the total, down to 10%. Since the number of SO samples is much higher than the number of HW samples, the latter were duplicated randomly to match the number of samples from the former. The incorporated image augmentation system for training was used to prevent overfitting on input images. Note also that experiments with homogeneous datasets were avoided since they were seen to decrease performance in earlier tests.

Validation and test were performed using HW dataset samples. Lastly, for Shallow Fusion we used a $\lambda = 0.1$ after testing three instances of the full architecture on the SM dataset and keeping the value that gave better output results.

8.4.1 Quantitative Results

This section is devoted to explaining the results obtained with the aforementioned training strategies. This is, Shallow, Deep and Candidate Fusion using a LM pretrained with the SM or the ASM Dataset. Numerical results are provided using the Symbol Error Rate (SER(%)) metric, which is defined as

$$SER(\%) = \frac{I + R + S}{T} \cdot 100 \tag{8.6}$$

where I, R and S are the number of token insertions, removals and substitutions in order to obtain the ground truth sequence from the predicted sequence and T is the length of the ground truth sequence. Lower values mean better results.

Table 8.2: Summary of performed experiments and results in SER(%) (Lower is better). The table header indicates the proportion of Synthetic scores against Handwritten scores. The "Pre" column indicates the LM pretraining dataset.

Model	Pre	90-10	80-20	70-30	60-40	50-50	40-60	30-70	20-80	10-90	0-100
CNN + BLSTM[10]	-	-	-	-	-	-	-	-	-	-	**56.20**
Seq2Seq Baseline [10]	-	60.03	-	-	66.20	-	43.38	-	37.86	34.56	**31.79**
Seq2Seq + Deep LM	SM	31.30	28.52	29.87	29.37	28.05	**26.11**	27.74	27.37	28.32	-
Seq2Seq + Shallow LM	SM	36.79	32.91	33.27	33.36	31.76	32.75	30.87	30.72	**30.58**	-
Seq2Seq + Cand. LM	SM	33.50	28.93	28.64	28.08	27.48	26.82	27.23	26.61	**25.80**	-
Seq2Seq + Deep LM	ASM	28.24	29.53	27.82	27.36	25.95	27.21	25.63	**25.15**	25.54	-
Seq2Seq + Shallow LM	ASM	35.34	34.75	36.67	**32.42**	34.23	34.52	33.76	33.79	35.13	-
Seq2Seq + Cand. LM	ASM	32.07	28.61	28.71	27.55	27.71	27.20	27.77	28.04	**25.73**	-

Table 8.2 shows the results obtained from all of our experiments. Given the fact that Seq2Seq model pre-training on the SM gave results well below 1% SER(%), we believe it is not worth to experiment with the addition of a LM when transcribing synthetic samples. Instead, we show test results using the training strategy in 8.4 and two baseline models: the BLSTM + CTC model and the LM-less Seq2Seq model [10]. All results are obtained using the HW test partition as input.

Best baseline results are 56.20% and 31.79% of SER(%) for BLSTM + CTC and Seq2Seq respectively. However, authors comment in the paper that there might be overfitting in the best result of the former model because training was done only with handwritten samples. When training with a mix of synthetic and real data, the authors state an increase from 56.20 SER(%) to 74.40 SER(%).

Our proposed models obtained mostly better results than those from the Baseline. Candidate and Deep Fusion are the better performing architectures, with best results (in bold in Table 8.2) between 5 and 6 SER(%) points below the baseline. Shallow Fusion obtained best results on par with the baseline.

The general pattern is that earlier iterations perform worse than latter ones. There are a few exceptions, which are the SM version of Deep Fusion and the ASM version of Shallow Fusion, which obtain better results in intermediate phases. This might be caused by the fact that the model might be entering local minima, which it may leave after further epochs.

Another general remark is that models pretrained with the ASM dataset seem to perform slightly better, with a 0.96 SER(%) improvement in Deep Fusion and a 0.07 one in Candidate Fusion, although this difference could be also attributed to optimisation since it is not substantial.

8.4.2 Discussion

Numerical proof is found that a LM does help improve recognition results in historical handwritten music scores, especially when using Candidate or Deep Fusion. However, we agree that it is not easy to assess their differences outside of a subjective qualitative study.

Expectedly, LM lowers the presence of certain syntactic mistakes (for instance, tokens that require a specific successor) or provides information on tokens that appear frequently. There is, however, a set of possible recognition mistakes that the LM was initially presumed to be able to correct which we found it unable to. The most relevant was enforcing the beat of the bar that is being recognised. It can be argued that at no point in the measures that comprise the dataset the time signature is indicated aside from its very beginning, but since the training dataset is written exclusively in a 4/4 time signature, the LM might have adapted to measures adding up to a beat value. Perhaps this is due to the purely statistical approach taken with the LM, so some postprocessing (based on music notation rules) may be needed for approaching such consistency checks.

Other "artistic" aspects of music cannot be corrected with the LM, such as the pitch and duration of notes, which can only be predicted up to a certain point based on its frequency of appearance. This was expected and, unsurprisingly, most noteheads have been predicted on the most common range within the original score.

A final remark is that we have observed that the adjustment strategy attempted with the ASM dataset showed no significant improvement. Instead, in order to better align training and test datasets without overfitting, more data should be used for training. A common issue when trying to collect data for this purpose is that most common transcriptions of old music adapt their notation style to current trends, which defeats the purpose of using such data for recognition.

8.5 Conclusions

This work successfully explored the integration of LMs into a Seq2Seq OMR architecture for recognising historical handwritten scores. An improvement of around 6 SER(%) points from the baseline was obtained when using a Deep Fusion mechanism, lowering it to 25.15 SER(%). This was achieved by reinforcing the model's capacity to keep consistency on predicted sequences. Thus, we can conclude that the integration of language models into OMR Seq2Seq architectures is a promising research direction worth exploring.

In the next chapter 9 we will go a step further in the contextual analysis. Concretely, we will present a method based on primitive detection and graph neural networks which relates the primitives, taking into account the music notation rules.

9 | Object Detection and Graph Neural Network

Learn how to see. Realize that everything connects to everything else.

– Leonardo da Vinci

In this chapter we explore the use of graph neural networks for musical score recognition. First, because graphs are suited for n-dimensional representations, and second, because the combination of graphs with deep learning has shown a great performance in similar applications. Our methodology consists of: First, we will detect each isolated / atomic symbols (those that can not be decomposed in more graphical primitives) and the primitives that form a musical symbol. Then, we will build the graph taking as root node the notehead and as leaves those primitives or symbols that modify the note's rhythm (stem, beam, flag) or pitch (flat, sharp, natural). Finally, the graph is translated into a human-readable character sequence for a final transcription and evaluation. Our method has been tested on more than five thousand measures, showing promising results.

9.1 Introduction

Since many music symbols can be drawn in different position and with different shapes, it is necessary to decompose those musical symbols into subparts. In other words, not all the symbols are atomic, some of them can be divided into smaller symbols called primitives. For example, a quarter note is composed of a full notehead and a stem, an eighth note is composed of a full notehead, a stem and a beam or a flag (see Figure 9.1).

In order to make the OMR system to understand which primitives form a symbol, a structure that relates them is needed. Music has a large variability of symbols, combining rhythm and melody, which is even increased in polyphonic scores. But, despite this variability, it shares a common set of primitives (e.g. stem, notehead, beam, flag, etc). Graphs are suitable for music description because of its flexibility and n-dimensional representation power. Music does not have a fixed representation (e.g. the measures do not have to be equal to each other), but each primitive will be related to others following the rules of music theory.

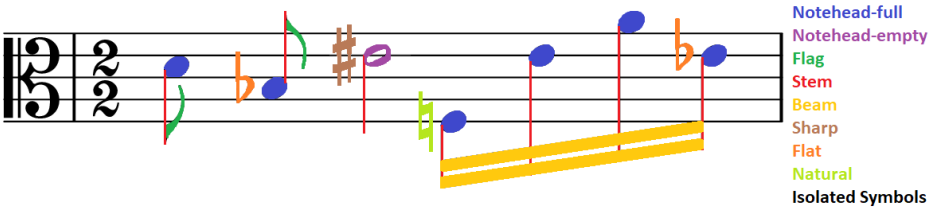


Figure 9.1: Graphic primitives. Isolated / atomic symbols shown in black color, whereas music primitives are shown in colors.

For the above reasons, in this chapter we present *Musigraph*, a graph-based OMR system for music scores. Inspired by the success of Graph Neural Network models, we explore their adaptation to optical music recognition. Our method is based on Convolutional Neural Networks and Graph Neural Networks. As far as we know, this is the first OMR method based on GNNs. Besides, given the few labelled available music scores, we have created a labeled synthetic dataset to train such deep learning systems. We provide the groundtruth for music object detection and graph recognition.

The rest of this paper is organized as follows. First, section 9.2 is devoted to describing the architecture. Section 9.3 discusses the experimental results. Finally, conclusions and future work are drawn in Section 9.4.

9.2 The Musigraph model

As explained before , music recognition can be seen as a two-dimensional problem. The music score has to be read from left to right, but taking into account the rhythm and the pitch of each symbol. Thus, our approach will be composed of two steps. First, we will detect each atomic symbol or primitive by an object detector. Note that each of these primitives are just compounding parts of the symbols we consider on a music score. Second, we will relate the primitives detected by a graph neural network. Figure 9.2 shows the full pipeline.

We use the same set of music measures to train the object detector and the graph neural Network. During training, the graph neural network uses the bounding boxes obtained by the groundtruth. Concretely, to train the graph neural network we need to initialize the graph by specifying the edges (*i.e.* connecting nodes between each others), and later, the GNN will decide if the edge should exists or not. At test time, we use the bounding boxes provided by the object detector. So, once the object detector is trained, at test time, we obtain the candidate locations of each primitive and symbol, which will be postprocessed for improving the bounding-box prediction accuracy. Finally, these detections will be the input of the graph neural network during the test time.

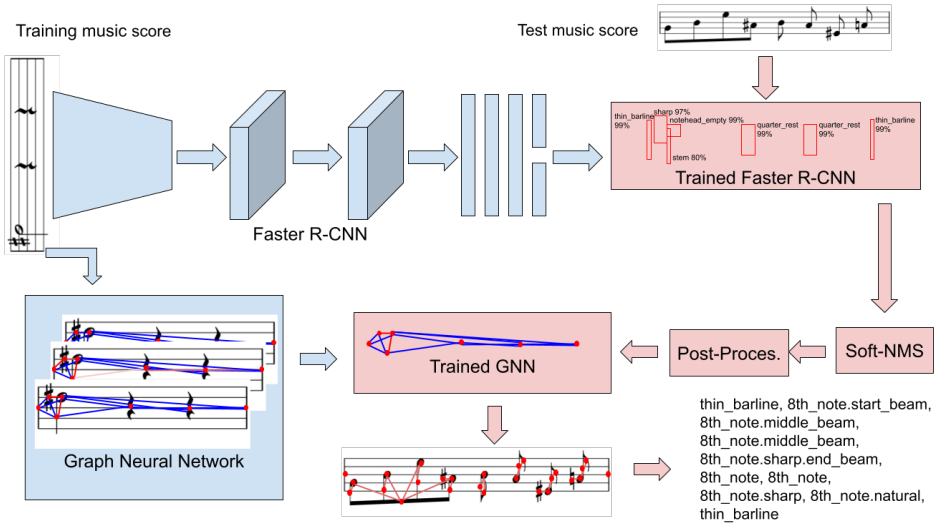


Figure 9.2: Architecture Pipeline. In blue the training process, in red, testing.

9.2.1 Object Detector

The object detector will be used in the test phase. Note that during the training, we use the bounding boxes provided in the groundtruth to ensure that the graph neural network learns properly. Specifically, for object detection we have used the well-known library *Detectron 2* by Facebook AI Research [139]. This library provides the current state-of-the-art detection algorithms. Particularly, we have used the Faster R-CNN, which consists of a CNN with a final ROI pooling to extract a fixed-length feature vector from each region proposal. Finally, the fully connected layer is divided in two branches; a category softmax and a bounding box regression.

Once all the bounding boxes have been detected by the Faster R-CNN, we have used the soft non-maximum suppression (soft-NMS) algorithm [14]. The main difference with the classic NMS is that it removes a detection if the confidence is lower than a threshold. Moreover, in the soft-NMS when the degree of overlap reaches a threshold and the confidence is lower, it is not directly suppressed, instead, its confidence only is reduced. Algorithm 1 shows this difference (in blue the soft-NMS and in red the classic NMS).

Moreover, we have improved the beam and stem detections using morphological operations. Stems have been detected through the noteheads detections. By applying a vertical projection in the closest area of the notehead's bounding box, we find peaks that allow us to detect stems, a very thin and elongated primitive. For beam detection, we have eroded and dilated the image using as a kernel a

Algorithm 1 Classic and soft non-maximum suppression algorithm. Reprinted from [14]

Input: $\mathscr{B} = \{b_1, .., b_n\}, \mathscr{S} = \{s_1, .., s_n\}$
  $\mathscr{B}$ is the list of bounding boxes
  $\mathscr{S}$ is the list of confidence scores
  $\mathscr{N}_t$ is the NMS threshold
  $\mathscr{D} \leftarrow \{\}$
  while $\mathscr{B} \neq empty$ **do**
    $m \leftarrow \arg\max \mathscr{S}$
    $\mathscr{M} \leftarrow b_m$
    $\mathscr{D} \leftarrow \mathscr{D} \cup \mathscr{M}; \mathscr{B} \leftarrow \mathscr{B} - \mathscr{M}$
    for bi in $\mathscr{B}$ **do**
      if $iou(\mathscr{M}, b_i) \geq \mathscr{N}_t$ **then**
        $\mathscr{B} \leftarrow \mathscr{B} - b_i; \mathscr{S} \leftarrow \mathscr{S} - s_i$
      end if
      $s_i \leftarrow s_i f(iou(\mathscr{M}, b_i))$
    end for
  end while
  return $\mathscr{D}, \mathscr{S}$

vertical and horizontal line respectively; and then, we find those regions with an aspect ratio in which the width is three or more times the height.

Finally, having the noteheads' bounding boxes, we perform a horizontal projection to detect the staff lines, which allows us to recognize the pitch of each note.

9.2.2 Graph Neural Network

A graph is a symbolic data structure describing relations (*edges*) between a finite set of objects (*nodes*). Let L_V and L_E be a finite or infinite label sets for nodes and edges, respectively. A *graph* g is a $4-$tuple $g = (V, E, \mu, \nu)$ where, V is the finite set of *nodes*; $E \subseteq V \times V$ is the set of *edges*; $\mu \colon V \rightarrow L_V$ is the node labelling function; and, $\nu \colon E \rightarrow L_E$ is the edge labelling function.

GNNs introduced by Gori and Scarselli [60, 124] were the first attempt to generalize neural networks to graphs. Then, Bruna *et al.* [15] proposed a new formulation based on the spectral graph theory. Later, the works of Henaff *et al.* [72], Defferrard *et al.* [41] and Kipf *et al.* [82] addressed these computational drawbacks. In its simplest form, a GNN layer $G_c(\cdot)$ is defined as:

$$h^{(k+1)} = G_c(h^{(k)}) = \rho\left(\sum_{B \in \mathcal{A}^{(k)}} B h^{(k)} \Theta_B^{(k)}\right) \tag{9.1}$$

where $h^{(k)}$ is the node hidden state at the k-th layer, ρ is a non-liniarity such as ReLU$(\cdot)$, $\mathcal{A}$ is a set of graph intrinsic linear operators that act locally on the graph signal and Θ are learnable parameters. In most cases, $\mathcal{A}$ it only contains the adjacency matrix [123].

More recently, Gilmer *et al.* [57] proposed the *Message Passing Neural Networks* (MPNNs) as a general supervised learning framework for graphs. MPNNs define its layers in terms of a message and update functions. Similarly, Hamilton *et al.* [69], proposed a graphSAGE layer which SAmples and aggreGatEs node features.

Given a graph g, our proposed graph neural network architecture $\varphi(\cdot)$ has N GNN layers and is defined using graphSAGE [69] layers, with mean aggregation function and ReLU activations.

In order to initialise the graph we have tried different input graphs by changing the heuristics about how to relate the nodes to each other. In all cases, the nodes are the symbols or the primitives. During training, these correspond to the groundtruth, whereas in testing, these are the ones detected by the Faster R-CNN. The edges connect the nodes between them according to the heuristic selected, which is detailed next.

1. K-Nearest Neighbours (KNN) graph: The graph is initialized connecting each node with the K nodes closer to itself. We have used the Euclidean distance expressed in Equation 9.2.

$$distance = \sqrt{(X_2 - X_1)^2 + (Y_2 - Y_1)^2} \tag{9.2}$$

2. Music Heuristic: This heuristic consists on analysing the music scores (frequent relations and distance) to avoid meaningless relationships. For each primitive we consider which other primitives are usually related to it, and how far they are. Thus, when the graph is constructed all this information is taken into account to avoid relations between primitives that are too far apart. Furthermore, we avoid meaningless primitive relations, in other words, we avoid the relation of atomic symbols (they are already a unit by themselves, for example a rest) with other elements.

Once the graph neural network is built, each node is embedded in a learned feature vector according to its detected class and bounding box using a fully-connected layer. Then, after each GNN layer (all but the last), the previous hidden state is concatenated. Afterwards, each node feature is normalized using the L_2-norm. Finally, an edge score is computed by performing an element-wise dot product between features of u (source node) and v (target node).

Our GNN has been trained using the binary cross entropy objective with the edge scores and its corresponding ground-truth. Our model has been optimized with the Adam [81] optimizer.

The end of the full pipeline finishes once the model is trained and the primitives are

transformed into symbols through the detected graph for an evaluation at symbol level.

9.3 Experimental Validation

This section describes the experiments performed to validate our approach. On the one hand, we will show and discuss the object detection results, and on the other hand, we will show and analyze the graph neural network results. To test this approach we have created a new dataset for object detection and Graph Neural networks detailed in section 3.2.4.

9.3.1 Object detection results

To evaluate the object detection method we have used the mean average precision(mAP) [100, 101]. The mAP is defined by

$$mAP = \frac{1}{n}\sum_{k=1}^{k=n} AP_k \tag{9.3}$$

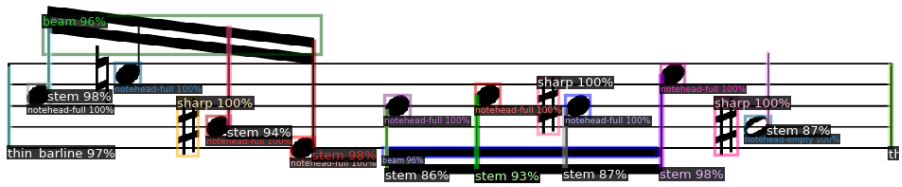
where n is the number of classes and AP_k is the average precision of class k.

Quantitative Results:

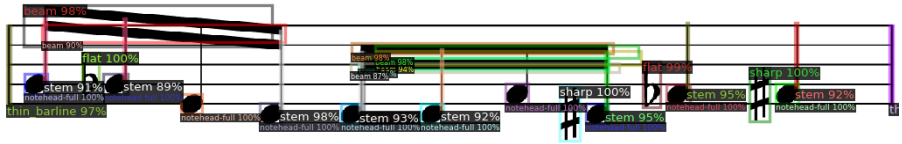
Table 9.1 shows the comparison between the object detection method baseline and the post processing, including the non-maximum suppression and the beam and stem detection improvement. The first column shows the list of primitives and symbols. The second column shows the average precision using the Faster R-CNN from Detectron 2 per each primitive or symbol. And finally, the third column indicates the average precision after the non-maximum suppression and the beam and stem betterment. The last row of the table shows the mean average precision per each method. From the Table 9.1 we can observe that the Faster R-CNN performs very well in all symbols and primitives except those primitive in which one side is much larger than the other. For example, stems are very large and thin with lots of close primitives, whereas beams have the same particularity but horizontally. From the last column, we observe that the morphological operations have improved more than 15 points in the case of beams and almost 10 points in the case of stems. This betterment will help the creation of the graph because, when one node is not detected by the Faster R-CNN, the graph will not be able to recognize the complete symbol (*e. g.* if the network detects a notehead and stem and a beam is missed, the graph will recognize the note as a quarter-note instead of an 8th-note).

Table 9.1: Detectron results using the Faster R-CNN. We show the results of the baseline and improving the stem and beam detection.

	AP%	**AP % after processing**
16th_flag	99.73	99.73
16th_rest	99.48	99.48
8th_flag	99.76	99.76
8th_rest	99.85	99.85
beam	84.48	100
c-clef	100	100
f-clef	100	100
flat	99.26	99.26
half_rest	99.56	99.56
natural	97.11	97.11
notehead-empty	99.93	99.93
notehead-full	99.48	99.48
quarter_rest	99.90	99.90
sharp	99.43	99.43
stem	79.71	88.85
thin_barline	99.17	99.17
timeSig_2-2	100	100
timeSig_cut	100	100
mAP	97.60	98.97



(a)



(b)

Figure 9.3: Object detection results.

Qualitative Results:

Figure 9.3 shows some qualitative results from the Faster R-CNN before applying the post processing. In Figure 9.3a we can observe one of the main problems of the Faster R-CNN: the missdetections of beams (note that only one is detected in each group of 16th notes). In Figure 9.3b we can observe that the third notehead in each 16th group of notes has missed the stem. Apart from this, in the second group of 16th notes multiples beams are detected and the non-maximum suppression is needed.

9.3.2 Graph Neural Network results

Quantitative Results:

Table 9.2 demonstrates the results of the Graph Neural Network using the different initialization options. The results are provided in terms of Music Error Rate (MER) at symbol level, which means that we first convert the different primitives into a symbol (*e. g.* flag+stem+notehead-full is converted into a 8th note). The Music Error Rate is defined as

$$MER = \frac{I + R + S}{T} \tag{9.4}$$

where I, R and S are the number of insertions, deletions and substitutions to obtain the groundtruth sequence. T is the length of the groundtruth. Lower values mean better results. The first columns shows the different graph initialization techniques explained in section 9.2. The second column shows the mean percentage of Music Error Rate and the standard deviation after five iterations.

Table 9.2: Results using our Graph Neural Network. We show which graph initialization has been used and the Music Error Rate. Note that the lower the better. The first number is the mean of the five executions, whereas the standard deviation is shown between parenthesis. All results use the Faster R-CNN and the post-processing to detect the nodes at test time.

Graph initialization	**MER %**
KNN	10.01 (± 0.72)
Music Heuristic	**5.32** (± 0.43)

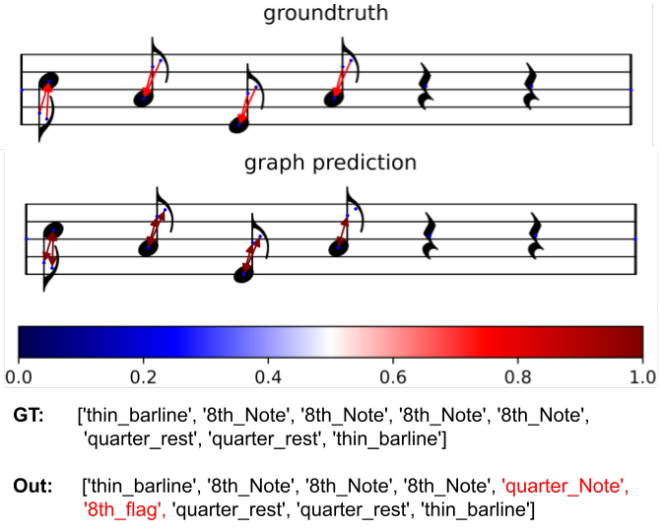
From Table 9.2 we can observe that using the Music Heuristic as graph constructor, we obtain very good results (around 5% of MER). However, it has to be taken into account that KNN does not use any musical information, so having one error for every ten is quite understandable, even if it is almost twice the previous one.

Qualitative Results:

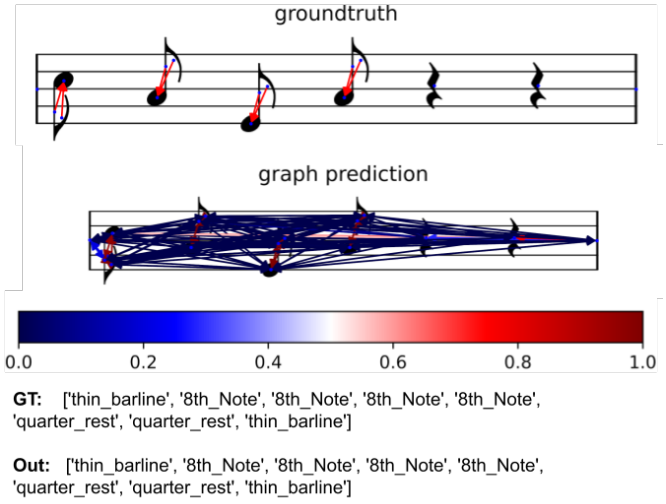
Figures 9.4 and 9.5 show some qualitative results from all the pipeline. The first row of each image shows the groundtruth, the blue points are the centers of the nodes and the arrows, the edges. The second row shows the edges detected by the Graph Neural Network. The blue arrows mean that this edge was created by the graph initialiser, but the GNN has decided that the edge is not relevant. Oppositely, the red arrows are the relevant edges. Finally the third row is the graph translated into text as a final result. Figure 9.4a and 9.5a shows the results using the music heuristic in the graph constructor. Figure 9.4b and 9.5b shows results using the K-Nearest Neighbours. In Figure 9.4a we can observe that the graph creator has not related the last flag with the stem, so the network is not able to predict this relation and finally it is detecting a quarter note. In contrast, in Figure 9.4b, the KNN initialisation has many more relations, so this measure is perfectly recognized. In addition, in Figure 9.5a the recognition is perfect but in Figure 9.5b, the last notehead has an edge with the beam but the network has decided that edge is not important.

9.4 Conclusions

In this chapter we have proposed Musigraph, an Object Detection and Graph Neural Network architecture for recognizing music scores. We have demonstrated that our model obtains very good results. With more than 98% of mAP in the object detection task and with almost 5% of Music Error Rate as a final result, our model proves to be a promising technique for recognizing music scores, or to help musicians when transcribing them.

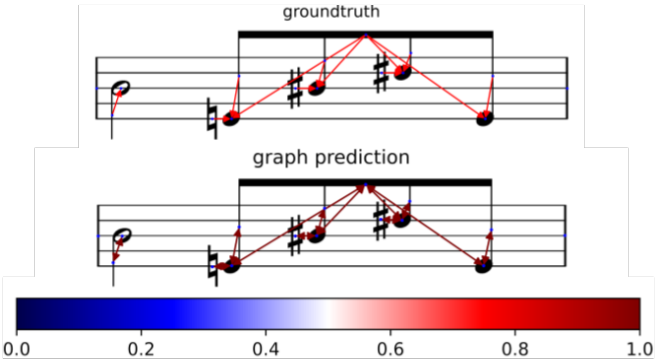


(a) Music Heuristic



(b) K-Nearest Neighbours

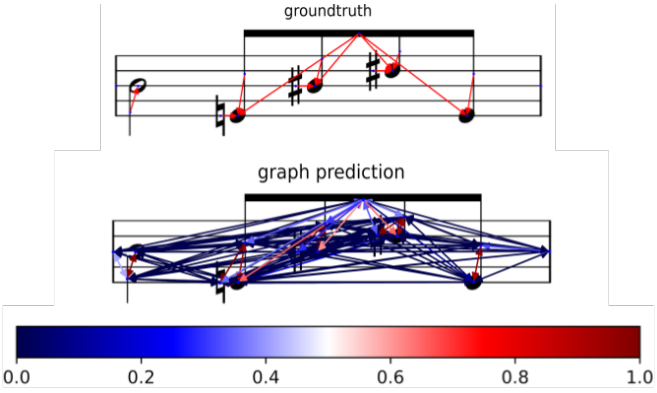
Figure 9.4: Final qualitative results of a measure comparing the graph initialiser.



GT: ['thin_barline', 'half_Note', '8th_Note.natural.startBeam', '8th_Note.sharp.middleBeam', '8th_Note.sharp.middleBeam', '8th_Note.endBeam', 'thin_barline']

Out: ['thin_barline', 'half_Note', '8th_Note.natural.startBeam', '8th_Note.sharp.middleBeam', '8th_Note.sharp.middleBeam', '8th_Note.endBeam', 'thin_barline']

(a) Music Heuristic



GT: ['thin_barline', 'half_Note', '8th_Note.natural.startBeam', '8th_Note.sharp.middleBeam', '8th_Note.sharp.middleBeam', '8th_Note.endBeam', 'thin_barline']

Out: ['thin_barline', 'half_Note', '8th_Note.natural.startBeam', '8th_Note.sharp.middleBeam', '8th_Note.sharp.endBeam', 'quarter_Note', 'thin_barline']

(b) K-Nearest Neighbours

Figure 9.5: Final qualitative results of another measure comparing the graph initialiser.

10 | Conclusions

Think about a piece of music - some great symphony - we don't expect it to get better as it develops, or that its whole purpose is to reach the final crescendo. The joy is found in listening to the music in each moment.

– Alan Watts

In this last chapter, we summarize the main contribution of this thesis to the computer vision literature, specifically to the optical music recognition research field. We will then discuss the benefits and limitations of the developed methods. Finally, based on the experience obtained during this thesis, we will propose how to improve some of the methods or create new lines of research.

10.1 Summary of the Contributions

During this thesis, we have focused our research on the recognition of music scores through different methodologies. On the one hand, the methods detailed in the first part are based on recurrent networks such as Long Short-Term Memory or Sequence to Sequence models. On the other hand, we take advantage of the music context to improve the previous methodologies or to propose some others (e.g. GNNs).

In chapter 3, we have described the datasets that have been created during this thesis. These are from different varieties: from simple datasets, such as a monophonic and printed dataset, to more complicated datasets, such as the old handwritten one. In particular, we have created four different datasets which covered the research needs at the time they were created. The monophonic printed dataset includes relations between primitives, a monophonic printed one that emulates the appearance of old scores, a polyphonic dataset with a modern appearance, a handwritten dataset with polyphonic and monophonic staff lines, and finally a monophonic old handwritten dataset.

In chapter 4, the first OMR method is presented. We address the recognition of printed monophonic staff lines through Recurrent Neural Networks. We propose

to use Long Short-Term Memory Neural Networks. We have split the output of the network in two parts. On the one hand we obtain the rhythm in each time step. On the other hand, the network provides us the pitch per time step.

Next, in chapter 5, this first OMR method is improved by adding a Convolutional Neural Network before the recurrent block. This, combined with a novel data augmentation technique for staff lines that consists of mixing measures and transferring the learning obtained by training printed scores, allows us to deal with more difficult scores such as handwritten ones.

In chapter 6, we have proposed an OMR method based on sequence to sequence models with attention mechanism. The seq2seq model consists on two parts. The encoder is formed by a features extraction block, which feeds a bidirectional gated recurrent unit (bGRU) neural network, whereas the decoder is formed by a one-directional multilayered GRU. The method also incorporates an attention mechanism which aligns our feature representations with our decoding steps. In addition, we generate a synthetic dataset with the appearance of old scores so that the method can handle old real scores.

The part concerning the benefit of using music context starts in chapter 7, with a first application based on dendrograms. This method consists in detecting the primitives with mathematical morphology and then, joining them to form compound music notes. We use dendrograms to check whether the groups of primitives that have been assembled are valid or not according to the music notation theory.

Then, in chapter 8, and given the difficulty in recognizing old music scores, we have improved the seq2seq model detailed in chapter 6 by incorporating Language Models. We have applied three different language models which basically model the probability distribution of possible tokens at time step t conditioned by predictions at time steps *1* to *t-1*.

Finally, in chapter 9, we propose to combine object detection models (for detecting atomic symbols or primitives) with Graph Neural Networks. The objective of this architecture is to take advantage of the relationality offered by graphs to combine the detected primitives to form music symbols. In this case, the music context help the network to avoid wrong relations from the music notation point of view.

10.2 Discussion

Throughout these years of research in optical music recognition, some contributions have been made that will be discussed next.

In Chapter 3 we have proposed four datasets. The first three datasets are intended for methods that do not require to label the exact position of each element in the image, *i.e.* methods such as recurrent networks. However, the last one is created to satisfy the necessities of various methods such as object detection and graph

neural networks. In the future, music datasets should tend to be created and labelled independently of the task or type of methods.

Some of the proposed OMR methods are restricted to monophonic scores. For example, the methods presented in Chapters 4 and 5 can only detect one musical symbol per time step. In other words, these methods will only be able to recognise monophonic scores. Since rhythm and pitch are obtained separately, the system does not know which pitch corresponds to which note if they are in the same column, as might happen in a chord. The same occurs if two ligatures were to appear at the same time; in such case, one of them would also be lost in detection.

The sequence to sequence model described in Chapter 6 works well for recognizing measures. It can be seen, how the method achieves good results even with the complexity of the old scores. However, as it has been demonstrated in text recognition, it is not appropriate for long sequences, *i.e.* seq2seq will suffer to properly recognize long staff lines.

In Chapter 7 and 9 it has been observed that the musical context helps to improve the recognition. However, these methods have the limitation that they are very sensitive to the accuracy of the object detector. An inaccurate detection of primitives will make the OMR method to significantly decrease the overall performance.

Finally, the language models described in Chapter 8 can effectively improve the baseline results. However, they also have some limitations. On the one hand, being a measure-sliced dataset, the language model does not know how many beats a measure has to sum up, so it will guess it purely statistically. On the other hand, it will try to modify artistic elements written by the composer, if any, whenever these are not frequent (so, they have a low probability in the probabilistic model).

10.3 Open Challenges

Along this thesis some lines of research in optical music recognition have been addressed. Despite all the contributions and the latest publications, the research on music score recognition still has a long way to go. Next, given the experience acquired along this thesis in this field, we detail some lines of research that have not yet been investigated, or not sufficiently researched, and could be addressed in the coming years.

General Dataset As we have seen in chapter 3, although there are nowadays a good amount of datasets for music recognition, all of them are designed to be used by a specific type of method. Given the limitations of getting annotated real data, a possible line of research would be the generation of synthetic data with handwritten or printed appearance. This dataset should be labelled in a way to allow the training of different type of methods, *i.e.* the data would be given in various formats. Therefore, works in the literature could be compared with each other.

Layout Analysis Music can be written using more than one staff at a time (e.g. voices written in different staves). An essential line to investigate is layout analysis in order to determine how to correctly recognise these polyphonic music scores. For this purpose, a segmentation free recognition method would be the most interesting to be investigated. For example, given the two dimensionalities of music notation and the representation power of graphs, it could be the best choice to recognize this kind of music scores. The nature of graphs, which can easily create relations between primitives demonstrates that it is suitable for addressing music score recognition without the limitations of a proper segmentation step.

Transfer Learning To preserve the music heritage, it is important to address the recognition of old scores existing in archives. Since old scores do not have labelled data to ease the training of deep learning models, transfer learning needs to be investigated. Indeed, if we could create realistic synthetic data that resembles the real target dataset, we could better adapt the OMR system to recognize such real scores. As a result, one could enjoy unknown scores because there are many music compositions in archives, churches or theatres that have never been listened because that they have not been yet transcribed nor edited.

Music context or Notation Rules Further research is needed towards the incorporation of music notation rules to solve ambiguities and improve the performance. As it has been observed in the music context part, the incorporation of some music knowledge in the model could improve the overall performance. For example, the time measure can be used to check if the number of beats in a bar unit is correct. Indeed, the more musical information the method has, the better it will work and the better the final result will be.

Final Output Finally, a standard output format would allow a standardised assessment among research works, but also, it would benefit the development of software applications. For example, the conversion of the output of the OMR architectures into a MIDI file, MEI or MusicXML formats, which can be listened and edited with music editors (such as PhotoScore), would bring the benefits of optical music recognition research to end users, including musicologists or archivists who want to share the works stored in their music archives.

List of Publications

Share your knowledge. It is a way to achieve immortality.

– Dalai Lama XIV

During the years researching on Optical Music Recognition, several publications have been developed on this topic and others. In this chapter, the Scientific Communications are listed.

Journals

1. **Arnau Baró**, Pau Riba, Jorge Calvo-Zaragoza and Alicia Fornés. From Optical Music Recognition to Handwritten Music Recognition: a Baseline. In *Pattern Recognition Letters*, Vol.123, pp.1–8, 2019, (Q2)

International Conferences

1. **Arnau Baró**, Pau Riba and Alicia Fornés. Towards the recognition of compound music notes in handwritten music scores. In *15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pp.465–470, 2016.

2. **Arnau Baró**, Pau Riba, Jorge Calvo-Zaragoza and Alicia Fornés. Optical Music Recognition by Recurrent Neural Networks. In *12th IAPR International Workshop on Graphic Recognition (GREC)*, pp.25–26, 2017.

3. Alicia Fornés, Verónica Romero, **Arnau Baró**, Juan Ignacio Toledo, Joan Andreu Sánchez, Enrique Vidal, Josep Lladós. ICDAR2017 Competition on Information Extraction in Historical Handwritten Records. In *14th International Conference on Document Analysis and Recognition (ICDAR)*, pp. 1389-1394, 2017.

4. **Arnau Baró**, Pau Riba, Alicia Fornés. A Starting Point for Handwritten Music Recognition. In *1st International Workshop on Reading Music Systems (WoRMS)*, pp. 5-6, 2018.

5. **Arnau Baró**, Jialuo Chen, Alicia Fornés, Beata Megyesi. Towards a generic unsupervised method for transcription of encoded manuscripts. In *3rd International Conference on Digital Access to Textual Cultural Heritage (DATeCH)*, pp. 73-78, 2019.

6. **Arnau Baró**, Carles Badal, Alicia Fornés. Handwritten Historical Music Recognition by Sequence-to-Sequence with Attention Mechanism. In *17th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pp. 205-210, 2020.

7. **Arnau Baró**, Carles Badal, Pau Torras, Alicia Fornés. Handwritten Historical Music Recognition through Sequence-to-Sequence with Attention Mechanism. In *In 3rd International Workshop on Reading Music Systems (WoRMS)*, pp. 55-59, 2021.

8. Pau Torras, **Arnau Baró**, Lei Kang, Alicia Fornés. On the Integration of Language Models into Sequence to Sequence Architectures for Handwritten Music Recognition. In *22nd International Society for Music Information Retrieval Conference (ISMIR)*, pp. 690-696, 2021.

9. **Arnau Baró**, Pau Riba and Alicia Fornés. Musigraph: Optical Music Recognition through Object Detection and Graph Neural Network. In *18th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, Accepted, 2022.

Lecture Notes in Computer Science

1. **Arnau Baró**, Pau Riba, Jorge Calvo-Zaragoza and Alicia Fornés. Optical Music Recognition by Long Short-Term Memory Networks. In *Graphic Recognition. Current Trends and Evolutions*, Vol.11009, pp.81–95, 2018.

Bibliography

[1] Aviad Aberdam, Ron Litman, Shahar Tsiper, Oron Anschel, Ron Slossberg, Shai Mazor, R. Manmatha, and Pietro Perona. Sequence-to-sequence contrastive learning for text recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 15302–15312, 2020.

[2] Narendra Ahuja and Sinisa Todorovic. From region based image representation to object discovery and recognition. In *Structural, Syntactic, and Statistical Pattern Recognition*, volume 6218, pages 1–19, 2010.

[3] María Alfaro-Contreras, Antonio Ríos-Vila, Jose J. Valero-Mas, José M. Iñesta, and Jorge Calvo-Zaragoza. Decoupling music notation to improve end-to-end optical music recognition. *Pattern Recognition Letters*, 158:157–163, 2022.

[4] María Alfaro-Contreras and Jose J. Valero-Mas. Exploiting the two-dimensional nature of agnostic music notation for neural optical music recognition. *Applied Sciences*, 11, 2021.

[5] Carles Badal. Pau llinàs i la vida a santa maria del pi (1711-1749): biografia i inventari. *Revista Catalana de Musicologia*, IX:153–173, 2017.

[6] David Bainbridge and Tim Bell. The challenge of optical music recognition. *Computers and the Humanities*, 35:95–121, 2001.

[7] Arnau Baró, Pau Riba, Jorge Calvo-Zaragoza, and Alicia Fornés. Optical music recognition by long short-term memory networks. In *Graphic Recognition. Current Trends and Challenges. LNCS*, pages 81–95, 2018.

[8] Arnau Baró, Pau Riba, Jorge Calvo-Zaragoza, and Alicia Fornés. From optical music recognition to handwritten music recognition: A baseline. *Pattern Recognition Letters*, 123:1–8, 2019.

[9] Arnau Baró, Pau Riba, and Alicia Fornés. Towards the recognition of compound music notes in handwritten music scores. In *International Conference on Frontiers in Handwriting Recognition*, pages 465–470, 2016.

[10] Arnau Baró, Carles Badal, and Alicia Fornés. Handwritten historical music recognition by sequence-to-sequence with attention mechanism. In *International Conference on Frontiers in Handwriting Recognition*, pages 205–210, 2020.

[11] Nicolas Bell. Byzantine music and musical manuscripts. `https://brewminate.com/byzantine-music-and-musical-manuscripts/`, 2018. Accessed: 2022-07-26.

[12] Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. Greedy layer-wise training of deep networks. In *Advances in Neural Information Processing Systems*, pages 153–160, 2007.

[13] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the International Conference on Machine Learning*, pages 41–48, 2009.

[14] Navaneeth Bodla, Bharat Singh, Rama Chellappa, and Larry S. Davis. Soft-nms – improving object detection with one line of code. In *Proceedings of the IEEE International Conference on Computer Vision*, 2017.

[15] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.

[16] John Burgoyne, Johanna Devaney, Laurent Pugin, and Ichiro Fujinaga. Enhanced bleedthrough correction for early music documents with recto-verso registration. In *International Society for Music Information Retrieval*, pages 407–412, 01 2008.

[17] John Ashley Burgoyne, Ichiro Fujinaga, and J. Stephen Downie. *Music Information Retrieval*, chapter 15, pages 213–228. John Wiley & Sons, Ltd, 2015.

[18] John Ashley Burgoyne, Yue Ouyang, Tristan Himmelman, Johanna Devaney, Laurent Pugin, and Ichiro Fujinaga. Lyric extraction and recognition on digital images of early music sources. In *International Society for Music Information Retrieval*, pages 723–727, 2009.

[19] Donald Byrd and Jakob Grue Simonsen. Towards a standard testbed for optical music recognition: Definitions, metrics, and page images. *New Music Research*, pages 169–195, 2015.

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

[21] Jorge Calvo-Zaragoza, Jan Hajič Jr., and Alexander Pacha. Understanding optical music recognition. *ACM Computing Surveys*, 53:1–35, 2020.

[22] Jorge Calvo-Zaragoza and Jose Oncina. Recognition of pen-based music notation: The homus dataset. In *Proceedings of the International Conference on Pattern Recognition*, pages 3038–3043, 2014.

[23] Jorge Calvo-Zaragoza and Jose Oncina. Recognition of pen-based music notation with probabilistic machines. In *International Workshop on Machine Learning and Music*, 2014.

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

[25] Jorge Calvo-Zaragoza and David Rizo. Camera-primus: Neural end-to-end optical music recognition on realistic monophonic scores. In *International Society for Music Information Retrieval*, pages 248–255, 2018.

[26] Jorge Calvo-Zaragoza and David Rizo. End-to-end neural optical music recognition of monophonic scores. *Applied Sciences*, 8:1–23, 2018.

[27] Jorge Calvo-Zaragoza, David Rizo, and Jose M. Iñesta. Two (note) heads are better than one: pen-based multimodal interaction with music scores. In *International Society for Music Information Retrieval*, 2016.

[28] Jorge Calvo-Zaragoza, Alejandro H Toselli, and Enrique Vidal. Handwritten music recognition for mensural notation with convolutional recurrent neural networks. *Pattern Recognition Letters*, 128:115–121, 2019.

[29] Jorge Calvo-Zaragoza, Alejandro Héctor Toselli, and Enrique Vidal. Handwritten music recognition for mensural notation: Formulation, data and baseline results. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 1081–1086, 2017.

[30] Jorge Calvo-Zaragoza, Jose J. Valero-Mas, and Antonio Pertusa. End-to-end optical music recognition using neural networks. In *International Society for Music Information Retrieval*, 2017.

[31] Vicente Bosch Campos, Jorge Calvo-Zaragoza, Alejandro Héctor Toselli, and Enrique Vidal-Ruiz. Sheet music statistical layout analysis. In *International Conference on Frontiers in Handwriting Recognition*, pages 313–318, 2016.

[32] Francisco Castellanos, Jorge Calvo-Zaragoza, and Jose Iñesta. A neural approach for full-page optical music recognition of mensural documents. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 59,73, 2020.

[33] Sukalpa Chanda, Debleena Das, Umapada Pal, and Fumitaka Kimura. Offline hand-written musical symbol recognition. In *International Conference on Frontiers in Handwriting Recognition*, pages 405–410, 2014.

[34] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *EMNLP*, pages 1724–1734, 2014.

[35] Jan Chorowski, Dzmitry Bahdanau, Dmitriy Serdyuk, Kyunghyun Cho, and Yoshua Bengio. Attention-based models for speech recognition. In *Advances in Neural Information Processing Systems*, page 577–585, 2015.

[36] Daniel Codina. La música vocal d'església en el barroc. In *Història de la Música Catalana, Valenciana i Balear II. Barroc i Classicisme*, pages 27–29. Barcelona: Edicions 62, 1999.

[37] Bertrand Coüasnon and Bernard Rétif. Using a grammar for a reliable full score recognition system. In *International Computer Music Association*, 1995.

[38] Rui Miguel Filipe da Silva. Mobile framework for recognition of musical characters. Master's thesis, Universidade do Porto, 2013.

[39] Christoph Dalitz, Michael Droettboom, Bastian Pranzas, and Ichiro Fujinaga. A comparative study of staff removal algorithms. In *Pattern Analysis and Machine Intelligence*, pages 753–766, 2008.

[40] Biblioteca de Catalunya. Fons musical de la catedral de barcelona (biblioteca de catalunya). `https://mdc.csuc.cat/digital/collection/musicatedra`. Accessed: 2022-08-02.

[41] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in Neural Information Processing Systems*, pages 3844–3852, 2016.

[42] Arnaud F. Desaedeleer. Reading sheet music. Master's thesis, University of London, 2006.

[43] Matthias Dorfer, Jan Hajič jr., Andreas Arzt, Harald Frostel, and Gerhard Widmer. Learning audio-sheet music correspondences for cross-modal retrieval and piece identification. In *International Society for Music Information Retrieval*, pages 22–33, 2018.

[44] Igor Dos Santos Montagner, Roberto Hirata, and Nina S.T. Hirata. A machine learning based method for staff removal. In *Proceedings of the International Conference on Pattern Recognition*, pages 3162–3167, 2014.

[45] Sergio Escalera, Alicia Fornés, Oriol Pujol, Petia Radeva, Gemma Sánchez, and Josep Lladós. Blurred Shape Model for binary and grey-level symbol recognition. *Pattern Recognition Letters*, 30:1424–1433, 2009.

[46] Sergio Escalera, Alicia Fornés, Oriol Pujol, Josep Lladós, and Petia Radeva. Circular blurred shape model for multiclass symbol recognition. *IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society*, pages 497–506, 2011.

[47] Jonathan Feist. *Berklee Contemporary Music Notation.* Berklee Press, 2017.

[48] Alicia Fornés, Anjan Dutta, Albert Gordo, and Josep Lladós. CVC-MUSCIMA: a ground truth of handwritten music score images for writer identification and staff removal. *IJDAR*, 15(3):243–251, 2012.

[49] Alicia Fornés, Van Cuong Kieu, Muriel Visani, Nicholas Journet, and Anjan Dutta. The icdar/grec 2013 music scores competition: Staff removal. In *Graphics Recognition. Current Trends and Challenges. LNCS.*, pages 207–220, 2014.

[50] Alicia Fornés, Josep Lladós, Gemma Sánchez, and Dimosthenis Karatzas. Rotation invariant hand drawn symbol recognition based on a dynamic time warping model. *International Journal on Document Analysis and Recognition*, 13:229–241, 2010.

[51] Alicia Fornés, Josep Lladós, and Gemma Sánchez. Old handwritten musical symbol classification by a dynamic time warping based method. In *International Workshop on Graphics Recognition*, 2008.

[52] Volkmar Frinken and Horst Bunke. Continuous handwritten script recognition. In *Handbook of Document Image Processing and Recognition*, pages 391–425. Springer, 2014.

[53] Ichiro Fujinaga and Andrew Hankinson. Single Interface for Music Score Searching and Analysis (SIMSSA). In *International Conference on Technologies for Music Notation and Representation*, pages 109–115, 2015.

[54] Antonio-Javier Gallego and Jorge Calvo-Zaragoza. Staff-line removal with selectional auto-encoders. *Expert Systems with Applications*, 89:138–48, 2017.

[55] Susan E. George. Online pen-based recognition of music notation with artificial neural networks. *Computer Music Journal*, 27:70—-79, 2003.

[56] Susan E. George. Wavelets for dealing with super-imposed objects in recognition of music notation. In *Visual Perception of Music Notation*, pages 78—107, 2004.

[57] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In *Proceedings of the International Conference on Machine Learning*, pages 1263–1272, 2017.

[58] Xavier Glorot, Antoine Bordes, and Y Bengio. Deep sparse rectifier neural networks. *Journal of Machine Learning Research*, 15:315–323, 2010.

[59] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[60] M. Gori, G. Monfardini, and F. Scarselli. A new model for learning in graph domains. In *IEEE International Joint Conference on Neural Networks*, volume 2, pages 729–734, 2005.

[61] Elaine Gould. *Behind Bars: The Definitive Guide To Music Notation*. Faber Music, 2011.

[62] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural 'networks. In *Proceedings of the International Conference on Machine Learning*, volume 2006, pages 369–376, 2006.

[63] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6645–6649. IEEE, 2013.

[64] Alex Graves and Jürgen Schmidhuber. Offline handwriting recognition with multidimensional recurrent neural networks. In *NIPS*, pages 545–552, 2009.

[65] Caglar Gulcehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, Loic Barrault, Huei-Chi Lin, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. On using monolingual corpora in neural machine translation. *arXiv preprint arXiv:1503.03535*, 2015.

[66] Lluís Gómez and Dimosthenis Karatzas. Multi-script text extraction from natural scenes. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 467–471, 2013.

[67] Jan Hajič jr. and Pavel Pecina. Detecting noteheads in handwritten scores with convnets and bounding box regression. *CoRR*, abs/1708.01806, 2017.

[68] Jan Hajič jr. and Pavel Pecina. The MUSCIMA++ Dataset for Handwritten Optical Music Recognition. In *ICDAR*, pages 39–46, 2017.

[69] Will Hamilton, Zhitao Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*, volume 30, pages 1024–1034, 2017.

[70] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[71] Donald O. Hebb. *The organization of behavior: A neuropsychological theory*. Wiley, 1949.

[72] Mikael Henaff, Joan Bruna, and Yann LeCun. Deep convolutional networks on graph-structured data. *arXiv preprint arXiv:1506.05163*, 2015.

[73] George Heussenstamm. *The Norton Manual of Music Notation*. W. W. Norton & Company, 1987.

[74] Geoffrey E. Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006.

[75] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997.

[76] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *Proceedings of the International Conference on Machine Learning*, 2015.

[77] Graham Jones, Bee Ong, Ivan Bruno, and Kia Ng. Optical music imaging: Music document digitisation, recognition, evaluation, and restoration. *Interactive Multimedia Music Technologies*, pages 50–79, 01 2007.

[78] Lei Kang, Pau Riba, Mauricio Villegas, Alicia Fornés, and Marçal Rusiñol. Candidate fusion: Integrating language modelling into a sequence-to-sequence handwritten word recognition architecture. *Pattern Recognition*, 112:107790, 2021.

[79] Lei Kang, Marçal Rusiñol, Alicia Fornés, Pau Riba, and Mauricio Villegas. Unsupervised adaptation for synthetic-to-real handwritten word recognition. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, 2020.

[80] Lei Kang, J. Toledo, Pau Riba, Mauricio Villegas, Alicia Fornés, and Marçal Rusiñol. Convolve, attend and spell: An attention-based sequence-to-sequence model for handwritten word recognition. In *Proceedings of the German Conference on Pattern Recognition*, pages 459–472, 2019.

[81] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations*, 2015.

[82] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *Proceedings of the International Conference on Learning Representations*, 2017.

[83] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, volume 25, pages 1–9, 2012.

[84] Yan LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1:541–551, 1989.

[85] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *Proceedings of the European Conference on Computer Vision*, pages 740–755, 2014.

[86] Josep Lladós, Marçal Rusiñol, Alicia Fornés, David Fernandez, and Anjan Dutta. On the influence of word representations for handwritten word spotting in historical documents. *International Journal of Pattern Recognition and Artificial Intelligence*, 26, 2012.

[87] Toshiaki Matsushima, Sadamu Ohteru, and Shuji Hashimoto. An integrated music information processing system: Psb-er. *Proceedings of the international computer music conference*, pages 191–198, 1989.

[88] Warren S. McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5:115–133, 1943.

[89] Tomáš Mikolov, Martin Karafiát, Lukáš Burget, Jan Černockỳ, and Sanjeev Khudanpur. Recurrent neural network based language model. In *Eleventh annual conference of the international speech communication association*, 2010.

[90] Hidetoshi Miyao and Minoru Maruyama. An online handwritten music symbol recognition system. *International Journal of Document Analysis and Recognition (IJDAR)*, 9:49–58, 2007.

[91] Igor S. Montagner, Nina S.T. Hirata, and Roberto Hirata. Staff removal using image operator learning. *Pattern Recognition*, 63:310–320, 2017.

[92] Savitri Apparao Nawade, Shivanand Rumma, Rajmohan Pardeshi, and Mallikarjun Hangarge. Old handwritten music symbol recognition using radon and discrete wavelet transform. In Niranjan N. Chiplunkar and Takanori Fukao, editors, *Advances in Artificial Intelligence and Data Engineering*, pages 1165–1171, 2021.

[93] Kia Ng. Music manuscript tracing. In *Graphics Recognition Algorithms and Applications*, pages 330–342, 2002.

[94] Kia Ng, Alex Mclean, and Alan Marsden. Big data optical music recognition with multi images and multi recognisers. In *Electronic Visualisation and the Arts*, 2014.

[95] Alexander Pacha. Omr-datasets. `https://apacha.github.io/OMR-Datasets/`, 2021. Accessed: 2022-07-28.

[96] Alexander Pacha and Jorge Calvo-Zaragoza. Optical music recognition in mensural notation with region-based convolutional neural networks. In *ISMIR*, pages 240–247, 2018.

[97] Alexander Pacha, Kwon-Young Choi, Bertrand Coüasnon, Yann Ricquebourg, Richard Zanibbi, and Horst M. Eidenberger. Handwritten music object detection: Open issues and baseline results. In *International Workshop on Document Analysis Systems*, pages 163–168, 2018.

[98] Alexander Pacha and Horst Eidenberger. Towards a universal music symbol classifier. In *International Workshop on Graphics Recognition*, pages 35–36, 2017.

[99] Alexander Pacha and Horst Eidenberger. Towards self-learning optical music recognition. In *International Conference on Machine Learning and Applications*, pages 795–800, 2017.

[100] R. Padilla, S. L. Netto, and E. A. B. da Silva. A survey on performance metrics for object-detection algorithms. In *International Conference on Systems Signals and Image Processing*, pages 237–242, 2020.

[101] Rafael Padilla, Wesley L. Passos, Thadeu L. B. Dias, Sergio L. Netto, and Eduardo A. B. da Silva. A comparative analysis of object detection metrics with a companion open-source toolkit. *Electronics*, 10, 2021.

[102] Emilia Parada-Cabaleiro, Anton Batliner, Alice Baird, and Björn W. Schuller. The seils dataset: Symbolically encoded scores in modernancient notation for computational musicology. In *International Society for Music Information Retrieval*, pages 575–581, 2017.

[103] Emilia Parada-Cabaleiro, Anton Batliner, and Björn W. Schuller. A diplomatic edition of il lauro secco: Ground truth for omr of white mensural notation. In *International Society for Music Information Retrieval*, pages 557–564, 2019.

[104] Emilia Parada-Cabaleiro, Maximilian Schmitt, Anton Batliner, and Björn W. Schuller. Musical-linguistic annotation of il lauro secco. In *International Society for Music Information Retrieval*, pages 461–467, 2018.

[105] Fabrizio Pedersoli and George Tzanetakis. Document segmentation and classification into musical scores and text. *International Journal on Document Analysis and Recognition*, 19:289–304, 2016.

[106] Roberto M. Pinheiro Pereira, Caio E.F. Matos, Geraldo Braz Junior, João D.S. de Almeida, and Anselmo C. de Paiva. A deep approach for handwritten musical symbols recognition. In *Symposium on Multimedia and the Web*, page 191–194, New York, NY, USA, 2016.

[107] João Caldas Pinto, Pedro Vieira, and João M. Sousa. A new graph-like classification method applied to ancient handwritten musical symbols. *Document Analysis and Recognition*, 6:10–22, 2003.

[108] Telmo Pinto, Ana Rebelo, Gilson A. Giraldi, and Jaime S. Cardoso. Music score binarization based on domain knowledge. In *Pattern Recognition and Image Analysis*, pages 700–708, 2011.

[109] Laurent Pugin. Optical music recognitoin of early typographic prints using hidden markov models. In *International Society for Music Information Retrieval*, 2006.

[110] Laurent Pugin, John Burgoyne, and Ichiro Fujinaga. Reducing costs for digitising early music with dynamic adaptation. In *European conference on Research and Advanced Technology for Digital Libraries*, pages 471–474, 08 2007.

[111] Laurent Pugin, John Ashley Burgoyne, and Ichiro Fujinaga. Map adaptation to improve optical music recognition of early music documents using hidden markov models. In *International Society for Music Information Retrieval*, 2007.

[112] Laurent Pugin and Tim Crawford. Evaluating omr on the early music online collection. In *International Society for Music Information Retrieval*, 2013.

[113] L. Rabiner and B. Juang. An introduction to hidden markov models. *IEEE ASSP Magazine*, 3:4–16, 1986.

[114] Marc'aurelio Ranzato, Christopher Poultney, Sumit Chopra, and Yann L. Cun. Efficient learning of sparse representations with an energy-based model. In *Advances in Neural Information Processing Systems*, pages 1137–1144, 2007.

[115] Ana Rebelo, G. Capela, and Jaime S. Cardoso. Optical recognition of music symbols: A comparative study. *International Journal on Document Analysis and Recognition*, 13:19–31, 2010.

[116] Ana Rebelo, Ichiro Fujinaga, Filipe Paszkiewicz, Andre R. S. Marcal, Carlos Guedes, and Jaime S. Cardoso. Optical music recognition: state-of-the-art and open issues. *International Journal of Multimedia Information Retrieval*, 1:173–190, 2012.

[117] Ana Rebelo, Jakub Tkaczuk, Ricardo Sousa, and Jaime S. Cardoso. Metric learning for music symbol recognition. In *International Conference on Machine Learning and Applications and Workshops*, volume 2, pages 106–111, 2011.

[118] Pau Riba, Andreas Fischer, Josep Lladós, and Alicia Fornés. Learning graph distances with message passing neural networks. In *Proceedings of the International Conference on Pattern Recognition*, pages 2239–2244, 2018.

[119] Pau Riba, Alicia Fornés, and Josep Lladós. Towards the alignment of handwritten music scores. In *Graphics Recognition. Current Trends and Challenges. LNCS.*, 2015.

[120] Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–408, 1958.

[121] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533–536, 1986.

[122] Antonio Ríos-Vila, Jorge Calvo-Zaragoza, and José M. Iñesta. Exploring the two-dimensional nature of music notation for score recognition with end-to-end approaches. In *International Conference on Frontiers in Handwriting Recognition*, pages 193–198, 2020.

[123] Victor Garcia Satorras and Joan Bruna Estrach. Few-shot learning with graph neural networks. In *Proceedings of the International Conference on Learning Representations*, 2018.

[124] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.

[125] Eleanor Selfridge-Field. *Beyond MIDI: The Handbook of Musical Codes*. MIT Press, 1997.

[126] Elona Shatri and György Fazekas. Doremi: First glance at a universal omr dataset. In *International Workshop on Reading Music Systems*, 2021.

[127] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 09 2014.

[128] Mahmood Sotoodeh, Farshad Tajeripour, Sadegh Teimori, and Kirk Jorgensen. A music symbols recognition method using pattern matching along with integrated projection and morphological operation techniques. *Multimedia Tools Applications*, 77:16833—-16866, 2018.

[129] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *NeurIPS*, pages 3104–3112, 2014.

[130] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[131] Shubham Toshniwal, Anjuli Kannan, Chung-Cheng Chiu, Yonghui Wu, Tara N Sainath, and Karen Livescu. A comparison of techniques for language model integration in encoder-decoder speech recognition. In *SLT*, pages 369–375, 2018.

[132] Timothy Tsai, Daniel Yang, Mengyi Shan, Thitaree Tanprasert, and TTeerapat Jenrungrot. Using cell phone pictures of sheet music to retrieve midi passages. In *IEEE Transactions on Multimedia*, 2020.

[133] Lukas Tuggener, Ismail Elezi, Jürgen Schmidhuber, and Thilo Stadelmann. Deep watershed detector for music object recognition. In *International Society for Music Information Retrieval*, pages 271–278, 2018.

[134] Lukas Tuggener, Yvan Putra Satyawan, Alexander Pacha, Jürgen Schmidhuber, and Thilo Stadelmann. The deepscoresv2 dataset and benchmark for music object detection. In *Proceedings of the International Conference on Pattern Recognition*, pages 9188–9195, 2021.

[135] Eelco van der Wel and Karen Ullrich. Optical music recognition with convolutional sequence-to-sequence models. In *International Society for Music Information Retrieval*, pages 731–737, 2017.

[136] Cuihong Wen, Ana Rebelo, Jing Zhang, and Jaime Cardoso. A new optical music recognition system based on combined neural network. *Pattern Recognition Letters*, 58:1 – 7, 2015.

[137] Christoph Wick, Christian Reul, and Frank Puppe. Calamari - A High-Performance Tensorflow-based Deep Learning Package for Optical Character Recognition. *Digital Humanities Quarterly*, 14, 2020.

[138] Yi-Chao Wu, Fei Yin, and Cheng-Lin Liu. Improving handwritten chinese text recognition using neural network language models and convolutional neural network shape models. *Pattern Recognition*, 65:251–264, 2017.

[139] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019.

[140] Frank Zalkow, Angel Villar Corrales, TJ Tsai, Vlora Arifi-Müller, , and Meinard Müller. Tools for semi-automatic bounding box annotation of musical measures in sheet music. In *International Society for Music Information Retrieval*, 2019.

This work has been partially supported by the FI fellowship AGAUR 2018 FI_B 00546, 2019 FI_B1 00196 and 2020 FI_B2 00149 (with the support of the Secretaria d'Universitats i Recerca of the Generalitat de Catalunya and the Fons Social Europeu). We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.