

Mobile framework for recognition of musical characters

Rui Miguel Filipe da Silva

Mestrado em Multimédia da Universidade do Porto

Orientador: Ana Maria Silva Rebelo (Doutor)

Coorientador: Jaime dos Santos Cardoso (Professor Doutor)

Junho de 2013

© Rui Miguel Filipe da Silva, 2013

Mobile framework for recognition of musical characters

Rui Miguel Filipe da Silva

Mestrado em Multimédia da Universidade do Porto

Aprovado em provas públicas pelo Júri:

Presidente: José Miguel Santos Araújo Carvalhais Fonseca (Professor Doutor)

Vogal Externo: Luis Gustavo Pereira Marques Martins (Professor Doutor)

Orientador: Ana Maria Silva Rebelo (Doutor)

Resumo

Em termos de conhecimento humano, a música pode ser uma das poucas coisas que nos segue desde os tempos pré-históricos. Como forma de representação, a notação musical tem sido sempre a principal fonte de expressão musical para sistemas não auditivos. Programas análogos aos sistemas de reconhecimento de caracteres ópticos, chamados sistemas de reconhecimento ótico de música (OMR), têm estado sob intenso desenvolvimento há diversos anos. Nos dias de hoje, uma interessante aplicação do OMR consiste no reconhecimento *online*, onde permite uma conversão automática de texto assim que é escrito num aparelho digital especial. Tomando em consideração a actual proliferação de pequenos aparelhos electrónicos e com o aumento da potência de computação, tal como em tablets ou em smartphones, a exploração destes recursos aplicados ao OMR podem ser úteis de forma a ultrapassar as limitações do reconhecimento *offline*. A maior vantagem em usar sistemas *online* está relacionado com: eliminação de problemas de segmentação causados por símbolos sobrepostos e extra informação acerca do espaço temporal dos símbolos.

Esta dissertação representa um paço em frente no estado da arte na área de optical music recognition com as seguintes contribuições: Protótipo para capturar símbolos musicais de forma livre; construção de uma base de dados de utilizadores com diferente experiência musical; análise estatística dos símbolos desenhados à mão pelos utilizadores; estudo de um classificador usando informação dos desenhos; e estudo comparativo entre os métodos online e offline.

Abstract

In terms of human past knowledge, music may be one of the few things which we are certain that follow us since pre-historic times. As a way of representation, musical score notation has always been the main source of musical expression for non-hearing systems. Programs analogous to optical character recognition systems, called optical music recognition (OMR) systems, have been under intensive development from many years. Nowadays, an interesting application of OMR concerns the online recognition, which allows an automatic conversion of text as it is written on a special digital device. Taking into consideration the current proliferation of small electronic devices with increasing computation power, such as tablets and Smartphones, the exploration of such features applied to OMR may be useful in order to overcome the limitations of offline recognition. The main advantages in using online systems are related to: elimination of segmentation issues caused by overlapped symbols, and extra knowledge about spatial-temporal information of symbols.

This thesis represents a step forward on the state-of-the art of optical music recognition with the following contributions: A prototype for capturing gesture-free of musical symbols; the construction of a database collected for a set of users with different musical expertise; statistical analyze of the hand-drawn symbols from the users; study of an classifier using information from the user strokes; and a comparative study between the online and the offline methods.

Acknowledgments

I would like to first express my gratitude to my advisor Dr. Ana Rebelo for her continuous support, help, motivation, patience and advice through the all process of the thesis. I honestly believe that I could not reach nearly as half as I did, if I couldn't count with her support.

I would also like to thank all the members of VCMi group, for receiving me with open arms, and giving their constant support and wisdom. A thank to Professor Jaime Cardoso for his advice and expertise in some very important aspects during this project.

A thanks to all my dear friends from the masters who also fought this battle with me, since the day one, who filled me throughout these months with great memories to look back to years to come.

An obvious thank to all the people who helped in the process of filling the database in order to have something to show today.

Finally, I would like to thank my fathers for their constant support, affection and encouragement since the day I was born. And all the people that I forgot to mentioned but that should also be here.

If it were not for you all I would certainly not be here today. Thank you!

Rui Miguel Filipe da Silva

Contents

Introduction	1
1.1 OMR: Offline versus Online	2
1.2 OMR System Project.....	2
1.3 Research Objectives	3
1.4 Contributions and Related Publications	3
1.5 Technological Description	4
1.6 Thesis Structure.....	4
State-of-the-Art	5
2.1 OMR Revision	5
2.2 Online Overview	6
2.3 Music Notation.....	7
Creation of the Database	9
3.1 Application development	9
3.1.1 Main Menu	9
3.1.2 New User / Update User.....	10
3.1.3 Draw Symbols	10
3.2 Database	12
3.2.1 Description	12
3.2.2 Statistical Study.....	14
3.2.2.1 Users	14
3.2.2.2 Draws	15
3.3 Editing User Draws from the database.....	34
Background Knowledge.....	37
4.1 Technological Description	37
4.1.1 Neural Networks Overview.....	37
4.1.2 HBF49	40
4.1.3 AvgRF	41
Experimental Testing.....	43
5.1 Symbol Classification	43

5.1.1	Dataset Creation	43
5.1.1.1	Online	43
5.1.1.2	Offline	45
5.1.2	Neural network creation	46
5.1.2.1	Online	46
5.1.2.2	Offline	47
5.2	Results	47
5.2.1	Online – All classes	48
5.2.2	Online – Sub classes	48
5.3	Comparative Study	49
Conclusion.....		51
6.1	Conclusion.....	51
6.2	Future Work	53
References		55
Papers		58
8.1	HB49 Features.....	58
8.1.1	Starting and ending points position	58
8.1.2	First point to last point vector	58
8.1.3	Closure	58
8.1.4	Angle of initial vector	58
8.1.5	Inflexions.....	59
8.1.6	Proportion of downstroke trajectory	59
8.1.7	Number of strokes	59
8.1.8	Bounding box diagonal angle:	59
8.1.9	Trajectory length	59
8.1.10	Deviation	59
8.1.11	Average direction	59
8.1.12	k-Perpendicularity, k-angle	59
8.1.13	Absolute angle histogram	60
8.1.14	2D histograma	60
8.1.15	Hu moments	60
8.1.16	Convex hull features	60
8.2	Online Database of Hand-draw Musical Symbols	60

List of Figures

Figure 1 - Example of variation of notes	8
Figure 2 - Main Menu image snapshot	10
Figure 3 - Drawing window image snapshot	11
Figure 4 - Finger movement	13
Figure 5 - G clef class and sub class	13
Figure 6 - Graphic of the different countries	14
Figure 7 - Understanding of music by the user	15
Figure 8 - Drawing with “garbage”	34
Figure 9 – Garbage examples	35
Figure 10 - Histogram of C1 (letter ‘a’) and C2 (letter ‘b’). [NNBishop]	38
Figure 11 - Separation between C1(letter 'a') and C2 (letter 'b'). [NNBishop]	38
Figure 12 - Function example of a neural network	40
Figure 13- Creation of the matrix	44
Figure 14 - Beamed notes separation	44
Figure 15 - Example of the NN methods: a – cNN, b – mixNN [Jaime2005]	46

List of Tables

Table 1 - Music experience by sex	14
Table 2 - list of symbols and their characteristics	17
Table 3 - Files replaced	35
Table 4 - Estimative rates results	48
Table 5 - Results of the performance from the sub-classes	48

Glossary

ANN	Artificial Neural Network
CD-CMG	Context-driven Constraint Multiset Grammars
HMM	Hidden Markov Model
JPEG	Joint Photographic Experts Group
MLP	Multi layer perceptron
NN	Neural Network
OCR	Optical Character Recognition
OMR	Optical Music Recognition
PDA	Personal digital assistant
PNG	Portable Network Graphics
SVM	Support Vector Machine

Chapter 1

Introduction

In terms of human past knowledge, music may be one of the few things which we are certain that follow us since pre-historic times.

Over the years, music has increased its importance around the globe, reaching people in very different ways (cultural, religious, educational, and so forth), being an important part of their lives. In fact, there is no record of any society who does not have a musical culture of his own, and its own way to express it, despite all cultural and social known divergences.

Since very early, people had the need to express themselves in different ways in music and for that, save those expressions in a more liable source besides their brain.

Musical score notation has always been the main source of musical expression for non-hearing systems. It is a worldwide standard representation of music writing, grouped with a lot of different symbols and ways to represent a very diversity of sound representations.

Being a language that has suffered a lot of changes across the years, it needed to be adjusting to the civilization needs, and to the means that were possibly in the time being.

Technology has grown exponentially along the years, and writing score pieces in paper, start to be a secondary option when you have a large fan of possibilities in a computer, such as: editable scores, perfectly formatted and archived, immediate sound response and so forth. This led to many composers to start use computer as their main way to compose music.

But it is a fact that most of the composers, in an age of global technology, still use the traditional “pen and paper” to write his pieces, saying that it still is the most intuitive, easier and faster way to do so [S Macé, 2005].

Scanning process it is in fact one of the most used ways to preserve the musical culture into a computer, where we achieve ease duplication, distribution and digital processing. But a system of recognition is imperative if we want use tools like search, recover or analysis.

A way to change this mood, and still benefits from the traditional computer functionalities, is using OMR (or Optical Music Recognition, Music OCR). OMR is the application of optical

character recognition to interpret sheet music or printed scores into editable or playable form, but dealing with user hand-drawn strokes in the context of document it became a complex problem, and yet to be solved [ACapela2008,AREbelo2009, Cardoso2009a,Cardoso2009b].

1.1 OMR: Offline versus Online

The main objectives of an OMR system are the recognition, the representation and the storage of musical scores in a machine-readable format. An OMR program should thus be able to recognize the musical content and make the semantic analysis of each musical symbol of a music work. In the end, all the musical information should be saved in an output format that is easily readable by a computer. An OMR system can be divided according to the input data into two categories: offline and online. One of the prominent differences is related to the type of music notation: online recognition mandatorily deals with handwritten music works. This imposes a high level of complexity in the process due to the wider variability of the objects. Besides, while an offline OMR system recognizes the symbols in a digitized music score, usually, obtained with a scanner, an online OMR system recognizes the symbols almost simultaneously when they are introduced in the system by a pen-based interface, being a real time process. The main advantages in these kinds of systems are related to: elimination of segmentation issues caused by overlapped symbols, and extra knowledge about spatial-temporal information of symbols.

This work proposes to conjugate the universal “pen-paper” metaphor with the news forms of mobile technology, as we see with the growing popularity of portable small devices, such smartphones and tablets, and with their power of computing. As pen-based interfaces are in wide expansion, there is a lack of applications taking advantage of this intuitive and ergonomic way to draw musical scores, where the user composes musical scores in a traditional way by drawing the symbols on the screen. The user benefits from the traditional computer functionalities and also avoids the loss of quality and neatness of the paper document.

It may be useful to explore these characteristics in OMR, with the purpose to overcome the limitations of the *offline* mode.

1.2 OMR System Project

The project Automatic Recognition of Handwritten Music Scores (PTDC/EIA/71225/2006) initiated in 2007 by Instituto de Engenharia de Sistemas e Computadores do Porto and Escola Superior de Música e das Artes do Espectáculo was the starting point for creating an OMR system that addresses some of the common problems in this area with already several published articles (e.g. [ACapela2008, AREbelo2009, Cardoso2009a,Cardoso2009b]). Nowadays, an interesting application of OMR concerns the

online recognition, which allows an automatic conversion of text as it is written on a special digital device. Taking into consideration the current proliferation of small electronic devices with increasing computation power, such as tablets and Smartphones, the exploration of such features applied to OMR may be useful in order to overcome the limitations of offline recognition.

This thesis is part of a longer project that intends to investigate and develop algorithms to recognize musical characters in real time to obtain a digital, easy-to-manage version of the original scores. The creation of a tool capable of recognizing music notation anywhere, including in rehearsals and while composers, conductors or musicians are on tour, will be very important and significant in the advancement of technology.

The main factors that motivate the research on this area are: (1) the use of both local and global information in the recognition procedure in order to assign to the system all the knowledge needed to perform at the end a semantic correction; (2) the non-predefined pattern technique giving total freedom of writing to the user; and (3) the useful investigation with high commercialization potential.

1.3 Research Objectives

This thesis has the aim to create an OMR software where users can create music sheets in a natural way. This interface should be capable of recognizing an unlimited number of calligraphies. In this manner it will encompass detection and recognition of handwritten musical symbols in dynamic context. Hence, it aims to overcome the inherent problems of this type of writing using the most recent techniques of machine learning and artificial intelligence.

1.4 Contributions and Related Publications

This thesis represents a step forward on the state-of-the art of optical music recognition with the following contributions:

1. A prototype for capturing gesture-free of musical symbols;
2. The construction of a database collected for a set of users with different musical expertise;
3. Statistical analyze of the hand-drawn symbols from the users
4. Study of an classifier using information from the user strokes
5. And, a comparative study between the online and the offline methods

The work related to this thesis already resulted in the publication of the following paper: “Online database of hand-draw musical symbols”, Rui Silva, Jaime S. Cardoso, and Ana

Rebello, in 2nd PhD. Students Conference in Electrical and Computer Engineering – StudECE 2013 [RuiSilva2013].

1.5 Technological Description

All prototype systems are programed in Android developer language, therefore, all methods using to draw paths, create or save data, will be using methods specific from android. The Code is programed to run to an android system equal or above 2.3.3 (API 10 - Gingerbread), which means 95% of the market [APIandroid].

In Matlab we will analyze and edit the data from the database, and develop algorithms to serve as base of recognition.

Other important point is using fan of types of classifiers to implement in programmable system. We will use a set of methods to segment the data and implement into an ANN (Artificial Neural Network) to segment and validate in order to create the classifiers. This will be explained with detail along the thesis.

1.6 Thesis Structure

This thesis is organized in 6 chapters. After the Introduction, the description of what has been done along the years in OMR field, with more focus on the online methods, will be made in Chapter 2.

In Chapter 3 it is made a description of the application done to record hand draw symbols, and we also take a closer look to the database, describing the process of creation, record, and edition. The technology being use and its importance in this work will be explained in Chapter 4.

The Chapter 5 start to describe the process of getting the data from the users draws, following by describing the neural network, create the classifiers and show the experimental results with a comparative study; both for online and offline mode.

This thesis finishes with Chapter 6 where conclusions are presented and future work is discussed

Capítulo 2

State-of-the-Art

In this section we will describe what has been done along the years in the OMR field, mainly on the online recognition process. We also explained a little of music notation.

2.1 OMR Revision

For centuries, music has been shared and remembered by two traditions: aural transmission and in the form of written documents, normally called musical scores. Programs analogous to optical character recognition systems are called optical music recognition (OMR) systems and have been under intensive development for many years. However, the results to date are far from ideal. Each of the proposed methods emphasizes different properties and therefore makes it difficult to effectively evaluate its competitive advantages [Rebelo2012].

The research field of OMR began with [Pruslin66] and [Prerau1970] and has undergone much important advancements since then.

It have been presented to the scientific community some surveys and summaries: [George2004] made a description of the existing methods for online handwritten input of music notation; [Jones2008] presented a study in music image, which included digitalization, recognition and restoration, with a well detailed list of hardware and software in OMR together with an evaluation of three OMR systems; [Arebelo2009] presented a pattern recognition study applied to music notation; and in [Rebelo2012] a complete overview of the literature concerning the automatic analysis of printed and handwritten musical scores was provided.

Most of the advance commercial products including *Notescan* in *Nightingale*, *Midiscan* in *Finale*, *Photoscore* in *Sibelius*, among others, cannot identify all the musical symbols. But

still, these products have their main focus on recognition of impress score music, producing good results for them, but not with handwriting scores.

The main objectives of an OMR system are the recognition, the representation and the storage of musical scores in a machine-readable format. An OMR system can be divided according to the input data into two categories: offline and online.

One of the main differences is related to the type of music notation: online recognition mandatorily deals with handwritten music works. While an offline OMR system recognizes the symbols into a digitized music score, and online OMR system recognizes the symbol almost simultaneously when they are introduced into the system by a pen-based interface, being a real time process.

2.2 Online Overview

A method for data input in a digital system through a digital pen applied to musical software appeared at the end of the 1990's with the [PalmPilot1997] produced by Pal Inc. The pilot was small handheld PDA where musical characters were inserted through a pen using a technique called Graffiti [GraffitiPalmOS]. The first approach to the problem was to treat a digital pen as an interface of localization and selection, in place of the mouse, as exemplified by the work developed by [Silberger1996]. Thereafter, the input data process relied on the concept of standard movements. This method depends on the learning of particular movements by the user, each one corresponding to a different music symbol [Anstice96, Forsberg98, Miyao2007]. However, the question is if this type of interface will be in fact of musical notation, since all the methods completely avoid the problem of free written style, restricting the issue to a creation of a simplified symbolic musical convention.

Most of the systems were developed to work with only 1 stroke (*unistroke*), forcing some of them to have its own dictionary of symbols: Presto [Jamie96], a system that makes it possible to realize musical scores faster than in the usual way, with a set of gestures designed to be learned easily; The Music Notepad [Forsberg98], that allowed the user to edit music with a mix of various pen gestures; Macè et al [Smace2005] constrained the pen-input to some predefined areas with reference to the note head to recognize the symbols, proposing a set of gestures that is almost as the usual ones, not being a completely unistroke system.

The main goal of an online OMR system should be the full capacity of the digital universe to recognize and interpret the dynamic handwriting, regardless of various calligraphies and strategies of writing, without restricting the creative process of the composer. The works developed by [George2004, Taubman2005] were the first to deal with the ordinary music notation in the problem of online recognition. [George2004] proposed a non-parametric statistical method for pattern recognition (ANN – Artificial Neural Networks) to guide the

identification and classification of the musical symbols, but incapable of recognize new objects. Taubman in 2005 present a system called MusicHand[Taubman2005], aiming to help in the process of pitch and symbol segmentation, where the system upon entry of an unknown stroke, make the decision by comparison with a set of trained gestures, without hesitation or coaxing the system. Neither of these uses any knowledge about music rules to help the recognition process. In this manner, our proposed methodology aims to overcome this limitation.

Other authors as [Mitobe2004, Lee2010] used Hidden Markov Models (HMMs) to classify handwritten music symbols. Mitobe et al [Mitobe2004] constructed abstract strokes that users must learn, whereas Lee et al [Lee2010] proposed a non-gestural process.

In [Macé2009], it is suggested that is necessary to exploit dedicated recognizers, because it is not possible to use a unique recognizer for all symbols that a musical score can contain, so that its recognizers be able to interpret subsets of this symbols. Sébastien Macé and Eric Anquetil in [Macé2012], propose a different formalism, which they focus on the interpretation of online hand-drawn structured documents, i.e., documents that have predefined structure (musical scores for instance). Designing a new class of visual grammars, called *context-driven constraint multiset grammars* (CD-CMG) and exploring both structural and statistical recognition approaches, they take the imprecise nature of hand-drawing into account

In overview, most of the documentation uses a unistroke (gestural approach, just one stroke) system, forcing the user to learn a new language, being an influence on the process of composition. On the existing *multistroke* (non-gestural approach, more than one stroke) systems, there is a lack of important values that are not collected from a specific draw and we could benefit with some new features that could exploit more from the dynamic user handwriting. Also, most of the systems do not use a full (or almost full) dictionary of symbols, confining the user to the most commons ones.

2.3 Music Notation

Music score notation is a representation system based on hearing perception through the use of symbols. This allows an interpreter to execute a specific music in a determine way; define by the composer or person who made the arrangement.

Music has come from several years ago, and exist diverse types of music notations, since Old Greece to Byzantine Empire going to the Arabia world and the beginnings of the Europe. Now a days, the more used notation had origin from the Europe classical music, more properly the Western graphic system [Williams1903] The modern notation system is based on using symbols above a score of 5 lines readied from the left to the right and above to low. Following the vertical positions in relation with the lines, it is define the standard frequency of the note (C

to B note), and the duration is defined by the shape of the symbol being used, we can see and example in figure 1.

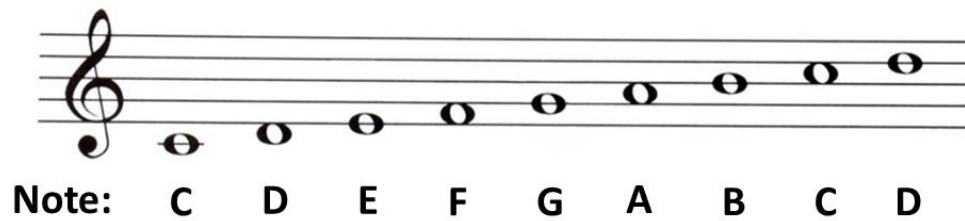


Figure 1 - Example of variation of notes

This system besides allowing giving characteristics of duration and note, also allows representing a diverse set of others characteristics like: intensity, expression or even specific techniques of instrumental execution.

Our focus in this project is in symbol classification, and for that we choose a set of symbols to be used along the project, that we think are the most important and could represent the main composition fundamentals: pitch, rhythm, tempo and articulation. The symbols encompasses: lines, clefs, notes, rests, breaks, accidentals, time signatures, note relationships, dynamics, articulation marks, ornaments, repetition and codas [Heussenstamm].

Capítulo 3

Creation of the Database

To obtain a robust online recognition system we need in the first place to set-up a dataset that can cover as much as possible the different handwritings from different people. This process exists in order to have a set of different hand draws, enabling the system to recognize a large fan of different handwrite draws.

Towards this goal we developed an android-based application and in this chapter we will describe thoroughly the individual modules of the prototype that resulted from this thesis.

3.1 Application development

The application made in android, has 3 main forms: Main Menu, New User and Draw Symbols. These forms intent to create and save data of the user hand draw and his personal information, we will next describe each one.

3.1.1 Main Menu

In the Main Menu of the system is where all users are presented for order of creation. For common a user, there are two options in this menu: the user can make a registration for him, clicking in the “New User” button, or he can continue the point where he left off by clicking on his name. This last option enables a user to finish his drawing in other time, and it is also important for safeguard, in a case of an unexpected shutdown of the application. If a user has already drawn all of his symbols, the system says that the user has reached the end of his process. In this panel it is also where the administrator can reach the edit form to the personal information of any user by clicking on his name and then confirming with a specific password. We can see an image snapshot of the menu in figure 2.

Mobile framework for recognition of musical characters

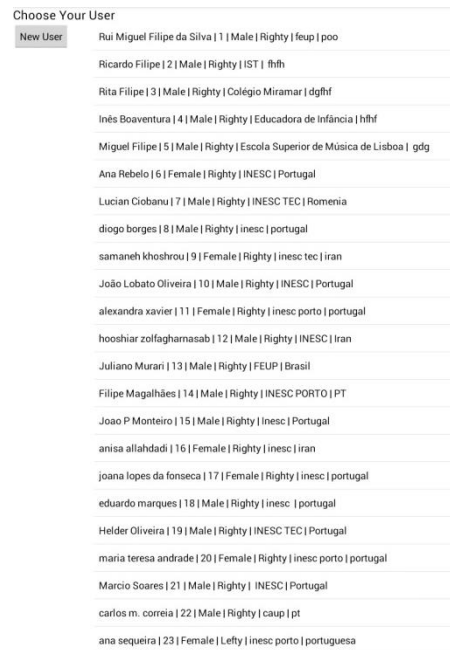


Figure 2 - Main Menu image snapshot

3.1.2 New User / Update User

Having a system with a user's hand draw, it is convenient to save some of the personal information of that user. In this section the user has to fill all of the fields, with is personal information, in order to advantage to the next stage. In case of any missing field, the systems ask to fill the remaining ones. If the point was to Update User, the fields are automatic filled with the information of the user being edited, and the administrator has the possibility to change what he wants and save the new information.

The fields will be explained with more detail in the Database section.

3.1.3 Draw Symbols

Most of the important things of the application happen in this section. The intention here is that the user looks at the symbol shown above and tries to copy it below with his own hand writing. Besides the representative figure of the symbol, it is also shown the family and the name of it, in case of doubts or just curiosity.

When the user finish his draw and clicks on the “Save” button to record the information of the symbol, the system shows another one and clear the space for the user the draw the new symbol. This process repeats until the users reach the end and draws all the symbols.

The symbol to draw, the description and even the background of the draw panel, change according to the symbol being drawn.

While the user is drawing, the system records important information about the path made, that is also going to be more explain in the Database section, and when he saves the draw, the system records that information into the database

It is a fact that when we give a pen to the user with different thickness, he draws his symbols in a different way, according to the thickness of pen. So, for each user we define a set of 3 sizes of stroke, to get 3 different types of handwritings, for each symbol. The order of the thickness is not always the same for every user; it changes depending on his id, avoiding distorted data due to the fatigue, hurry, experience or speed.

The button “Delete” is used to erase the draw when the user is not happy with it and want to re-draw it again; and button “Back” is used to go to the previous draw, where the system re-draw the symbol made by the user, and let him make changes if he want. We can see a snapshot of the window Draw Symbols in figure 3.

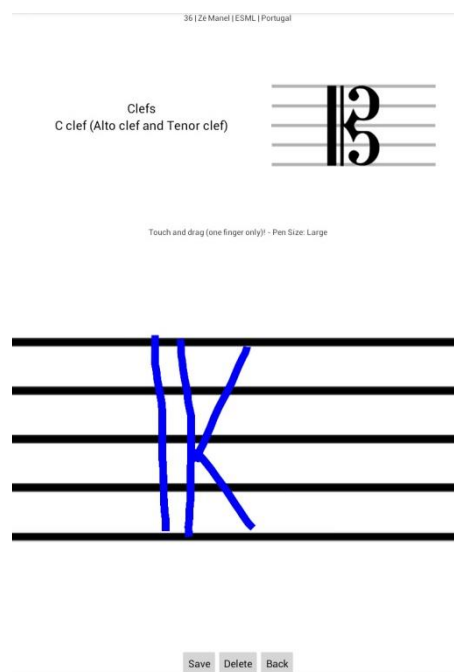


Figure 3 - Drawing window image snapshot

3.2 Database

3.2.1 Description

In an initial phase, the database was created using SQLite API, recommended by the Android documentation, being ideal for structured data. But it became completely inadvisable for our purpose when the time to record the data information into the database was between 30 and 180 seconds for every draw (depending on the size of the draw), which in a user perspective way, it is too much time to dispense. Another disadvantage was the problem of division of works; when we needed to uninstall the application, the database was also deleted together with all the recorded data. We also needed a simpler way to use and edit the database information than the SQLite system. For that we decide to save the data files into a txt file, due its portability, easy writing, universal recognition, and above all, its speed of reading and writing.

The files were saved into the internal memory of the device, separated by folders according to the users. The “Users” file has all the information from every user, that is: id, name of the user, date of registration, reference/institution/school, hand preference (right or left), age, country, sex (male or female) and sheet music knowledge (1 to 5).

Every folder is defined by the id of the user. Inside every folder there is all the symbols made by that user and also some additional information about the draws: the size of the stroke used, date of the drawing done, the hardware model of the machine and the resolution being used.

Every user needed to draw 252 symbols, repeating 3 times a sequence of 84 symbols, with some changes in some symbols (for example, with the leg up or down) and 3 different thicknesses. For every draw we collect a stream of points in the draw, and with each one of them we define 5 parameters:

- Action - if the user touch the screen he executes a down action, which we represent as 0; if he drags his finger he executes a move action, which we represent as 1; and if he lift his finger he executes an up action, which we represent as 2;
- X axis – position X of the point, 1 to the maximum X position of the device
- Y axis – position Y of the point, 1 to the maximum Y position of the device
- Pressure – pressure made by the user on the device; this measure is not very accurate, but generally ranges from 0 (no pressure at all) to 1(normal pressure), although values higher than 1 may be generated depending on the calibration of the input device

Mobile framework for recognition of musical characters

- Time – retrieve the time that the event occurred in Android time base “uptimeMillis()”; this event returns the milliseconds of non-sleep uptime since boot.

When exists an action of “Up” or “Down”, the android device recognize immediately that information, but when exists a type of “Move”, we can only get a certain number of points of that movement, what constitutes a consistent stream varies on the type of device. So for every type of device we can get a different set of pointers of the stream.

If a user only draw a single stroke, the database will be formed with one 0, a set of 1s and one 2; therefore if he draw a symbol with more than 1 stroke, the file will have more than one set of “0/1/2”, as we can understand by the representative figure 4.

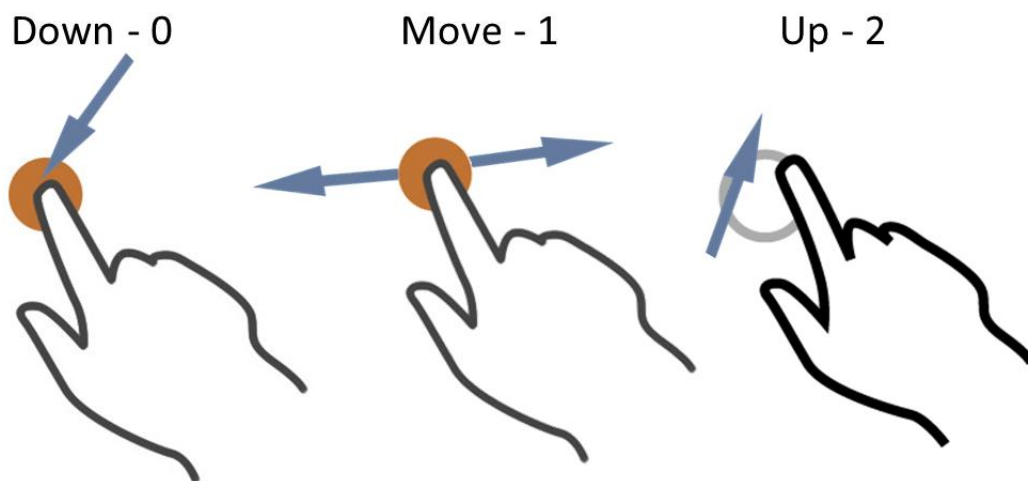


Figure 4 - Finger movement

With this information, besides the fact that we can re-write the symbol in the future, we can get a structural and temporal context of every draw.

We define a set of 20 classes representing all of the symbols, according to their meaning. In the classes with more than 1 symbol, we can define a set of subclasses to find a specific symbol into that class. So if we want to find the G clef, we would need to enter the class 4 and the subclass 1 to identify that symbol, as we can see in figure 5. In table 2, we show all of the symbol classes and subclasses.

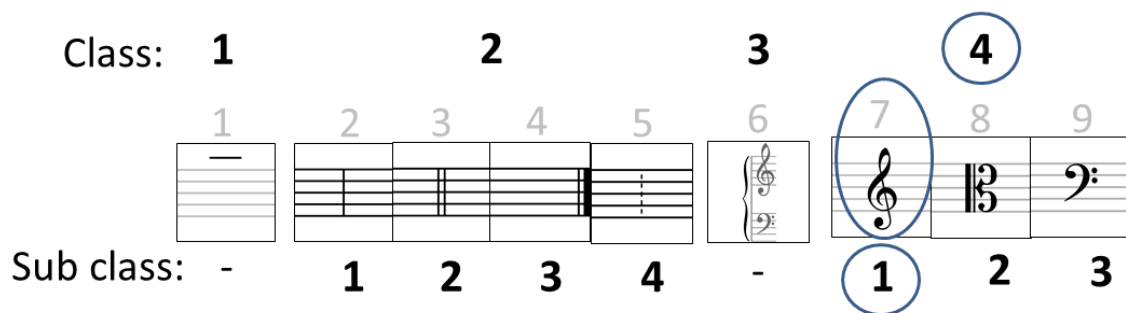


Figure 5 - G clef class and sub class

3.2.2 Statistical Study

3.2.2.1 Users

With information collected, we analyze the data and came with several interesting points.

We collect data from 50 users with ages between 13 and 70 from different countries and different cultures, with an average age of 30 years old.

We have a high 96% of right handed users in the database and only a 4 % of left handed.

We have 45 persons from Portugal, and 1 from Iran, Italia, Greece, Argentina and Spain, as we can see in the graphic with the figure 6.

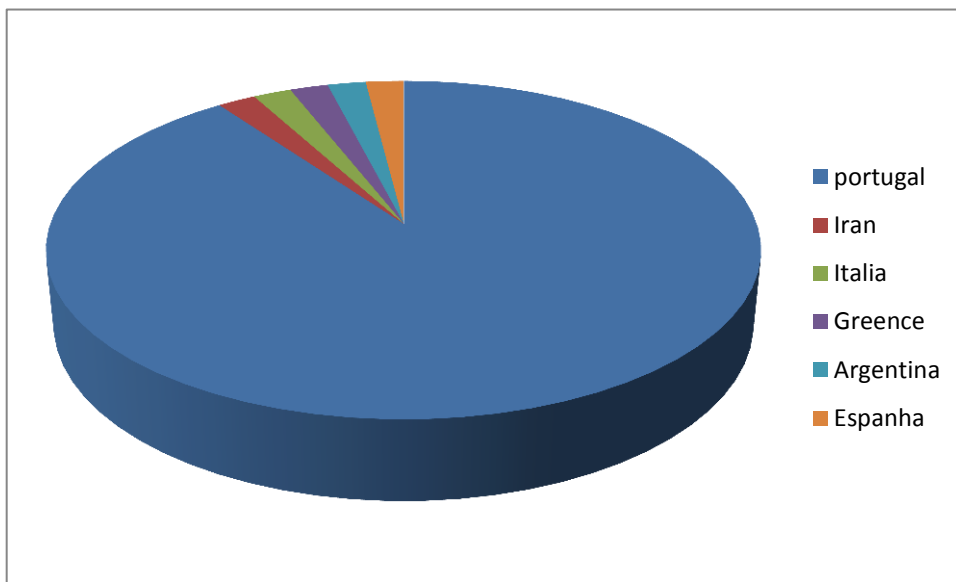


Figure 6 - Graphic of the different countries

In the database 34 people are men and 16 are women, a rate of 68% to 32%. From all people, we separate the people that understand music with 4 or 5, a user who understand more or less with 3 and a user who understand very little or nothing with 1 or 2. Based on these factors we saw that 22 users understand music, 18 do not and 10 understand more or less, as we can see in the figure 7. We also saw the number of men and women with every different experience, as we can see in the table 1.

Table 1 - Music experience by sex

Music Experience	1		2		3		4		5	
Men	0	0%	9	56%	8	80%	9	75%	8	80%
Women	2	100%	7	44%	2	20%	3	25%	2	20%

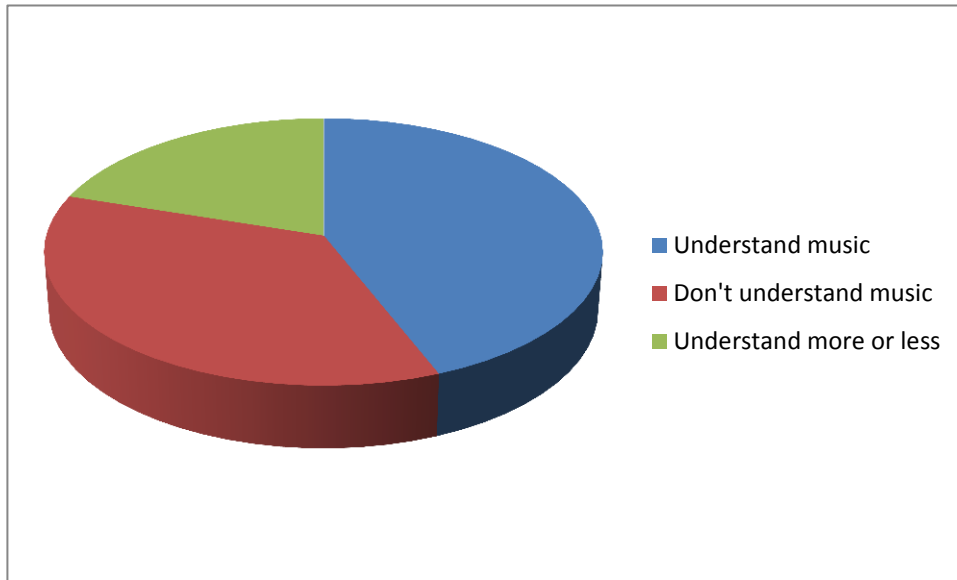


Figure 7 - Understanding of music by the user

3.2.2.2 Draws

The drawings made by the users also have some useful information that we can explore.

All the drawing process for each user took between 17 and 90 minutes, depending on the user, but with an average time of 30 minutes.

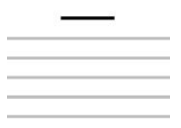
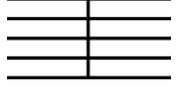
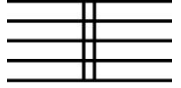
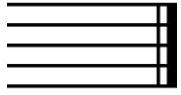
We also collect some interesting information like:

- The smallest draw has only 5 lines
- The biggest draw has 997 lines with 5 strokes
- The draw with more strokes has 64 strokes

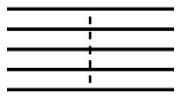




In the table 2 we can see the symbols being used in the database, where we show the standard image of the symbol, the name of it, as also its class and sub-class. We check the *Lowest Stroke* for a specific class, comparing all the users' draws, as well for the *Biggest Strokes*, *Average Strokes*, *Lowest Lines*, *Biggest Lines*, *Average Lines*, *Lowest Time*, *Biggest Time*, and *Average Time*, for all the symbols. Understanding the *Strokes* as a number of times that the user clicks and lifts his finger, the *Lines* as the number of lines saved in the draw file and the *Time* as the duration of the drawing process. The *Biggest Time* could be influenced by pauses or distractions.

Mobile framework for recognition of musical characters






Table 2 - list of symbols and their characteristics

Symbo l	Image	Name	Class	Sub- class	Lowest Strokes	Biggest Strokes	Average Strokes	Lowest Lines	Biggest Line	Average Lines	Lowest Time	Biggest Time	Average Time
1		Ledger or Leger lines	1	-	1	2	1	12	180	30	184	3363	581
2		Bar Lines	2	1	1	2	1	13	181	45	179	5623	875
3		Double bar line	2	2	2	4	2	26	179	75	463	4973	1864
4		Bold double barline	2	3	1	28	4	33	922	253	606	49875	5951






Mobile framework for recognition of musical characters

Symbol	Image	Name	Class	Sub-class	Lowest Strokes	Biggest Strokes	Average Strokes	Lowest Lines	Biggest Line	Average Lines	Lowest Time	Biggest Time	Average Time
5		Dotted bar line, Dotted barline	2	4	3	9	5	20	144	51	458	10578	2461
6		Accolade, brace	3	-	1	2	1	29	204	84	470	3774	1558
7		G clef	4	1	1	3	1	45	511	149	714	8679	2583
8		C clef	4	2	1	21	4	56	882	328	1216	45175	9021
9		F clef	4	3	3	13	4	40	457	124	1091	15456	4049






Mobile framework for recognition of musical characters

Symbol	Image	Name	Class	Sub-class	Lowest Strokes	Biggest Strokes	Average Strokes	Lowest Lines	Biggest Line	Average Lines	Lowest Time	Biggest Time	Average Time
10		Breve	5	1	3	9	5	50	418	130	1059	20098	4161
11		Semibreve	5	2	<u>1</u>	<u>12</u>	1	23	275	55	383	6805	959
12		Minim	5	3	1	4	2	28	227	79	468	6519	1765
13		Crotchet	5	4	1	32	2	29	654	195	459	19617	3925
14		Quaver	5	5	1	44	3	38	922	207	615	30846	4772






Mobile framework for recognition of musical characters

Symbol	Image	Name	Class	Sub-class	Lowest Strokes	Biggest Strokes	Average Strokes	Lowest Lines	Biggest Line	Average Lines	Lowest Time	Biggest Time	Average Time
15		Semiquaver	5	6	2	42	4	49	743	213	1025	31495	5124
16		Demisemiquaver	5	7	3	29	5	58	864	235	1252	30633	6121
17		Hemidemisemiquaver	5	8	4	30	6	66	942	270	1606	34575	7276
18		Rest Breve	6	1	1	64	4	68	799	290	1238	25634	6254
19		Rest Semibreve	6	2	1	30	3	20	479	164	424	18441	3762

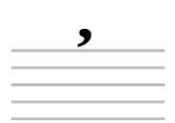




Mobile framework for recognition of musical characters

Symbol	Image	Name	Class	Sub-class	Lowest Strokes	Biggest Strokes	Average Strokes	Lowest Lines	Biggest Line	Average Lines	Lowest Time	Biggest Time	Average Time
20		Rest Minim	6	3	1	28	3	18	595	176	346	14738	4065
21		Rest Crotchet	6	4	1	7	1	20	420	100	333	14956	2471
22		Rest Quaver	6	5	1	6	1	21	473	104	330	11860	2251
23		Rest Semiquaver	6	6	2	10	3	30	479	162	658	17085	4074
24		Rest Demisemiquaver	6	7	3	16	4	38	717	226	883	20859	5675


Mobile framework for recognition of musical characters

Symbol	Image	Name	Class	Sub-class	Lowest Strokes	Biggest Strokes	Average Strokes	Lowest Lines	Biggest Line	Average Lines	Lowest Time	Biggest Time	Average Time
25		Rest Hemidemisemiquaver r	6	8	4	20	6	42	940	286	1130	31447	7339
26		Beamed notes	7	1	4	6	4	59	835	174	1420	32631	5227
27		Beamed notes	7	2	4	6	4	47	638	106	1426	16967	3721
28		Beamed notes	7	3	4	6	4	50	491	109	1259	41533	3815
29		Dotted note	8	-	1	5	1	5	147	36	82	5038	892

Mobile framework for recognition of musical characters

Symbol	Image	Name	Class	Sub-class	Lowest Strokes	Biggest Strokes	Average Strokes	Lowest Lines	Biggest Line	Average Lines	Lowest Time	Biggest Time	Average Time
30		Breath mark	9	1	1	13	1	8	292	62	120	8935	1476
31		Caesura	9	2	1	7	2	19	171	37	341	3831	896
32		Flat	10	1	1	3	1	25	267	59	384	6043	1313
33		Sharp	10	2	4	7	4	41	222	67	841	8626	1938
34		Natural	10	3	1	12	3	32	309	77	727	11918	2139






Mobile framework for recognition of musical characters

Symbo l	Image	Name	Class	Sub- class	Lowest Strokes	Biggest Strokes	Average Strokes	Lowest Lines	Biggest Line	Average Lines	Lowest Time	Biggest Time	Average Time
35		Double sharp	10	4	1	14	2	17	362	52	292	11526	1501
36	<i>0</i>	Number 0	11	1	1	2	1	21	143	47	328	3939	817
37	<i>1</i>	Number 1	11	2	1	3	1	15	108	41	267	2990	874
38	<i>2</i>	Number 2	11	3	1	1	1	18	124	46	274	2067	775
39	<i>3</i>	Number 3	11	4	1	1	1	23	133	48	371	2510	808






Mobile framework for recognition of musical characters

Symbo l	Image	Name	Class	Sub- class	Lowest Strokes	Biggest Strokes	Average Strokes	Lowest Lines	Biggest Line	Average Lines	Lowest Time	Biggest Time	Average Time
40	4	Number 4	11	5	1	4	2	16	165	49	264	4313	1008
41	5	Number 5	11	6	1	3	2	27	144	53	431	3596	1054
42	6	Number 6	11	7	1	2	1	22	108	42	349	1894	701
43	7	Number 7	11	8	1	2	2	18	91	39	266	3399	857
44	8	Number 8	11	9	1	3	1	25	135	60	404	2626	1058



Mobile framework for recognition of musical characters

Symbol	Image	Name	Class	Sub-class	Lowest Strokes	Biggest Strokes	Average Strokes	Lowest Lines	Biggest Line	Average Lines	Lowest Time	Biggest Time	Average Time
45		Number 9	11	10	1	2	1	20	111	50	313	2081	870
46		Common time	12	1	1	7	1	17	458	72	257	9223	1357
47		Alla breve or Cut time	12	2	2	4	2	25	365	90	530	8250	2048
48		Equal sign	13	-	2	4	2	13	79	27	263	3509	668
49		Tie	14	1	1	2	1	16	80	39	265	1726	699






Mobile framework for recognition of musical characters

Symbol	Image	Name	Class	Sub-class	Lowest Strokes	Biggest Strokes	Average Strokes	Lowest Lines	Biggest Line	Average Lines	Lowest Time	Biggest Time	Average Time
50		Glissando or Portamento	14	2	1	2	1	17	458	79	280	10856	1438
51		Tuplet	15	-	1	4	2	32	189	83	508	6490	1950
52		Arpeggiated chord	16	-	1	2	1	15	507	81	224	11987	1542
53		Piano	17	1	1	3	2	21	312	63	361	7649	1373
54		Mezzo piano	17	2	1	7	3	53	376	120	924	12819	2687






Mobile framework for recognition of musical characters

Symbo l	Image	Name	Class	Sub- class	Lowest Strokes	Biggest Strokes	Average Strokes	Lowest Lines	Biggest Line	Average Lines	Lowest Time	Biggest Time	Average Time
55	<i>mf</i>	Mezzo forte	17	3	1	8	3	52	442	115	1082	13751	2672
56	<i>f</i>	Forte	17	4	1	3	3	23	340	54	444	8296	1120
57	<i>sfz</i>	Sforzando	17	5	1	9	4	60	633	126	1346	16643	3406
58		Crescendo	18	1	1	2	1	17	83	37	266	1573	681
59		Diminuendo	18	2	1	2	1	17	70	36	255	1883	647

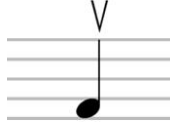




Mobile framework for recognition of musical characters

Symbol	Image	Name	Class	Sub-class	Lowest Strokes	Biggest Strokes	Average Strokes	Lowest Lines	Biggest Line	Average Lines	Lowest Time	Biggest Time	Average Time
60		Forte-piano	17	6	1	7	3	45	396	103	890	16042	2673
61		Staccato	19	1	1	4	1	5	157	38	100	4298	903
62		Staccatissimo or Spiccato	19	2	1	15	3	6	492	99	62	10577	2587
63		Accent	19	3	1	2	1	14	61	28	212	1455	530
64		Tenuto	19	4	1	1	1	7	35	16	77	712	300



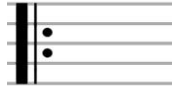


Mobile framework for recognition of musical characters

Symbol	Image	Name	Class	Sub-class	Lowest Strokes	Biggest Strokes	Average Strokes	Lowest Lines	Biggest Line	Average Lines	Lowest Time	Biggest Time	Average Time
65		Marcato	19	5	1	2	1	13	64	31	192	2527	575
66		Left-hand pizzicato or Stopped note	19	6	2	4	2	14	58	28	305	2395	690
67		Snap pizzicato	19	7	2	6	2	31	136	56	530	6794	1402
68		Natural harmonic or Open note	19	8	1	2	1	15	97	34	263	2638	591
69		Fermata (Pause)	19	9	2	3	2	22	262	58	427	5578	1640




Mobile framework for recognition of musical characters

Symbol	Image	Name	Class	Sub-class	Lowest Strokes	Biggest Strokes	Average Strokes	Lowest Lines	Biggest Line	Average Lines	Lowest Time	Biggest Time	Average Time
70		Up bow or Sull'arco	19	10	1	3	1	16	64	28	249	2720	545
71		Down bow or Giù arco	19	11	1	17	3	27	395	117	425	9334	2688
72		Trill	19	12	1	5	3	30	207	65	621	11075	1918
73		Mordent	19	13	1	5	1	27	340	62	437	9681	1267
74		Mordent (lower)	19	14	2	6	2	29	244	75	557	9378	1938

Mobile framework for recognition of musical characters

Symbol	Image	Name	Class	Sub-class	Lowest Strokes	Biggest Strokes	Average Strokes	Lowest Lines	Biggest Line	Average Lines	Lowest Time	Biggest Time	Average Time
75		Turn	19	15	1	7	1	27	291	62	420	9518	1120
76		Tremolo	20	1	1	7	3	25	311	47	485	6619	1282
77		Repeat sign	20	2	4	18	6	50	643	255	1732	18362	6845
78		Simile mark	20	3	3	11	3	24	386	94	784	13341	2870
79		Simile mark	20	4	5	17	6	45	669	152	1344	24424	5531

Mobile framework for recognition of musical characters

Symbo l	Image	Name	Class	Sub- class	Lowest Strokes	Biggest Strokes	Average Strokes	Lowest Lines	Biggest Line	Average Lines	Lowest Time	Biggest Time	Average Time
80		Volta brackets	20	5	1	4	1	26	210	68	397	4415	1230
81	<i>D.C.</i>	Da capo	20	6	3	7	5	58	255	105	1202	54453	3753
82	<i>D.S.</i>	Dal segno	20	7	3	7	5	58	254	115	1258	13200	3427
83		Segno	20	8	2	8	4	47	997	139	900	23993	4044
84		Coda	20	9	3	6	3	54	241	100	915	7119	2349

3.3 Editing User Draws from the database

When facing information by a hand draw symbol, there are a lot of problems we need to face before making a real analysis.

Despite being an important point to have different kind draws from different people, we cannot have control of how the user draws his symbol. From little points made intentional, to the lack of experience in touch environment or even the need to write exactly like in a paper, we have a lot of data that is not good for analyze. These “errors” must be fixed before entering the phase of segmentation, in order to have a most cohesive database. We can see an example of a drawing error in image 8.

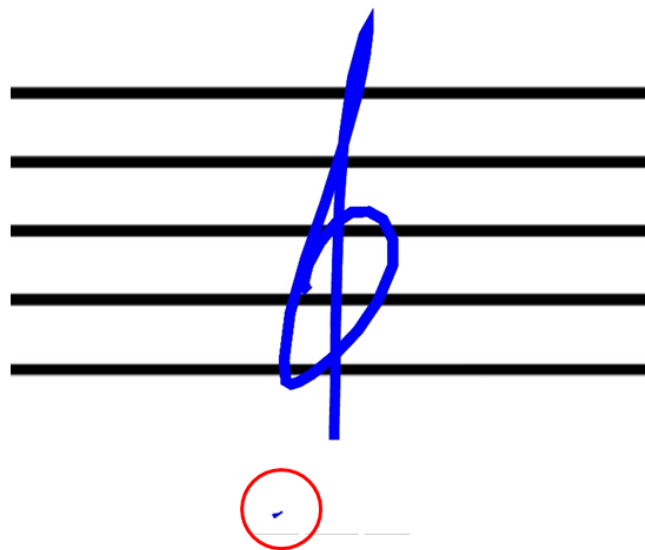


Figure 8 - Drawing with “garbage”

The first approach was to redraw every draw, and see one by one if there was any “garbage” that needed to be erased from the draw; in this phase we focus essentially in the draws done intentionally by the user (at least the ones that we could see). But that was not enough because there were still symbols that were too small to be drawn, but enough to be recorded. For that, we created a code in Matlab to see in every stroke of every draw, if the percentage of ‘1s’ between the ‘0’ and the ‘2’ were too small to even be a draw, and in that cases, making that stroke a non-usable stroke. This process made the rate of recognition rise dramatically.

The mistakes done by the user were driven by 5 possible facts: the user is mistaken and draws in the wrong place that was not supposed to (example 1 in the figure 9); the user tried to draw a single point with a single fast touch (trying to draw a small circle), but the way that the

android system was made, did not make possible to recognize that kind of gesture, forcing the user to make another circle beyond the previous one; the system did not recognize the draw at the first time; the user exceed the space of drawing (example 2 in the figure 9); or even when trying to save the draw in the “Save” button, the user touch a little above and instead of clicking in the button , he touches in the drawing panel (example 3 in the figure 9).

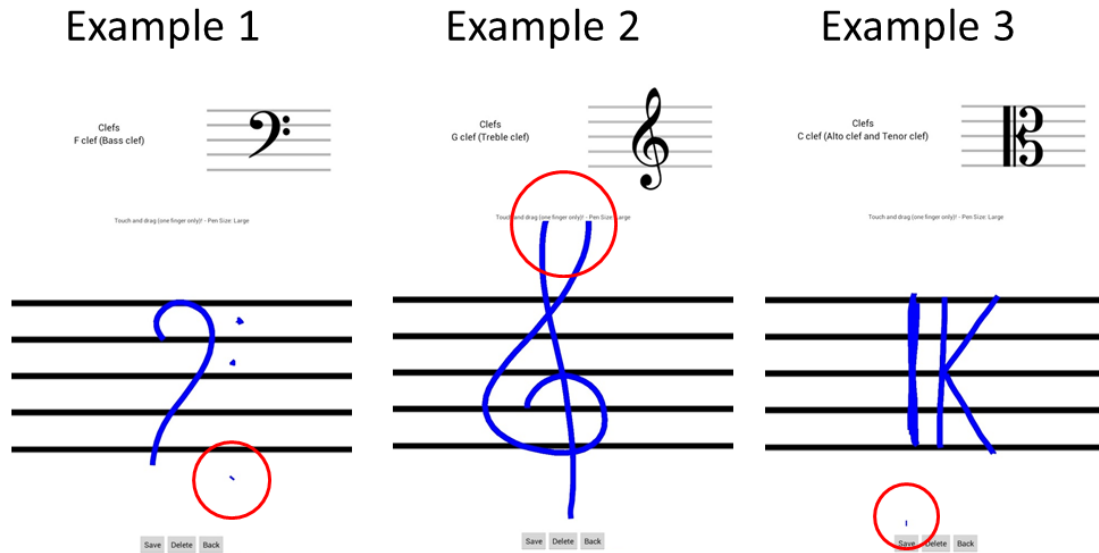


Figure 9 – Garbage examples

After that, we were faced with a problem of files with 0 Kb (Quilobyte) of disk size, which means that the user save the file without drawing anything. This point could be easily be prevented with a condition in the Android application, forcing the user to draw anything in order to proceed to the next symbol, but that function was not implemented at the moment. For solving this problem we copy one of the other 2 equal symbols created and past it with the name of the file with the problem. The solution is not ideal, but taking into account that we did not had another chance to be with every drawer in the database for them to redraw, we think that was the most viable. For documentation purpose the symbols exchanged can be seen in table 3.

Table 3 - Files replaced

User Number	Original Draw	Exchanged by
11	117	201
18	30	198
24	36	204
26	161	245
38	162	246

Mobile framework for recognition of musical characters

The final problem needed to be fixed, was not directly fault of any user, but more of the Android system. When a user draws his symbol in a very slightly way, but not lifting his finger, the android system save the parameter 'Action' with values '1' and then '0', when is just supposed to save only '1s' until it reaches a '2', following the logic of 0/1/2, explain in the "Creation of the Database" section. For that, we just locate all the '0s', which are not followed by '2' or that start a draw, and substitute for '1s', creating a normal logic of action. This event happens especially with people with less ability (or experience) to draw in the touch screen.

Capítulo 4

Background Knowledge

In this chapter we will first make a brief description of the technology being use, explained the importance of them in this work.

4.1 Technological Description

The term pattern recognition encompasses a wide range of information processing problems of great practical significance, from speech recognition and classification of handwriting characters. Many of these problems are solved effortlessly by humans, but their solution using computer has proved to be immensely difficult [NNBishop].

The most general, and most natural, framework in which to formulate solutions to pattern recognition problem is a statistical one, which recognize the probabilistic nature both of the information we seek to process, and of the form in which we should express the results.

Concerning problems with pattern recognition, the majority of the applications use several feature methods to achieve his purpose and normally is categorized according to the type of learning procedure used to generate the output value.

We will next describe some of those methods and how we used them in this work.

4.1.1 Neural Networks Overview

Neural Network (NNs) are networks of neurons, for example, as found in a real biological brain. Artificial Neurons are representations of the neurons found in brains, being physical devices, or purely mathematical constructs. Artificial Neural Networks (ANNs) are networks of Artificial Neurons, and for that, constitute crude approximations to parts of real brains.

From a practical point of view, an ANN is just a parallel computational system consisting of many simple processing elements connected together in a specific way in order to perform a particular task. [Jaime2005]

In order to introduce many of the fundamental concepts of statistical pattern recognition, we will use the example from Christopher M. Bishop book [NNBishop], with a problem of distinguishing hand-written versions of the characters 'a' and 'b'. Supposing that we already have the images into the computer, we seek an algorithm that can distinguish as reliably as possible between the two characters. We shall suppose that we provided with a large number of examples of images corresponding to both 'a' and 'b', which has already been classified by human, such collection is referred as a *data set*. Having a large number of input variables can bring some problems for pattern recognition systems. So normally we combine input variables together to make a smaller number of new variables called *features* (construct by hand or be derived from the data by automated procedures). In this example we could evaluate the height and the width of the character, since the letter 'a' has some differences from the letter 'b' in that fields. And for that we might expect that characters 'b' to have larger values of the ratio, than the characters corresponding to 'a'. But it suffers a problem, that there is still a significant overlap of the histograms, as we can see in the image from Bishop book [NNBishop], in figure 10.

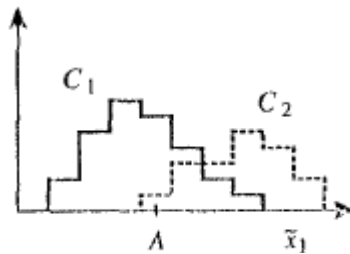


Figure 10 - Histogram of C1 (letter 'a') and C2 (letter 'b'). [NNBishop]

So it is possible that some of the symbols would be misclassified. One way to improve the situation is to consider a second feature and try to classify new images with both features together. We will then see that a few examples are still incorrectly classified, but the separation of the patterns is much better, as we can see in the figure 11.

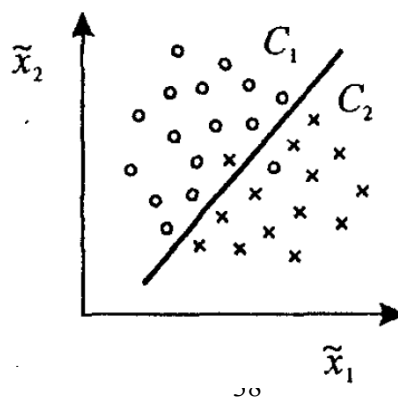


Figure 11 - Separation between C1 (letter 'a') and C2 (letter 'b'). [NNBishop]

We could continue to consider ever larger number of (independent) features in the hope of improving the performance indefinitely.

Adding too many features can lead to a worsening of performance. Furthermore, for many real pattern recognition applications, it is the case that some overlap between the distributions of the classes is inevitable

With handwritten characters there is a considerable variability in the way the characters are drawn. We are forced to treat the measured variables as random quantities, and we cannot expect a perfect classification. Instead we could aim to build a classifier that has the smallest probability of making a mistake.

With the help of a dataset of examples, the mapping is therefore modeled in terms of some mathematical function which contains a number of adjustable parameters, whose values are determined with the help of the data.

The importance of neural networks is that they offer a very powerful and very general framework for representing non-linear mappings from several input variables to several output variables, where the form of the mapping is governed by a number of adjustable parameters. The process of determining the values for these parameters on the basis of the data is called *learning* or *training*, and for these reasons the data set of examples is generally referred to as a *training set*.

The advantages of using Artificial Neural Networks undergo by: are extremely powerful computational devices; massive parallelism makes them very efficient; they can learn and generalize from training data – so there is no need for enormous feats of programming; they are particularly fault and noise tolerant, so they can cope with situations where normal symbolic systems would have difficulty [Jaime2005].

One of the basic goals for neural network research goes to the engineering goal of building efficient systems for real world applications. This may make machines more powerful, relieve humans of tedious tasks, and may even improve upon human performance.

They adapt the strengths/weights of the connections between neurons so that the final output activations are correct, as see in figure 12.

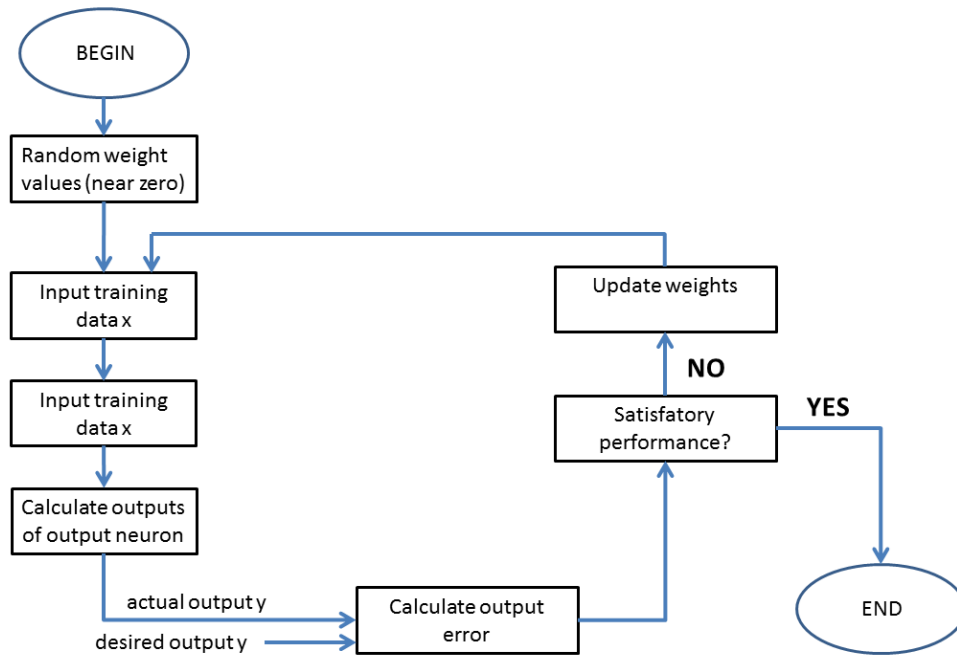


Figure 12 - Function example of a neural network

Further, we will describe where and when do we use a Neural Network.

4.1.2 HBF49

Unlike image based approaches for classification, where the pattern is represented as a purely visual signal, the online nature of input signal, however contains rich information about the dynamics of the drawing (order and number of strokes, writing direction, speed and pressure). When working with this kind of information we face a problem different from the off-line segmentation. We present here a work done by Adrien Delaye and Eric Anquetil, where they introduce a set of 49 features, called *heterogeneous baseline feature set* (HBF49) [HBF49], covering diverse aspects of patterns characteristics, and presenting the following properties:

- Ability to describe unconstrained pen-based input (number of strokes, writing order, direction);
- Comparable/better performance with respect to state-of-the-art results on various benchmarking datasets, by using a standard support vector machine (SVM) classifier;
- Dealing with writer-dependent (WD) or writer-independent (WI) experimental settings;
- Limited in size (reasonably low number of 49 features)

They do not claim to be the best possible set of features for universal symbol recognition. However they are confident that it provides an accurate and robust description of symbols in a greater variety of context thanks to a complete coverage of the patterns properties.

For homogeneous representation, they deliberately chose to exclude features based on pressure (pressure variations, and off-strokes measure) or temporal information (speed, acceleration). For that the data collected of *time* and *pressure* are not information considered in these features. However some points can always be detected as pen-up points: they denote points of the trajectory where the contact between pen and surface was interrupted.

The classification in HBF49 is separated into 3 major sections: Preprocessing, Dynamic features and Visual features. Prior to the feature extraction itself, they expose the pre-processing operations applied on the input patterns. These operations are simple to perform and guarantee a better stability of extracted features, for any type of input pattern. They first do a *Linear scaling and translation* where the input pattern is rescaled so that its maximal dimension is equal to a normalized size. They chose the normalized dimension as 128 in all experiments. Then they do a *Trajectory resampling*, where they spatially resample the points, so that the points on the resampled trajectory are equidistant. The imposed distance between two points in the resampled trajectory is fixed to 8, this value being related to the box dimension of 128.

The Dynamic features, model the writing process, focusing on how the pattern was accomplished by the writer, by implicitly or explicitly describing the order of the points, the writing direction, the number and order of strokes, and so on. Here we have a total of 14 dynamic features: *Starting and ending points position*, *First point to last point vector*, *Angle of initial vector*, *Inflexions*, *Proportion of downstrokes trajectory* and *Number of strokes*.

The second type of features is *Visual features*, in the sense that they do not depend on the writing process, but focus on the appearance of the writing result. Here are the mains sections: *Bounding box diagonal angle*, *Trajectory length*, *Deviation*, *Average direction*, *Curvature and perpendicularity*, *K-Perpendicularity*, *K-angle*, *Absolute angle histogram*, *Relative angle histogram*, *2D histogram*, *Hu moments*, *Convex hull features*.

HB49 is designed as a generic representation of symbols, without consideration of drawing constraints or domain specificities, can serve as basis for designing universal systems.

These features can be seen with more detail in the paper [HB49], or in the formulas description in Appendix A.

4.1.3 AvgRF

In the article “Online Signature Verification and Recognition” [AvgRF] they proposed a method of representing online signatures by interval-valued symbolic features, based on representing global features of online signatures of a class in the form of interval-valued data. Features of samples signatures of a class have considerable intraclass variations. They propose

to have an effective representation to capture these variations through their assimilation by the use of interval-valued feature vector called symbolic feature vector.

Let μ , be the mean of the feature values obtained from all the samples of the class:

$$\mu_{jk} = \frac{1}{n} \sum_{i=1}^n f_{ik} \quad (1)$$

And σ , be the standard deviation:

$$\sigma_{jk} = \left[\frac{1}{n} \sum_i^n \left(f_{ik} - \mu_{jk} \right)^2 \right]^{\frac{1}{2}} \quad (2)$$

The interval representation depends on the mean and standard deviation of respective individual features. The interval represents the upper and lower limits of a feature value of a signature class in the knowledgebase. This individual's feature-dependent threshold forms a good measure to compare the features in an online reference for verification and recognition, calling it 'AvgRD'.

We believed that this information could improve the rate of recognition if associated to HBF49 in the neural network.

Capítulo 5

Experimental Testing

We will describe all the process from getting the users draws data from the database until the creation of the classifiers.

5.1 Symbol Classification

Before comparing the online recognition results with the offline ones, we need to see that the parameters of input collected are not the same as one to one another, implying a different way to process each one. In the next section we will describe the difference between them.

5.1.1 Dataset Creation

5.1.1.1 Online

Having the edited version of the database, now it is time to structure the data in order to create the *dataset* for the neural network.

We made a code in Matlab to read every txt file from the database.

Reading every txt files from the database, we analysed the symbol and extracted the 49 features from the HBF49 functions, and we connect with the specific class of the symbol (as described in “Creation of Database” section). For every user we have 252 symbols, and for every symbol we have a matrix of 1x50 (49 features and the number of the class). Then doing the same for all 50 users, we get a matrix of 12600x50 elements. We can see an example scheme in figure 13 that illustrates this matter.

Having the final matrix of all the symbols, it is time make the recognition.



Figure 13- Creation of the matrix

We also need to create a matrix for every class, with more than 1 symbol, in order to recognize the symbols within. This is a simple process except for the “Beamed Symbols”, the symbols 26, 27 and 28. Every one of this symbol has 4 different symbols within. The user needed to draw 4 lines of the beamed note, but in order to avoid repetition on drawing (saving some time to the user), we decide to only give the user 1 draw of 4 beamed notes, and we would get all the information from the symbol in order to remove what we want from that one single draw, as seen in the figure 14.

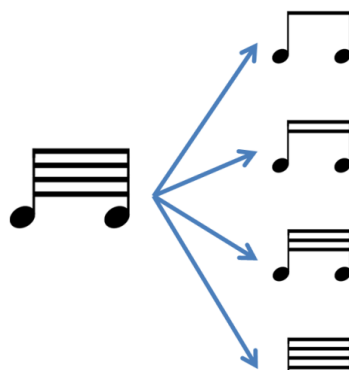


Figure 14 - Beamed notes separation

Analyzing the file, when we have a set of 4 strokes in the beamed note draw, the separations it is pretty intuitive, because we just need to get each one of them and combine them in order to create the 4 floors of beamed notes. But we face a problem when we have more than 4 strokes, implying that we need to see how the draw was drawing, and define which stroke belong to each level, and define the level to the specific stroke or strokes. For the class 7, we have 3 symbols, repeating 3 times, and every one of them extract 4 symbols, giving a total of 1800 symbols (3 symbols x 3 repetitions x 4 extractions x 50 users).

5.1.1.2 Offline

In the offline mode, we are making the recognition from the drawing images.

For that, we needed to re-draw all the symbols and save them into a standard lossless data image compressed file (JPEG¹/PNG²), because what we have is just information of how to draw them. We make a code in Android developer language, to do that process. It would be possible to also redraw any symbol in other software, like Matlab; but since the drawing data was collected from an android system, the drawing from android would reproduce a more reliably image than other software, because it uses a specific drawing functions, curvatures and image approaches.

The algorithm started to open the txt file from the database, and save all the information needed. After that it sees the maximum and minimum position values of the symbol and adjusts the symbol to the top left corner. Creates a clean bitmap and draw in it with the information saved before, using the size of the pen saved in the database. The file is closed and saved into a specific folder. We compressed the symbol into a PNG file, because it would occupied less space and we did see no significant different between one and another. This process was repeated for every symbol in the database.

¹ JPEG – Joint Photographic Experts Group

² PNG – Portable Network Graphics,

5.1.2 Neural network creation

5.1.2.1 Online

In order to see if the methods described would increase the rate of recognition, we implement 3 different ways the neural network. We concatenate the HBF49 and the AvgRF before creating the neural network, which we call ‘cNN’; naming ‘mixNN’ we use the HBF49’s 49 features to create the neural network and the AvgRF only entered in the last layer of the network; and in last we create a simple neural network, using only the HBF49. We can see in figure 15 an example of the previous described methods.

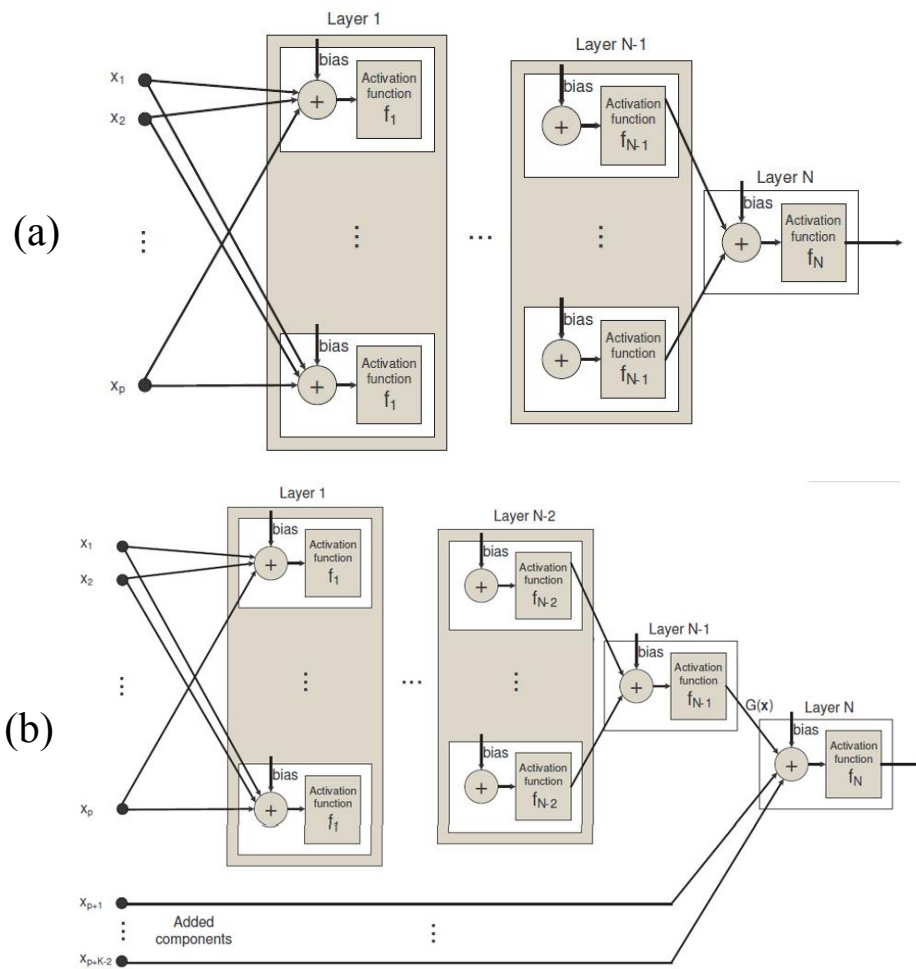


Figure 15 - Example of the NN methods: a – cNN, b – mixNN [Jaime2005]

We used a Multi-Layer Perceptron (MLP³) classifier with one hidden layer and a sigmoid function as activation function. To obtain all the NNs the available dataset was randomly split into training and test sets, with 60% and 40% of the data, respectively. No special constraint was imposed on the distribution of the categories of symbols over the training and test sets, ensuring only that at least one example of each category was present in the training set. The best parameterization of each model was found using the training and validation sets being the expected error estimated on the best set by a 4-cross validation scheme. The 49 features of the matrix have a specific class that goes from 1 to 20 (has explained in “Creation of the Database” section), this 20 classes are going to be the output layer of the network.

We create a neural network using all of the specific methods and some will be describe next. Besides the final neural network created, with all the important information for classification, we also set the estimate rates of performance; data that we will describe in the next section.

5.1.2.2 Offline

In the offline process, we do not have a set of features, but a set of images. We use the Otsu’s method [Sezgin2004] of global threshold to convert the image to binary. The image is resized to 60X60 in order that all the symbols have the same size and be possible to compare them.

Then, we create a simple neural network with the same parameters as the one used by the online mode (using the images as input) to classify the symbols, extracting also the estimate performance rates, as we will describe in the next section. In an offline mode, it is not possible to use the same methods, because we do not have dynamic features as in online mode. Taking into account that our focus is on the online recognition, we did not feel the need to implement other type of offline classifiers.

5.2 Results

In this section we will show and analyze the results from the all the operations done in the previous section. We also will make a comparative study between the online and the offline mode.

³ MLP – a MLP is a layered structure consisting of nodes or units (called neurons) and one-way connections or links between the nodes of successive layers.

5.2.1 Online – All classes

We set these parameters for all the classifiers:

- Train size = 0.25;
- Number of runs = 50;
- Number of neurons = 40:5:70

We can see in table 4, the results from all the estimative rates for all the 20 classes.

Table 4 - Estimative rates results

	Online						Offline	
	cNN		mixNN		Neural Network		Neural Network	
Performance expected in real life (%)	85		64		79		26	
IC at 99% for the performance	85	85	64	65	79	80	26	27
IC at 99% for standard deviation	0,95	1,61	1,65	2,79	0,92	1,56	1,50	2,53
Number of neurons in the hidden layer	90		58		90		58	

5.2.2 Online – Sub classes

We previous saw the results from the classification of all the classes, but now we need to see the classification for every sub class, in order to see the rate of performance for each one, as we can see in table 5.

Table 5 - Results of the performance from the sub-classes

	cNN	mixNN	Neural Network
sub-Class	Performance	Performance	Performance
2	98	94	97
4	99	99	97
5	77	60	66
6	81	63	74
7	98	89	98
9	100	99	99

10	98	93	95
11	87	86	90
12	99	49	99
14	100	49	100
17	80	73	79
19	92	87	90
20	92	87	89

5.3 Comparative Study

As we can see in the previous section of results, the best result was from the cNN method, where it got a “performance expected in real life” of 85%, an interval of confidence for the performance between 84,56% and 85,48%, and also an interval for standard deviation between 0,94 and 1,6.

We can also see that Neural Network got better results than *MixNN*, giving us information that in the process of the creation of the neural network it is not a good idea to implement the AvgRF in the last layer of the network, since HBF49 alone worked better. Also we can see that the neural network works better when we concatenate the HBF49 with the AvgRF before creating the neural network.

The idea of implemented the 2 features from AvgRF into the classification turned out to be good idea, increasing the rate of performance in 6 %.

In the subclass table of recognition rates we can get clearly high rate of recognition results, as expected. Since there are many symbols to classify it is normal that the rates go higher. As we can see in the table 5, cNN is also the best choice for all the symbols in the subclass. With classes with fewer symbols, the rates cross the 90% and some even reach up to 100%.

In the subclass, the worst rate of recognition is in class 5 where we got 77%. This class contains all of the notes, from the ‘Breve’ to the ‘Hemidemisemiquaver’. This rate can be explain by the fact that most of these symbols are very similar to one another, and also because, every symbol have at least 2 variations (for example, with the leg up and down), worsening the classification.

The rates of recognition in class 17 are also not very good, where we got 80%. This class is where it sets the dynamical words of music, like ‘piano’ and ‘forte’. We can explain the low rate here by the fact that the handwriting of words be less standardized than the musical notation

dictionary, which can lead to a more complex variation between users handwriting. The class 11 follows the same logic as the class 17, with an average performance of 87%

In comparison between offline and online, we can see that the online mode works better than the offline, having a difference of 60%. Even not using the average and the standard deviation from AvgRF, we still got a difference of 53%.

The low rate of the offline method could be improved if we tried other classifiers, some of them are described in the Chapter 2, but has said before our focus is on dynamical recognition of the online mode. We could try to improve the rate of recognition if we define the matrix for the draws in another way. We resize all images to 60x60, this implies stretching for most of the images, therefore a changing the real shape of the symbol, and not in a global way. Every draw is changing differently, not following a width or a height, making symbols alike becoming completely different and symbols completely difference becoming alike.

Chapter 6

Conclusion

In this chapter we will make some considerations of this thesis, taking some conclusions and also present a possible future work in this area.

6.1 Conclusion

The work developed in this thesis had the target to overcome the several issues that affect handwritten musical symbol recognition. Based on the off-line state-of-art, we believed that we could focus our efforts in the online recognition and trying to overcome some of the off-lines problems, like elimination of segmentation issues caused by overlapped symbols. This also brings some advantage such as, extra knowledge about spatial-temporal information of symbols.

We created an application based on android for capturing gesture-free of musical symbols. This application had a system to record the drawing path made by the user and save vital information of each draw. This draws were saved into a database of user draws. With the information collected we were, besides be able rewrite the symbol, we got a lot of information from each draw, which could be used in very different areas of study.

For the classification, we did not use neither the time variation nor the pressure (despite being record in the database for future use), but both could be useful to improve the recognition system. With the *time*, we could segment the symbol by his average velocity or acceleration in certain parts of the symbol. This could be also used to identify a person through a specific draw, by the acceleration imposed on the draw, and also the draw itself. Both with *pressure*, we could also use it as an addition to the segmentation. Other interesting matter with *pressure* is that we

could use it for statistical analysis, to evaluate types of emotions from the way of drawing, creating a logical profile of a person based on the draw data information.

We saved 252 musical handwriting draws from 50 different users, and also some important personal information from the user.

We made a statistical study from the user's information, where we could associate that information with the kind of draws made, and analyze data like: sex, country, music knowledge in sheet music, handwriting preference and age.

We did not get much information about the difference in culture, because the majority of the users were Portuguese, and we did not have a solid number of users with different cultures to make a solid argument. Overall the percentage of users with musical experience were superior to the rest of the users, increasing the diversity of handwriting draws, because users with experience in music try not copy the symbol, instead they write like they used to. This situation can decrease the rate of recognition, but enrich the quality of the database.

We saw a clear difference from users with music experience and users with no experience in the way of drawing, the time depending, the number of strokes, the proximity to that standard music symbol and even the number of errors done.

We did a description of the symbols, based on the user draws, showing some information with highest, lowest and average values, creating an interesting way to see how the users usually write the symbols and how we should expect a symbol to be drawn in the future.

Editing the database is a process indispensable when dealing with touch screen methods. Almost all the users made at least 1 mistake when writing in the application, especially when they faced a long period of drawing. We saw that when we fix the "errors" the rates increased substantially.

Being this a project with focus on pattern recognition, it is advisable the use of Neural Networks, and in the case of Online classification, it is almost mandatory. NN are extremely powerful computational devices which they can learn and generalize from training data.

We use the methods from HBF49 [HBF49] to classify the draws into a set of features and the AvgRF [AvgRF] to help in the process of segmentation.

HBF49 has a set of features very useful when dealing with unconstrained pen-based input. It provides an accurate and robust description of the symbol in a greater variety of context, covering the complete set of patterns properties. Unfortunately, they exclude features based on pressure (pressure variations, and off-strokes measure) or temporal information (speed, acceleration). We are also using AvgRF interval-valued symbolic features for classification, where we use the Mean and the *Standard* Deviation. We set up 3 types of data to enter the neural network: one where the HBF49 enters in the first layer with AvgRF, which we call *cNN*; other when the *AvgRF* instead enters in the last layer of the network; and other when the

network is dataset from the HBF49 features. The network using a 4-cross validation scheme and a Multi-Layer Perceptron classifier with one hidden layer, has an output of 20 classes.

With the offline dataset, we retrieve the images from the database and create a simple neural network with the same parameters as the online. Since our focus in the beginning was in online recognition, we did not try to implement classifiers on the offline dataset in order to improve the recognition. And we believe that the online recognition have more potential to recognize a symbol with the highest rate than the offline mode.

We achieve the best rate in online mode of 85% by cNN and the worst in mixNN with 64%. In comparison with the offline mode, the results were very superior, where we see rates in the 26% for an offline neural network.

In the subclasses rates, we were expected some classes to have the rates a little higher than 77% and 81%, but we reached several in rates between 98% and 100 %

The results of the rate of performance were globally satisfactory. We were expecting higher rates for some of the sub classes, but still a very good rate of recognition.

This work is the agglomerate of all the process involved, and can be used in the future as a framework for future development.

6.2 Future Work

Following the conclusions of this work we will describe the improvements that we would like to implement in the future and our vision for this project.

Our immediate purpose is to implement the result classifiers into android, in order to create a software for recognition. This process is not very far from now, since that the classifiers are already created, we just need to create a neural network in Android, implement the HBF49 features in android, and make the selection by means of the weights.

Other also important point is to try to improve the rates of recognition with other methods and correcting some of the flaws in this project.

Other focus is to create a second database using different tools to save the information, where the user has to write a whole score into a mobile device, using options of slide, zoom, easy composing, using pressure or time to vary the kind of ways of drawing in some different way.

Also based on the previous idea, we would like to create a new intuitive software for composition using new accessible tools and new different ways of drawing.

Other important vision in the future, is to implement other types of media recognition, like external sound composition, OMR offline, or some dispositive of automatic motion capture

Mobile framework for recognition of musical characters

(like Kinect or Leap Motion), to create a new way of composing using gestures into portable devices.

References

[Macé2005] S Macé, E Anquetil, E Garrivier, B Bossis. “A Pen-Based Musical Score Editor” - In proceedings ICMC, Barcelona, 2005

[ACapela2008] A. Capela, J. S. Cardoso, A. Rebelo, and C. Guedes, “Integrated recognition system for music scores,” in Proceedings of the International Computer Music Conference, 2008.

[Rebelo2010] A. Rebelo, G. Capela, and J. S. Cardoso. Optical recognition of music symbols: A comparative study. International Journal on Document Analysis and Recognition, 13:19–31, 2010.

[Cardoso2009a] Jaime S. Cardoso, Artur Capela, Ana Rebelo, Carlos Guedes, and Joaquim F. Pinto da Costa. Staff detection with stable paths. IEEE Transactions on Pattern Analysis and Machine Intelligence, 31(6):1134–1139, 2009.

[Cardoso2009b] Jaime S. Cardoso and Ana Rebelo. Robust staffline thickness and distance estimation in binary and gray-level music scores. In Proceedings of The Twentieth International Conference on Pattern Recognition (ICPR 2010), pages 1856–1859., 2010.

[RuiSilva2013] Rui Silva, Jaime S. Cardoso, and Ana Rebelo , “Online database of hand-draw musical symbols”, in 2nd PhD. Students Conference in Electrical and Computer Engineering – StudECE 2013

[APIandroid] Android 2012, Plataform Versions, <http://developer.android.com/about/dashboards/index.html#Platform>, access in 28 June 2013

[Rebelo2012] A. Rebelo, I. Fujinaga, F. Paszkiewicz, A. R. S. Marcal, C. Guedes, and J. S. Cardoso, “Optical Music Recognition - State-of-the-Art and Open Issues,” International Journal of Multimedia Information Retrieval, 2012.

[Pruslin66] Automatic recognition of sheet music, 1966. In D. Blostein and H. Baird. A Critical Survey of Music Image Analysis. In Structured Document Image Analysis, pages 405–434, Springer-Verlag, Heidelberg, 1992.

[Prerau1970] Optical music recognition using projections, 1988. In D. Blostein and H. Baird. A Critical Survey of Music Image Analysis. In Structured Document Image Analysis, pages 405–434, Springer-Verlag, Heidelberg, 1992.

[George2004] S. George. Pen-based Input for On-line Handwritten Music Notation. IRM Press, 2004.

[Jones2008] G. Jones, B. Ong, I. Bruno, and K. Ng. Optical music imaging: Music document digitisation, recognition, evaluation, and restoration. In Interactive Multimedia Music Technologies, pages 50–79. Hershey: IGI Global, 2008.

[ARebelo2009] A. Rebelo, G. Capela, and J. S. Cardoso. Optical recognition of music symbols: A comparative study. International Journal on Document Analysis and Recognition, 13:19–31, 2010.

[PalmPilot1997] Alsop, Stewart, —Innovative Graffiti Might Actually Make PDAs Useable, InfoWorld, September 26, 1994, p.130.

[GraffitiPalmOS] Silberger, K. 1996, —Putting Composers in Control. IBM Think Research 34(4). Available at http://domino.research.ibm.com/comm/wwwr_thinkresearch.nsf/pages/musiceditor496.html

[Silberger1996] k. Silberger. Putting composers in control. IBM Think Research, 34(4), 1996. Available online at http://domino.research.ibm.com/comm/wwwr_thinkresearch.nsf/pages/musiceditor496.html

[Anstice96] J. Anstice, T. Bell, A. Cockburn and Martin Setchell. The Design of a Pen-Based Musical Input System. In Proceedings of the Sixth Australian Conference on Computer-Human Interaction, pages 260-267, 1996.

[Forsberg98] A. Forsberg, M. Dieterich and R. Zeleznik. The Music Notepad. In Proceedings of the ACM Symposium on User Interface and Software Technology (UIST), pages 203-210, 1996.

[Miyao2007] H. Miyao and M. Maruyama. An online handwritten music symbol recognition system. International Journal on Document Analysis and Recognition, 9:49-58, 2007.

[Jamie96] Jamie Ansticel, Tim Bell, Andy Cockburn, Martin Setchel. “The Design of a Pen-Based Musical Input System”, Computer-Human Interaction, Proceedings., Sixth Australian Conference, Page(s): 260 – 267, 1996

[Forsberg98] A. Forsberg, M. Dieterich and R. Zeleznik. "The Music Notepad", ACM Symposium on User Interface Software & Technology, pp. 203-210, 1998.

[SMace2005] S. Macè, E. Anquetil and B. Couasnon, "A generic method to design pen-based systems for structured document composition: Development of a musical score editor," in Proceedings of the First Workshop on Improving and Assessing Pen-Based Input Techniques. pp. 15-22 Edinburgh, September 2005.

[George2004] S. George. Pen-based Input for On-line Handwritten Music Notation. IRM Press, 2004.

[Taubman2005] G. Taubman. MusicHand: A handwritten music recognition system. Honors thesis, 2005.

[Mitobe2004] Y. Mitobe, H. Miyao and M. Maruyama, "A fast HMM algorithm based on stroke lengths for on-line recognition of handwritten music scores," in Proceedings of the ninth International Workshop on Frontiers in Handwriting Recognition (IWFHR-9 2004), IEEE Computer Society, 2004.

[Lee2010] Kian Chin Lee; Phon-Amnuaisuk, S.; Choo Yee Ting; , "Handwritten music notation recognition using HMM — a non-gestural approach," International Conference on Information Retrieval & Knowledge Management, pages 255-259, 2010.

[Macé2009] Sébastien Macé, Eric Anquetil . "Eager interpretation of on-line hand-drawn structured documents: The DALI methodology" - Pattern Recognition, Volume 42, Issue 12, December 2009, Pages 3202–3214

[Macé2012] Macé, Sébastien, Eric Anquetil and Bruno Bossis. "Pen-Based Interaction for Intuitive Music Composition and Editing." Intelligent Music Information Systems: Tools and Methodologies. IGI Global, 2008. 261-288. Web. 7 Dec. 2012.

[Williams1903] Williams, Charles Francis Abdy (1903). "The Story of Notation." New York: Charles Scribner's Sons.

[Heussenstamm] George Heussenstamm, *The Norton Manual of Music Notation* (New York and London: W. W. Norton & Company), p.16

[NNBishop] C. M. Bishop. Pattern Recognition and Machine Learning (Information Science and Statistics). Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[Jaime2005] Jaime S. Cardoso , "Classification of Ordinal Data", MSc in Mathematical Engineering, FCUP, November 2005.

[HBF49] Adrien Delaye, Eric Anquetil. "HBF49 feature set: A first unified baseline for online symbol recognition" - Pattern Recognition, Volume 46 Issue 1, January, 2013, Pages 117-130

[AvgRF] Guru DS, Prakash HN. "Online signature verification and recognition: an approach based on symbolic representation" - IEEE Trans Pattern Anal Mach Intell. June 2009 (vol. 31 no. 6) pp. 1059-1073

[Sezgin2004] M. Sezgin and B. Sankur (2004). "Survey over image thresholding techniques and quantitative performance evaluation". Journal of Electronic Imaging 13 (1): 146–165.

Appendix A

Papers

Here we will show the functions from HBF49 used to define the dataset and also the accepted paper involving this work.

8.1 HBF49 Features

This set of images will describe the functions used to collect the 49 characteristics from the user draw.

8.1.1 Starting and ending points position

$$f_1 = \frac{x_1 - c_x}{l} + \frac{1}{2}, \quad f_2 = \frac{y_1 - c_y}{l} + \frac{1}{2}$$

$$f_3 = \frac{x_n - c_x}{l} + \frac{1}{2}, \quad f_4 = \frac{y_n - c_y}{l} + \frac{1}{2}.$$

8.1.2 First point to last point vector

$$f_5 = \|v\|, \quad f_6 = \frac{\vec{v}_x \cdot \vec{u}_x}{\|v\|}, \quad f_7 = \frac{\vec{v}_y \cdot \vec{u}_y}{\|v\|},$$

8.1.3 Closure

$$f_8 = \frac{\|v\|}{L}.$$

8.1.4 Angle of initial vector

$$f_9 = \frac{\vec{w}_x \cdot \vec{u}_x}{\|w\|}, \quad f_{10} = \frac{\vec{w}_y \cdot \vec{u}_y}{\|w\|}.$$

8.1.5 Inflexions

$$f_{11} = \frac{1}{w} \left(x_m - \frac{x_1 + x_n}{2} \right), \quad f_{12} = \frac{1}{h} \left(y_m - \frac{y_1 + y_n}{2} \right)$$

8.1.6 Proportion of downstroke trajectory

$$f_{13} = \sum_{k=1}^p L_{k_i, k_j},$$

8.1.7 Number of strokes

$$f_{14} = K$$

8.1.8 Bounding box diagonal angle:

$$f_{15} = \arctan \frac{h}{w}$$

8.1.9 Trajectory length

$$f_{16} = L \quad f_{17} = \frac{w+h}{L}$$

8.1.10 Deviation

$$f_{18} = \frac{1}{n} \sum_{i=1}^n \|s_i \mu\|.$$

8.1.11 Average direction

$$f_{19} = \frac{1}{n-1} \sum_{i=1}^{n-1} \arctan \left(\frac{y_{i+1} - y_i}{x_{i+1} - x_i} \right)$$

8.1.12 k-Perpendicularity, k-angle

$$f_{22} = \sum_{i=k+1}^{n-k} \sin^2(\theta_i^k), \quad f_{23} = \max_{i=1+k}^{n-k} \theta_i^k.$$

8.1.13 Absolute anglehistogram

$$f_{24} = \frac{h_1 + h_5}{n_a}, \quad \dots \quad f_{27} = \frac{h_4 + h_8}{n_a}$$

8.1.14 2D histograma

$$f_{32} = \frac{1}{n} \sum_{i=1}^n \mu_{11}(s_i) \quad \dots \quad f_{40} = \frac{1}{n} \sum_{i=1}^n \mu_{33}(s_i).$$

8.1.15 Hu moments

$$\begin{aligned} f_{41} &= v_{02} + v_{20}, \\ f_{42} &= (v_{20} - v_{02})^2 + 4v_{11}^2, \\ f_{43} &= (v_{30} - 3v_{12})^2 + (3v_{21} - v_{03})^2, \\ f_{44} &= (v_{30} + v_{12})^2 + (v_{21} + v_{03})^2, \\ f_{45} &= (v_{30} - 3v_{12})^2(v_{30} + v_{12})[(v_{30} + v_{12})^2 - 3(v_{21} + v_{03})^2] \\ &\quad + (3v_{21} - v_{03})(v_{21} + v_{03})[3(v_{30} + v_{12})^2 - (v_{21} + v_{03})^2], \\ f_{46} &= (v_{20} - v_{02})[(v_{30} + v_{12})^2 - (v_{21} + v_{03})^2] + 4v_{11}(v_{30} + v_{12})(v_{21} + v_{03}), \\ f_{47} &= (3v_{21} - v_{03})(v_{30} + v_{12})[(v_{30} + v_{12})^2 - 3(v_{21} + v_{03})^2] \\ &\quad - (v_{30} - 3v_{12})(v_{21} + v_{03})[3(v_{30} + v_{12})^2 - (v_{21} + v_{03})^2]. \end{aligned}$$

8.1.16 Convex hullfeatures

$$f_{48} = \frac{A_H}{w * h}, \quad f_{49} = \frac{L^2}{A_H}$$

8.2 Online Database of Hand-draw Musical Symbols

We will now show the accepted paper involving this work.

Online Database of Hand-draw Musical Symbols

Rui Silva ^{#1}, Jaime S. Cardoso ^{*2}, Ana Rebelo ^{#3}

[#] INESC TEC - Porto, Faculty of Engineering of the University of Porto
Campus da FEUP, Rua Dr.Roberto Frias, 378, 4200 – 465 Porto, Portugal

¹ slyzer.rui@gmail.com

² jaime.cardoso@fe.up.pt

³ ana.m.rebelo@inescporto.pt

Abstract— Programs analogous to optical character recognition systems, called optical music recognition (OMR) systems, have been under intensive development from many years. Nowadays, an interesting application of OMR concerns the online recognition, which allows an automatic conversion of text as it is written on a special digital device. This work intends to create an online database of hand-draw musical symbols in order to study and develop classifiers to recognize musical characters in real time.

Keywords—optical music recognition, online omr, sheet music draw recognition, hand-draw musical symbols

I. INTRODUCTION

In terms of human past knowledge, music may be one of the few things which we are certain that follow us since pre-historic times. As a way of representation, musical score notation has always been the main source of musical expression for non-hearing systems.

A musical score notation is a worldwide standard representation of music writing, grouped with a lot of different symbols and ways to represent a very diversity of sound representations. For purposes of safeguard, digitization has been the most common tool being used to preserve a musical score, offering easy duplications, distribution and digital processing. However, a machine-readable symbolic format from the music scores is needed to facilitate operations such as search, retrieval and analysis.

OMR – Optical Music Recognition has gain a lot of contributions along the years that come since the 80's [1], this research, mostly on offline mode, make us aware that OMR is still an important and complex field where knowledge from several field intersects.

Technology has grown exponentially along the years, and write score pieces in paper, start to be a secondary option when you have a large fan of possibilities in a computer, such as: editable scores, perfectly formatted and archived, as immediate sound response and all the other possibilities with a segmented standard music files, like MIDI (Instrument Digital Interface) or MusicXML (Music Extensible Markup Language). This led to some composers to start use computer as their main way to compose music. However, the reality is that, in an age of global technology, most of the composers still use the traditional “pen and paper” to write his pieces, saying that it still is the most intuitive, easier and faster way to do so [1].

This work proposes to conjugate the universal “pen-paper” metaphor with the news forms of mobile

technology, as we see with the growing popularity of portable small devices, such smartphones and tablets, and their power of computing. As pen-based interfaces are in wide expansion, there is a lack of applications taking advantage of this intuitive and ergonomic way to draw musical scores, where the user composes musical scores in a traditional way by drawing the symbols on the screen. This work propose to create an online database of hand-draw musical symbols, in order to save the data getting in a structural and temporal context, which is not possible in the offline mode.

II. STATE-OF-ART

A method for date input in a digital system through a digital pen applied to musical software appeared at the end of the 1990's with the Palm Pilot device [2] produced by Palm Inc. The Pilot was a small handle PDA (Personal Digital Assistant) where musical characters were inserted through a pen using a technique called Graffiti [2]. The first approach to online musical characters recognition was to create a digital pen as an interface of localization and selection, in place of the mouse, and exemplified by the work developed by [3]. Thereafter, the input data process relied on the concept of standard movements: the user needs to learn a new way to write music, not doing the same way as is used to do on paper [4, 5, 6]. Macè in [7] says that it is not possible to use a unique recognizer for all music symbols and suggest that is necessary to exploit dedicated recognizers, trying to use a mix of gestural and non-gestural symbols. Taubman in 2005 developed a system called MusicHand [8], aiming to help in the process of pitch and symbol segmentation, where the system upon entry of an unknown stroke, make the decision by comparison with a set of trained gestures, without hesitation or coaxing of the system. In 2010, Kian Chin Lee proposed a recognition process using a set of different HMMs (Hidden Markov Models) in a non-gestural approach where the users do not need to learn any special gestures for input [9].

Most of the documentation uses a *unistroke* (gestural approach, just one stroke) system, forcing the user to learn a new language, being an influence on the process of composition. On the existing *multistroke* (non-gestural approach, more than one stroke) systems, there is a lack of high level technics of image and pattern recognition, as also a set of features that were not exploited from the dynamic user handwriting.

The focus in this work will be on exploit most of the described flaws in a non-gestural approach and introduce other methods in order to improve the Online recognition process in OMR.

III. WORK

The project begun with a creation of a graphical user interface in android developer language, where were collected a several information of the user draws in order to create a user hand-draw database of music symbols.

The user begins to fill a set of parameters with his personal information in order to be recognized by the system. After that, with his own handwriting, he had to draw symbols, according with the images that the system was showing.

For every drawn symbol, it was saved: the action done by the user (touch, drag and lift), the position X and Y, the Time and the Pressure, for every amount of milliseconds (depending on the Android machine in use). For a complete set of 84 musical symbols, drawn 3 times, with a different thickness, for every user, we can obtain a set of different handwriting.

The choice done in the number of symbols, were based on the most common and globally ones in music notation. The different thickness was driven by the fact that everybody draws their symbols in a different way when the size of the pen is different, and we save that information in the database for further analysis. When a draw is finished, the system knows exactly where and when the path was made.

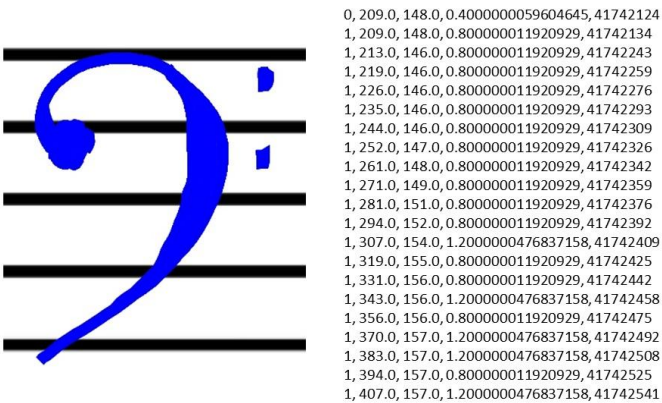


Fig. 1. Example of a hand-draw symbol and part of the data saved from it

We collected data from 50 users with ages between 21 and 70, from different origins and different music educations. From those users we saved the user name, date of registration, his institution or place of work and the hand writing preference. We describe the level of knowledge in sheet music handwriting of each user, from 1 to 5, receiving handwritings from different music experiences. For purposes of segmentation and symbol analysis we included also in the database the hardware model of the machine and the resolution being used.

IV. FUTURE WORK

With the data collected we will investigate and develop algorithms to recognize musical characters in real time to obtain a digital, easy-to-manage version of the original scores. More precisely, the methodologies for online OMR which will be explored in this work will encompass Support Vector Machines (SVMs), Neural Networks (NNS), Nearest Neighbour (kNN) and Hidden Markov Models (HMM).

These techniques will be used for a comparative and validation analysis with offline OMR procedures.

Moreover, the set of features used in HBF49[10] will be implemented in our database. These characteristics will enhance the advantages of using music symbols dynamically extracted.

The final purpose is to create a software with the possibility of instant recognition of the hand-drawn symbol and immediate transformation into a universal editable digital version. The aim is to eagerly know what symbol is been drawn before the user even finish it.

The process which is based on this technology, should be capable of recognize an unlimited number of different calligraphies, and obviously be an important step on the history of musical notation. This tool would be indispensable for composers, conductors, musicians and teachers.

V. ACKNOWLEDGMENTS

Part of this research was supported by the BEST CASE RL6 project Media Arts and Technologies (MAT) funded by ON.2.

REFERENCES

- [1] S Macé, E Anquetil, E Garrivier, B Bossis. "A Pen-Based Musical Score Editor" - In proceedings ICMC, Barcelona, 2005
- [2] Alsop, Stewart, "Innovative Graffiti Might Actually Make PDAs Useable", InfoWorld, September 26, 1994, p.130.
- [3] Silberger, K. 1996, "Putting Composers in Control." IBM Think Research 34(4). Available at http://domino.research.ibm.com/comm/wwwr_thinkresearch.nsf/pages/musiceditor496.html
- [4] H. Miyao and M. Maruyama. "An Online Handwritten Music Score Recognition System", Proceedings of the 17th International Conference on Pattern Recognition, Volume 1, pp. 461-464, 2004
- [5] Jamie Ansticel, Tim Bell, Andy Cockburnl, Martin Setchel. "The Design of a Pen-Based Musical Input System", Computer-Human Interaction, Proceedings., Sixth Australian Conference, Page(s): 260 – 267, 1996
- [6] A. Forsberg, M. Dieterich and R. Zeleznik. "The Music Notepad", ACM Symposium on User Interface Software & Technology, pp. 203-210, 1998.
- [7] Sébastien Macé, Éric Anquetil, "A Generic Approach for Pen-Based User Interface Development", Computer-Aided Design of User Interfaces V, 2007, pp 57-70
- [8] Taubman G, "Musichand: a handwritten music recognition System", technical report, 2005
- [9] Kian Chin Lee; Phon-Amnuaisuk, S.; Choo Yee Ting "Handwritten music notation recognition using HMM — a non-gestural approach", Information Retrieval & Knowledge Management, (CAMP), 2010 International Conference on, On page(s): 255 - 259
- [10] Adrien Delaye, Eric Anquetil. "HBF49 feature set: A first unified baseline for online symbol recognition" - Pattern Recognition, Volume 46 Issue 1, January, 2013, Pages 117-13

