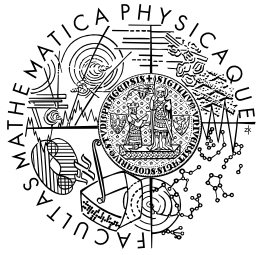


FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

## Report for GA UK Project

# ~~Multimodal~~ Optical Music Recognition using Deep Learning

GA UK Project no. 1444217
Principal Investigator Jan Hajič jr.
Project duration: 2017 – 2019

Prague 2020

# Executive summary

This technical report summarizes the contributions and findings of the GA UK project no. 1444217 on Optical Music Recognition (OMR).[1] **The project focused on extracting musical semantics, represented by MIDI, from handwritten sheet music.** The main output of the project is a series of peer-reviewed scientific publications that describe how the OMR system was created and evaluate its performance, both of its individual steps and of its performance as a whole.

The main success of the project is creating the first machine learning-based pipeline for recognition of handwritten music notation of arbitrary complexity, achieving state-of-the-art performance. Furthermore, it has significantly contributed to the growth and consolidation of the OMR scientific community, and it has contributed to some advances in cross-modal processing of sheet music and musical audio as well.

The report is structured as follows:

1. A brief introduction to OMR (Chap. 1)
2. A review of OMR state of the art (Chap. 2)
3. The summary of the project's contributions to OMR (Chap. 3)

The state of the art of OMR (Chap. 2) is reviewed in order for the specific value of the project's contributions to become clear (neither under-, nor over-estimated).

The third chapter (Chap. 3) then summarizes the work done in the project: details the main results and describes their importance. (The technical details can be found in the respective publications.) Overall, we are confident to say that the project has contributed significantly to the state of the art of Optical Music Recognition. Note that besides technical contributions resulting in peer-reviewed publications, the existence of the project enabled participation in conferences and workshops that led to establishing substantial "intangibles" in scientific community building: a workshop[2], a hitherto-missing tutorial with introductory materials for newcommers to the otherwise rather opaque field,[3] an up-to-date bibliography of OMR,[4] and a website centralizing OMR research news.[5]

[1] Note that while the project's official name adds the epithet *Multimodal* OMR, due to circumstances described in the project's annual reports, the overwhelming majority of work was done on OMR with image inputs only.

[2] `https://sites.google.com/view/worms2018`, `https://sites.google.com/view/worms2019`

[3] `http://ismir2018.ircam.fr/pages/events-tutorial-07.html`

[4] `https://github.com/OMR-Research/omr-research.github.io`

[5] `https://omr-research.net/`

# Disclaimer

**The text of this report is not original work.**

We freely acknowledge that the text of this report re-uses heavily the text from appropriate sections of the author's dissertation (defended in June 2019), most of which has been based on work done in this project. The report is essentially an updated version of the introductory and summary text of the thesis abridged to only refer to the work done within the project. This report (while of course accurate) makes no claim to be a *scientific* publication, so we see no reason to significantly re-write an already polished text that contains exactly the right information about the project. Chapter 1 is an amalgamation of chapters 1 and portions of chapter 3 of the thesis, chapter 2 re-uses the corresponding portions of chapter 3 of the thesis, and chapter 3 re-uses portions of chapter 4. The thesis chapter 2 which explains how music notation works – what it encodes about a musical composition (and, no less importantly, what it does *not* encode), and how it encodes this musical information – is included in this report as Appendix A.1, in case the reader wishes to refer to this detailed analysis of the music notation writing language (but for understanding the report, this should not be necessary).

# Contents


**1 Introduction to OMR** **4**
- 1.1 What does OMR do? . . . 5
  - 1.1.1 The different inputs of OMR . . . 6
- 1.2 Why is OMR difficult? . . . 8
- 1.3 The project in relation to OMR . . . 11

**2 OMR State of the Art** **12**
- 2.1 Methods . . . 13
  - 2.1.1 Preprocessing . . . 13
  - 2.1.2 Staff detection and removal . . . 15
  - 2.1.3 Object Detection . . . 16
  - 2.1.4 Notation Assembly . . . 16
  - 2.1.5 Constructing the output representation . . . 18
  - 2.1.6 End-to-end OMR . . . 18
  - 2.1.7 Interactive OMR . . . 19
  - 2.1.8 Online OMR . . . 19
- 2.2 Infrastructure of OMR . . . 20
  - 2.2.1 Commercial Software . . . 21

**3 Contributions** **23**
- 3.1 Music Notation Graph . . . 24
- 3.2 MUSCIMA++ . . . 26
- 3.3 The Recognition Pipeline . . . 27
  - 3.3.1 Object Detection . . . 28
  - 3.3.2 Detection in Full Pipeline . . . 31
  - 3.3.3 Notation Assembly and Semantics Inference . . . 34
  - 3.3.4 Full Pipeline Results . . . 35
  - 3.3.5 Applications: Retrieval and Digital Musicology . . . 37
- 3.4 Auxilliary contributions . . . 38
  - 3.4.1 Audio – sheet music cross-modal retrieval . . . 38
  - 3.4.2 OMR Scientific Community . . . 41

**Publications by GA UK 144217** **43**

**Bibliography** **44**

**A Attachments** **56**
- A.1 Reading Music . . . 56
  - A.1.1 Musical Semantics . . . 56
  - A.1.2 Music Notation . . . 60

# 1. Introduction to OMR

Optical Music Recognition (OMR) is the field of research that investigates how to computationally read music notation. Music notation is an established visual language that encodes music graphically; the role of OMR is to automatically understand this encoding and extract the encoded musical information from this graphical representation.

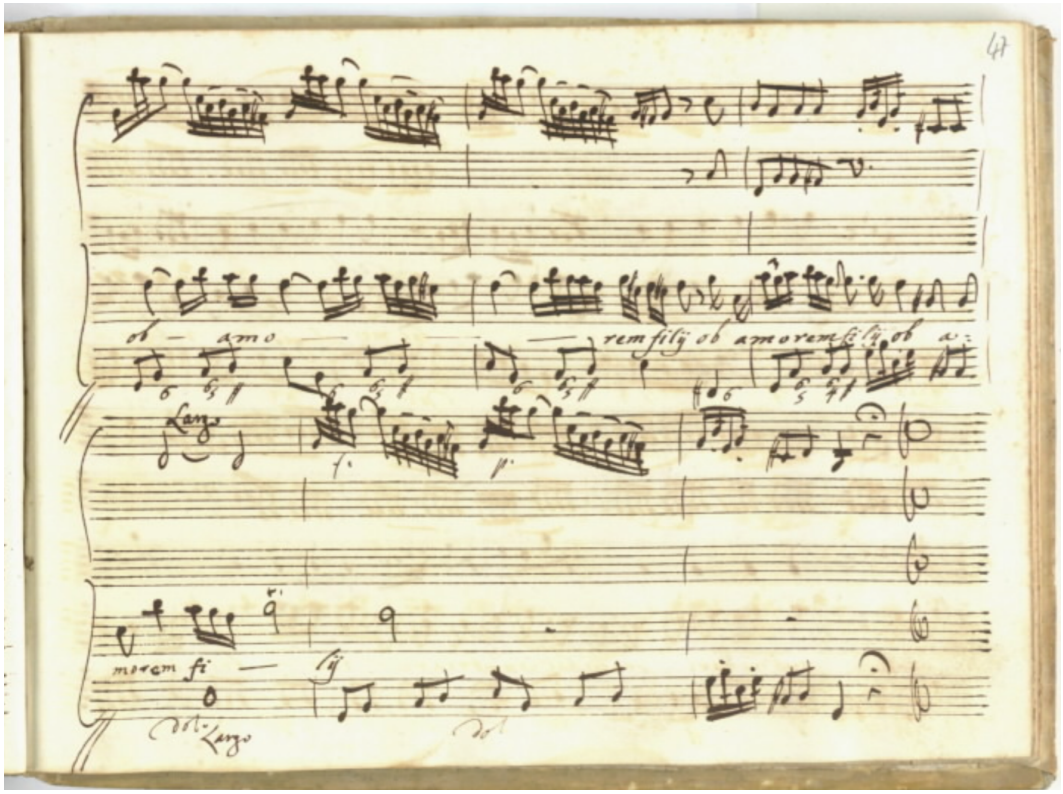



Figure 1.1: An example of a musical manuscript: a copy of G. B. Pergolesi's Stabat Mater, part X: Fac, ut portem Christi mortem.

What is the motivation for OMR?

In European culture, and wherever it has been able to reach, music notation is the primary way of transferring music from composer to performer. The Common Western Music Notation (CWMN) writing system evolved over the course of the 17th and 18th centuries and has since been used to encode tens or hundreds of thousands of compositions, one of the defining bodies of European cultural heritage (like the manuscript in Fig. 1.1). It is daily in use by musicians ranging from children to professionals, composers as well as performers, and reading music notation is one of the skills that belongs to a general education.

As the digital domain is becoming the primary domain for manipulation and dissemination of source materials, digitizing this body of cultural heritage becomes essential for meaningful preservation. There are commercial solutions for

automatically reading printed music (which can hope to garner enough users to become economically viable), but this software cannot cope with musical *manuscripts* – which are the more valuable component of the source material: many *compositions* are recorded only in manuscript form.[1]

Digitization efforts have been undertaken by institutions holding large collections of music scores, such as the SLUB[4] in Dresden or the Bavarian State Library in Munich,[5] or by organizations dedicated solely to facillitating access to scans and born-digital scores such as the IMSLP[6] and CPDL[7] projects. These projects lack, however, the capability to make digitally accessible not only an *image* of the music (which in and of itself is already extremely valuable), but also its *musical content*: essentially, what the given music would sound like. Having digital access to the music encoded by music notation in the given documents would enable novel ways of interacting with the accumlated body of music scores, such as musical "full-text" search,[8] re-typesetting old and contemporary manuscripts, creating full scores from collections where only parts for individual instruments survive – and vice versa, exporting parts for individual instruments from the full scores; cross-modal retrieval, digital musicology at scale and with access to music that has never been recorded, and cost-cutting tools for composers or music directors. Anecdotally, OMR remains one of the applications of computer science in the musical domain where the gap between obvious expectations and delivered results remains widest.

Besides the angle of cultural heritage preservation, dissemination, and deeper understanding, functioning OMR would singificantly decrease the costs of working with contemporarily-produced sheet music as well: many scores are being made available as PDFs, ususally scans; standard unsophisticated musical operations (such as exporting parts from orchestral or chamber scores, or transpositions) currently require re-writing the music in a notation software by hand, which is extremely time-consuming.

## 1.1 What does OMR do?

The process of *reading music* can be formulated as the process of correctly inferrng the *notes* encoded graphically using the *music notation* visual language

[1] The reasons for this are economical. Before the advent of personal computers and the proliferation of software such as Sibelius[2] or MuseScore,[3] music typesetting was a very costly endeavor reserved for authors and compositions with practically assurred chances of market success, or – in earlier times – with a particular printing privilege; therefore, most compositions never had the chance to be typeset.

[4] `https://www.slub-dresden.de/en/collections/music/`

[5] `https://www.bsb-muenchen.de/en/collections/music/`

[6] `https://imslp.org`

[7] `https://cpdl.org`

[8] The composition is often referred to in music and musicology as *musical text*; hence the term is indeed appropriate.

in a document that is commonly called the *score*. Notes are abstract musical objects that are determined by five attributes – pitch (on a piano, which key to press), duration (how long to hold it), loudness and timbre (which are not encoded in music notation, aside from signs for some instrument-specific playing techniques), and the fifth attribute is onset: when should one press the given key, in relation to the start of the composition. Recovering the ⟨`pitch, duration, onset`⟩ triplets is sufficient to then create a practical representation for further processing in most of the applications utilizing of OMR (such as searching for a piece based on a short melody); one widespread such representation is the MIDI file.[9] This is the first major part of the problem of Optical Music Recognition: extracting the *musical semantics*, defined as the set of these triplets.[10]

Apart from extracting the set of notes encoded by a music notation document, the second major task of OMR is recording *how* these notes were encoded: creating a digital representation of the *score* itself. This is a different objective: one may recover the musical semantics without explicitly recording information about how the semantics were encoded (e.g., one need not remember whether the stem of a half-note was oriented up, or down). Due to the nature of the music notation writing system, recovering the score itself requires a more complex representation than a set of triplets. Typical file formats for storing music notation are MusicXML,[11] or `*.mscz`, `*.sib` and other formats used by music notation editors. That this is a more complex task is perhaps best illustrated by the fact that the same semantics can be represented by many different configurations of music notation symbols, as attested to in Fig. 1.2.

### 1.1.1 The different inputs of OMR

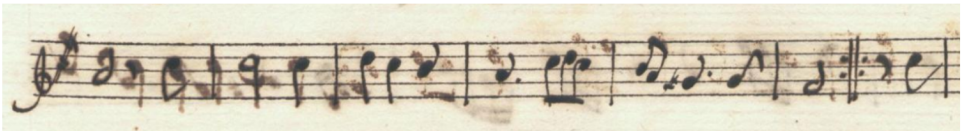
Orthogonally to their goals, OMR systems can be characterized by the types of input they are designed to process (see Fig. 1.3).

The first major difference lies in the **input signal**: we differentiate *offline* OMR, which processes an image, from *online* OMR, which processes the temporal signal from a touch-based device (such as writing with a stylus on a tablet). The latter is in principle simpler because the pen strokes represent a very good natural segmentation heuristic; however, the former is more broadly applicable: while online OMR has its place whenever a composer or arranger is willing to use a device that records the trajectory information information, it cannot deal with the stacks of sheet music that have already been written. An interesting combination, however, is to use online OMR in ground truth acquisition, as tracing the already written notation is much faster and more natural to qualified annotators (who presumably themselves have ample experience with *writing* mu-

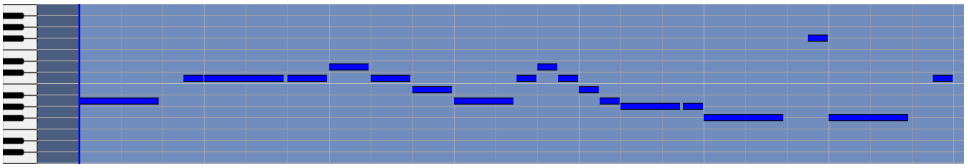
[9] `https://www.midi.org`

[10] Technically, since it is possible for two notes to share all three properties, one needs to assign an ID as well, in order for the notes to formally be a set.

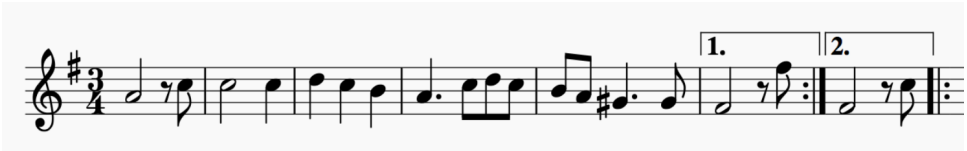
[11] `https://www.musicxml.com`



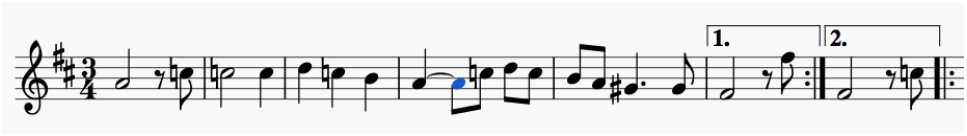
(a) Input: manuscript image.



(b) Replayable output: pitches, durations, onsets. Time is the horizontal axis, pitch is the vertical axis. This visualization is called a *piano roll*.




(c) Reprintable output: re-typesetting.




(d) Reprintable output: same music expressed differently

Figure 1.2: OMR for replayability and reprintability. The input (a) encodes the sequence of pitches, durations, and onsets (b), which can be expressed in different ways (c, d).

sic notation), as done by Calvo Zaragoza et al. [2016a]. The second distinction based on input signal is whether the music in question is typeset, or handwritten, with obvious implications for symbol intra-class symbol variability. Third, one must specify what type of music notation a system is designed to process: CWMN, mensural notation, choral square notation, tabulature (lute, modern guitar, North German organ...), etc.

A second major axis of classifying OMR systems by input is according to the **complexity of notation** they are able to process. This was described in depth by Byrd and Simonsen [2015]; we use a slightly different classification that nevertheless preserves the spirit of the original categories:

- *Monophonic:* each staff contains at most one voice; each simultaneity contains at most one note.
- *Homophonic:* each staff contains at most one voice; each simultaneity can contain more than one note.
- *Polyphonic:* each staff can contain multiple voices, but the staffs can still be processed in isolation.

- *Pianoform:* staffs contain multiple voices, and there is interaction between staffs (e.g., cross-staff beaming).

A third way of characterizing the inputs of OMR is by the image quality: both in terms of the underlying document, and in terms of the imaging process used to digitize the document. Problems that affect the underlying document are degradation over time (especially for archival materials) – most serious of which is bleedthrough – or outright damage to the material (stains and tears). The imaging process then ranges from high-quality scans from music libraries to mobile phone photos in sub-optimal lighting conditions.

An overview of the basic characterizations of OMR inputs is given in Fig. 1.3.

## 1.2 Why is OMR difficult?

OMR is still an open problem and satisfactory solutions are available only for limited sub-problems [Bainbridge and Bell, 2001, Rebelo et al., 2012, Novotný and Pokorný, 2015]. Beside the small size of the field and the accompanying non-technical challenges [Calvo Zaragoza et al., 2018], one reason why OMR is not solved to any satisfactory extent is its sheer difficulty [Byrd and Simonsen, 2015].[12]

While the straightforward intuitive description of OMR as "Optical Character Recognition for music". is appealing, this analogy is only superficially accurate in terms of the purpose of both OMR and OCR. Music notation has evolved into a very different writing system than the writing systems for natural languages: it is a system where one must recover *configurations* of symbols in order to be able to output the musical information that OMR is, by definition of its domain, *expected* to produce. Compared to OCR, which has to output the sequence of graphical symbols (including whitespace) and this already can be presented as useful input for downstream applications in Natural Language Processing, OMR

[12]That OMR is a difficult problem is attested to by the fact that problems connected to the inherent properties of music notation have been called "really rotten" in a publication title, already in 1989 [Clarke et al., 1989]!

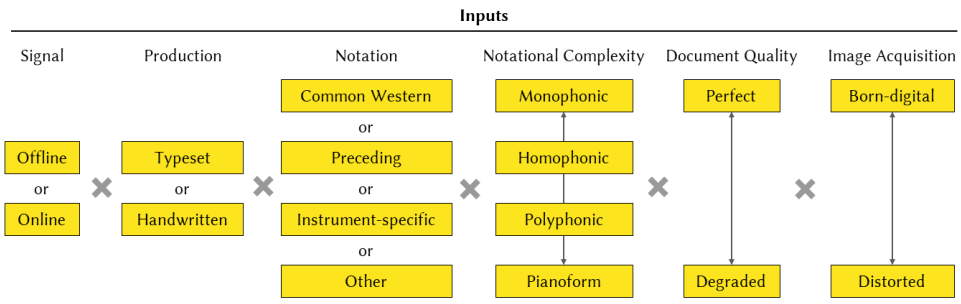



Figure 1.3: The basic ways of characterizing OMR inputs.

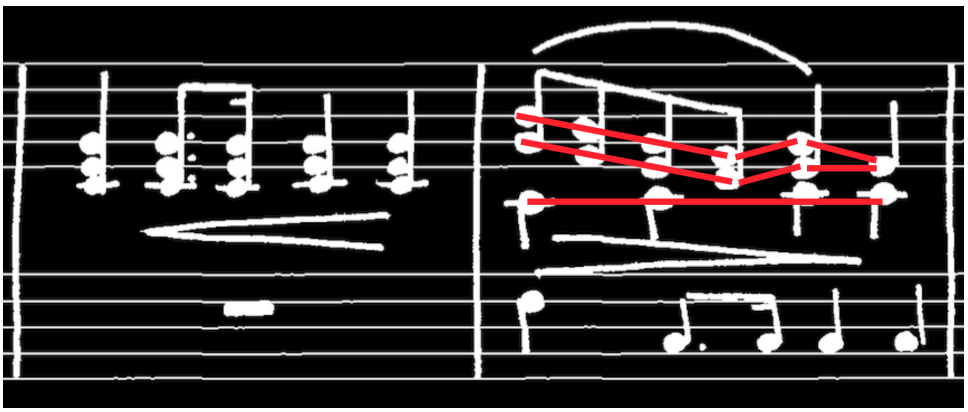


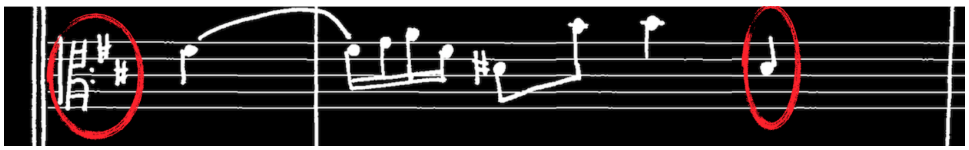
Figure 1.4: The presence of multiple voices (indicated with red lines) adds complications.

must, by virtue of the domain it operates on, perform additional steps in order to be considered useful. This is one fundamental reason why the analogy of OMR to OCR does not hold beyond a superficial similarity of purpose.

A furhter source of difficulty are the visual properties of music notation. According to Byrd and Simonsen [2015], CWMN is probably even the most complex known writing system, especially from the point of view of computer vision. The main reasons why the way CWMN is written makes OMR more difficult than OCR are:

- In order to correctly disambiguate individual symbols, and more generally in order to construct and interpret the symbol configurations correctly, both the horizontal and vertical dimensions are salient, in terms of both size and position.
- Graphical complexity is increased due to the fact that many symbols overlap (especially stafflines) [Bainbridge and Carter, 1997], and by design composite graphical structures are built (esp. beamed groups – see Fig. A.6).
- In handwritten music, besides vastly more varied symbol shapes, the variability of handwriting leads to a lack of reliable topological properties overall (Fig. 1.6) – symbols that should not touch start touching, and conversely gaps are left where symbols should touch or overlap.
- In polyphonic music, individual voices are written, in a sense, "over" each other (some symbols may be shared among multiple voices) – as opposed to OCR, where the ordering of the symbols is linear (Fig. 1.4).
- Recovering pitch and duration requires recovering long-distance relationships (Fig. 1.5).

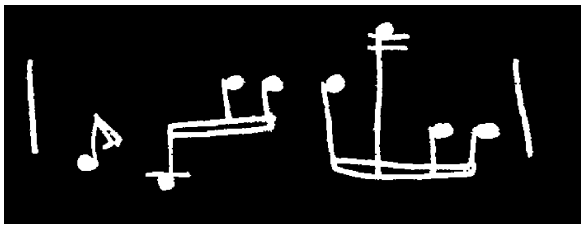
A further hindrance to OMR is that despite its intuitive appeal, the field is small, has had few resources and standards for reproducible OMR research, had little introductory literature for newcomers, and overall lacked internal cohesion.



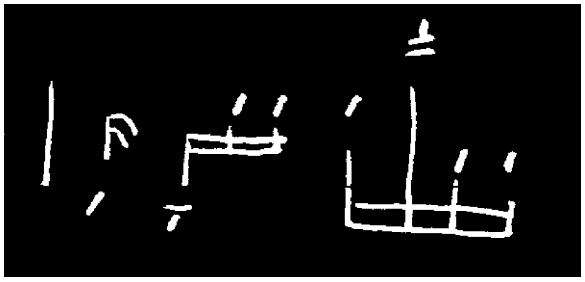
(a) The C-clef on the left influences how stafflines are interpreted with respect to the pitches they denote.

(b) A change of clef and key signature. Also, notice the sharp in the middle: it is valid up to the end of the measure.

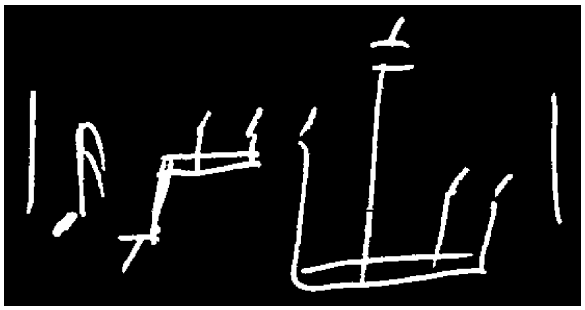
Figure 1.5: Long-distance relationships affecting pitch of the note on the right.



(a) Nice handwriting that follows topological constraints according to ideal printed CWMN.



(b) Disjoint notation primitives.



(c) Very hasty handwriting. Some noteheads may be very hard to distinguish from the stem.

Figure 1.6: The variety of handwriting. Taken from the CVC-MUSCIMA dataset [Fornés et al., 2012].

These challenges and issues have combined to make OMR a relatively immature field that provides few satisfactory solutions.

## 1.3 The project in relation to OMR

The project addresses the task of automatically reading musical manuscripts. Based on the author's previous findings in the "theory" of OMR and the resources he prepared: most importantly, the Music Notation Graph (MuNG[13] formalism for describing music notation that allows formulating the problem of musical manuscript recognition in a machine-learnable manner, and the associated MUSCIMA++ dataset Hajič jr. and Pecina [2017a], in the project, an OMR system that takes an image of handwritten music as input and outputs a MIDI file capturing the musical semantics encoded in the given score is built and evaluated both directly and in a retrieval setting.

In terms of the above introduction to OMR, the project created a solution to *replayability-oriented* OMR that operates on handwritten music notation of arbitrary complexity. The input score images exhibit no degradation, with a clear separation of foreground and background. (While the methods we used do not require that this be true, they would probably require accordingly more data to reach similar levels of performance.)

The inherent variability of mansucripts also points directly towards using statistical methods that can deal with the corresponding uncertainties; we apply machine learning techniques that form the current state of the art in computer vision in general, which is specifically deep learning [Schmidhuber, 2015, LeCun et al., 2015, Ian Goodfellow et al., 2016].

[13] `https://github.com/OMR-Research/mung`

# 2. OMR State of the Art

At its most general, while the "wishlist" of OMR applications exists from the earliest publications [Pruslin, 1966, Prerau, 1971, Fujinaga, 1988, Blostein and Baird, 1992] onwards, after more than 50 years of OMR research, few convincing results have materialized. The reasons for this state of affairs are several. First, despite its intuitive appeal, the field is small (some 500 publications to date), as it requires a combination of computer science expertise and relatively deep domain knowledge of music and music notation. Second, it follows that the field does not have too many resources and established methodologies. Most work on OMR has been focused into PhD theses [Fujinaga, 1996, Bainbridge, 1997, Fornés, 2009, Rebelo, 2012, Calvo Zaragoza, 2016], which is a form that offers little incentive for collaboration and establishing a research community that in turn establishes standards for evaluation and interoperability; therefore, it becomes difficult to build on previous work. Given the lack of standardized, practical evaluation methodologies [Byrd and Simonsen, 2015, Hajič jr. et al., 2016] and even the underlying understanding of *what* should be evaluated, the field cannot in good conscience precisely say what "the state of the art in OMR" is.

Having said that, there are survey papers available for OMR. The first such substantial paper is by Blostein and Baird [1992], which is the first attempt to systematize the field. The key survey paper for OMR up until 2012 is [Rebelo et al., 2012], which systematizes the many approaches and contributions to OMR. The underlying terminology of the field and an analysis of its structure and needs has been done by Byrd and Simonsen [2015]; a smaller but nevertheless useful review paper for developments up to the start of this project has been written by Novotný and Pokorný [2015]. Recently, the paper Understanding Optical Music Recognition [Calvo-Zaragoza et al., 2019][1] systematizes the field from the perspective of its *output*, in addition to Byrd and Simonsen [2015] characterizations by input and Rebelo et al. [2012] by method. Further new resources exist:[2] a list of OMR datasets[3], an OMR bibliography,[4] and a video series that introduces OMR.[5,6] What the survey papers have in common is the assessment that a complete OMR system still lies in the future.[7]

[1] Also created as part of the project; the arXiv version does *not* acknowledge the project; the version currently (31. 2. 2020) under review at the ACM Computational Surveys journal does.

[2] That the project's PI was heavily involved in.

[3] `https://apacha.github.io/OMR-Datasets/`

[4] Originally maintained by Fujinaga [2000], recently updated and verified: `https://github.com/OMR-Research/omr-research.github.io`.

[5] Presented at the ISMIR 2018 conference as a tutorial: `https://www.youtube.com/playlist?list=PL1jvwDVNwQke-04UxzlzY4FM33bo1CGS0`

[6] The latter two resources were to a significant extent co-created by the project PI.

[7] Incidentally, this is the title of[Bainbridge, 1994], "A complete optical music recognition system: Looking to the future".

With these limitations in mind, we turn to introduce the state of OMR, in terms of its methods and the available infrastructure.

## 2.1 Methods

In terms of methods, the problem is usually broken down into the following steps [Bainbridge and Bell, 2001, Fornés et al., 2006, Rebelo et al., 2012, Hankinson, 2014, Novotný and Pokorný, 2015]:

1. **Preprocessing.** This step involves image de-skewing, potentially binarization, and other steps that ensure the image is as normalized as possible for further processing.

2. **Staff detection and removal.** The staffs (horizontal objects consisting, usually, of 5 equally-spaced lines) are the "spines" along which music is read, so detecting them provides basic information about the layout of the sheet music. They are often then removed from the image, as they are responsible for most of the object overlap and crossing; once staffs are removed, segmentation can be done using some heuristics such as connected components. This is a step specific to processing music notation. The pipeline up to this step is depicted in Fig. 2.1.

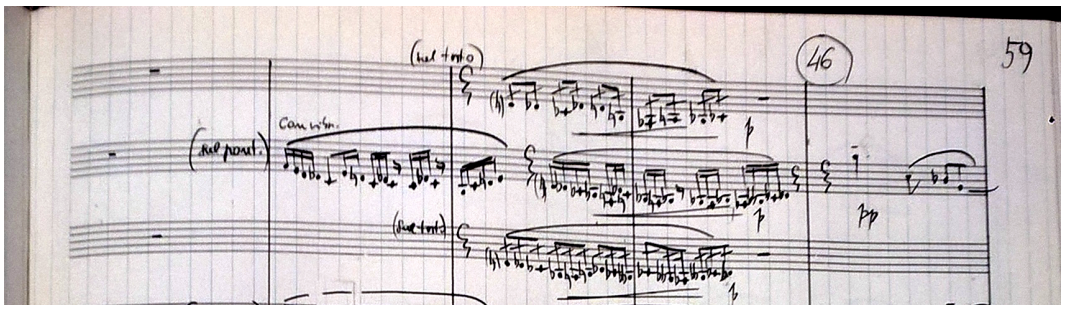
3. **Object detection.** The individual notation objects are then detected, either in two steps (segmentation and classification) in earlier approaches, or detected directly in more recent works, using deep learning. In our view, the distinction from the previous step is mostly a practical issue, not one of principle – stafflines are also symbols that must be detected – but the methods for detecting stafflines have historically been distinct; this is due both to their distinct characteristics and the fact that most methods relied on finding and removing stafflines before the remaining objects could be found.

4. **Notation assembly and semantics inference.** Given the featural nature of music notation as a writing system, the relationships of the individual detected objects to each other must be added (such as: accidentals must be associated with the right note or grouped into a key signature, beams must be correctly assigned to noteheads, etc.) and the musical semantics can thus be inferred, by applying the rules of music notation.

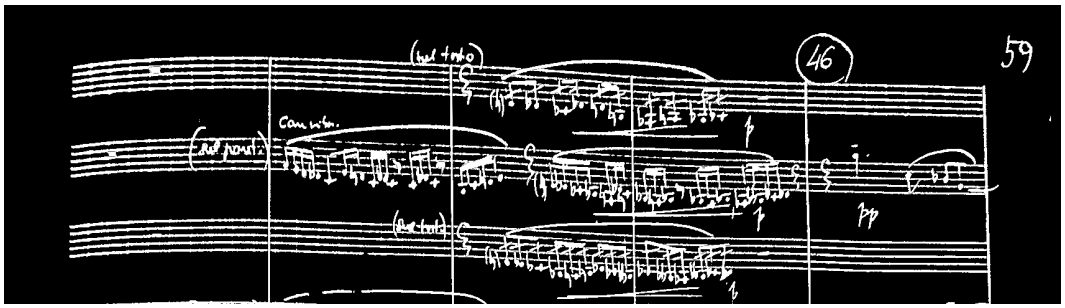
The final step is to construct the output representation in the required format.

### 2.1.1 Preprocessing

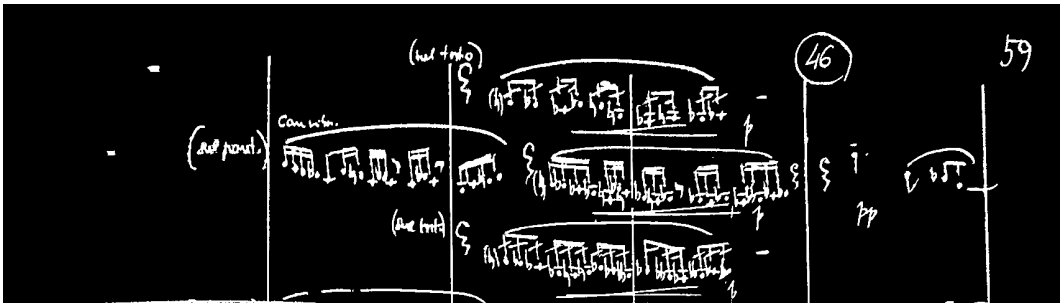
Preprocessing focuses on normalizing the input images in order for it to conform to the assumptions of the downstream parts of a given OMR system (e.g.,



(a) Original image.



(b) Binarized image: notation pixels as foreground.



(c) After staff removal.

Figure 2.1: The standard OMR pipeline from the original image through image processing including binarization, and staff removal. While staff removal is technically part of symbol recognition, as stafflines are symbols as well, it has until very recently been considered practical to recognize stafflines separately.

de-skewing, so that staffs are straight [Fujinaga, 1988]). The most important problem for OMR in this stage has been binarization [Rebelo et al., 2012]: selecting which pixels belong to the background, and which pixels are part of the notation. However, authors have also attempted to bypass binarization, especially before staffline detection [Rebelo and Cardoso, 2013, Calvo Zaragoza et al., 2016b]. Given the results of [Calvo Zaragoza et al., 2016b], we believe that with deep learning methods that work on raw pixel values instead of various morphological features, binarization has ceased to be a problem relevant to OMR. (Furthermore, the only dataset available at the beginning of the project was already binarized.)

Other preprocessing relates mostly to imperfections in the imaging process

(e.g., uneven lighting, deformations of the paper; with mobile phone cameras, limited depth-of-field may lead to out-of-focus segments of the image) and the quality of the underlying document (degradation, stains, especially bleedthrough) [Byrd and Simonsen, 2015]. Other than possibly some specific binarization techniques, or rather estimating the optimal settings of general binarization techniques, preprocessing is not specific to OMR.

### 2.1.2 Staff detection and removal

Staff detection and removal has seen a lot of activity, as it is a critical issue for OMR since its beginnings [Pruslin, 1966, Prerau, 1971, Fujinaga, 1988]. It is the only part of OMR where a competition has been organized [Fornés et al., 2011, Fornés et al., 2014], and an extensive dataset was created [Fornés et al., 2012]. Removing stafflines significantly simplifies the topology of the foreground regions, to the extend that connected components become a useful (if imperfect) heuristic for pruning the search space of possible segmentations [Fujinaga, 1996, Rebelo, 2012]. Furthermore, the vertical spacing of stafflines (staffspace height, measured in pixels) is the primary parameter that describes the scaling of the score: one can normalize scores by re-scaling to some fixed staffspace height (due to differences in staffline thickness relative to the height of a whole staff, using a sum of staffline and staffspace height is a more robust characteristic [Rebelo, 2012]).

Traditional staffline detection and removal methods exploit the fact that stafflines are by definition long and straight, or at least should be. The natural idea has been to detect them by searching for peaks in horizontal projections [Pruslin, 1966, Prerau, 1971], notably also by Fujinaga [1988]. A more general approach that also applies to grayscale images, not necessarily only to binarized inputs, was attempted by Cardoso et al. [2009] and Rebelo et al. [2013], Rebelo and Cardoso [2013], search for shortest "stable paths" through foreground areas from the left edge of the score to the right, also based on the assumption that the stafflines are the only extensive horizontal foreground objects. However, these results have been almost entirely superseded by convolutional networks Calvo Zaragoza et al. [2017c], Gallego and Calvo Zaragoza [2017], achieving robust results: both significantly outperforming previous results on the CVC-MUSCIMA dataset used for the competition [Fornés et al., 2012], and being applicable to different types of scores as well. Furthermore, yet more object detection methods using deep learning (such as those used in the project) have been found to not require staff *removal* at all [Pacha and Calvo Zaragoza, 2018, Hajič jr. et al., 2018a, Pacha et al., 2018b].

### 2.1.3 Object Detection

Object detection, whether with or without staffs removed, has been attempted mostly in two steps: a segmentation or localization step first, and a classification step next. Classification of musical symbols has produced near-perfect accuracy for both printed and handwritten musical symbols [Rebelo, 2012, Chanda et al., 2014, Wen et al., 2016], with baseline classification algorithms on raw pixels as features achieving close to 80 % accuracy [Calvo Zaragoza and Oncina, 2014]. However, it must be noted that different authors ignored various subsets of possible notation symbols, subject to data availability, and yet more importantly, they use disparate "alphabets" of music notation symbols. Some OMR researchers decompose notation into individual primitives (notehead, stem, flag) [Coüasnon and Camillerapp, 1994, Bainbridge and Bell, 1997, Bellini et al., 2001, Bainbridge and Bell, 2003, Fornés, 2005], while others retain the graphical "note" as a single visual object, and beamed groups are decomposed into the beam(s) and the remaining notehead+stem combinations of "quarter-like notes" [Rebelo et al., 2010, Rebelo, 2012, Pham et al., 2015]; in some literature that chooses this decomposition, beams are unfortunately not included at all [Calvo Zaragoza and Oncina, 2014, Chanda et al., 2014].

Segmentation of handwritten scores has until recently been yet more elusive. Most segmentation approaches such as projections [Fujinaga, 1988, 1996, Bellini et al., 2001] relied on topological constraints (such as: the notehead and stem touch) that do not necessarily hold even in printed music, much less in manuscripts. In response, a fuzzy approach to topological constraints has been proposed in [Rossant and Bloch, 2006], and morphological skeletons have been proposed instead [Roach and Tatem, 1988, Ng et al., 1999, Luth, 2002] as a basis for handwritten OMR. However, as with the other steps, general object detection methods based on deep learning (including those developed by the project) have recently brought previously unseen performance [Pacha et al., 2018b] that has since improved further.

### 2.1.4 Notation Assembly

The locations and classes of symbols on the page becomes the input to the **notation assembly** stage. Recall that music notation is a featural writing system: the essence of notation assembly lies in inferring the symbol *configurations* from the individual symbols and their locations.

However, it is not clear what the *output* of this stage is, or rather: the formulation of the output heavily depends on the assembly approach taken. This is because at this point in the recognition pipeline, one must start thinking about how to formally represent music notation. In strictly replayability-oriented applications, when using end-to-end learning, one may decide that no explicit representation is needed [Shi et al., 2017, van der Wel and Ullrich, 2017, Calvo

Zaragoza et al., 2017b, Calvo Zaragoza and Rizo, 2018]. However, in other cases, it is necessary to have a formal model of music *notation* in mind.

One such formalism are *context-free grammars*. This approach is rooted in the fact that music notation seems like it can be hierarchically decomposed, corresponding well to the notion of a non-terminal symbol. A page is split into systems, systems into staffs, staffs into measures, measures into notes, etc. Furthermore, there are strong visual syntactic rules for how to write valid music notation: e.g., every full notehead must have an associated stem; the stem is supposed to touch the notehead on the rightmost point (if it is pointing upwards), or leftmost (if pointing down); the half-rest is on top of the middle staffline, the whole rest is positioned "hanging" from below the 2nd staffline from the top; an inline sharp is at the height of the notehead, etc., that invite this line of thinking: one can easily imagine generation with non-terminals such as `quarter_note` $-->$ {`notehead-full, stem`} with additional attributes to make sure that, e.g., stems point in the right direction.

Using context-free grammars has been first attempted already in 1982 [Alfio Andronico and Alberto Ciampa, 1982] and several times since [Coüasnon and Camillerapp, 1994, Coüasnon and Rétif, 1995, Bainbridge and Bell, 2003, Szwoch, 2007], as it offers an elegant formalism with established inference algorithms. However, although it does to some extent simulate the human process of writing music ("I need to write a G4# quarter note" translates at the graphical level to "write a full notehead on 4th staffline, stem pointing upwards, sharp on the left of notehead"), the intuitively appealing top-down hierarchical decomposition of music notation into a tree structure is not necessarily an adequate representation of music notation itself: for instance, in the (relatively frequent) situation where two voices share a notehead, one either has to "invent" an overlapping notehead symbol so that the parse tree remains a tree, or let subtrees share leaves. This is a problem for parsers, as they rely on a pre-computed segmentation of the input. In response, graph grammars have been used [Fahmy and Blostein, 1993, Baumann, 1995, Reed and Parker, 1996, Fahmy and Blostein, 1998]. The core idea of graph rewriting is also being used by the Audiveris open-source OMR system [Bitteur, 2004].[8] However, the interest in such unified formalisms (and notation assembly in general) seems to have waned after 2000, in exchange for increased focus on staff removal and symbol classification.

The project uses a novel *notation graph* formalism developed previously by the PI that is both sufficiently strong to represent arbitrary music notation, and allows formulating assembly as a straightforward machine learning task; see section 3.1.

[8] A grpah is also used by Chen et al. [2015a]: a graph is built with edges directly connecting some notation primitives, but this was done for the purposes of preserving layout constraints when stretching and otherwise manipulating a score *without* fully recognizing it.

### 2.1.5 Constructing the output representation

Once the score is fully described and the musical semantics are inferred, what remains is to store the given score in the desired output format. The individual formats are each suitable for a different purpose: for instance, MIDI is most useful for interfacing different electronic audio devices, MEI[9] is great for editorial work, LilyPond[10] allows for excellent control of music engraving, MusicXML[11] offers greatest interoperability with music notation software. Many of these have associated software tools that enable rendering the encoded music as a standard musical score, although some – notably MIDI – do not allow for a lossless round-trip.[12]

For replayability-oriented applications, the most practical output format is MIDI, which can be straightforwardly created from the list of ⟨`pitch, onset, duration`⟩ triplets that the OMR system has inferred.

As the notation assembly step should resolve any remaining ambiguity, constructing the output representation should remain an engineering task (even though it may still be complex), not a part of the OMR process per se. However, this still depends on having an appropriate formalism for notation assembly output.

### 2.1.6 End-to-end OMR

With the advent of deep learning methods that require barely any feature engineering, a different approach than decomposing the problem into the standard pipeline can be taken: *end-to-end* recognition, where the intermediate stages of the process are not done explicitly and the corresponding intermediate results – especially the individual symbols and their locations – are never recorded. As the object detection subproblem is in principle hard in OMR, including issues with properly defining the set of symbols (see subsection 2.1.3), this approach is particularly appealing. It also widens the possibilities of using synthetic data generated on the fly during training. Recurrent networks offer the possibility of dealing with the long-range dependencies inherent in music notation, such as remembering which clef or key signature is valid for the particular location in the score.

Already before the advent of deep learning, the end-to-end approach has been elegantly applied using Hidden Markov Models by Pugin [2006a], for the recognition of monophonic mensural notation printed with movable type. For monophonic music, this approach was presented first by Shi et al. [2017] as

[9] `https://music-encoding.org`

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

[11] `http://www.musicxml.com/`

[12] That is: when converting a file from format A to B and then back from B to A, the result will not necessarily contain all the information of the original file in format A.

a side note for a recurent-convolutional model; an encoder-decoder model was used by van der Wel and Ullrich [2017], Calvo Zaragoza et al. [2017a] use a recurrent-convolutional model with Connectionist Temporal Classification loss. Unfortunately, no end-to-end models have so far been developed for polyphonic, much less pianoform music.

### 2.1.7 Interactive OMR

For replayability-oriented applications where the OMR output is supposed to be used as performance material for musicians, no errors are tolerated, and therefore OMR outputs will always be held "under suspicion" until reviewed and cleared by a qualified editor [Raphael and Wang, 2011]. Since the application requires human intervention anyway, there is little reason to limit the intervention to the endpoint of the recognition process, especially since low-level errors early in the recognition pipeline can have severe implications [Bellini et al., 2007], it would be useful to catch these errors as they happen, saving subsequent editing effort. This line of thought leads to *interactive* OMR systems, where the user is invited to intervene along the pipeline. Fujinaga [1996] proposed an adaptive system that learned from user feedback over time; the ideas have been implemented in the Gamera framework [Droettboom et al., 2002]. More recently, this framework has been adapted into the Rodan online infrastructure that allows for arbitrary interactive pipeline steps in the browser [Hankinson, 2014]. Outside of the Gamera/Rodan effort, Church and Cuthbert [2014] created an interface to let users correct misrecognized rhythmic patterns using correct measures elsewhere in the score. In contrast to these post-editing approaches, Calvo Zaragoza et al. [2016a] combine the musical score image with the signal from pen-based "tracing" of the symbols, merging the offline and online modalities of OMR. Chen and Duan [2016] incorporate human guidance directly into the recognition process, by letting the user control what elements of notation are allowed, in order to avoid false positives for rare situations that the editor can rule out for the given page; the resulting CERES tool allows quick re-recognition and incorporates visual feedback. What has *not* been attempted yet is Interactive OMR guided by audio input, even though playing the music in question seems to be the fastest and most natural way of providing user feedback: after all, musical instruments are exactly the interfaces intended for the interpretation of the musical score. Closest is the work on tracking audio in sheet music Dorfer et al. [2016, 2018c].

### 2.1.8 Online OMR

With the advent of touch-operated devices, especially in the realm tablets, there has been interest in *online* OMR that takes as its input signal the trajectory of a pen [Anstice et al., 1996, Miyao and Maruyama, 2004, Mitobe et al., 2004, Tsandilas, 2012, Calvo Zaragoza and Oncina, 2014, 2015, Calvo Zaragoza et al.,

2016a, Calvo Zaragoza and Jose Oncina, 2017, Sober Mira et al., 2017]. The advantage of this approach is that much more information is available to the OMR system: individual pen strokes are an important pre-segmentation heuristic, the order in which strokes are done will also be predictive of their meaning [Calvo Zaragoza and Jose Oncina, 2017]. This approach cannot on the other hand deal with the already accumulated body of written works. However, an elegant idea is to use online OMR to speed up data acquisition for offline OMR [Calvo Zaragoza et al., 2016a]: the user traces notation that has already been written on a touch interface, and the system thus has multiple signals available. This is much faster than tracing the notation elements individually, and it might make it feasible for untrained annotators to quickly create in-domain datasets for specializing OMR systems for individual collections. The MuRET tool by Rizo et al. [2018] implements this process.

## 2.2 Infrastructure of OMR

The individual steps of this pipeline have garnered the most attention in OMR literature, but three more important areas must be mentioned which underpin the overall state of the art: datasets, evaluation, and software.

Datasets have been scarce. There was no openly available dataset for object detection, for instance; much less for the full recognition pipeline. The only extensive dataset that has been available is the CVC-MUSCIMA staff removal and writer identification collection of 1000 scores (and eleven distortions, for a total of 12 000 images) by Fornés et al. [2012]. For symbol *classification* (not detectiononsted the follthe ionalon ture !), the HOMUS dataset [Calvo Zaragoza and Oncina, 2014] was the most extensive, with the advantage of providing inputs in both offline and online flavors, and also the only such dataset that was publicly available; even so, it contained only 32 different symbol classes (with the core alphabet of music notation, disregarding text, having more than 50 such classes).

During the last three years, scientific dataset have, however, become available. The first significant addition was the MUSCIMA++ dataset [Hajič jr. and Pecina, 2017c], which still remains the only dataset for full-pipeline recognition and for CWMN manuscripts and is one of the key prerequisites for the success of this project, but for symbol detection and partial semantics inference, there is the much more extensive – though printed and synthetic – DeepScores [Tuggener et al., 2018]. Third, the Capitan dataset of mensural notation has been made available [Pacha and Calvo Zaragoza, 2018] that supports symbol detection, although not semantics inference.

Evaluation remains a problem. While it is possible to evaluate individual subproblems of the OMR pipeline and clear up-to-date methodologies exist [Fornés et al., 2012, Rebelo, 2012, Calvo Zaragoza and Oncina, 2014, Pacha and Calvo Zaragoza, 2018, Hajič jr. et al., 2018a], evaluating an OMR system as a whole

is still problematic [Byrd and Simonsen, 2015, Hajič jr. et al., 2016]. The most extensive such effort was probably undertaken by Bellini et al. [2007], who directed annotators to manually label OMR mistakes according to a detailed list of possible errors; a different approach was undertaked by Szwoch [2008] and Hajič jr. et al. [2016], which seeks to develop automated metrics for directly comparing files in the MusicXML output format.

Finally, we mention software tools available for OMR research. The veritable Gamera system [MacMillan et al., 2001, 2002] has held reference implementations of various OMR methods, for instance staff removal up until the machine learning-based contributions of the last six years; it has been used to build also an OMR system for lute tabulatures [Dalitz and Karsten, 2005] and Byzantine chant notation [Dalitz et al., 2008]. Aside from Gamera, there is the Audiveris generic open-source system [Bitteur, 2004] and the Aruspix system for processing early music prints [Pugin, 2006b, Pugin et al., 2008]. In recent years, the OMR pipeline has been marshalled by the SIMSSA project [Fujinaga et al., 2014] under the Rodan system [Hankinson, 2014], which is openly available. Given the dominance of machine learning in computer vision, this includes interactive editors for creating ground truth: within the SIMSSA system, these are editors such as Pixel.js [Saleh et al., 2017] for creating pixelwise ground truth for binarization and staff deteciton, or Neume.js [Burlet et al., 2012] for manipulating recognition outputs of sqaure notation of neumes. Recently, the MuReT tool was also released that facillitates also pen-based data acquisition [Rizo et al., 2018].

The project author has also published the MUSCIMarker editor for ground truth acquisition [Hajič jr. and Dorfer, 2017], and throughout the run of the project, it has been extended with server-based recognition functionality.[13]

### 2.2.1 Commercial Software

Finally, we mention the available commercial software. The biggest players are PhotoScore[14] and SmartScore[15], each integrated into one of the major commercial notation editors (PhotoScore in Finale, SmartScore in Sibelius). Recently, the PlayScore software has emerged.[16] Given the sorry state of OMR evaluation and the "black box" nature of commercial software, it is not possible to measure their performance with more accuracy than anecdotal evidence.[17] This anecdotal evidence suggests that at least for high-quality printed scans, the performance of all commercial software has improved significantly during the last several years, to the extent that they can now actually be used in practice. However, at the time of writing only PhotoScore offers manuscript recognition functionality, and

[13] `https://github.com/omr-research/MUSCIMarker`
[14] `http://www.neuratron.com/photoscore.htm`
[15] `http://www.musitek.com/index.html`
[16] `www.playscore.co`
[17] See e.g. `https://omr-research.net/2019/11/04/assessing-playscore/`

it is rather bad. The Audiveris software is being gradually integrated into the MuseScore open-source notation editor.[18] For online OMR, the Neuratron NotateMe[19], StaffPad[20] and the MyScript back-end service[21] are available, and again the estimates of their usefulness are at best anecdotal and uncertain.

[18] https://www.musescore.com
[19] https://www.neuratron.com/notateme.html
[20] https://staffpad.com
[21] https://developer.myscript.com/music

# 3. Contributions

The project has contributed to the field of Optical Music Recognition the following:

- The `mung` software package for manipulating the MuNG representation of music notation (T2), musical semantics inference, and MIDI export.[1]
  [Hajič jr. and Pecina, 2017c,b, Hajič jr. and Dorfer, 2017]

- General music notation object detection with U-Nets (semantic segmentation), achieving (then-)state-of-the-art detection performance across multiple datasets.
  [Hajič jr. et al., 2018a, Pacha et al., 2018b]

- Notation assembly using pairwise MuNG edge/non-edge classification.
  [Hajič jr. and Pecina, 2017c, Hajič jr. et al., 2018a]

- **Full pipeline combining the previous contributions that produces MIDI from musical manuscripts from the MUSCIMA++ dataset, with a graphical interface with MUSCIMarker.**[2]
  [Hajič jr. and Pecina, 2017b, Hajič jr. and Dorfer, 2017]

- A tutorial on Optical Music Recognition presented at the ISMIR 2019 conference, available as a YouTube playlist.[3] [4]

- The project further contributed to the field by enabling the PI to participate in GREC 2017 OMR community discussion and subsequently serve as one of the General Chairs of the 1st International Workshop on Reading Music Systems (WoRMS),[5] a satellite event of the ISMIR 2018 conference.[6]
  [Calvo Zaragoza et al., 2018]

The key pre-requisite to building the pipeline in this project was the previous work of the PI on the Music Notation Graph formal representation of music notation, which was used for designing and creating the MUSCIMA++ dataset. Hajič jr. and Pecina [2017c]. This work provided then a (relatively) clear path how to design and run experiments leading to the recognition pipeline (M4). We therefore first introduce the MuNG representation, the MUSCIMA++ dataset, and then we can proceed to describe the corresponding recognition pipeline.

[1] `https://github.com/OMR-research/mung`
[2] `https://github.com/OMR-research/MUSCIMarker`
[3] `https://youtube.com/playlist?list=PL1jvwDVNwQke-04UxzlzY4FM33bo1CGS0`
[4] Note that ISMIR tutorials involve writing an extended abstract that undergoes review.
[5] `https://sites.google.com/view/worms2018`
[6] `https://https://ismir2018.ircam.fr/pages/events-at-a-glance.html`

Beside this main line of research, the project did make some contributions to multimodal sheet music-audio research (although not specifically OMR): together with CP JKU Linz, it co-produced the MSMD dataset[7] and the associated improvements in audio-based sheet music retrieval (and vice versa) [Dorfer et al., 2018a].[8]

## 3.1 Music Notation Graph

We have already prepared ground for the idea of the Music Notation Graph (MuNG) through the discussion of notation assembly methods and their limitations in section 2.1.4. A shared property of all the context-free grammar approaches [Alfio Andronico and Alberto Ciampa, 1982, Coüasnon and Camillerapp, 1994, Coüasnon and Rétif, 1995, Bainbridge and Bell, 2003, Szwoch, 2007] and graph rewriting systems [Fahmy and Blostein, 1993, Baumann, 1995, Reed and Parker, 1996, Fahmy and Blostein, 1998, Bitteur, 2004] in OMR so far is that they infer non-terminal "invisible" symbols that correspond to the hierarchy of abstract notation concepts (note, measure, voice...). This derives from the context-free grammar approach of building constituency trees. However, while this hierarchical approach is certainly appealing, especially given that this is how one usually learns to think about music and music notation, is it the best one can do in OMR? We of course propose that the answer is – *no*.

Rather than what could be termed a "constituency graph" of the previous approaches, in an analogy to the Prague school of Computational Linguistics, we apply the notion of a *dependency graph*. Instead of grouping music notation primitives under composite nodes, we link them to each other. We call this formalism simply a **Music Notation Graph** (abbreviated as MuNG). The vertices of this graph are music notation primitives (*not* notes!); oriented edges may link the vertices. The idea of assembling the music notation primitives into a notation graph is illustrated in Fig. 3.1.

The key property of this graph is that it is acquired for a given score, it holds all the information relevant to the music notation on the page in a fully disambiguated manner.

We can exploit the straightforward 1:1 relationship of "notehead" notation primitives to the abstract musical notes and use the rules of reading music (see Appendix A.1), which are deterministic, to unambiguously infer the musical semantics.[9]

Central to how MuNG is specified is the principle that each notehead-type node (full notehead, empty notehead, and all rests) has as its neighbors (im-

[7] `https://github.com/CPJKU/msmd`

[8] `https://github.com/CPJKU/audio_sheet_retrieval`

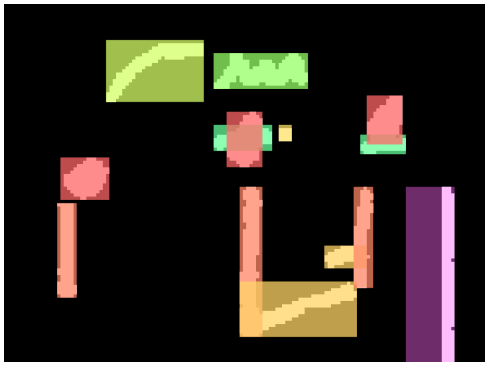
[9] This functionality is available in the `mung` library as the `mung.inference` module: `https://github.com/omr-research/mung`

mediate or close) all the notation primitives relevant to the decoding of the corresponding note. Recall that noteheads are the interface between music notation and the encoded notes: there is one note per notehead.[10] The musical semantics for each note are fully encoded through *configurations* of symbols associated with each notehead: the stafflines and ledger lines, clef, key signature and inline accidentals encode pitch; the notehead type, stem, flags or beams, augmentation dots and tuples encode its duration. Each of these elements can be simply captured by linking the notehead to the given primitive. The precedence relationships that are necessary to compute the onsets of notes can be captured as precedence edges in the notation graph that link noteheads which should be interpreted as consecutive notes.

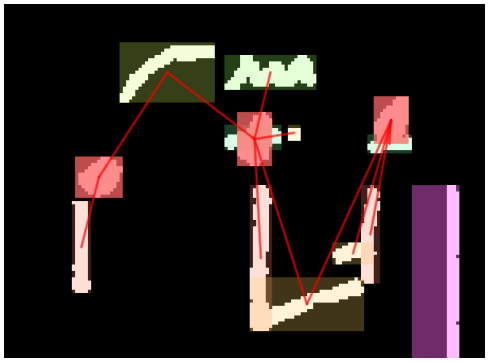
A significant advantage of this approach is that the separation between music *notation* as a visual language and musical *notes* and their semantics as abstract objects is retained: the notation graph is merely a description of the music notation on the page. The entire process of inferring semantics from the MuNG output of our notation assembly stage happens independently from the underlying image – and at the same time, all the information available in the score is fully disambiguated even *before* one starts thinking about the musical semantics. This separation makes it possible to deal with the image separately from the music encoded therein, and we conjecture that maintaining this principle is what allows formulating the OMR pipeline in terms of straightforward machine learning tasks.

The *disadvantage* of the dependency graph approach is that there are no tractable algorithms that we know of for generic graph inference from an image. However: it seems that these may not be required. First, for object detection, state-of-the-art generic models are capable of leveraging the neighborhood of an object to disambiguate it (such as the staccato dot, which is written below or above its corresponding notehead, vs. the augmentation dot, which positioned is to the right of a notehead or rest) *without* having explicit access to syntactic information (as indicated by Pacha et al. [2018b], Hajič jr. et al. [2018a] and especially by Pacha et al. [2018a]). With respect to notation assembly, we can make a strong independence assumption – that given the vertices of the graph (the music notation primitives), the edges of the graph are independent. This allows formulating the notation assembly as a binary classification problem over vertex pairs. On an average page of some 500–800 symbols, this would still amount to 250 000–640 000 decisions; however, in practice there are reasonable assumptions (such as the maximum distance between objects that may be related, and constraints on linked symbol classes) that help prune the space of

[10]The sole exception being notehead sharing across multiple voices; however, this is detectable from the presence of multiple stems. In case whole notes are shared, this is typeset as two consecutive empty noteheads significantly closer to each other than if they were to be played consecutively.



(a) Notation symbols: noteheads, stems, beams, ledger lines, a duration dot, slur, and ornament sign; part of a barline on the lower right. Vertices of the notation graph.



(b) Notation graph, highlighting noteheads as "roots" of subtrees. Noteheads share the beam and slur symbols.

Figure 3.1: Visualizing the detected symbols and the assembled notation graph on top of staff removal output. Colors of symbol bounding boxes encode symbol classes (noteheads in red, stems in orange, ledger lines in green, etc.). Using the edges of the notation graph in (b), the pitch and duration of the notes encoded by the noteheads (highlighted) can be unambiguously inferred (stafflines removed for clarity, although for encoding pitch, we would need to establish the relationship of the noteheads to stafflines). Assuming the music is monophonic, onset can be inferred from the ordering of the noteheads and the notes' durations.

decisions to an assymptotically linear instead of quadratic number of classifier runs. This already allows bringing the full power of current machine learning methods to bear on the assembly problem: already decision trees with simple features (bounding box relative distance and symbol class labels) achieve useful results, as described below in section 3.3.3. Furthermore, ongoing experiments with factoring the MuNG inference process from detected objects into independent decisions about individual edges seems to also provide satisfactory results.

While the idea of MuNG and the notehead-centric definition is hopefully clear and clearly motivated, there still remains a plethora of details to take care of: dealing with key signatures, time signatures, measure separators, etc. Further principles of MuNG definition are described in the publication [Hajič jr. and Pecina, 2017c], and in full detail the definition is available online in the form of annotation guidelines for the MUSCIMA++ dataset,[11].

## 3.2 MUSCIMA++

While the MuNG formalism provides a clear definition of the ground truth, the MUSCIMA++ dataset proivides human ground truth annotations for evaluation

[11] `https://muscimarker.readthedocs.io/en/latest/instructions.html`

and supervised learning of the MuNG-based OMR pipeline. The dataset is a subset of 140 of the 1000 pages of CVC-MUSCIMA [Fornés et al., 2012], consisting of a total of more than 91 254 manually annotated symbols (MuNG vertices) and 82 247 manually annotated relationships (MuNG edges), representing 23 349 notes [Hajič jr. and Pecina, 2017c]. (Further objects in the dataset are staffline and staffspace objects – acquired automatically from the CVC-MUSCIMA staff detection ground truth – and precedence relationships, inferred automatically and checked by hand.)

The CVC-MUSCIMA dataset contains a handwritten copy of a set of 20 pages of music done by 50 writers for the total 1000 pages; since there were 7 annotators available to create MUSCIMA++, 7 copies of each page were annotated, but selected so that all the 50 writers are represented in the resulting 140 pages as equally as possible (2 or 3 pages per writer). This ensures the high variability of handwriting in CVC-MUSCIMA is retained.

The 20 pages of CVC-MUSCIMA also contain notation of all levels of complexity, from monophonic to pianoform notation, including rare, yet important situations, such as cross-staff beaming, time signature, key signature and clef changes in the middle of a staff, complex beamed groups, non-standard tuples, and even an instance of notehead sharing between voices.

As CVC-MUSCIMA contains binarized images with staffs removed, the same kind of images remains as inputs in the project's recognition pipeline.

An example of a complex score and its MuNG representation, as visualized by the MUSCIMarker editor, is shown in Fig. 3.2.

For all experiments, a test set of 20 pages is used so that each of the 20 CVC-MUSCIMA pages is represented once and the test set *writers* do not appear in the training set (so at test time the model deals primarily with unseen handwriting styles).

With the MUSCIMA++ dataset of binary images manually annotated with MuNG ground truth in hand, we may now proceed to build the recognition pipeline itself.

## 3.3 The Recognition Pipeline

The OMR pipeline focuses on the later stages of the OMR pipeline: object detection, and notation assembly and semantics inference. As stated in the introductory chapter, we focus on a difficult setting in terms of processing manuscripts of arbitrary notation complexity, rather than on difficulties regarding image quality (which are of course in practice equally important, but not as inherent to the domain of music notation). The input images for our pipeline have already been binarized, and stafflines have been detected (and, if need be, removed). This is no more an entirely unreasonable expectation: convolutional networks have been shown to perform "layout analysis" (essentially, joint staffline detec-

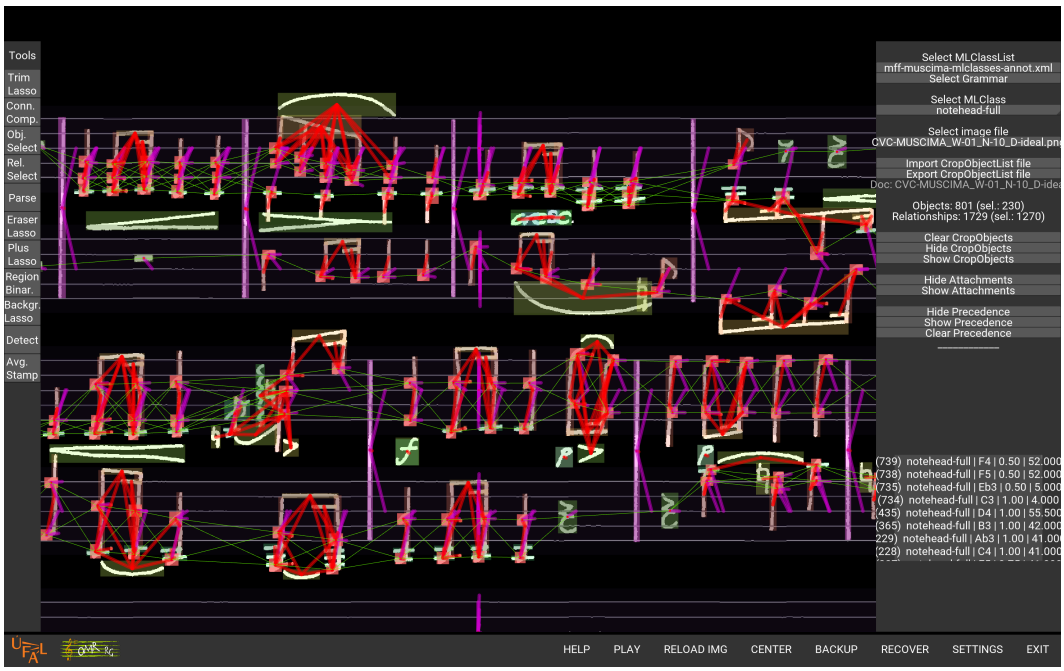



Figure 3.2: The interface of the MUSCIMarker tool. The notation graph in this example is taken from CVC-MUSCIMA image of page 10 by writer 01. Highlighted in red are manually added syntactic edges of the notation graph; in purple are automatically inferred edges pertaining to staff objects (stafflines, staffspaces and staffs). Precedence edges are weakly visible in green.

tion and binarization: semantic segmentation into background, stafflines, and notation symbols) very well [Calvo Zaragoza et al., 2017a,c,d, Gallego and Calvo Zaragoza, 2017].

### 3.3.1 Object Detection

We start the work on the recognition pipeline by testing state-of-the-art object detection techniques.

Detecting music notation primitives is a difficult class of the general object detection problem since music notation does not conform well to the usual assumptions of object detection models. The score images are very cluttered (on average, they contain some 650 objects). While the highest-priority symbols, noteheads, are relatively easy to detect, because their appearance is very distinct (by design: they should be the first thing that attract the eye of a musician!), other symbols present tricky detection issues, especially in handwriting. Some fixed-size symbols such as clefs are visually quite complex and very variable, and usually overlap with stafflines. Furthermore, clefs are by far not as frequent as noteheads, even though they are critical for decoding the semantics of all subsequent notes (as they define how stafflines are interpreted with respect to pitch). While these symbols at least have a (relatively) fixed size, there are others that

have in principle a relatively simple shape (straight thick line), but their size is variable (stems, especially in music with wide chords), or even size and orientation (beams, which are probably the most variable symbol class, and slurs, which can theoretically be extremely complicated shape, although in practice they rarely have an inflection point). Therefore, we prefer models that make as few assumptions as possible about the objects they are attempting to detect.

The least-assumptions state-of-the-art model that can be used for object detection is the **fully convolutional networks**. These networks first perform semantic segmentation (assigning a label to each pixel) and require a subsequent detection stage (such as peak picking, or thresholding and connected components). The network itself is completely free of assumptions about symbol shapes and sizes; the only important hyperpamater is the receptive field of the output pixels, defined implicitly by the width and amount of convolutional filters through which information passes to an output pixel. This contrasts with models such as Region Proposal Networks (of the Faster-RCNN family) that require pre-defining a set of *anchor boxes* and their spacing.

Specifically, we used the U-Net model [Ronneberger et al., 2015]. This model has an "hourglass" architecture reminiscent of autoencoders, but it is standard feedforward network; the output layer provides a value for each pixel (in this case, the probability of the given pixel belonging to the given symbol class). Given that the stages of the hourglass have the same size, residual connections are added between the corresponding stages. The network architecture is shown in Fig. 3.3.

Seeing as noteheads are the most important object to detect reliably, we began our efforts there. Without any post-filtering step, merely with thresholding at 0.5 and non-maxima suppression as the detection step on top of the probability map output by the model, the U-Net achieved on noteheads a recall of 0.97 and precision 0.99.

Given this convincing advantage of the U-Net on noteheads, which are the most important object to detect, we chose to follow up on the fully convolutional model and build general object detection based on the U-Net architecture. Each symbol class will be detected with its own U-Net trained specifically on that class (with the exception of networks with multi-channel outputs for certain rare symbols, see below).

The major drawbacks of U-Nets for musical symbol detection is that these models only perform semantic segmentation, not object detection per se: a detector must be added on top of the output symbol probability map. Two simplest options are thresholding and connected component search, and thresholding and non-maxima suppression. Since non-maxima suppression is prone to leave false positives in long symbols sucha as stems or slurs, and in larger complex symbols such as clefs, we choose connected component search, with thresholding at

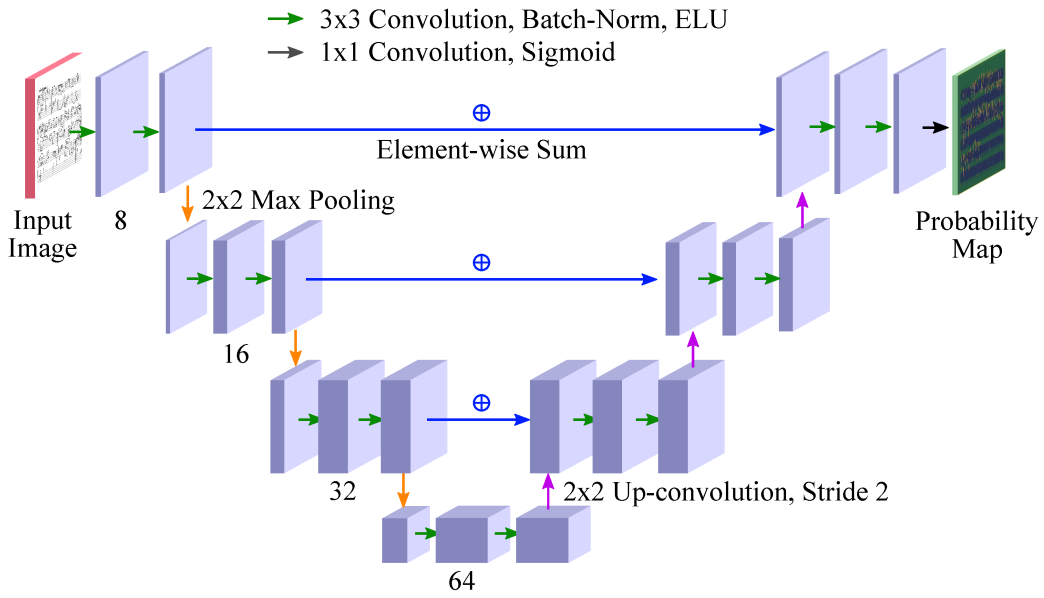



Figure 3.3: The U-Net architecture. Computation flows left-to-right; the "hourglass" shape is unrolled downwards with each 2x2 Max-Pooling layer (orange arrows); in the other direction are 2x2 up-convolution layers with a stride of 2. Blue arrows indicate the residual connections (implemented simply as elementwise sums) bewteen blocks of corresponding sizes. (Figure taken from [Pacha et al., 2018b].)

0.5.[12] Note that using a connected component detector implies that the model may merge objects from the same class that legitimately touch (such as some handwritten noteheads in chords) into a single symbol.

In a comparison to other general object detection models, Faster R-CNN [Shaoqing Ren et al., 2015] and RetinaNet [Lin et al., 2017], the advantages of U-Nets do result in better performance [Pacha et al., 2018b], as illustrated in Table 3.1.[13]

The advantage disappears for the Capitan dataset [Pacha and Calvo Zaragoza, 2018] of mensural notation, which uses a different symbol alphabet: instead of decomposing the graphical notes into primitives, its symbol classes correspond to the entire note: `longa`, `breve`, `semibreve`, `minima`, `semiminima`, etc. Furthermore, Spanish white mensural notation (of which the Capitan dataset comprises) does not allow many of the situations that make CWMN recognition difficult, such as beamed groups and polyphony on one staff. Fig. 3.4 illustrates

[12] Changing the threshold did not lead to improvements.

[13] The results are evaluated using Mean Average Precision (mAP) and Weighted Mean Average Precision (w-mAP), according to the object detection practices for the COCO dataset [Chen et al., 2015b]. The "mean" is taken over average precisions with true positives considered using different intersection-over-union thresholds: the most permissive is 0.5, and, using increments of 0.05, the cutoff for considering a detected object a true positive, the minimum intersection-over-union it must share with a ground truth object of the given class increases up to 0.95. In the weighted variant, the object classes are weighed by their support.

| | mAP / w-mAP (%) | | |
|---|---|---|---|
| | **DeepScores** | **MUSCIMA++** | **Capitan** |
| **Faster R-CNN** | 19.6 / 14.4 | 3.9 / 7.9 | 15.2 / 23.2 |
| **RetinaNet** | 9.8 / 1.9 | 7.7 / 4.9 | 14.5 / 34.9 |
| **U-Net** | 24.8 / 17.4 | 16.6 / 23.3 | 17.4 / 26.0 |

Table 3.1: Results in terms of mAP (%) and w-mAP (%) with respect to the dataset and object detector model following the COCO evaluation protocol. (Table reproduced from [Pacha et al., 2018b].)

the advantage of U-Nets on MUSCIMA++.[14]

We employ further two tricks for improving detection performance.

First, we deal with class imbalances. As the training process samples a 256x512-pixel window for each data point [Ronneberger et al., 2015, Dorfer et al., 2017, Hajič jr. et al., 2018a], for relatively rare symbols, often the window contains no pixel of the given target class, and useful signal is drowned out by noise in the initial stages of learning. In order to avoid this effect, if the sampled window does not contain any pixels from the target class, we uniformly sample a different one up to five times. (If after five samples we still found no foreground target pixel, we use the last sampled empty window.) A second trick for training the detection of rare symbols is letting them share features: with the U-Net model, this only requires adding an output channel to the training data and the model.

A different trick is used to deal with symbols that exhibit complex shapes – again, especially clefs. Instead of training against their true masks, we train against the *convex hulls* of these masks. Since we are at this point mainly trying to detect the presence of the object in a particular location, this approximation does not lower the upper bound on detection performance. At the same time, it simplifies the job of the up-convolution part of the network, as it does not have to "fill in" blanks inside the complex symbols; it decreases the chances that a single detected symbol will form two connected components after thresholding due to false negative pixels in its thin parts, and most importantly, it saves us from dealing with symbols that are legitimately composed from several connected components (such as f-clefs and c-clefs) or written erroneously as disconnected. The convex hull trick is illustrated in Fig. 3.5.

### 3.3.2 Detection in Full Pipeline

Since the objective of the pipeline is replayability (technically, producing a MIDI file), in the full-pipeline experiments we restricted detection to only those classes that are relevant for extracting the musical semantics. (As there is a separate

[14] In [Pacha et al., 2018b], both the quantitative and qualitative results are further discussed.

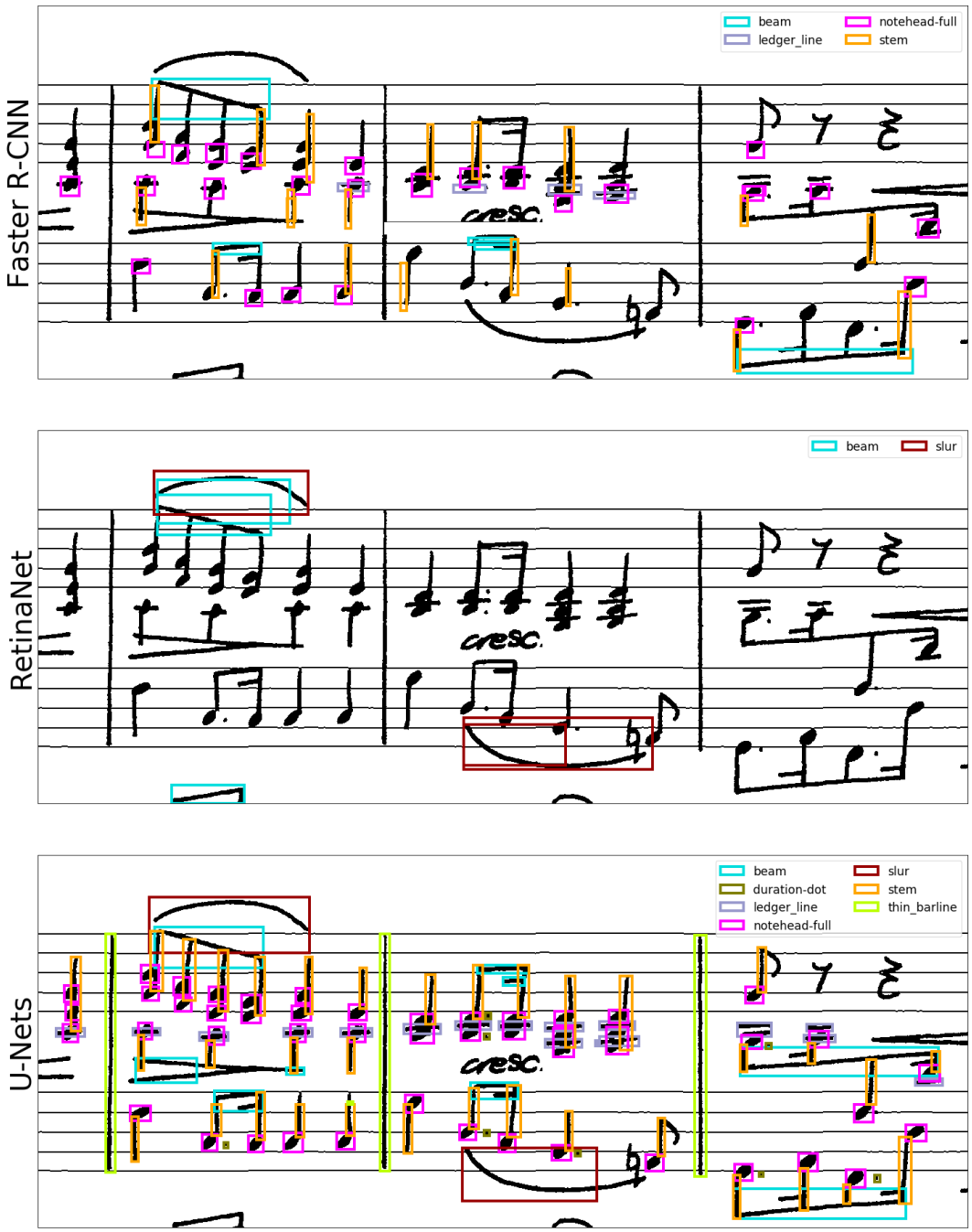



Figure 3.4: Example results in a complex notational situation. (Selected classes. Figure taken from [Pacha et al., 2018b].)

model trained for each class, however, this just means reducing the number of models).

The detection performance, reported simply as the detection f-score,[15] for in-

[15]The F-score is the harmonic mean of recall and precision. This way, it balances the need to avoid both false positives and false negatives, and penalizes systems that err too much to one side: a system with recall 1.0 and precision 0.1 will have an f-score of 0.18, while a system with recall 0.6 and precision 0.5 will have an f-score of 0.55.

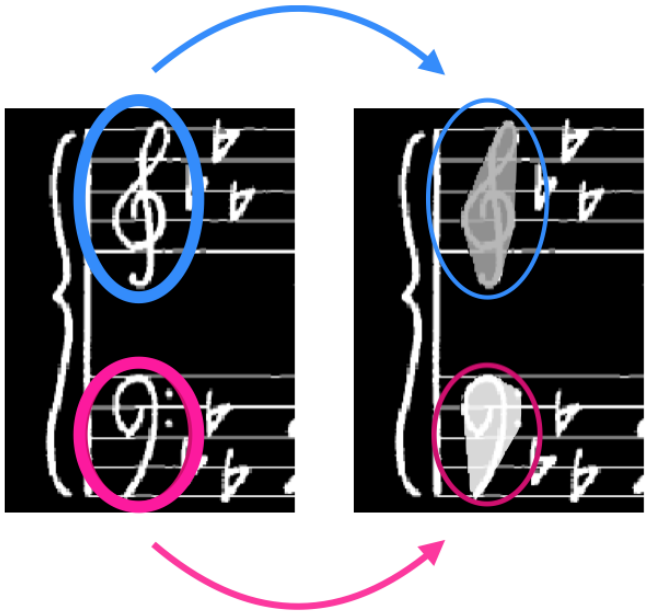


Figure 3.5: Modifying the targets for semantic segmentation training to convex hulls of the objects that we ultimately want to detect. Top: g-clef, bottom: f-clef.

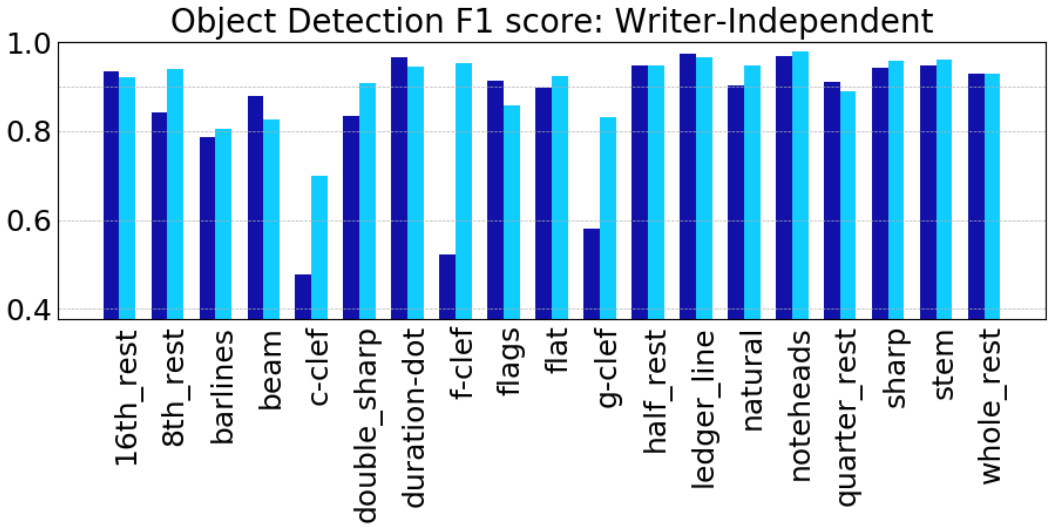



Figure 3.6: Detection f-score for symbols required for replayability: with "vanilla" U-Nets, and with tricks. Note the improvements especially for clefs: these are critical for pitch inference.

dividual replayability-oriented object classes, is reported in Fig. 3.6. The greatest improvements using the training tricks came for clefs, which were the most problematic symbols for the "vanilla" training. They were also the group of symbols that benefited most from being grouped into a multichannel model.

### 3.3.3 Notation Assembly and Semantics Inference

Next, we build the notation assembly stage and infer musical semantics.

Under the MuNG formalism, notation assembly is the task of inferring the graph edges given the (detected) nodes. The simplest thing one can do is to decompose this task into decision about individual edges (or non-edges): frame the task as binary classification over node pairs, and assume the edges (and non-edges) are independent.

The independence assumption is of course an over-simplification: e.g., a notehead may be connected to a beam either above, or below its position, but not both. However, breaking down the problem into independent decisions is a reasonable start.

What simplifies the situation further is that edges leading from objects to stafflines, staffspaces and their containing staffs can be even in manuscripts inferred near perfectly[16] using appropriate heuristics based simply on how the object overlaps with the stafflines and staffspaces. The only "magic number" that must be selected concerns the situation where a notehead has one part above a staffline and another part below the same staffline: if there is a large imbalance between these two parts, expressed as the ratio of the offset of the top of the notehead to the top of the staffline vs. the offset of the bottom of the notehead to the bottom of the staffline, it should be considered connected to the corresponding staffspace rather than to the staffline it overlaps. If this ratio is smaller than 0.2 or greater than 0.8, the notehead should be assigned to the staffspace; if the imbalance is not as large, it should be considered to lie on the staffline it overlaps.[17]

Given that the average image in MUSCIMA++ contains about 650 notation objects, if we were to consider the quadratic amount of $\langle$`from, to`$\rangle$ pairs in an image (recall that the MuNG has *oriented* edges, so the distinction between `from` and `to` is necessary), we would have to make over 400 000 decisions for each image. Fortunately, objects that are far from each other are quite certain to not be related. In MUSCIMA++, we found that if we only consider $\langle$`from, to`$\rangle$ pairs within a distance of $10*$ *staffspace_height* + *staffline_height*, we only discard 52 out of the 82247 related symbol pairs (excluding the relationships to stafflines and staffspaces) [Hajič jr. and Pecina, 2017c]. This reduces the number of $\langle$`from, to`$\rangle$ candidate pairs to a linear number, albeit with a significant multiplicative constant (about 8 – 15, based on the density of the handwriting).

For a $\langle$`from, to`$\rangle$ candidate object pair: the simplest features we can use are their classes $(class_f, class_t)$, and relative position of their bounding boxes: given that the bounding box of the `from` object is $B_f = (top_f, left_f, bottom_f, right_f)$,

[16] Only two noteheads in MUSCIMA++ had their relationship to the staff objects inferred incorrectly with the implementation in `mung.inference`.

[17] The fact that the relationships of notation objects to staffs can be inferred so deterministically is one of the few surprisingly *easy* things in OMR.

and the bounding box of the `to` object is $B_t = (top_t, left_t, bottom_t, right_t)$, the offset features are $(top_t - top_f, left_t - left_f, bottom_t - bottom_f, right_t - right_f)$. If an edge leads from the `from` object to the `to` object, the target is 1, otherwise it is 0. As positive examples, we take all related objects in the training data, as negative examples, we take simply all pairs that are within the threshold distance, but are not related.

We then train a decision tree [Leo Breiman et al., 1984]. Given a sufficient maximum depth for the tree, the model picks up by itself on the constraints imposed by symbol classes (for instance, there can never be an edge leading from a stem to a notehead, only in the opposite direction, accidentals are never associated with rests, etc.). Already this simple model achieves an f-score of 0.92 on edges on the MUSCIMA++ test set [Hajič jr. and Pecina, 2017c, Hajič jr. et al., 2018a] (as there are many more non-edges than edges, reporting overall accuracy would be overly optimistic, and we care only about the positive class anyway).

Some obvious assembly errors are caused by the simplified pairwise model that does not take any other objects than the ⟨`from, to`⟩ pair into account when making a decision. Especially (1) connecting a notehead to ledger lines both above and below the given notehead, and (2) connecting a notehead to beams both above it and below, unless it also has two related stems. However, these two can be relatively easily corrected. While these postprocessing heuristics based on "hard" constraints of music notation syntax did give quick improvement in these two specific cases, attempting other such patches did not improve overall results anymore.

We note that the MuNG formalism has allowed us to reach respectable assembly performance on handwritten music notation with gross oversimplifications of the rules of music notation, thanks to making straightforward machine learning techniques applicable.

A separate chapter are *precedence* edges. This aspect of the pipeline is somewhat underdeveloped: we simply order simultaneities linked to a staff left to right, and consider noteheads to belong to a simultaneity whenever they have an edge to a shared stem (this is the MuNG-based definition of what a chord is in music notation). This is probably the greatest limitation of our recognition pipeline.

### 3.3.4 Full Pipeline Results

All the results above matter little by themselves; what makes a (replayability-based) OMR system interesting is its ability to infer the musical semantics. A key advantage of MuNG is that once notation assembly is done, which happens without ever straying from the graphical layer of music notation into the layer of the semantics, one can infer the musical semantics unambiguously, using the

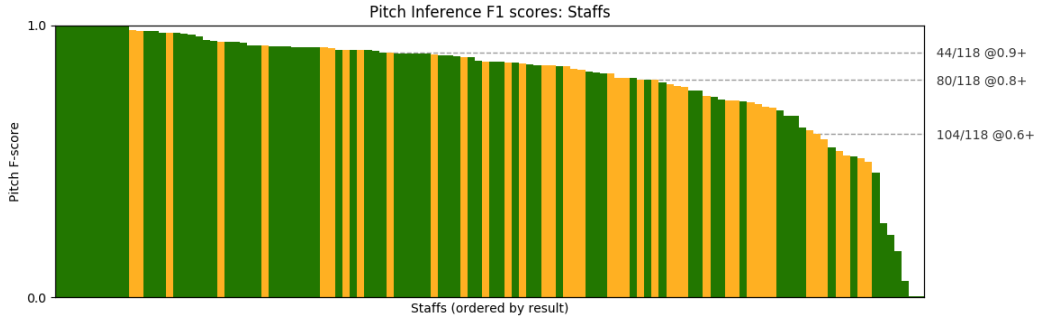



Figure 3.7: The pitch recognition f-score for the MUSCIMA++ writer-independent test set, broken down per individual staff. Monophonic staffs in green, homophonic and polyphonic in yellow.

rules for reading music.[18] The inferred notes are then encoded as a MIDI file.

The detection, assembly and semantics inference steps can also be run interactively from MUSCIMarker (with detection running over an socket connection, so that the detection model can be run remotely as a service in case a given machine does not have the necessary computing power).

We evaluate the full pipeline first intrinsically, based on its ability to correctly retrieve the musical semantics. In order to do this, we first have to align the output MIDI with the MIDI corresponding to the ground truth pipeline. (Since not all notes are detected properly, it is not straightforward to directly compare the detected notes with the ground truth notes: especially duration errors propagate by influencing the onsets of all subsequent notes.) We use Dynamic Time Warping (DTW) on sequences of simultaneities, using the ratio of pitches shared as the inverse cost function. Within each simultaneity, the notes are aligned from lowest to highest using a second round of DTW. The reason for using DTW to find the alignment between the recognition result and the ground truth is that it naturally finds an optimal *monotonous* alignment, that is, an alignment that does not violate the precedence relationships in neither the ground truth, nor the OMR output.

Given this alignment, we can compute how well the semantics were recovered. Unfortunately, for durations, the results were less than convincing (an f-score of less than 0.6). **For pitch, the overall f-score was 0.81.** The breakdown of pitch recognition f-score by individual staffs in the MUSCIMA++ test set is given in Fig. 3.7. Since duration errors propagate into onset errors, f-score for onsets cannot really be computed; however, it is implicitly encoded in the inferred precedence edges. These are always correct in monophonic and homophonic notation to the extent to which the symbols corresponding to notes or rests are detected. While we would like to compare our results, there is nothing to compare to: the project brought the first manuscript recognition results at the level of musical semantics.

[18] See Appendix A.1

| | MAP@1 | MAP@10 | MAP@49 |
|---|---|---|---|
| Page queries, OMR2OMR | 1.0 | 1.0 | 0.998 |
| Page queries, cross-modal | 1.0 | 1.0 | 0.998 |
| Snippet queries, OMR2OMR | 0.928 | 0.834 | 0.763 |
| Snippet queries, cross-modal | 0.606 | 0.610 | 0.577 |

Table 3.2: Results for page retrieval using page queries and snippet queries under two modalities: using OMR for creating the database and the query (OMR2OMR) or just for the database (cross-modal) and query with ground-truth MIDI. (Table reproduced from [Hajič jr. et al., 2018b].)

### 3.3.5 Applications: Retrieval and Digital Musicology

What can we do with a manuscript recognition system with this performance?

In [Hajič jr. et al., 2018b] we show that it should be possible to use this system to retrieve musical manuscripts copies. Since transfer learning for object detection is an unresolved issue, we are limited to the CVC-MUSCIMA dataset (of which MUSCIMA++ is a subset). At least for demonstration purposes, we select a subset of 7 of the 20 pages that it as confusing as possible (the music is as similar as possible to each other), in order to not artificially inflate the scores, and run recognition for each of these across all 50 writers, for a total of 350 pages. This is a toy dataset (although for manuscripts, we unfortunately cannot do better right now); any OMR system worth its salt should be able to retrieve duplicate pages perfectly. We use the DTW alignment cost as the similarity function of the retrieval system.

In Table 3.2, we report results for retrieving OMR outputs using queries constructed from OMR outputs, and also *cross-modal* results: querying the database of MIDI files obtained through OMR using ground truth MIDI files and snippets. Aside from page queries, we also attempt to retrieve pages using only music from a single staff, where results are less than perfect (but the task is significantly more difficult); the results there are worse, especially in the cross-modal setting that is sensitive to design limitations of the OMR system (that may "cancel out" when using OMR outputs both as the query and as the database).

The implications of the retrieval experiments are: despite the many limitations of our OMR pipeline, already it is a method for extracting the musical semantics from handwritten CWMN of arbitrary complexity that has potential real-world applications (identifying copies of manuscripts across archives) – to the extent to which training data will be available.

In summary, **the project built and published a functioning** (to the extent described above) **OMR system for handwritten music notation of arbitrary complexity,** and demonstrated the soundness of the MuNG representation for such full-pipeline OMR.

## 3.4 Auxilliary contributions

Outside of the main line of research on the project, some work was done on audio-sheet music multimodal retrieval and contributions were made to the functioning of the OMR scientific community.

### 3.4.1 Audio – sheet music cross-modal retrieval

The project has had some outputs in the sub-field of multimodal processing of audio and sheet music images, only not attempting OMR, but rather something different: retrieving corresponding segments of music across these two modalities *without* a need for a shared representation of the musical semantics, instead learning a joint latent space directly. This was done .[19]

The retrieval system in [Dorfer et al., 2018a] was trained using the MSMD open dataset.[20]. Both the sheet music and the audio is replicably synthesized from the digital scores of the Mutopia project[21], exploiting the advantages of the LilyPond format for music notation and the associated software stack.[22] The dataset contains 479 solo piano pieces of mostly classical music by 53 composers, for a total of 1,129 pages of music. The scores are available both as sheet images and as MIDI. What makes the dataset unique is that the modalities are automatically aligned at a fine-grained level: each individual notehead in the score images is linked to the corresponding MIDI event(s). The multimodal models then learn from snippet pairs centered around the aligned notehead/note event pairs. There is a total of 344,742 such aligned pairs.

The projection into a joint multimodal space is done using neural networks that are trained with a Canonical Correlation Analysis projection layer and ranking loss [Dorfer et al., 2018]. The learning setup is shown in Figure 3.8. Retrieval of pieces is done by mapping both the audio and sheet music snippets into this joint space, searching for nearest neighbors using Euclidean distance, and combining the snippet-wise results across all query snippets from a given query piece simply by voting (see Figure 3.9).

Performance is measured across the three different train/test splits of MSMD (one discards everything except the 173 pieces by J. S. Bach, keeping 50 of them as the test set; one that keeps the 173 Bach pieces as the test set, and one that selected random 100 compositions as the test set). Besides evaluation on the synthetic data, the system was also evaluated on retrieving real scores and audio

[19]Note that the learning setup and the retrieval method was the work of the first author of the project paper [Dorfer et al., 2018a] that the PI collaborated on. The project PI contributed the infrastructure for the experiments in the paper: the MSMD dataset, some insights into evaluation, and suggested using attention for the paper [Dorfer et al., 2018b].

[20]`https://github.com/CPJKU/msmd`

[21]`http://www.mutopiaproject.org/`

[22]`http://lilypond.org/`

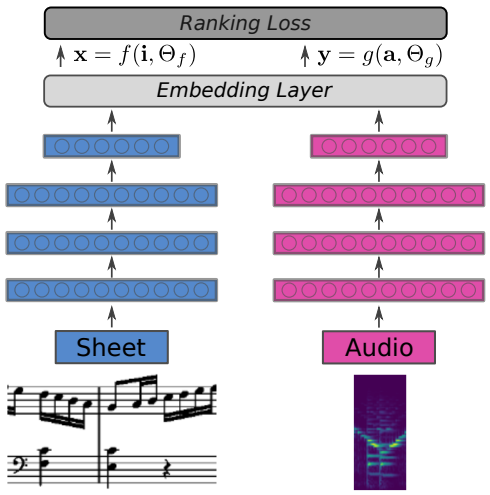



Figure 3.8: Audio-sheet music cross-modal learning setup with a differentiable Canonical Correlation Analysis layer that maximizes correlation between the corresponding representations of input snippets of sheet music and audio excerpts. The ranking loss then ensures that representations in the joint space differentiate between input snippets that do *not* corresnpond to each other. (The figure is reproduced from [Dorfer et al., 2018a].)

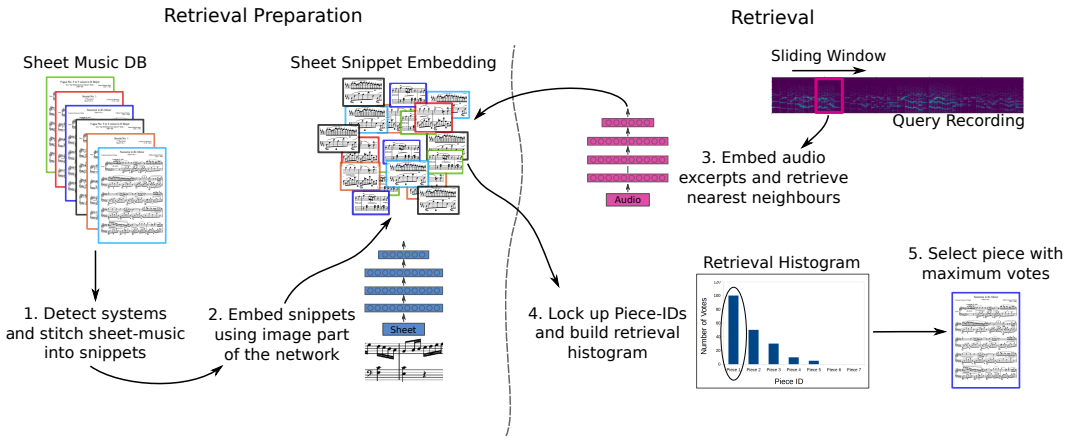



Figure 3.9: How pieces in the complementary modality are retrieved using the multimodal joint space. (The figure is reproduced from [Dorfer et al., 2018a].)

recordings, on a dataset of 193 compositions with various recordings by famous pianists and Henle editions of the scores.

We summarize the main results, which show that the multimodal learning created a useful retrieval tool that generalizes from synthetic data to real scores and, to some extent, real performances. (For details, see [Dorfer et al., 2018a].) When searching for a score based on an audio input, Recall@1 is 0.82 (meaning correct piece is retrieved rougly 82 % of the time as the top retrieval result) and Recall@5 is 0.95 (which signifies that 95 % of the time the correct piece is retrieved within the top 5). When using real scores, the system obtained similarly satisfactory performance with synthesized audio (in both directions), but not so

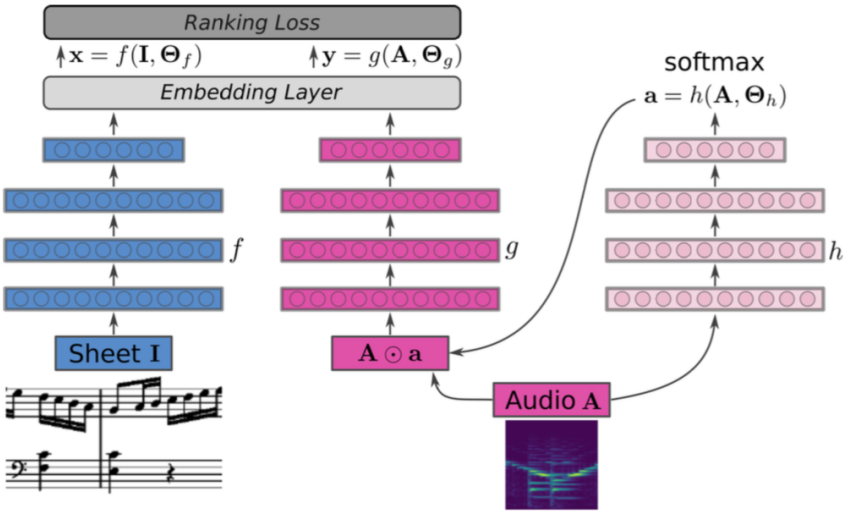



Figure 3.10: Adding a soft attention mechanism over the audio input bewteen the spectrogram and the first layer of the previously established multimodal architecture should allow the model to become more invariant to changes in tempo, which affect the "density" of events in the input spectrogram. (The figure is reproduced from [Dorfer et al., 2018b].)

when using real scores to search for recordings of real performances (Recall@1 at 0.46, Recall@5 0.70), and even worse when searching for real scores using the real performances (Recall@1 0.29, Recall@5 0.58, with almost sixty ).

### Tempo-invariant audio representation with attention

The substantially worse retrieval results on real performance audio, we suspect, has to do with the fact that time tends to be very flexible in real musical performance.

A limitation of the retrieval system is that the field of view into both modalities had a fixed size. This is a pronounced problem for the audio modality because of different recordings having different tempi. When the audio of the same sheet excerpt played at a different tempo is segmented into spectrogram excerpts with a fixed number of frames (and a fixed framerate), these will contain different amounts of musical content, relative to what the model has seen during training. At the same time, sheet music is *written* with the same notehead typesetting density both for slow and fast pieces. We would like our retrieval system to be, ideally, *tempo-invariant*. To use this, we attempted to use a soft *attention mechanism* [Dorfer et al., 2018b]. This mechanism is applied directly to the columns of the spectrogram before it is sent into the audio network (see Fig. 3.10).

The attention mechanism does indeed learn to react to tempo changes as one

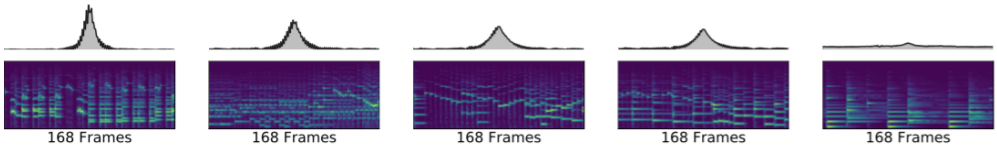



Figure 3.11: The attention mechanism is shown to train according to our intuition: at faster tempi, only a narrow part of the spectrogram is "let through" for the multimodal mechanism to learn, corresponding to the fact that the given fixed-width snippet of sheet music is played very quickly. As the music becomes slower, the attention mechanism allows focusing on a longer time range in the input spectrogram, so that the roughly the same amount of musical events is processed as in the faster music. (The figure is taken from [Dorfer et al., 2018].)

would expect: when note onsets are spread further from each other in time, as is the case when playing the same part of the music at a slower tempo, the attention mechanism is capable of adapting to what it believes is the representative counterpart to the sheet image snippet in question (see Fig. 3.11).

Adding the attention mechanism improves performance already at the snippet level he performance, improving Recall@1 from 0.41 to 0.48 and Recall@5 from 0.64 to 0.68. However, we realized that attention also allows using a larger audio window without causing imbalance in the relative importance of the audio and image input modalities, in order to deal specifically with slow tempi. Doubling the audio excerpts from 84 spectrogram frames to 168 improved Recall@1 to 0.55 and Recall@5 to 0.77.[23]

### 3.4.2 OMR Scientific Community

As a function of the international collaboration established through the project, the PI was able to contribute significantly to the coalescing of the field of OMR into an true scientific community, with a shared publication venue, introductory materials for newcomers, centralized resources, and, similarly to the related Digital Libraries for Music community, a place as a part of the broader Music Information Retrieval community. Three elements were critical to building this (sense of) community:

- The GREC 2017 discussion group, organized by Alicia Fornés. The PI actively participated in this group and co-wrote its report [Calvo Zaragoza et al., 2018]. The outputs of this OMR roundtable established guidance for further community-building activities, namely:
- The 1st International Workshop on Reading Music Systems (WoRMS), which took place as a satellite event of the ISMIR 2018 conference in

[23] As [Dorfer et al., 2018] was a smaller workshop paper, the full battery of retrieval experiments on whole pieces was not done.

Paris.[24] The PI served as one of the general chairs of the workshop. The 2nd WoRMS was held as a satellite event of ISMIR 2019 in Delft.[25]

- The tutorial "Optical Music Recognition for Dummies" selected for presentation at the ISMIR 2018 conference in Paris, which serves as extensive introductory material for newcomers to OMR. The tutorial is made available online.[26]

[24] http://ismir2018.ircam.fr/pages/events-at-a-glance.html
[25] https://ismir2019.ewi.tudelft.nl/?q=satellite-events
[26] https://youtube.com/playlist?list=PL1jvwDVNwQke-04UxzlzY4FM33bo1CGS0

# Publications by GA UK 144217


Matthias Dorfer, Jan Hajič jr. and Gerhard Widmer. On the Potential of Fully Convolutional Neural Networks for Musical Symbol Detection. *14th International Conference on Document Analysis and Recognition / GREC*, Kyoto, Japan, pp. 53–54, 2017. ISBN 978-1-5386-3586-5, doi: 10.1109/ICDAR.2017.274.

Alexander Pacha, Jan Hajič jr. and Jorge Calvo-Zaragoza. A Baseline for General Music Object Detection with Deep Learning. *Applied Sciences*, Vol. 8, No. 9, pages 1488–1488, Basel, Switzerland, 2018. ISSN 2076-3417.

Jan Hajič jr., Matthias Dorfer, Gerhard Widmer, Pavel Pecina. Towards Full-Pipeline Handwritten OMR with Musical Symbol Detection by U-Nets. *Proceedings of the 19th Conference of the International Society for Music Information Retrieval*, pages 225–232, Paris, France, 2018. ISBN 978-2-9540351-2-3.

Jan Hajič jr., Marta Kolárová, Alexander Pacha, Jorge Calvo-Zaragoza. How current optical music recognition systems are becoming useful for digital libraries. *Proceedings of the 5th International Conference on Digital Libraries for Musicology*, pages 57–61, Paris, France, 2018. ISBN 978-1-4503-6522-2.

Matthias Dorfer, Jan Hajič, jr.; Andreas Arzt, Harald Frostel, Gerhard Widmer. Learning Audio–Sheet Music Correspondences for Cross-Modal Retrieval and Piece Identification. *Transactions of the International Society for Music Information Retrieval*, 2018, vol. 1, pages 22—33. ISSN 2514-3298.

Matthias Dorfer, Jan Hajič, jr., Gerhard Widmer. Attention as a Perspective for Learning Tempo-invariant Audio Queries. *JMLR.org. Proceedings of the 35 th International Conference on Machine Learning*: JMLR.org, 2018.

Jorge Calvo-Zaragoza, Jan Hajič jr., Alexander Pacha. Discussion Group Summary: Optical Music Recognition. *Lecture Notes in Computer Science, Graphics Recognition. Current Trends and Evolutions. 12th IAPR International Workshop, GREC 2017, Kyoto, Japan, November 9-10, 2017, Revised Selected Papers*, Vol. 11009, No. 1, pages. 152-157, Basel, Switzerland, 2018. ISBN 978-3-030-02284-6, ISSN 0302-9743.

Jorge Calvo-Zaragoza, Jan Hajič jr. and Alexander Pacha. Understanding Optical Music Recognition. *ACM Computational Surveys; Manuscript under review (revision).*

# Bibliography


Alfio Andronico and Alberto Ciampa. On Automatic Pattern Recognition and Acquisition of Printed Music. In *International Computer Music Conference*, Venice, Italy, 1982. Michigan Publishing. URL `http://hdl.handle.net/2027/spo.bbp2372.1982.024`.

Jamie Anstice, Tim Bell, Andy Cockburn, and Martin Setchell. The design of a pen-based musical input system. In *6th Australian Conference on Computer-Human Interaction*, pages 260–267, 1996. doi: 10.1109/OZCHI.1996.560019.

David Bainbridge. A complete optical music recognition system: Looking to the future. Technical report, University of Canterbury, 1994. URL `https://ir.canterbury.ac.nz/handle/10092/14874`.

David Bainbridge. *Extensible optical music recognition*. PhD thesis, University of Canterbury, 1997. URL `http://hdl.handle.net/10092/9420`.

David Bainbridge and Tim Bell. Dealing with superimposed objects in optical music recognition. In *6th International Conference on Image Processing and its Applications*, number 443, pages 756–760, 1997. ISBN 0 85296 692 X. doi: 10.1049/cp:19970997.

David Bainbridge and Tim Bell. The Challenge of Optical Music Recognition. *Computers and the Humanities*, 35(2):95–121, 2001. ISSN 1572-8412. doi: 10.1023/A:1002485918032.

David Bainbridge and Tim Bell. A music notation construction engine for optical music recognition. *Software: Practice and Experience*, 33(2):173–200, 2003. ISSN 1097-024X. doi: 10.1002/spe.502.

David Bainbridge and Nicholas Paul Carter. Automatic reading of music notation. In H. Bunke and P. Wang, editors, *Handbook of Character Recognition and Document Image Analysis*, pages 583–603. World Scientific, Singapore, 1997. doi: 10.1142/9789812830968_0022.

Stephan Baumann. A Simplified Attributed Graph Grammar for High-Level Music Recognition. In *3rd International Conference on Document Analysis and Recognition*, pages 1080–1083. IEEE, 1995. ISBN 0-8186-7128-9. doi: 10.1109/ICDAR.1995.602096.

Pierfrancesco Bellini, Ivan Bruno, and Paolo Nesi. Optical music sheet segmentation. In *1st International Conference on WEB Delivering of Music*, pages 183–190. Institute of Electrical & Electronics Engineers (IEEE), 2001. ISBN 0769512844. doi: 10.1109/wdm.2001.990175.

Pierfrancesco Bellini, Ivan Bruno, and Paolo Nesi. Assessing Optical Music Recognition Tools. *Computer Music Journal*, 31(1):68–93, 2007. doi: 10.1162/comj.2007.31.1.68.

Hervé Bitteur. Audiveris, 2004. URL `https://github.com/audiveris`.

Dorothea Blostein and Henry S. Baird. *A Critical Survey of Music Image Analysis*, pages 405–434. Springer Berlin Heidelberg, 1992. ISBN 978-3-642-77281-8. doi: 10.1007/978-3-642-77281-8_19.

Gregory Burlet, Alastair Porter, Andrew Hankinson, and Ichiro Fujinaga. Neon.js: Neume Editor Online. In *13th International Society for Music Information Retrieval Conference*, pages 121–126, Porto, Portugal, 2012. URL `http://ismir2012.ismir.net/event/papers/121_ISMIR_2012.pdf`.

Donald Byrd and Jakob Grue Simonsen. Towards a Standard Testbed for Optical Music Recognition: Definitions, Metrics, and Page Images. *Journal of New Music Research*, 44(3):169–195, 2015. ISSN 0929-8215. doi: 10.1080/09298215.2015.1045424.

Jorge Calvo Zaragoza. *Pattern Recognition for Music Notation*. PhD thesis, 2016.

Jorge Calvo Zaragoza and Jose Oncina. Recognition of pen-based music notation with finite-state machines. *Expert Systems with Applications*, 72:395–406, 2017. ISSN 0957-4174. doi: 10.1016/j.eswa.2016.10.041.

Jorge Calvo Zaragoza and Jose Oncina. Recognition of Pen-Based Music Notation: The HOMUS Dataset. In *22nd International Conference on Pattern Recognition*, pages 3038–3043. Institute of Electrical & Electronics Engineers (IEEE), 2014. doi: 10.1109/ICPR.2014.524.

Jorge Calvo Zaragoza and Jose Oncina. Clustering of strokes from pen-based music notation: An experimental study. *Lecture Notes in Computer Science*, 9117:633–640, 2015. ISSN 0302-9743. doi: 10.1007/978-3-319-19390-8_71.

Jorge Calvo Zaragoza and David Rizo. End-to-End Neural Optical Music Recognition of Monophonic Scores. *Applied Sciences*, (4), 2018. ISSN 2076-3417. doi: 10.3390/app8040606. URL `http://www.mdpi.com/2076-3417/8/4/606`.

Jorge Calvo Zaragoza, David Rizo, and José Manuel Iñesta. Two (note) heads are better than one: pen-based multimodal interaction with music scores. In J. et al. Devaney, editor, *17th International Society for Music Information Retrieval Conference*, pages 509–514, New York City, 2016a. ISBN 978-0-692-75506-8. URL `https://wp.nyu.edu/ismir2016/wp-content/uploads/sites/2294/2016/07/006_Paper.pdf`.

Jorge Calvo Zaragoza, Gabriel Vigliensoni, and Ichiro Fujinaga. Document Analysis for Music Scores via Machine Learning. In *3rd International workshop on Digital Libraries for Musicology*, pages 37–40, New York, USA, 2016b. ACM, ACM. ISBN 978-1-4503-4751-8. doi: 10.1145/2970044.2970047.

Jorge Calvo Zaragoza, Antonio Pertusa, and Jose Oncina. Staff-line detection and removal using a convolutional neural network. *Machine Vision and Applications*, pages 1–10, 2017a. ISSN 1432-1769. doi: 10.1007/s00138-017-0844-4.

Jorge Calvo Zaragoza, Jose J. Valero Mas, and Antonio Pertusa. End-to-end Optical Music Recognition using Neural Networks. In *18th International Society for Music Information Retrieval Conference*, Suzhou, China, 2017b. ISBN 978-981-11-5179-8. URL `https://ismir2017.smcnus.org/wp-content/uploads/2017/10/34_Paper.pdf`.

Jorge Calvo Zaragoza, Gabriel Vigliensoni, and Ichiro Fujinaga. One-step detection of background, staff lines, and symbols in medieval music manuscripts with convolutional neural networks. In *18th International Society for Music Information Retrieval Conference*, Suzhou, China, 2017c. ISBN 978-981-11-5179-8. URL `https://ismir2017.smcnus.org/wp-content/uploads/2017/10/162_Paper.pdf`.

Jorge Calvo Zaragoza, Gabriel Vigliensoni, and Ichiro Fujinaga. Staff-Line Detection on Grayscale Images with Pixel Classification. In Luís A. Alexandre, José Salvador Sánchez, and João M. F. Rodrigues, editors, *Pattern Recognition and Image Analysis*, pages 279–286, Cham, 2017d. Springer International Publishing. ISBN 978-3-319-58838-4. URL `https://link.springer.com/chapter/10.1007%2F978-3-319-58838-4_31`.

Jorge Calvo Zaragoza, Jan Hajič jr., and Alexander Pacha. Discussion Group Summary: Optical Music Recognition. In Alicia Fornés and Lamiroy Bart, editors, *Graphics Recognition, Current Trends and Evolutions*, Lecture Notes in Computer Science, pages 152–157. Springer International Publishing, 2018. ISBN 978-3-030-02283-9. doi: 10.1007/978-3-030-02284-6_12.

Jorge Calvo-Zaragoza, Jan Hajič Jr., and Alexander Pacha. Understanding optical music recognition, 2019.

Jamie dos Santos Cardoso, Artur Capela, Ana Rebelo, Carlos Guedes, and Joaquim Pinto da Costa. Staff Detection with Stable Paths. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(6):1134–1139, 2009. ISSN 0162-8828. doi: 10.1109/TPAMI.2009.34.

Sukalpa Chanda, Debleena Das, Umapada Pal, and Fumitaka Kimura. Offline Hand-Written Musical Symbol Recognition. *14th International Conference on Frontiers in Handwriting Recognition*, pages 405–410, 2014. doi: 10.

1109/ICFHR.2014.74. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6981053.

Liang Chen and Kun Duan. MIDI-assisted egocentric optical music recognition. In *Winter Conference on Applications of Computer Vision*. Institute of Electrical and Electronics Engineers Inc., 2016. ISBN 9781509006410. doi: 10.1109/WACV.2016.7477714.

Liang Chen, Rong Jin, and Christopher Raphael. Renotation from Optical Music Recognition. In *Mathematics and Computation in Music*, pages 16–26, Cham, 2015a. Springer International Publishing. doi: 10.1007/978-3-319-20603-5_2.

Xinlei Chen, Hao Fang, Tsung-Yi Lin, Ramakrishna Vedantam, Saurabh Gupta, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO captions: Data collection and evaluation server. *arXiv preprint arXiv:1504.00325*, 2015b.

Maura Church and Michael Scott Cuthbert. Improving Rhythmic Transcriptions via Probability Models Applied Post-OMR. In Hsin-Min Wang, Yi-Hsuan Yang, and Jin Ha Lee, editors, *15th International Society for Music Information Retrieval Conference*, pages 643–648, 2014. URL http://www.terasoft.com.tw/conf/ismir2014/proceedings/T116_357_Paper.pdf.

Alastair T. Clarke, B. Malcom Brown, and M. P. Thorne. Coping with some really rotten problems in automatic music recognition. *Microprocessing and Microprogramming*, 27(1):547–550, 1989. ISSN 0165-6074. doi: 10.1016/0165-6074(89)90108-7. URL http://www.sciencedirect.com/science/article/pii/0165607489901087. Fifteenth EUROMICRO Symposium on Microprocessing and Microprogramming.

Bertrand Coüasnon and Jean Camillerapp. Using Grammars To Segment and Recognize Music Scores. In *International Association for Pattern Recognition Workshop on Document Analysis Systems*, pages 15–27, Kaiserslautern, Germany, 1994. URL ftp://ftp.idsa.prd.fr/local/IMADOC/couasnon/Articles/das94.ps.

Bertrand Coüasnon and Bernard Rétif. Using a grammar for a reliable full score recognition system. In *International Computer Music Conference*, pages 187–194, 1995. URL https://pdfs.semanticscholar.org/3b97/949f436f929ed11ee76358e07fa1a61d2e01.pdf.

Christoph Dalitz and Thomas Karsten. Using the Gamera framework for building a lute tablature recognition system. In *6th International Conference on Music Information Retrieval*, pages 478–481, London, UK, 2005. URL http://ismir2005.ismir.net/proceedings/2012.pdf.

Christoph Dalitz, Georgios K. Michalakis, and Christine Pranzas. Optical recognition of psaltic Byzantine chant notation. *International Journal of Document Analysis and Recognition*, 11(3):143–158, 2008. ISSN 1433-2825. doi: 10.1007/s10032-008-0074-4. URL https://doi.org/10.1007/s10032-008-0074-4.

Matthias Dorfer, Andreas Arzt, and Gerhard Widmer. Towards Score Following In Sheet Music Images. In Michael I. Mandel, Johanna Devaney, Douglas Turnbull, and George Tzanetakis, editors, *17th International Society for Music Information Retrieval Conference*, pages 789–795, 2016. ISBN 978-0-692-75506-8. URL https://wp.nyu.edu/ismir2016/wp-content/uploads/sites/2294/2016/07/027_Paper.pdf.

Matthias Dorfer, Jan Hajič jr., and Gerhard Widmer. On the Potential of Fully Convolutional Neural Networks for Musical Symbol Detection. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 02, pages 53–54, New York, USA, Nov 2017. IAPR TC10 (Technical Committee on Graphics Recognition), IEEE Computer Society. ISBN 978-1-5386-3586-5. doi: 10.1109/ICDAR.2017.274.

Matthias Dorfer, Jan Hajič jr., Andreas Arzt, Harald Frostel, and Gerhard Widmer. Learning Audio–Sheet Music Correspondences for Cross-Modal Retrieval and Piece Identification. *Transactions of the International Society for Music Information Retrieval*, 1(1):22–33, 2018a. doi: 10.5334/tismir.12.

Matthias Dorfer, Jan Hajič jr., and Gerhard Widmer. Attention as a Perspective for Learning Tempo-invariant Audio Queries. In *ICML 2018 Joint Workshop on Machine Learning for Music*, Stockholm, Sweden, 2018b. URL https://arxiv.org/pdf/1809.05689.pdf.

Matthias Dorfer, Florian Henkel, and Gerhard Widmer. Learning To Listen, Read And Follow: Score Following As A Reinforcement Learning Game. In *19th International Society for Music Information Retrieval Conference*, pages 784–791, Paris, France, 2018c. ISBN 978-2-9540351-2-3. URL http://ismir2018.ircam.fr/doc/pdfs/45_Paper.pdf.

Matthias Dorfer, Jan Schlüter, Andreu Vall, Filip Korzeniowski, and Gerhard Widmer. End-to-end cross-modality retrieval with cca projections and pairwise ranking loss. *International Journal of Multimedia Information Retrieval*, 7(2):117–128, 2018. doi: 10.1007/s13735-018-0151-5. URL https://doi.org/10.1007/s13735-018-0151-5.

Michael Droettboom, Ichiro Fujinaga, Karl MacMillan, G. Sayeed Chouhury, Tim DiLauro, Mark Patton, and Teal Anderson. Using the Gamera framework for the recognition of cultural heritage materials. In *Joint Conference on*

*Digital Libraries*, pages 12–17, London, UK, 2002. URL http://droettboom.com/papers/p74-droettboom.pdf.

Hoda M. Fahmy and Dorothea Blostein. Graph Grammar Processing of Uncertain Data. In *Advances in Structural and Syntactic Pattern Recognition*, pages 373–382. World Scientific, 1993. doi: 10.1142/9789812797919_0031.

Hoda M. Fahmy and Dorothea Blostein. A graph-rewriting paradigm for discrete relaxation: Application to sheet-music recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, 12(6):763–799, 1998. doi: 10.1142/S0218001498000439.

Alicia Fornés. Analysis of Old Handwritten Musical Scores. Master's thesis, Universitat Autònoma de Barcelona, 2005. URL http://www.cvc.uab.es/~afornes/publi/AFornes_Master.pdf.

Alicia Fornés. *Writer Identification by a Combination of Graphical Features in the Framework of Old Handwritten Music Scores*. PhD thesis, Universitat Autònoma de Barcelona, 2009. URL http://www.cvc.uab.es/~afornes/publi/PhDAliciaFornes.pdf.

Alicia Fornés, Josep Lladós, and Gemma Sánchez. Primitive Segmentation in Old Handwritten Music Scores. In Wenyin Liu and Josep Lladós, editors, *Graphics Recognition. Ten Years Review and Future Perspectives*, pages 279–290, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. ISBN 978-3-540-34712-5. doi: 10.1007/11767978_25.

Alicia Fornés, Anjan Dutta, Albert Gordo, and Josep Llados. The ICDAR 2011 Music Scores Competition: Staff Removal and Writer Identification. In *International Conference on Document Analysis and Recognition*, pages 1511–1515, 2011. doi: 10.1109/ICDAR.2011.300.

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. *International Journal on Document Analysis and Recognition*, 15(3):243–251, 2012. ISSN 1433-2825. doi: 10.1007/s10032-011-0168-2.

Alicia Fornés, Van Cuong Kieu, Muriel Visani, Nicholas Journet, and Anjan Dutta. The ICDAR/GREC 2013 Music Scores Competition: Staff Removal. In Bart Lamiroy and Jean-Marc Ogier, editors, *Graphics Recognition. Current Trends and Challenges*, pages 207–220, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg. ISBN 978-3-662-44854-0. URL https://link.springer.com/chapter/10.1007/978-3-662-44854-0_16.

Ichiro Fujinaga. Optical Music Recognition using Projections. Master's thesis, McGill University, 1988. URL https://www.researchgate.net/profile/

Ichiro_Fujinaga/publication/38435306_Optical_music_recognition_using_projections/links/546ca7980cf24b753c628c6e.pdf.

Ichiro Fujinaga. Exemplar-based learning in adaptive optical music recognition system. In *International Computer Music Conference*, pages 55–56, Hong Kong, 1996. ISBN 962-85092-1-7. URL http://hdl.handle.net/2027/spo.bbp2372.1996.015.

Ichiro Fujinaga. Optical Music Recognition Bibliography. Website, 2000. URL http://www.music.mcgill.ca/~ich/research/omr/omrbib.html.

Ichiro Fujinaga, Andrew Hankinson, and Julie E. Cumming. Introduction to SIMSSA (Single Interface for Music Score Searching and Analysis). In *1st International Workshop on Digital Libraries for Musicology*, pages 1–3. ACM, 2014. doi: 10.1145/2660168.2660184.

Antonio-Javier Gallego and Jorge Calvo Zaragoza. Staff-line removal with selectional auto-encoders. *Expert Systems with Applications*, 89:138–148, 2017. ISSN 0957-4174. doi: 10.1016/j.eswa.2017.07.002. URL http://www.sciencedirect.com/science/article/pii/S0957417417304712.

Jan Hajič jr. and Matthias Dorfer. Handwritten Optical Music Recognition: a Working Prototype. In *Extended abstracts for the Late-Breaking Demo Session of the 18th International Society for Music Information Retrieval Conference*, Suzhou, China, 2017. URL https://ismir2017.smcnus.org/lbds/Hajic2017.pdf.

Jan Hajič jr. and Pavel Pecina. In Search of a Dataset for Handwritten Optical Music Recognition: Introducing MUSCIMA++. *Computing Research Repository*, abs/1703.04824:1–16, 2017a. URL http://arxiv.org/abs/1703.04824.

Jan Hajič jr. and Pavel Pecina. Groundtruthing (Not Only) Music Notation with MUSICMarker: A Practical Overview. In *14th International Conference on Document Analysis and Recognition*, pages 47–48, Kyoto, Japan, 2017b. doi: 10.1109/ICDAR.2017.271.

Jan Hajič jr. and Pavel Pecina. The MUSCIMA++ Dataset for Handwritten Optical Music Recognition. In *14th International Conference on Document Analysis and Recognition*, pages 39–46, Kyoto, Japan, 2017c. doi: 10.1109/ICDAR.2017.16.

Jan Hajič jr., Jiří Novotný, Pavel Pecina, and Jaroslav Pokorný. Further Steps towards a Standard Testbed for Optical Music Recognition. In Michael Mandel, Johanna Devaney, Douglas Turnbull, and George Tzanetakis, editors, *17th International Society for Music Information Retrieval Conference*, pages

157–163, New York, USA, 2016. New York University, New York University. ISBN 978-0-692-75506-8. URL https://wp.nyu.edu/ismir2016/event/proceedings/.

Jan Hajič jr., Matthias Dorfer, Gerhard Widmer, and Pavel Pecina. Towards Full-Pipeline Handwritten OMR with Musical Symbol Detection by U-Nets. In *19th International Society for Music Information Retrieval Conference*, pages 225–232, Paris, France, 2018a. ISBN 978-2-9540351-2-3. URL http://ismir2018.ircam.fr/doc/pdfs/175_Paper.pdf.

Jan Hajič jr., Marta Kolárová, Alexander Pacha, and Jorge Calvo Zaragoza. How Current Optical Music Recognition Systems Are Becoming Useful for Digital Libraries. In *5th International Conference on Digital Libraries for Musicology*, pages 57–61, Paris, France, 2018b. ACM. ISBN 978-1-4503-6522-2. doi: 10.1145/3273024.3273034. URL http://doi.acm.org/10.1145/3273024.3273034.

Andrew Hankinson. *Optical music recognition infrastructure for large-scale music document analysis*. PhD thesis, McGill University, 2014. URL http://digitool.library.mcgill.ca/webclient/DeliveryManager?pid=130291.

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

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521: 436 EP –, 05 2015. URL https://doi.org/10.1038/nature14539.

Leo Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth, 1984. ISBN 0-534-98053-8.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal Loss for Dense Object Detection. *CoRR*, abs/1708.02002, 2017. URL http://arxiv.org/abs/1708.02002.

Nailja Luth. Automatic Identification of Music Notations. In *2nd International Conference on WEB Delivering of Music*, 2002. ISBN 0769518621. doi: 10.1109/WDM.2002.1176212.

Karl MacMillan, Michael Droettboom, and Ichiro Fujinaga. Gamera: A structured document recognition application development environment. In *2nd International Symposium on Music Information Retrieval*, pages 15–16, Bloomington, IN, 2001. URL https://jscholarship.library.jhu.edu/handle/1774.2/44376.

Karl MacMillan, Michael Droettboom, and Ichiro Fujinaga. Gamera: Optical music recognition in a new shell. In *International Computer Music Conference*,

pages 482–485, 2002. URL http://www.music.mcgill.ca/~ich/research/icmc02/icmc2002.gamera.pdf.

Youichi Mitobe, Hidetoshi Miyao, and Minoru Maruyama. A fast HMM algorithm based on stroke lengths for on-line recognition of handwritten music scores. In *9th International Workshop on Frontiers in Handwriting Recognition*, pages 521–526, 2004. doi: 10.1109/IWFHR.2004.2.

Hidetoshi Miyao and Minoru Maruyama. An online handwritten music score recognition system. In *17th International Conference on Pattern Recognition*. Institute of Electrical & Electronics Engineers (IEEE), 2004. doi: 10.1109/icpr.2004.1334164.

Kia Ng, David Cooper, Ewan Stefani, Roger Boyle, and Nick Bailey. Embracing the Composer : Optical Recognition of Handwrtten Manuscripts. In *International Computer Music Conference*, pages 500–503, 1999. URL https://ci.nii.ac.jp/naid/10011612045/en/.

Jiri Novotný and Jaroslav Pokorný. Introduction to Optical Music Recognition: Overview and Practical Challenges. In Pokorny J. Necasky M., Moravec P., editor, *Annual International Workshop on DAtabases, TExts, Specifications and Objects*, pages 65–76. CEUR-WS, 2015. URL http://ceur-ws.org/Vol-1343/paper6.pdf.

Alexander Pacha and Jorge Calvo Zaragoza. Optical Music Recognition in Mensural Notation with Region-Based Convolutional Neural Networks. In *19th International Society for Music Information Retrieval Conference*, pages 240–247, Paris, France, 2018. ISBN 978-2-9540351-2-3. URL http://ismir2018.ircam.fr/doc/pdfs/32_Paper.pdf.

Alexander Pacha, Kwon-Young Choi, Bertrand Coüasnon, Yann Ricquebourg, Richard Zanibbi, and Horst Eidenberger. Handwritten Music Object Detection: Open Issues and Baseline Results. In *13th International Workshop on Document Analysis Systems*, pages 163–168, 2018a. doi: 10.1109/DAS.2018.51.

Alexander Pacha, Jan Hajič jr., and Jorge Calvo Zaragoza. A Baseline for General Music Object Detection with Deep Learning. *Applied Sciences*, 8 (9):1488–1508, 2018b. ISSN 2076-3417. doi: 10.3390/app8091488. URL http://www.mdpi.com/2076-3417/8/9/1488.

Viet-Khoi Pham, Hai-Dang Nguyen, and Minh-Triet Tran. Virtual Music Teacher for New Music Learners with Optical Music Recognition. In *International Conference on Learning and Collaboration Technologies*, pages 415–426. Springer, 2015. doi: 10.1007/978-3-319-20609-7_39.

David S. Prerau. Computer pattern recognition of printed music. In *Fall Joint Computer Conference*, pages 153–162, 1971.

Denis Pruslin. *Automatic recognition of sheet music*. PhD thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA, 1966.

Laurent Pugin. Optical Music Recognitoin of Early Typographic Prints using Hidden Markov Models. In *7th International Conference on Music Information Retrieval*, pages 53–56, Victoria, Canada, 2006a. URL `http://ismir2006.ismir.net/PAPERS/ISMIR06152_Paper.pdf`.

Laurent Pugin. Aruspix: an Automatic Source-Comparison System. *Computing in Musicology*, 14:49–59, 2006b. ISSN 1057-9478. URL `https://dialnet.unirioja.es/servlet/articulo?codigo=3476563`.

Laurent Pugin, Jason Hockman, John Ashley Burgoyne, and Ichiro Fujinaga. Gamera versus Aruspix – Two Optical Music Recognition Approaches. In *9th International Conference on Music Information Retrieval*, 2008. URL `http://ismir2008.ismir.net/papers/ISMIR2008_247.pdf`.

Christopher Raphael and Jingya Wang. New Approaches to Optical Music Recognition. In Anssi Klapuri and Colby Leider, editors, *12th International Society for Music Information Retrieval Conference*, pages 305–310, Miami, Florida, 2011. University of Miami. URL `http://ismir2011.ismir.net/papers/OS3-3.pdf`.

Ana Rebelo. *Robust Optical Recognition of Handwritten Musical Scores based on Domain Knowledge*. PhD thesis, University of Porto, 2012. URL `http://www.inescporto.pt/~arebelo/arebeloThesis.pdf`.

Ana Rebelo and Jamie dos Santos Cardoso. Staff Line Detection and Removal in the Grayscale Domain. In *12th International Conference on Document Analysis and Recognition*, pages 57–61, 2013. doi: 10.1109/ICDAR.2013.20.

Ana Rebelo, G. Capela, and Jamie dos Santos Cardoso. Optical recognition of music symbols. *International Journal on Document Analysis and Recognition*, 13(1):19–31, 2010. ISSN 1433-2825. doi: 10.1007/s10032-009-0100-1.

Ana Rebelo, Ichiro Fujinaga, Filipe Paszkiewicz, Andre R.S. Marcal, Carlos Guedes, and Jamie dos Santos Cardoso. Optical music recognition: state-of-the-art and open issues. *International Journal of Multimedia Information Retrieval*, 1(3):173–190, 2012. doi: 10.1007/s13735-012-0004-6.

Ana Rebelo, André Marçal, and Jamie dos Santos Cardoso. Global constraints for syntactic consistency in OMR: an ongoing approach. In *International Conference on Image Analysis and Recognition*, 2013.

URL http://www.inescporto.pt/~jsc/publications/conferences/2013ARebeloICIAR.pdf.

K. Todd Reed and J. R. Parker. Automatic Computer Recognition of Printed Music. In *13th International Conference on Pattern Recognition*, pages 803–807, 1996. ISBN 081867282X. doi: 10.1109/ICPR.1996.547279.

David Rizo, Jorge Calvo Zaragoza, and José M. Iñesta. MuRET: A Music Recognition, Encoding, and Transcription Tool. In *5th International Conference on Digital Libraries for Musicology*, pages 52–56, Paris, France, 2018. ACM. ISBN 978-1-4503-6522-2. doi: 10.1145/3273024.3273029. URL http://doi.acm.org/10.1145/3273024.3273029.

JW W Roach and J E Tatem. Using domain knowledge in low-level visual processing to interpret handwritten music: an experiment. *Pattern Recognition*, 21(1):33–44, 1988. ISSN 0031-3203. doi: 10.1016/0031-3203(88)90069-6. URL http://www.sciencedirect.com/science/article/pii/0031320388900696.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III*, pages 234–241, Cham, 2015. Springer International Publishing. ISBN 978-3-319-24574-4. doi: 10.1007/978-3-319-24574-4_28. URL https://doi.org/10.1007/978-3-319-24574-4_28.

Florence Rossant and Isabelle Bloch. Robust and Adaptive OMR System Including Fuzzy Modeling, Fusion of Musical Rules, and Possible Error Detection. *EURASIP Journal on Advances in Signal Processing*, 2007(1):081541, 2006. ISSN 1687-6180. doi: 10.1155/2007/81541.

Zeyad Saleh, Ke Zhang, Jorge Calvo Zaragoza, Gabriel Vigliensoni, and Ichiro Fujinaga. Pixel.js: Web-Based Pixel Classification Correction Platform for Ground Truth Creation. In *14th International Conference on Document Analysis and Recognition*, pages 39–40, Kyoto, Japan, 2017. doi: 10.1109/ICDAR.2017.267.

Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In Corinna Cortes, Neil D. Lawrence, Daniel D. Lee, Masashi Sugiyama, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 28: Annual*

*Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 91–99, 2015.

Baoguang Shi, Xiang Bai, and Cong Yao. An End-to-End Trainable Neural Network for Image-Based Sequence Recognition and Its Application to Scene Text Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(11):2298–2304, 2017. ISSN 0162-8828. doi: 10.1109/TPAMI.2016.2646371.

Javier Sober Mira, Jorge Calvo Zaragoza, David Rizo, and José Manuel Iñesta. Pen-Based Music Document Transcription. In *14th International Conference on Document Analysis and Recognition*, pages 21–22, Kyoto, Japan, 2017. IEEE. doi: 10.1109/ICDAR.2017.258.

Mariusz Szwoch. Guido: A Musical Score Recognition System. In *9th International Conference on Document Analysis and Recognition*, pages 809–813, 2007. doi: 10.1109/ICDAR.2007.4377027.

Mariusz Szwoch. Using MusicXML to Evaluate Accuracy of OMR Systems. In *International Conference on Theory and Application of Diagrams*, pages 419–422, Herrsching, Germany, 2008. Springer, Springer-Verlag. ISBN 978-3-540-87729-5. doi: 10.1007/978-3-540-87730-1_53.

Theophanis Tsandilas. Interpreting Strokes on Paper with a Mobile Assistant. In *25th Annual ACM Symposium on User Interface Software and Technology*, pages 299–308, Cambridge, Massachusetts, USA, 2012. ACM. ISBN 978-1-4503-1580-7. doi: 10.1145/2380116.2380155.

Lukas Tuggener, Isamil Elezi, Jürgen Schmidhuber, Marcello Pelillo, and Stadelmann Thilo. DeepScores - A Dataset for Segmentation, Detection and Classification of Tiny Objects. In *24th International Conference on Pattern Recognition*, Beijing, China, 2018. doi: 10.21256/zhaw-4255. URL `https://arxiv.org/abs/1804.00525`.

Eelco van der Wel and Karen Ullrich. Optical Music Recognition with Convolutional Sequence-to-Sequence Models. In *18th International Society for Music Information Retrieval Conference*, Suzhou, China, 2017. ISBN 978-981-11-5179-8. URL `https://arxiv.org/abs/1707.04877`.

Cuihong Wen, Jing Zhang, Ana Rebelo, and Fanyong Cheng. A Directed Acyclic Graph-Large Margin Distribution Machine Model for Music Symbol Classification. *PLoS ONE*, 11(3):1–11, 2016. doi: 10.1371/journal.pone.0149688.

# A. Attachments

## A.1 Reading Music

We have stated that "music notation is [...] a visual language that encodes music graphically; the role of OMR is to automatically extract the encoded musical information from this graphical representation". In other words, the role of OMR is to automatically *read music*. What is this process of reading music? What is it that OMR is actually trying to achieve, in more specific terms that can be formalized for automation?

We proceed by introducing the two layers that take part in this process: the layer of musical semantics, which is the target of the process, and the layer of music notation, which is its input, and show how these relate to each other.

### A.1.1 Musical Semantics

First, we define what we mean by this "encoded musical representation".

The music that is encoded with Common Western Music Notation (CWMN) can be conceptualized as a structure of *notes in time*. This is not necessarily the only conceptualization of music, but it is the one that CWMN is designed to communicate. Notes form further structures, such as voices or phrases, but for the purposes of OMR, we restrict the conceptualization of musical semantics to a *set* of notes.

We must also clarify what "in time" means. Musical semantics are concerned not with wallclock time measured in seconds, but with *musical time*. Musical time is an abstract concept measured in units called *beats* that can be further subdivided into regular parts. Musical time is only projected into wallclock time during performance – primarily by means of *tempo*, which sets a certain baseline rate of beats per minute. However, in reality, the projection is wildly non-linear, and it is at the discretion of the performer; a linear projection would result in a robotic, boring performance.

A *note* is an abstract object that is defined by four properties: its *pitch*, *duration*, *strength*, and *timbre*. When a musician plays a note, these properties are translated into a fundamental frequency (pitch) maintained for a certain time (duration) with certain perceptual loudness (strength) and spectral characteristics (timbre). The notes are placed on the axis of *musical time* with a fifth parameter, their *onset*, which governs during performance when the note should start.

Next to notes, there are *rests*: periods of silence. They can be regarded as pseudo-notes that only have temporal parameters: onset and duration.

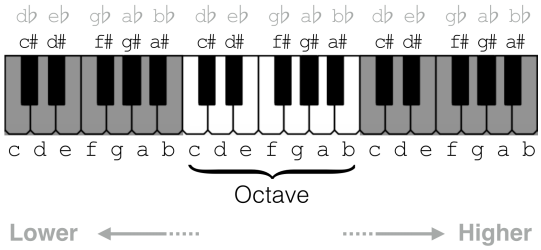



Figure A.1: The values of pitch, illustrated on a piano keyboard. One octave period is indicated.

### Pitch

Formally, *pitch* is a categorical variable whose values are linearly ordered from lowest to highest. A piano-centric schematic of how notes are indexed by pitch is provided in Fig. A.1. The distance between neighboring pitches is called the *semitone* and corresponds to neighboring keys. The values of pitch are expressed in terms of ⟨`step, octave, accidental`⟩ triplets:

- *Step*, which is traditionally one of A, B[1], C, D, E, F, and G. The distance between neighboring steps is a whole tone (two semitones), except for B-C and E-F, which are only a semitone from each other.
- One period of this pattern of steps is called the *octave*, which is the second of the three determining parameters of pitch in post-medieval musical tradition. Octaves are usually numbered in the English-speaking tradition 1 to 8, from lowest to highest, with the "middle c" on a piano falling into octave 4. Although steps are traditionally named from A to G, octaves – somewhat confusingly – usually start with C and end with B, again especially where German pedagogical tradition is prevalent.
- The third pitch descriptor is the *accidental*. The accidental can shift the pitch, as determined by the step and octave, by one semitone upwards (indicated using a *sharp* sign: `#`), one semitone down (indicated using a *flat* sign: `b`), and sometimes by two semitones, indicated with a double sharp (an `x`-like symbol) and analogously a double flat (written usually as *bb*).

We emphasize again that pitch is an *abstract* object: it is not yet the fundamental frequency of the corresponding tone that would sound during performance. The relationship from pitch to a fundamental frequency during performance is governed by the *tuning* used. The middle A (A4) is first tied to a particular frequency; in modern times, this is usually 440 Hz.[2] The second component of projecting pitch to fundamental frequency is the *temperament*: the

[1] In German tradition, which is also prevalent in the Czech Republic, B is called H.

[2] In practice, however, this is a much more complicated topic. Orchestras that include many wind instruments sometimes tune at 442 Hz, but in the baroque period, this standard varied:

difference in frequency between individual semitones. In modern times, all the semitone distances between neighboring pitches are the same (this is called *equal temperament*); as indicated by Fig. A.1, this implies that, e.g., an F4 with a sharp (`#`) refers to the same frequency as a G4 with a flat (`b`): F4 and G4 are one tone away from each other, and the semitone upwards from F4 to F4# is the same as the one downwards from G4 to G4b. If a different temperament was used, e.g. the historical *meantone temperament*[3], the distance from F4 to F4# would not be exactly half that of the distance from F4 to G4, and vice versa (the distance from G4 down to G4b would also be slightly smaller than the whole tone distance F4–G4 divided by two).[4]

The take-away from this explanation should be that pitch is an abstract concept expressed usually with a step, octave and accidental, that pitch of a note and frequency of a performed tone are two distinct concepts, and that music notation is concerned with encoding pitch, not frequency.

### Duration

The duration of a note is theoretically any rational number, but in practice – such that is encoded using music notation – it also becomes a categorical variable. As stated above (section A.1.1), duration is expressed in terms of *beats*, the basic units of musical time. In CWMN, these values are powers of 2, mostly negative: 8, 4, 2, 1, 1/2, 1/4, 1/8, 1/16, 1/32 and 1/64, rarely higher (1/128). The usual terminology calls a 4-beat note a *whole note*, a note that has a duration of 2 beats is called a *half-note*, a note that lasts for one beat is a *quarter note*, etc. (The most common grouping of beats into regular units called *measures* is 4, hence the name "whole" for the 4-beat note, as it lasts for the whole measure.) Additionally, music notation has ways to encode durations that do not fit into this sequence of powers of two, so that durations like 1/3 of a beat can be expressed as well. The usual durations one encounters (from a whole note to a 64th note) are usually depicted in music notation as shown in Fig. A.2.

Duration applies to rests as well as to notes.

---

for most of music from the time of J. S. Bach or J. D. Zelenka, setting A4 at 415 Hz (roughly a semitone lower) is more appropriate, and for earlier music, many more tunings were used, both lower (390 Hz) and higher (467 Hz). Furthermore, current musicological and organological research suggests that much of Mozart's or Beethoven's music was originally performed with A4 = 430 Hz...

[3]There are very good musical reasons for this, not just chasing the nebulous concept of "authenticity": meantone temperament provides very different timbre and character for different harmonies (although some become unusable), and these differences were actively exploited by composers of the time; when such compositions – especially those that feature the voice – are performed in equal temperament, they lose much of their dramatic power.

[4]On a keyboard instrument, when using meantone, one then has to choose whether to tune the given black key as the appropriate F4#, or G4b. Many so-called "well temperaments" were developed to allow some compromises in this respect; the title of J. S. Bach's "Well-Tempered Clavier" refers to one of these many compromise options, not to the equal temperament of today. Pitch and tuning is a fascinating topic!

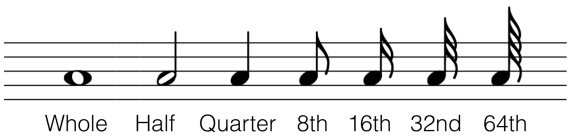



Figure A.2: The types of notes according to duration. Two half-notes takes as long as one whole note, one half-note takes the same number of beats as two quarter-notes, etc.

### Onset

The onset of a note is the point in musical time at which the note should start being played. The elementary rule for determining the onset of a note is: once the previous note ends, the next note begins. In order to determine the onset of a note, one must therefore know the sequence into which the notes are organized. This sequence is called a *voice*. Assuming a composition with a single voice (monophonic music), the first note of the voice has an onset of 0, and the onset of the $i$-th note in the given voice is computed as $\mathbf{onset}(i-1)+\mathbf{duration}(i-1)$. This is the important concept of *precedence*. In the written score, notes are encoded by precedence from left to right according to the positions of the corresponding graphical notes. When more than one voice is present, in music that is called *polyphonic*, a simple left-to-right ordering is not sufficient; for correctly inferring precedence, one must correctly assign notes to voices.

The concept orthogonal to precedence is *simultaneity*: a simultaneity is a set of notes that shares the same onset. Simultaneity can happen within a voice, where all the participating notes share the same preceding note (or members of the preceding simultaneity); if all simultaneities with more than one member are such that the notes belong to the same voice, we call the music *homophonic*.

### Strength and Timbre

The note parameters of strength and timbre can be mostly left out for the purposes of OMR, as they are barely encoded in music notation. Strength is rudimentarily conveyed with dynamic markings such as *piano* or *forte*, and some symbols that convey how strength should change over a sequence of notes (*crescendo* for increasing, *decrescendo* for gradually decreasing strength); timbre is expressed with sporadic textual expressive markings or instrument-specific techniques. These two parameters are left for the most part at the discretion of the performer.

Importantly, for the motivating applications of OMR listed above, strength and timbre need not be a part of the OMR outputs. **Therefore, for the purposes of this project, we simplify the definition of a note to just the triplet** ⟨`pitch, duration, onset`⟩, and the task of recovering musical seman-

tics then becomes the task of recovering the set of such triplets. This simplified representation is already rich enough to have the OMR system output files in the widely used MIDI format, which in turn serves as an input of many of the downstream applications of OMR – especially full-text search in music archives and other applications oriented towards digital libraries, more broadly curating and making accessible sheet music collections, and musicology overall.

### A.1.2 Music Notation

Now that we have introduced the layer of *musical semantics*, we turn our attention to the music notation, specifically Common Western Music Notation (CWMN) visual language and its relationship to these semantics. We introduce music notation terminology factored into subsets of the music notation alphabet according to which aspect of the musical semantics defined above the given symbols help encode.

The elementary interface between the "written" layer and the "semantic" layer, or music notation and the notes that it encodes, are *noteheads*. The general rule is that **one notehead encodes one note**.[5] Noteheads are round[6] objects with a fixed size.[7] They are the central and most frequent music notation primitives.

In the context of OMR, one must be carefull to distinguish the abstract musical *note* object with the composite graphical objects that are often also called notes. These graphical objects serve as a useful pedagogical concept, but for the purposes of OMR they are not well-defined. Whenever we wish to refer to the graphical objects, we will explicitly use the term *graphical note*.

A reference example of actual music notation is given in Fig. A.3.

Suppose we have correctly identified noteheads, and therefore we know how many notes are encoded in a given notation document. It remains to find their parameters that are unambiguously recorded by music notation: pitches, durations, and onsets. The nature of the music notation visual language is *featural*: these properties are encoded not using individual primitives, but using the *configurations* of these primitives. We describe which primitives participate in encoding which property, and explain how.

[5] In polyphonic music, this rule may get broken: multiple voices on a single staff may share pitch and have a graphically compatible duration, in which case a notehead can encode two notes; this would be evident to the reader from the presence of two stems attached to a single notehead.

[6] Mostly. As is the case with nearly everything in music notation, exceptions abound. In this case, CWMN allows for different notehead shapes, especially in the latter half of the 20th century: these are most widely used to indicate certain playing techniques that affect timbre. Percussion parts are most prone to non-round noteheads. However, as these are still (relatively) rare, we do not consider non-round noteheads in this project.

[7] Except for *grace* noteheads: these encode ornamental notes that do not have a duration of their own.

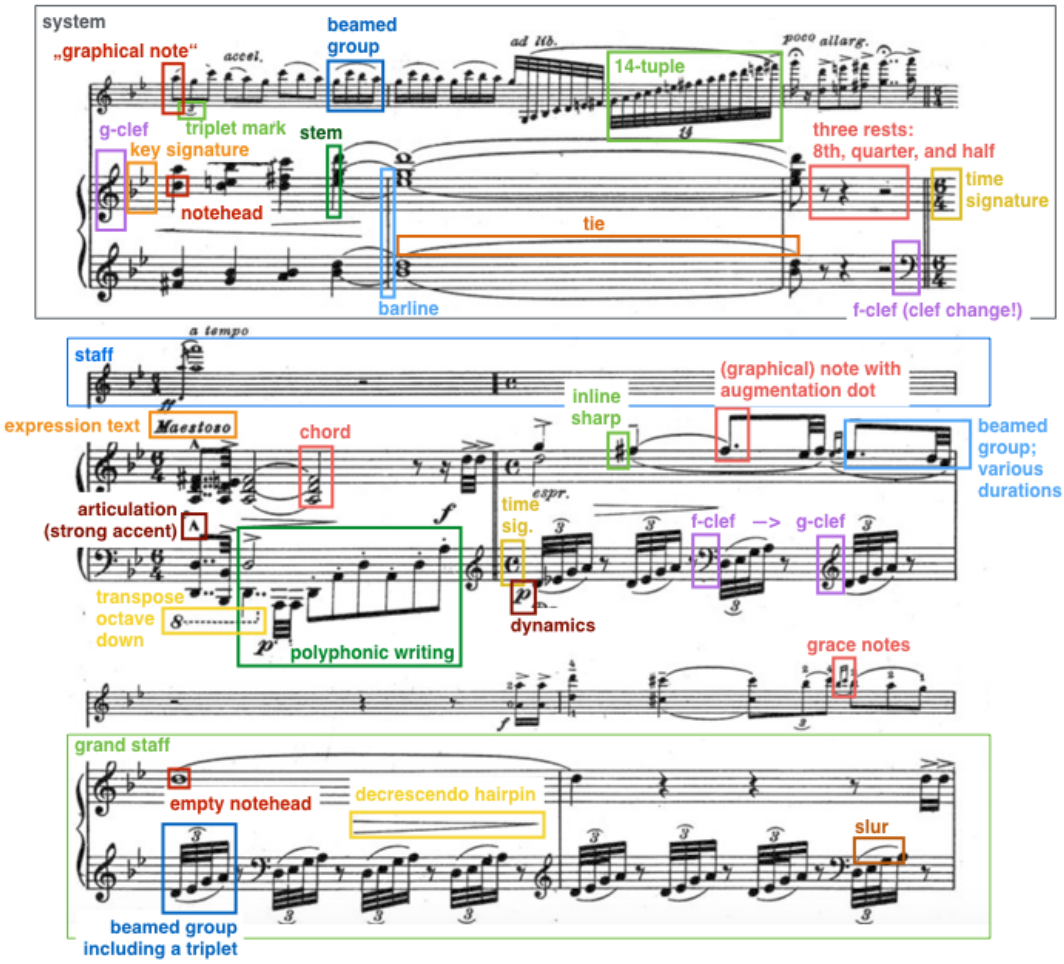



Figure A.3: An example of real-world notation, with the individual notation elements marked (not exhaustive). Taken from: Ernst Bloch, Baal Schem: Drei chassidische Stimmungen for violin and piano, II: Improvisation (Nigun).

### Encoding Pitch

Pitch, described as a ⟨`step, octave, accidental`⟩ triplet, is encoded using the following primitives:

- stafflines and spaces between them, which combine into staffs,
- ledger lines,
- clefs (g-clef, f-clef, c-clef),
- accidentals (sharp, flat, double sharp, double flat, natural),
- measure separators (thin and thick barlines and barline groups).

Three notes with the same pitch are shown in Fig. A.4.

Since the times of Guido of Arrezzo (12th century), the *staff*, consisting of a number of equidistant parallel stafflines and the spaces between them, is the basic

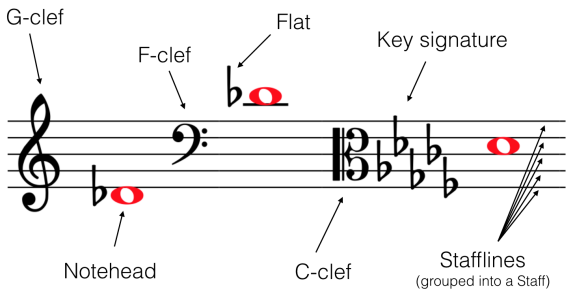



Figure A.4: The elements encoding pitch: three notes with the same pitch written in different ways. Noteheads red, for clarity.

layout object in music notation. **Music is read left-to-right** per *system*; each system consists of a number of *staffs* that are read concurrently; each staff consists of a certain number of *stafflines* and the *staffspaces* between those lines (plus two surrounding the outer stafflines of the staff). Overwhelmingly often, in post-1650 scores, staffs are built from groups of five stafflines, with six staffspaces between and around them.[8] **Noteheads are placed on stafflines or into staffspaces.** Each notehead is positioned on exactly one staffline or staffspace (not both). Moving a notehead to the neighboring staffspace (or, vice versa, staffline) changes its pitch by a step (not necessarily by a semitone; see subsection A.1.1).

*Ledger lines* are used when the pitch of a note is more extreme than can be recorded with the given limited number of stafflines and staffspaces. Ledger lines are short horizontal lines parallel to the staff that simulate the presence of a given number of additional stafflines for the given note; they are interpreted exactly as additional stafflines would be.

The *clef* symbols are also attached to stafflines (not staffspaces, however). **Clefs govern how positions on the staff are interpreted in terms of step and octave.** The *f-clef* denotes the staffline on which noteheads will be interpreted to encode notes with the pitch F3; overwhelmingly often, this would be the second staffline from the top. This clef is also known as the bass clef, as it is most often used to encode lower-sounding music, such as the left hand's part in piano literature or the bass part in vocal music. The *g-clef*, also known the violin clef, denotes the staffline corresponding to the pitch G4; this clef is overwhelmingly often placed on the fourth line from the top, and is used for music in the higher ranges (right hand on the piano, soprano and later also alto parts, the eponymous violin music, etc.). The *c-clef*, sometimes called the viola clef, denotes the staffline corresponding to the middle C (C4), and as its name suggests it is used mostly with alto or tenor instruments such as the viola. The c-clef usually appears in viola parts on the middle of the five standard stafflines,

[8]The widest exception today is modern plainchant notation with four stafflines, and single-line percussion parts.

but in trombone, French horn, or cello music, it often appears also on the second line from the top, and in pre-1750 music it can appear on any of the stafflines.[9]

Clefs are most often present at the beginnings of staffs, but it is perfectly valid to encounter clefs anywhere on a staff; all clefs are valid for interpreting notehead positions on the given staff only to the right of the given clef.

Thus, from the position of the clef, and the position of the notehead on the staff (in terms of which staffline, staffspace, or ledger line the notehead is placed on), one can assign to each notehead the `step` and `octave` parts of its pitch. What remains is the `accidental`. This is encoded by *accidental* primitives (again, one must distinguish between the semantic property of `accidental` and the subset of music notation primitives – sharps, flats, etc. – that encode this semantic property). These primitives are the *sharp* (modifies pitch upwards by semitone), the *flat* (downwards), the *double sharp* and *double flat* that modify pitch twice as much, and the *natural*, which cancels any accidental that might have been valid for the given note. Similarly to clefs, accidentals are also valid to the right of their horizontal position on the staff, but they have two flavors – *inline* accidentals, and *key signature* accidentals.

*Inline accidentals* apply to notes encoded by noteheads on the same staff position as the accidental by convention up until the end of the measure, denoted by the next barline or barline group.[10] Using this convention, if one wants to cancel an inline accidental at a given note (e.g., first encode F4#, then F4), one would use the *natural* sign to cancel, from then onward, the effect of the accidental that would apply at that point.

Accidentals in *key signatures* (which are simply groups of accidentals that are not associated with any specific notehead) differ from their inline counterparts in that their validity does *not* expire with the next barline: key signature accidentals stay valid unless overriden temporarily by an inline accidental, or permanently by a new key signature. They are usually re-stated at the beginning of each staff.

Note also that the inline accidentals and key signature changes imply that reading music is a *stateful* process: in order to correctly read the next note, one must remember some information about the previous notes and music notation elements that have already been read.

This analysis should now clarify why the notes corresponding to the three highlighted noteheads in Fig. A.4 actually encode the same pitch.

### Encoding Duration

Duration is encoded using:

[9]Surprisingly, in early music, the g-clef and the f-clef only appear more frequently in non-standard positions in French baroque scores.

[10]Before roughly 1670, the convention was to apply the inline accidental only to the next note on the given staff position, and this has become relevant again for atonal music in the 20th century.

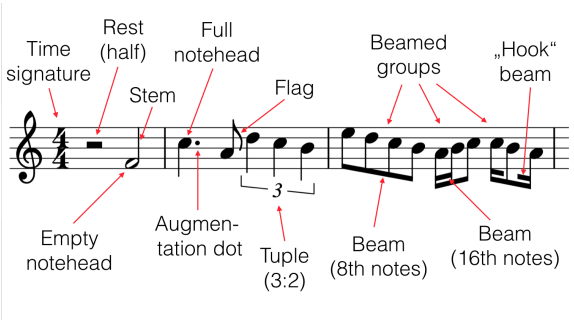



Figure A.5: The elements encoding duration.

- notehead type (full vs. empty),
- stem,
- flags and beams,
- augmentation dots (absent, one, two),
- tuples.

Recall that duration is a categorical variable that determines for how many beats (units of musical time) a note should be held. The values of duration are drawn mostly from powers of 2: 4 beats, 2 beats, 1, 1/2, 1/4, 1/8 and 1/16 (rarely is a note written shorter than 1/16th of a beat). This is all we need to care about in OMR; music notation does *not* encode the relationship of beats to wallclock time. The names for durations are somewhat misleading: a 4-beat note is a *whole* note, a 2-beat note is commonly called a *half* note, a 1-beat note is a *quarter* note, etc.

See Fig. A.2 for how these duration values are prototypically encoded in CWMN. An empty notehead without a stem denotes a whole note; with a stem, it encodes a half note. A full notehead with a stem encodes a quarter note (1 beat); in CWMN, full noteheads are required to have stems. Smaller values are then encoded with *flags* or *beams*. There can be more than one flag (or beam) associated with a notehead; each associated flag or beam halves the duration of the encoded note (a note encoded with a full notehead, stem and no flag has a duration of 1 beat, with one flag it will have a duration of 1/2 beat, with two flags, 1/4 of a beat, etc.). Beams are used instead of flags prototypically when there is more than one consecutive note with sub-beat duration; they connect the respective graphical notes into *beamed groups*. For duration values from outside this row – especially when dividing a larger value into three equal parts instead of two – the *tuple* symbols are used. The elements that govern duration are depicted in Fig. A.5.

Beams and beamed groups are some of the most problematic elements of music notation for OMR. Individual beams can have wildly different lengths, thicknesses and angles; the way they are combined into groups relies on just a few rules, but can lead to visually complex structures with many overlapping

Figure A.6: A somewhat complex beamed group. Note how it changes stem sizes, which are otherwise mostly fixed (at 3.5∗*staffspace_height*). The first two and the last note have a 16th duration (two relevant beams), the third and fourth notes are 32nd notes (three beams).

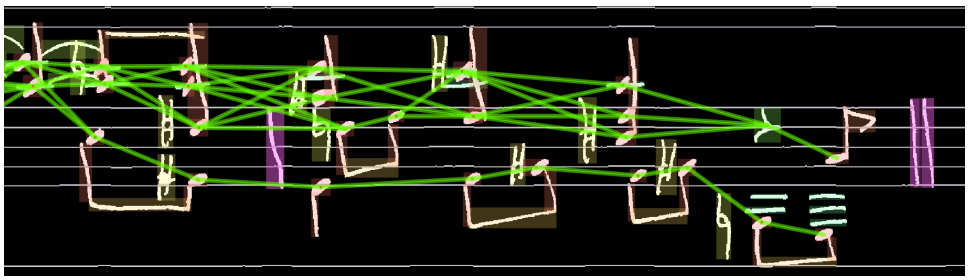
Figure A.7: A more complex beamed group situation in a 17th century violin manuscript (H. I. F. Biber, Mystery sonata IV).

elements, and especially in manuscripts, beamed groups are prime suspects where the topological constraints on printed music notation get violated. A moderately complex beamed group is depicted in Fig. A.6; a tricky real-life example from an early music manuscript is depicted in Fig. A.7

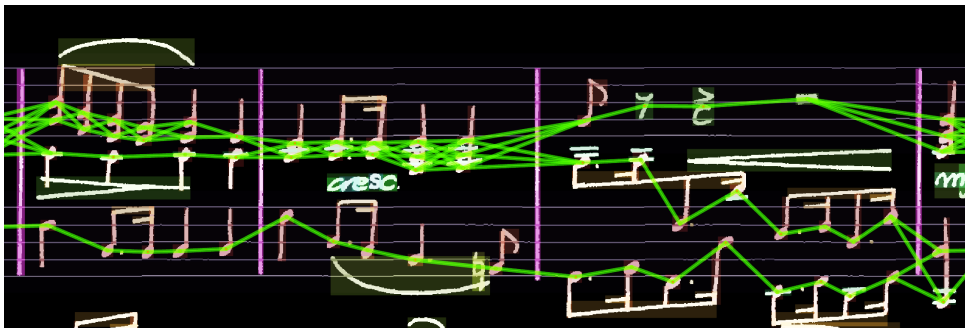
**Encoding Onset**

Onset is encoded by how notes are ordered by precedence and assigned to voices. Ordering notes by precedence within a voice is done simply by ordering the corresponding noteheads left-to-right within each staff (and ordering systems downwards). A simultaneity with more than one member within a voice is encoded by making all noteheads that correspond to its member notes share a single stem. (In case of whole notes, where no stem is used, the empty noteheads are either stacked on top of each other, or, if they lie on a neighboring staffline and staffspace, they are stacked very close to each other without the noteheads overlapping.)

In case of polyphonic music on a single staff, correctly finding the predecessor(s) of a note is a difficult problem to solve in general. When the number of voices in a staff is limited to two, most often the voices can be differentiated by



(a) Precedence in polyphonic guitar music: note the variable number of voices, some with chords.




(b) Precedence in complex pianoform music (green connecting lines, left to right). Notice the voice that is written across staffs.

Figure A.8: Examples of notation where precedence is complicated.

stem direction (rests are then usually positioned significantly above their usual position for the top voice, and below their usual position for the bottom voice); however, when more than two voices are present on a staff, this approach fails. One can partly rely on the heuristic that whenever voices cross (a voice that is typically below the other has a note above the one that is currently in the upper voice), stem directions are set to reflect this. However, resolving precedence in polyphonic music in general is an open problem; examples of complex situations are given in Fig. A.8. (Note also that in Fig. A.7, there is a rare situation where the noteheads that are part of one beamed group are not necessarily encoding each other's predecessor notes.)