Master research Internship

Internship report

Segmentation and recognition of symbols for printed and handwritten music scores

Domain: Document and Text Processing, Computer Vision and Pattern Recognition

Author:
Kwon-Young Choi

Supervisors:
Bertrand Coüasnon
Yann Ricquebourg
Intuidoc, Irisa, France
Richard Zanibbi
RIT, USA

Abstract: An OMR system objective is to transcribe an image of a music score into a machine readable format. A music grammar can be used in order to model the structural knowledge of music scores. Music scores can have a high density of symbols and the different preprocessing step can lead to wrongly segmented components. However, we don't want to tell the grammar how a music symbol should be segmented. Statistical model like convolutional neural network are much more suited for localizing and classifying music symbols In this report, we propose a way to combine a music grammar with a convolutional neural network architecture capable of doing localization and classification of music symbols. The grammar is used to generate small contextual zone that will be then fed to the neural network. This grammar can then also be used to bootstrap a dataset generation for the training of neural networks. We demonstrate the use of our model on the concrete task of recognizing accidentals in a music score and obtain 93.4% of localization accuracy and 96.2% of classification accuracy.

Contents

1 Introduction

Optical Music Recognition (OMR) can be described as a sub field of Document and Image Recognition. It has been studied for decades, starting from Pruslin [Pruslin, 1966] in 1966 and is still an active area of research. An OMR system objective is to transcribe an image of a music score into a machine readable format. A music score is constituted of many symbols arranged in a complex bi-dimensional structure. Music symbols has the property to be relatively simple, however a score has the particularity that almost every symbols are connected by five lines. Since 1994, the research team Intuidoc from the Irisa laboratory has developed a generic way of recognizing structured documents by using the DMOS system and have applied it on orchestral music scores. The team has already formalized the music notation grammar into rules understandable by the DMOS system. But there are still problems during the low-level image recognition task. Indeed, because of the density of information in a musical score and because of all preprocessing phases applied to the image, there are often cases of broken and overlapping musical symbols. In these cases, a simple classifier of well segmented symbols is not sufficient to correctly segment and localize the symbols.

However, the literature on statistical models like neural networks 2.3.2 shows us that such models has probably the capability to learn to localize and recognize such symbols. Despite their performances on local localization and classification tasks, we can't expect to train a neural network to recognize an entire page of music score. This is where the music grammar is very interesting for their capability to work with very complex, hierarchical structures. This complex structure and knowledge can then be used to generate local contextual zone that will then be fed to neural networks However, we don't want to tell the grammar how a symbol should be segmented. Furthermore, the grammar should hand down all the graphical recognition task to a neural network that will be able to localize and classify music symbols.

In this report, we will first review the state of the art of OMR in section 2.1 by introducing how a music score is structured, then we will review the state of the art of OMR by presenting all steps classically used in OMR at section 2.2: pre-processing, staff recognition and removal, segmentation and recognition of music symbols, music notation reconstruction. We will also introduce the reader to statistical models like hidden markov model and convolutional neural network used for symbols localization and classification in section 2.3 and 2.4. The DMOS system will also be introduce in section 2.6. We will then explained in section 3 our proposition to combine a simple music grammar and a convolutional neural network architecture in order to localize and recognize music symbols with a concrete application to recognize an accidental. Finally, we will expose in section 4 different experiments in order to demonstrate that we are able to localize and recognize a music symbols in an image with a local context.

2 Related works

In this section we will first define what is Optical Music Recognition and its main issues. Then we will describe all the steps commonly used for implementing an OMR system. In a second step, we will describe techniques for merging the recognition and segmentation of music symbol by using hidden markov model and neural network. We will also describe the use of such neural network on optical character recognition and object localization and recognition. Finally, we review in more detail the DMOS method as a generic method of document recognition.

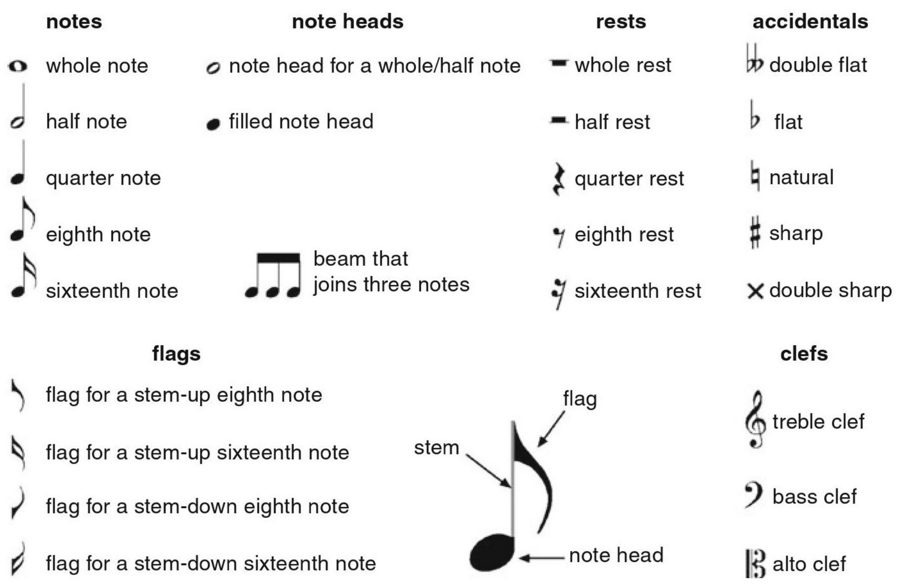


Figure 1: List of the main musical symbols [Fornés and Sánchez, 2014].

2.1 Definition of Optical Music Recognition

Nowadays, digitizing scores is a common practice to save and share music. But the format of these documents cannot be directly interpreted by a computer. That is why Optical Music Recognition systems have been developed for converting an image of a score to a format that is understandable by a computer. This computer vision task could be therefore classified as a hybrid domain between Optical Character Recognition and image recognition. OMR is principally used to save time because manually transcribing a score in machine readable format is very time consuming. Many music composers still find awkward to directly write music with a music composer software and prefer to write their music on paper. The use of OMR in these situations could integrate into the working flow of music composer their habit to write music on paper and automatically transcribe them in a computer readable format.

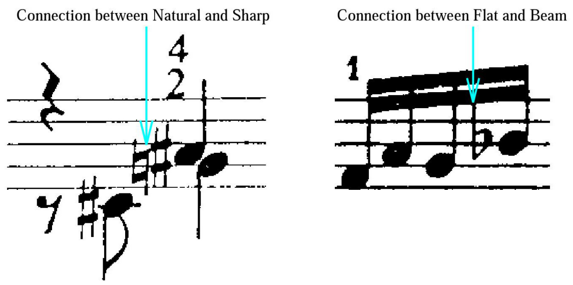
A score is a structured document used to formalize music. Its organization is hierarchical, for example:

- Different staves are used for different instruments or voices;
- Each staff contain five lines;
- A staff contains music symbols;
- A music symbol like a note is constructed by the association of graphical primitives like a dot, stem and flags . . .

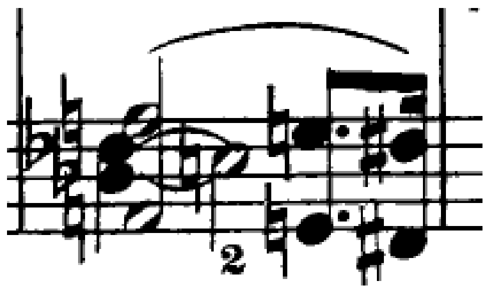
There are multiple symbols in musical notation resumed in figure 1: clefs, notes and rests, breaks, accidentals and key signatures, time signatures, dynamics . . .

OMR systems are faced with multiple problems such as:

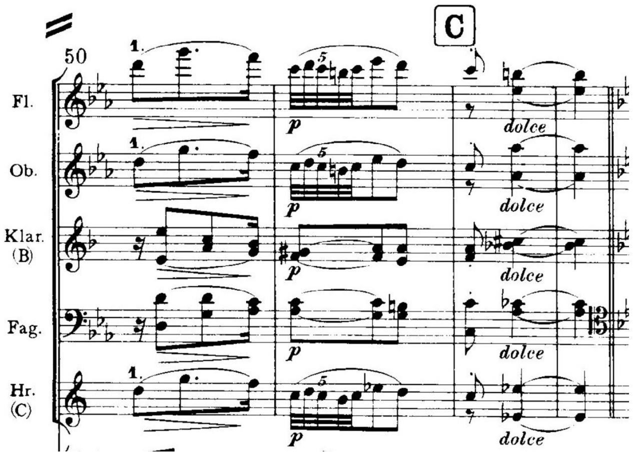
- The high density of symbols;


Connection between Natural and Sharp
Connection between Flat and Beam

(a) Examples of symbols that should not touch [Coüasnon et al., 1995].



(b) Examples of lots of cascading accidentals.



(c) Example of vertical synchronization of an orchestral score.

Figure 2: Examples of some OMR main problems.

- High connectivity between symbols;
- High variation of symbols;
- Overlapping and broken symbols like in figure 2b.

A score has a unidirectional direction of reading that corresponds to the time flow, but multiple things can happen simultaneously. For example, when there are multiple instruments in the same score, all music symbols are vertically synchronized across staves as illustrated in figure 2c. As a consequence, the output of an OMR system is not just a one dimensional sequence of symbols. All graphical symbols must be localized horizontally and vertically in order to deduce their relations between each other.

Authors like in [Rebelo et al., 2012] or [Fornés and Sánchez, 2014] generally describe an OMR system by identifying multiple steps: pre-processing, staff recognition and removal, symbol segmentation, symbol recognition, music notation reconstruction. A typical architecture of an OMR system is resumed in figure 3 at page 4. We will first review all these different steps, however, as said in [Couasnon and Camillerapp, 1995], an OMR system constituted of these sequential steps, also described as a bottom-up system, is very limited because the segmentation and recognition is done

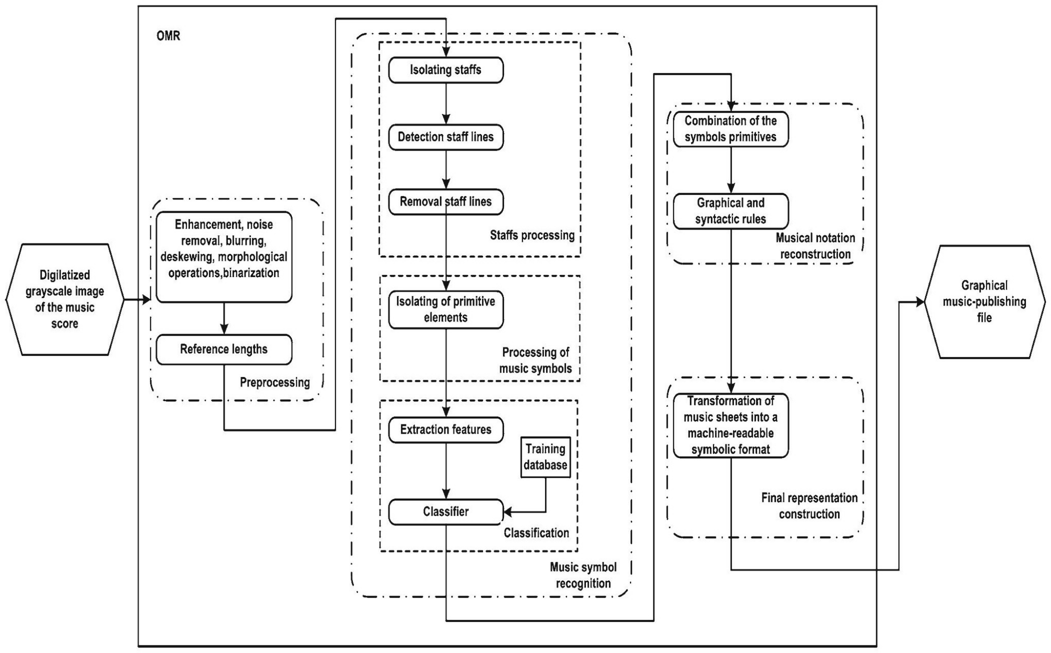

OMR
Digitalized grayscale image of the music score
Enhancement, noise removal, blurring, deskewing, morphological operations,binarization
Reference lengths
Preprocessing
Isolating staffs
Detection staff lines
Removal staff lines
Staffs processing
Isolating of primitive elements
Processing of music symbols
Extraction features
Training database
Classifier
Classification
Music symbol recognition
Combination of the symbols primitives
Graphical and syntactic rules
Musical notation reconstruction
Transformation of music sheets into a machine-readable symbolic format
Final representation construction
Graphical music-publishing file

Figure 3: Typical architecture of an OMR system [Rebelo et al., 2012].

without using the context. That's why we will also present more in details the DMOS system at the end of this part because of its special architecture that can use the context for segmenting and recognizing music symbols. Also, we will particularly focus our attention on symbol segmentation and recognition because it is the central theme of this internship.

2.2 Different step of OMR

2.2.1 Pre-processing

The pre-processing phase of OMR consists of making some operations on the raw image to ease the latter steps of OMR. Before anything, the image has to be converted into a binary image. This operation is a classical step of computer vision tasks. Then, optional operation can be applied to enhance the quality of the score.

- Skew correction for staff line: used for simplifying the staff recognition and removal step;
- Noise removal for bad scanning and digitization, for example see the figure 4 on page 5;
- Some other morphological operations ...

2.2.2 Recognition and removal of music staves

One characteristic of a music score is that most of the symbols are connected by the five lines that constitute a staff. Lines and spacing are used for describing a note pitch, therefore it is essential to recognize these lines for the later music notation reconstruction step. Also, these lines are an

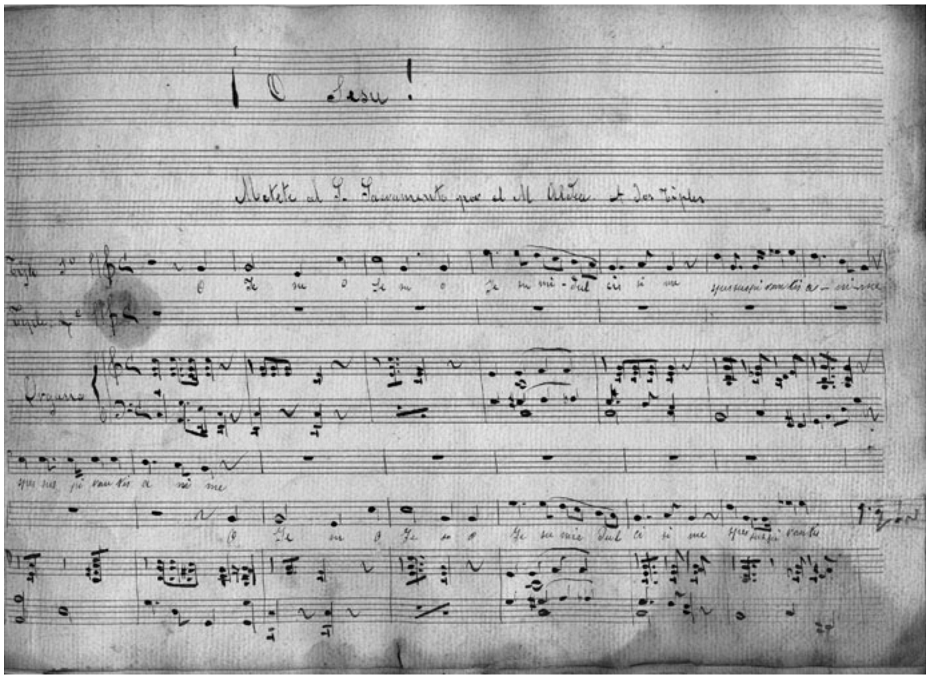


Figure 4: Old and damaged handwritten score [Fornés and Sánchez, 2014].

obstacle for almost every classical technique of segmentation and recognition used in computer vision. This is why, most OMR systems first remove these lines.

A fast way to do this is to do a horizontal projection of a music staff. The resulting vertical position of an intensity peak is an indicator of the position of a line. It is worth noting that projections techniques do not work well if the score contains a high density of symbols or if lines are curved. Also, the removal task is often done with a thickness criterion and often result in breaking symbols. Candidate assemblage and contour tracking are another way to remove a staff that are relatively fast to compute and can sometimes deal with overlapping symbols. Graph path search techniques can produce really good results, but are slow to compute. Finally, detection of lines in an image can be done by using complex filters like a Hough transform or a Kalman filter like the one used in the DMOS system [d'Andecy et al., 1994]. The Kalman filter used in DMOS has the capability to recognize curved line, useful to detect handwritten line.

2.2.3 Music symbol segmentation and recognition

Music symbol segmentation and recognition are a very important part of an OMR system. It is still a challenging part because of the high number and high variability of music symbols and its complex bi-dimensional structure. This complexity is often aggravated because of bad digitization and paper degradation for ancient scores like the figure 4. Furthermore, symbols are often deteriorated because most of OMR systems do the staff removal step that often leads on breaking symbols. Broken symbols then make the segmentation steps more challenging, even more if the context information is not taken into account.

Techniques used by OMR systems during this step are very diverse and many solutions have been proposed. However, most of OMR systems first does a segmentation step and then apply recognition techniques to extract the correct symbol. A simple way of segmenting a score is by first extracting graphical primitives like lines, blobs, circles . . . and then joins these graphical primitives for the recognition stages. Symbols can be extracted and recognized by using template matching techniques. But these kinds of techniques have many drawbacks as they are slow to compute and have a really low robustness when recognizing a symbol. One way to recognize a symbol from a bounding box candidate is by computing simple operation that will result in simple features of the region of interest and then compare these features with those same features computed from symbols models. This same philosophy can be used with different techniques by using symbol descriptors like centroids, Zernike moments, and decision trees.

The use of classifiers is yet another method to recognize a music symbol and four of them are compared in [Rebelo et al., 2009]. One of them is a Hidden Markov Models and they have been extensively used for optical character recognition and speech recognition. They have attempted to apply a Hidden Markov Model for recognizing very simple and old scores following the proposition of [Pugin, 2006]. Their proposition has the interesting property to merge the segmentation task and the recognition task. In the contrary, the following classifiers also presented in [Rebelo et al., 2009] need a prior segmentation before the recognition can take place. This constraint is very limiting because many errors can happen in the segmentation phase because of broken and overlapping symbols. Indeed, the use of the context is essential in correctly segmenting a music symbols. The use of simple neural networks like a multi-layered perceptron also have been reviewed for recognizing music symbols. The classic backward-propagation algorithm has been used for the training algorithm. Disappointing results was produced as they were outperformed by some simple classifiers as the k-NN classifiers. Nowadays, new neural network architectures have been proposed in similar domain like OCR, Speech Recognition or Computer Vision, but to our knowledge, they have never been used in the field of OMR. The k-nearest neighbor algorithm used in this context is the simplest classification algorithm used. They implemented it using Euclidean distance and it gave the second best result after the Support Vector Machine (SVM) classifiers. SVM used a radial basis function network and at the time of the study, it was the classifier that gave the best result. For this internship, we are looking for new ways of recognizing broken and overlapping symbols and one of our hypothesis is that by merging the segmentation and recognition phase, we will enable the recognition of those badly segmented symbols and consequently improve the overall recognition rate of music symbols. That is why we will review more in detail the HMM proposition in section 2.3.1.

2.2.4 Music notation reconstruction

The music notation reconstruction has the specific task to interpret the spatial relationship between different primitives or symbol recognized. This kind of algorithm has to deal with the complex two dimensional structure of music notation. As a consequence the positional information of a music symbol is very important as well as the context of the symbol.

OMR system has used fuzzy models or grammars to formalize musical notation. We can differentiate two kinds of grammars. Rule based grammar, or graph based grammar. Most of the systems that use a grammar for formalizing the music grammar are only using an ascending recognition methods. All steps of preprocessing, staff recognition and removal, symbol segmentation, symbol recognition and finally music notation reconstruction using a grammar take place sequentially and therefore limit the use of the context in the segmentation and recognition phase. In this work, we will focus on rule based grammar by presenting the DMOS system [Couasnon, 2001] in section 2.6

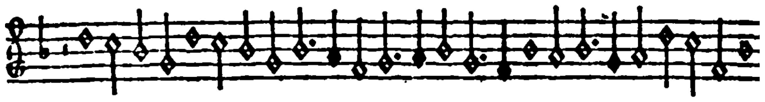


Figure 5: Example of old score recognized in [Pugin, 2006].

as it is the system that will be used during this internship.

2.3 Merging recognition and segmentation phase for symbol classification

Our hypothesis is that by merging the segmentation task at the symbol level, we will enable the recognition of broken and overlapping symbols and therefore improve the recognition rate of our system. An attempt has been made in this direction by using a Hidden Markov Model for recognizing old and very simple scores.

2.3.1 Hidden Markov Model applied to OMR

Authors in [Pugin, 2006] present a Hidden Markov Model for recognizing old and very simple scores as show in figure 5. They decided to avoid the staff removal step and the segmentation step because they considered that these steps are important sources of errors. They trained their model with a dataset of 240 pages of music scores containing 52,178 characters corresponding to 175 classes symbols. The scores were printed with two music fonts made up of very different graphical symbols. Partitions were manually annotated with graphical information. The recognition process used imitates speech recognition as they used a sliding window for scanning the score. They simplified the problem by only having a succession of notes from left to right. As a consequence, their system couldn't handle chords because the model was sequential and unidirectional. They decided to use 6 features from the raw score image and used the Baum-Welch algorithm for training the HMM. Here is the list of the 6 features: number of connected components, second and third features are functions of the gravity centers of the image, the largest black element, the smallest white element, total area of black elements. The recognition rate obtained on the training data was around 96% but this rate is highly dependent on the feature chosen, the sliding window width, . . . However, authors don't explicitly precise if the training data and the validation data are the same or different. If the two sets are the same, the generalization capability of this system is called into question.

This method has the advantage to avoid explicit segmentation by using a Hidden Markov Model. But this model will be hard to be applied to more complex scores as they have simplified the problem by using only unidimensional scores. Indeed, the objective of this internship is to work on bigger and complex orchestral scores. The HMM is also known for having difficulties to use context information because of their architecture principle. This is why, we will introduce in section 2.3.2 another way to make a joint segmentation and recognition using convolutional neural network.

2.3.2 Convolutional neural network

A convolutional neural network is a kind a neural network specifically designed to work with images. These networks compute small local features of the input data and gradually build more complex features by stacking multiple convolutional layers.

A typical architecture of a convolutional neural network is composed of two type of layers. A convolutional layer that can take two dimensional data as input and apply multiple convolutional

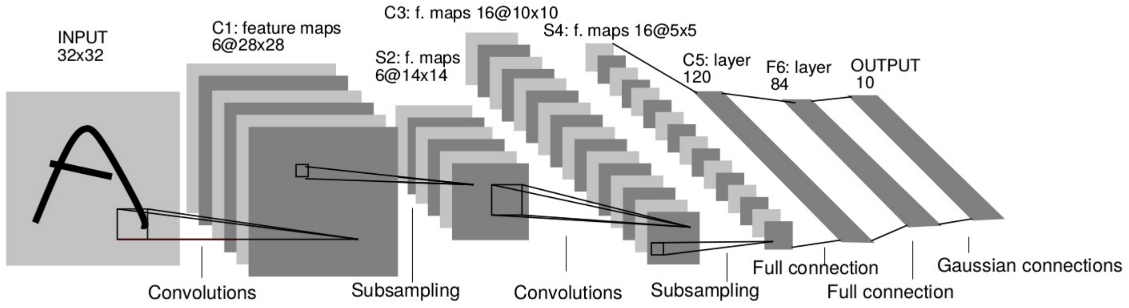


Figure 6: Convolutional neural network with successive convolution and subsampling [Lecun et al., 1998].

filters. The weight of each neurons constitute the convolutional kernels of each filters. To compute the output of the layer, the results of the convolution is biased and feed to an activation function. A convolutional layer can therefore be seen as an extractor of learned features. Following a convolutional layer, a pooling layer is used to reduce the dimension of the input data. This operation can be viewed as a subsampling operation and is used to create high-level features as the network architecture goes deeper. Finally, a fully connected layer of neuron, also called a dense layer is applied to calculate the output of the architecture. This layer organization is illustrated in figure 6, which is the architecture named LeNet-5 proposed by Yann le Cun in [Lecun et al., 1998] to recognize digit.

These two layers are used consecutively and repeatedly in order to make a deep neural network capable of doing, for example, classification and regression problems. However, some activation function and objective function will be more adapted for doing either classification or regression task. Often, we will choose the rectify activation function $\varphi(x) = max(0, x)$ for convolutional layer and a final softmax function $\varphi(X)_j = \frac{e^{X_j}}{\sum_{k=1}^{K} e^{X_k}}$ in conjunction of an objective function that will minimize the categorical cross entropy of the prediction The rectify activation function $\varphi(x) = max(0, x)$ for convolutional layer and a final softmax function $\varphi(X)_j = \frac{e^{X_j}}{\sum_{k=1}^{K} e^{X_k}}$ will particularly well suited for doing classification task. And often, we will try to minimize the categorical cross entropy between the predictions and the targets during the training. On the other hand, when doing a regression task, we will try to minimize the mean squared error.

2.3.3 Spatial transformer network

As we saw earlier, a convolutional neural network is a very powerful class of model to be applied on image classification and regression problems. The use of simple pooling layer is used to reduce the dimension of the data flow in the network. This can guarantee a relative spatial invariance of the features in the deeper level of the network. However, in the early layers of the network, all features are closely linked to a small local context in the image. This leads to the fact that a convolutional neural network is dependent on the spatial position of an object in the image. In [Jaderberg et al., 2015], the author present a new kind of neural layer allowing the network to make an explicit spatial manipulation on the input feature map. The manipulation of the input feature map is only conditional on the feature map itself and no other ground truth data is used to train the network. One advantage of this layer is that it can be seamlessly integrated into an existing neural network layer. This network can be used to improve the accuracy of a classification task by cropping

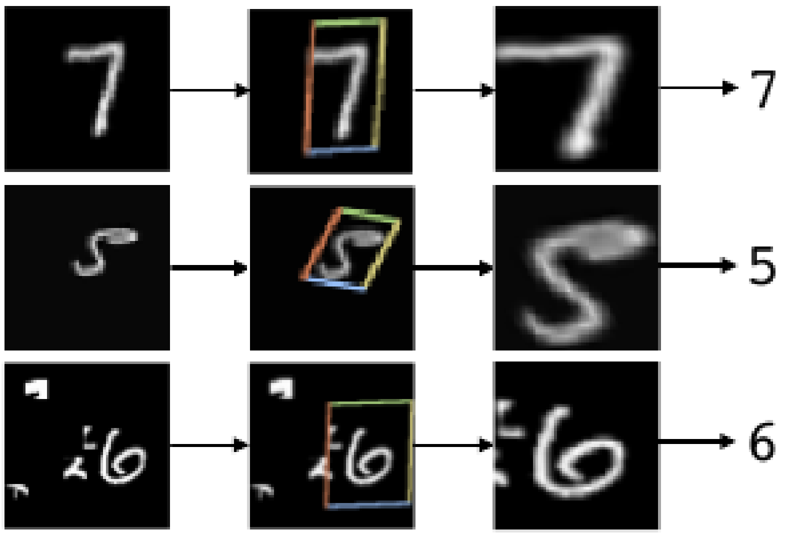


Figure 7: Example of output of the spatial transformer network. The first column is the network input, the second column show the geometrical transformation applied to the input image, the third column is the output of the geometrical transformation and finally the last column is the class prediction of the network.

and scaling the appropriate region of an image in order to remove the background noise of an object like presented in figure 7. The author presented other kind of applications like a co-localization task where the spatial transformer network has to localize an object of unknown class present in multiple images. This network can also be used to make a spatial attention mechanism by applying a downsample factor that will allow more computation efficiency by reducing the input resolution of the classification part.

The architecture of a spatial transformer module is divided into three part. A localization network, a grid generator and a sampling module, see figure 8. The localization network goal is to produce the parameters of the transformation that will be applied to the input feature map. This regression task is solved by using either a fully-connected neural network or a convolutional neural network. The second part, the grid generator, is a special layer used to map the output grid of pixels to the input grid of pixels. This mapping can be performed by using any kind of geometrical transformation: affine transformation, thin plate transformation ... However, this transformation has to be differentiable in respect to their parameters. This allows the grid generator to be also differentiable and to backpropagate the gradient through the network. Also, this grid generator module can take a downsample factor, allowing the module to reduce the size of the output data and producing an attention mechanism. Finally, the last layer, called the sampling module, samples the input feature map following the grid generated by the grid generator. Again, one can used any kind of sampling method as long as the sampling method is derivable. It is because all the part of the spatial transformer network is differentiable that it can be seamlessly integrated into existing architecture of neural network.

The author presents a series of experiment to validate his model and has used data coming from the MNIST dataset. This MNIST dataset is dataset of handwritten digits containing 10 classes (0 to 9 digits) with a training set of 60,000 images and a test set of 10,0000 images. He first use

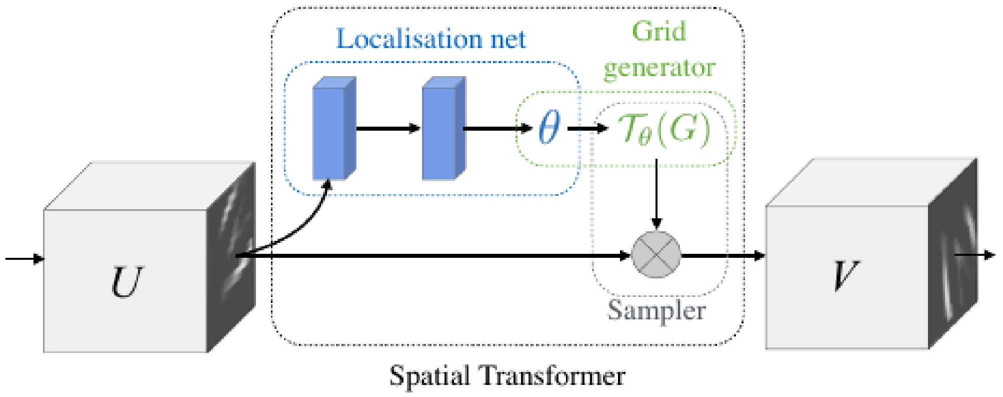


Figure 8: Architecture of the spatial transformer network.

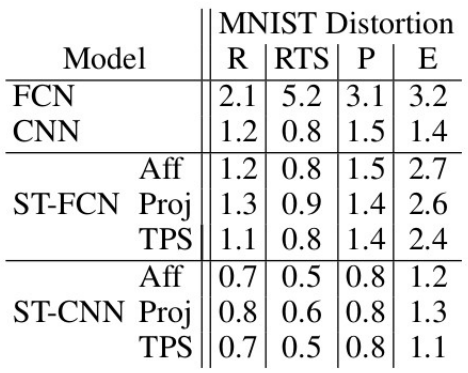
multiple distorted MNIST dataset produced by rotating, scaling, translating the digit and also by doing more complex transformation like projective transformation and elastic warping. He also used a cluttered MNIST dataset where the digit is into a bigger image containing noise. The classification part and localization network was tested with a fully connected neural network and a convolutional neural network making four combinations possible. Also, he compares the results of network having or not a spatial transformer, making a total of six possible experiments. The results in figure 1 shows that adding a spatial transformer network improve significantly the classification accuracy of the overall network. The transformation used by the spatial transformer network is an affine transformation allowing the network to crop, scale and rotate the image. This shows that by only taking appropriate region of an image, the classification task is better performed.

2.4 Joint object localization and recognition techniques

After reviewing some techniques used in OCR and Speech Recognition, it could be interesting to see what kinds of recognition systems are used for object recognition in an image. Nowadays, deep neural networks are widely used in object recognition and are known to be the state-of-the-art in image recognition [Erhan et al., 2014]. We will also review another proposition [Ba et al., 2014] that concentrate on modeling an effective visual attention model.

2.4.1 Deep Neural Network

In [Erhan et al., 2014], they used a Deep Neural Network for detecting and localizing multiple scalable object in an image. A common way to localize different object in an image is to train one object detector by class to recognize. As the number of classes grows, it becomes computationally difficult to scale up the system. The originality of this works is to train one single Deep Neural Network that is *class agnostic* and use it to localize any object of interest in an image. This Deep Neural Network is structured with multiple Convolutional Neural Networks and multiple max-pooling layers. The output is normalized with a soft-max layer. The recognition system proposed has the capability to output a bounding box to localize objects and a confidence score about the label of all objects detected. This proposition's strength is its property to handle naturally multiple instance of an object in an image. This network was benchmarked by using the Pascal Visual Object Classes (VOC) Challenge [Everingham et al., 2009]. They used the 2007 edition to evaluate



Model		MNIST Distortion			
		R	RTS	P	E
FCN		2.1	5.2	3.1	3.2
CNN		1.2	0.8	1.5	1.4
ST-FCN	Aff	1.2	0.8	1.5	2.7
	Proj	1.3	0.9	1.4	2.6
	TPS	1.1	0.8	1.4	2.4
ST-CNN	Aff	0.7	0.5	0.8	1.2
	Proj	0.8	0.6	0.8	1.3
	TPS	0.7	0.5	0.8	1.1

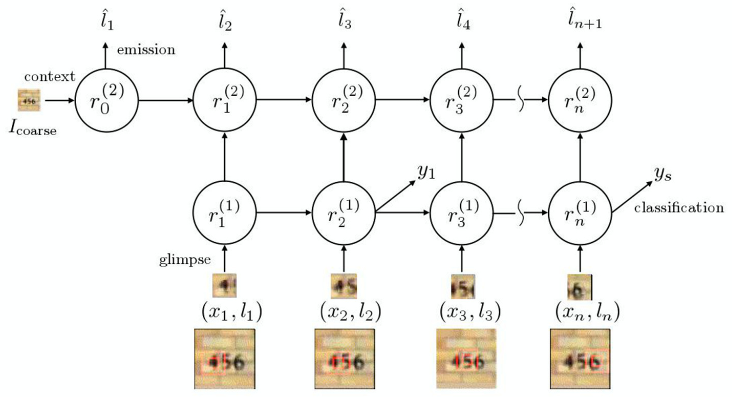
Table 1: Results of the MNIST experiment of [Jaderberg et al., 2015]. The results presented here is the error rate of each experiments. FCN means Fully Connected Network, CNN means Convolutional Neural Network. Aff, Proj, TPS are respectively the affine transform, projective transform and thin plate transformation used by the spatial transformer network. Different operation were used in order to produce different datasets : R rotation, RTS rotation translation and scaling, P projective distortion and E elastic distortion.

the model and the 2012 edition to train the model. The result presented shows that this system can be quite competitive although they do not achieve the best results. However, the computational cost of this system is much lower than the best recognition system.

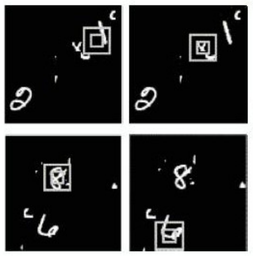
In the context of OMR, this kind of network could be interesting as it can localize precisely with a bounding box an object in an image. The fact that the bounding box is scalable is also interesting because there are multiple musical symbols that can be used at different sizes. Still, this kind of neural network can't yet localize a symbol inside of an entire page of music sheet. This is why the use of the grammar like the one used in this internship 3.1 is very interesting in order to facilitate the localization task of the network.

2.4.2 Multiple Object Recognition with Visual Attention

Authors of [Ba et al., 2014] proposed an attention-based model inspired by the human vision with the purpose to recognize multiple objects in images. Studies on human vision show that a human gradually construct its environment representation by focusing its attention on successive small parts of the image. This principle is used here by using a Deep Recurrent Neural Network that processes an image by taking *glimpse* at different resolutions. A glimpse is a small part of the image centered on a special location. Furthermore, they used reinforcement learning to train the network to learn about how to scan the image. The model can learn to localize and recognize multiple objects, although it was only trained with class labels. Jimmy Ba, Volodymyr Mnih and Koray Kavukcuoglu first validated the model by localizing and recognizing pairs of MNIST numbers as shown in figure 9b. The error rate of this experiment was 5% and it significantly outperformed the other models. To also test their model on a real world problem, they evaluated the system by using the model to transcribe house numbers from the SVHN dataset [Netzer et al., 2011]. The training set was composed of 200,000 images and the validation set of 5000 images. They obtained the lowest error rate of 3.9%. This model showed to be more accurate than state of the art convolutional networks, although the model is using fewer parameters and less computation. A diagram of this model is



(a) Deep recurrent attention model [Ba et al., 2014].



(b) Using the visual model to recognize MNIST numbers [Ba et al., 2014].

Figure 9: Visual attention recurrent model and examples of recognition of MNIST numbers

shown in figure 9a.

The Visual Attention model process the image in N sequential steps. For each step, the model scans a new location and extracts a glimpse at this location. From this glimpse, the model updates its internal representation of the image. The model uses a special classifier for recognizing objects. Finally, the model output a new location to scan from. The model is divided successively in sub-component and each of those components is a neural network. The first network is the glimpse network used to extract features from the glimpse with two input layer: a convolutional layer that take a glimpse as an input and a fully connected layer that take the location of the glimpse as an input. These two layers are then combined by multiplying the output of each layer. The next network is a recurrent network composed of two recurrent layers composed themselves of LSTM units. Its work is to aggregate information extracted from successive glimpses while preserving the spatial information. Then a third network called the emission network is used to predict the location of the next glimpse. A context network is used to compute an initial state of the system and provide the location of the first glimpse. Finally, a classification network is used to output a prediction from the RNN layer. It is composed of a fully connected hidden layer and a softmax output layer.

This kind of model could be useful in our OMR system. The fact that this model can locate and recognize symbol, although the network was only trained with class labels is interesting because in OMR, we only have databases containing class label symbols. Furthermore, this system can locate an object in an image, making it ideal for an OMR system. Without having this system to learn how to read a complex score, we could use the grammar to make a first rough segmentation and then use this recognition system for locating accidentals, key signatures, time signature, silences... Although this proposition could have been very interesting to study, I didn't have the time to experiment this model during this internship.

2.5 Multi-lingual printed and handwritten text recognition system

We already highlighted the fact that OCR is a source of inspiration for OMR, but the main difference is that the structure to recognize is much simpler in OCR. A score is a bi-dimensional structure

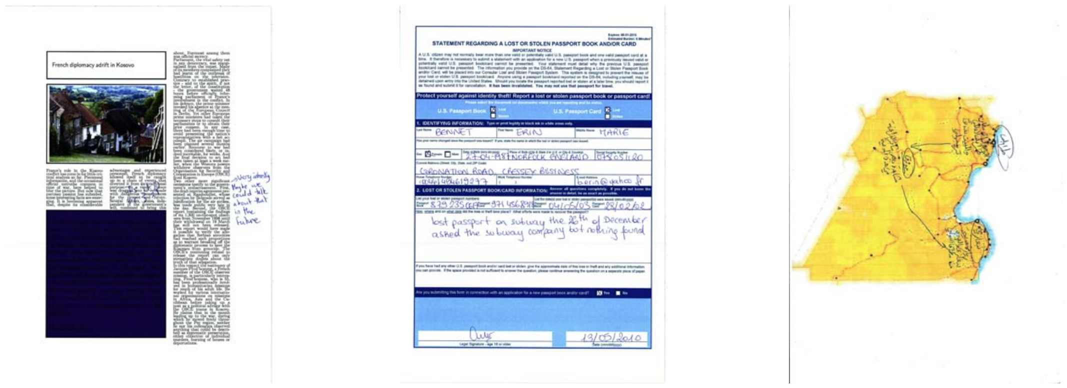


Figure 10: Example of a document from the Maurdor Challenge [Moysset et al., 2014].

with a notion of vertical synchronization and this notion is especially important for orchestral scores where there are many parallel staves for each instruments. However, this is only true at the symbol level. In fact, if we zoom out and consider an entire document, the recognition process can be much more complex. For example, the Maurdor evaluation is a computer vision and document recognition contest with some very hard data to recognize like the figure 10. A complete document processing chain is composed of: document layout analysis, write type identification, language identification, text recognition, logical organization extraction, information extraction. The A2IA team has submitted a text recognition module that ranked first at the second Maurdor evaluation for both printed and handwritten text recognition for French, English and Arabic [Moysset et al., 2014]. They used a Long Short Term Memory Recurrent Neural Network, with little changes:

- Adapted the sub-sampling filter size to fit the image resolution;
- Tuned hidden layer sizes for the dataset.

One important improvement to the training phase was the use of a powerful regularization technique called dropout.

However, the interesting part here is the fact that the DMOS system was used in the precedent module called the module layout analysis, and was used to detect line of text. In a complete recognition system, DMOS would be used to extract line of text that will be then fed to this recognition module. In the context of this internship, we could also use the powerful grammar present in the DMOS system to roughly segment a score and generate zones with broken and overlapping symbols. Then, we could feed these zones to a recognition system like the one we saw previously.

2.6 DMOS: a generic document recognition method

DMOS, "Description and MOdification of Segmentation", is a general method for the recognition of structured documents. Its main purpose is to separate the semantic and graphic knowledge of the program that do the recognition task because of the well-known paradox in computer vision that there is a "discrepancy between the way knowledge describe an object and the way objects have to be recognized" [Couasnon, 2001].

The system implemented in [Couasnon, 2001] is mainly separated in two parts. First, the system uses a grammar for formalizing the semantic and graphical organization of the documents. This grammar is defined in EPF (Enhanced Position Formalism) and then compiled to a Prolog dialect called $\lambda Prolog$. There are two levels of grammar:

- A physical level: formalization of the graphical organization of the documents;
- A logical level: work on the semantic level of the document.

The grammar first manipulates *terminals* that are simple graphical units like a line segment or a component (connected or not). It's by agglomerating these terminals that the grammar construct complex structures that will describe the graphical and the semantic organization of the document.

The second part is where the originality of the approach is found. A classical computer vision system that uses a grammar only uses it at the end of the recognition chain. First, some tools for segmenting and recognizing a symbol is used. Then, labels recognized are fed into the grammar for detecting possible recognition errors and performing the reconstruction of the semantic notation of the document. In the DMOS system presented by [Couasnon, 2001], the grammar totally guide the segmentation and recognition of the document. This grammar decides on how to segment the document, then feed the candidates into classifiers that will confirm the label of the candidate. The grammar can then check if the recognition is consistent, and if it is not, it can redo the segmentation step for finding a better solution. The grammar can therefore use the context for segmentation *and* recognition tasks.

For validating the modeling capacity of DMOS and its advantageous use of the context, this method has been initially implemented for recognizing orchestral scores. Furthermore, this system has been applied and validated on more than 700 000 complex documents of different types: archive documents, handwritten letters, forms, mathematical formula . . . However, this system is still having difficulties in recognizing broken and overlapping symbols in scores. That is why we propose in the following section a combination of syntactic recognition using DMOS and a localization and recognition module using convolutional neural networks. The grammar used by DMOS will generate zone with local context and pilot the neural recognition module in order to localize and classify broken and overlapping symbols. We also show a way to bootstrap the generation of a dataset by using a music grammar.

3 Proposed method for joint localization and recognition of music symbols

In this section, we present our proposition to do a joint localization and recognition of music symbols using a neural network. Neural networks, and especially convolutional neural networks, are a very powerful class of model for doing image localization and recognition tasks. However, the dimension of the image is static inside a neural network (inscribe in the architecture of the network). Furthermore, the dimension of the input data is directly connected the computation cost of the network. This is why it is currently impossible to use a convolution neural network on, for example, an entire page of music sheet. To resolve this problem, we used a simple music grammar that will enable us to generate a dataset of symbols, well segmented and badly segmented, with a small context. To resolve this problem, we used a simple music grammar that will enable us to generate a zone hypothesis where the network will have to localize, classify the music symbols and reject if there are no symbols. During this internship, we will only be using a minimal grammar in order to localize and classify accidentals. This will be enable us to make experiments to validate our model of neural network. We now present this grammar.

3.1 Simple music Grammar for accidental localization

In order to produce a dataset allowing to train a neural network to localize and to classify, we used the music grammar introduced in [Coüasnon and Camillerapp, 1994]. This grammar was adapted for the newer version of the DMOS system. As we introduced earlier, the DMOS system delegate all the low level recognition and segmentation process of the image to a specific framework called "Vision Precoce". This framework allowed us to extract multiple visible clue in a score:

- Staff lines: the staff lines was extracted by using a kalman filter and put into a specific layer;
- Note heads: note heads were extracted by using heuristics on the size and area of the note head; These note heads were also put into a specific layer to ease the music notation reconstruction process;
- Vertical segments: vertical segments presents in the score were also detected by using the kalman filter. These segment represents potential stem for the grammar;
- After the staff removal step, we can then extract all the remaining connected components that will be used to construct a note with an accidental. Before starting the construction of the musical notation with this grammar, we first filter all the connected components of the music score using a simple classifier of well segmented symbols presented in section 3.3.2.

Next, we will use grammar rules to construct a representation of the music notation. In the context of this grammar, DMOS also use the concept of layers. Each layers can be used to represent a different resolution and can be used to regroup components in order to ease the work of the grammar. A special operator `USE_LAYER layer FOR rule` is used to change the current layer for a certain rule. Other special operator are used, especially the operator `AT (zone) && rule` in order to search a component relative to another component.

First of all, we extracted the staff lines of the score. This allowed us to deduce the height of the staff interline which a very important parameter for the grammar and the for the dataset generation. These staff lines were put into a separate layer and a special connected component is build for each set of five staff lines to construct a complete staff.

```
% Detect five consecutive staff Line and construct an object CCportee that will
    represent a staff
detectPortee CCportee ::=
    detectLigne Ligne1 &&
    detectLigne Ligne2 &&
    detectLigne Ligne3 &&
    detectLigne Ligne4 &&
    detectLigne Ligne5 &&
    ``(constructCCPortee Ligne1 Ligne5 CCportee).
```

Once a staff has been built, we try to construct a note by first searching a potential stem. If a vertical segment has been found, then change the current layer to the note head layer in order to find a note head.

```
note_H NuGr Note _TypeTete TypeNote Duree SegHampe CCaff ::=
    % Searching for a potential stem
    hampe haut SegHampe &&
    % Change the current layer to the layer that contains note head and for a note
    head
    USE_LAYER(nomResol "Tete") FOR(tete_de_note_H CCtete Tete SegHampe) &&
    ‘‘(cons_elemVoixNot Note TypeNote NuGr Duree haut 1 [Tete]  CCtete SegHampe).
```

To search for a note head, we need to search at one of the extremity of the stem. If the note head is situated at the bottom of the stem, we will need to search in a zone that will be close, at the left and low extremity of the stem. The `AT (procheGextrB SegHampe` operator does that in the following code:

```
tete_de_note_H CCtete Tete SegHampe ::#
    AT(procheGextrB SegHampe) && (tete CCtete) &&
    USE_LAYER(nomResol "Music") FOR(alterationL CCtete SymbAlter) &&
    ‘‘(cons_tete Tete CCtete 0 _HautNote SymbAlter _NbPt _LstSymbPt 0 [])))).
```

Finally, when a note has been constructed, we search an accidental once again with `AT (procheGmemeL)` operator at the left, on the same line of the note head:

```
alterationL CCtete SymbAlter ::=
    AT(procheGmemeL CCtete) && alteration01 CCtete SymbAlter.
```

With this simple grammar, we can only construct a representation of a note with or without an accidental. In addition of this simple grammar, we used a simple convolutional neural network classifier presented in section 3.3.2 to filter connected component and put a label on accidentals.

3.2 Dataset generation using the music grammar

With the grammar presented in section 3.1, we generated a dataset that will be used to train a neural network that will localize and recognize an accidental. When the accidental is well segmented, the grammar won't normally have any problem to construct the note with an accidental. However, when the accidental is not well segmented, the grammar won't be able to construct a correct representation of the note. Nonetheless, the grammar still have multiple information that could be of great use for the neural network. For example, we know that the vertical position of the accidental is directly dependent of the vertical position of the associated note head, see figure 11. However, the horizontal position of the accidental is dependent of the local context of the score, see figure 12.

When the grammar is searching an accidental, it already knows the position of the note head. So it would be very interesting to send this information to the neural network to ease his localization task. To do this, we took thumbnail of region that could contain an accidental. The vertical position of the thumbnail is centered on the vertical position of the note head and is horizontally positioned at the left of the note head. The size of the thumbnail is decided from the height of the staff interline. The height of an accidental is around three times the height of an interline, so we took a squared image of 6 times the height of an interline to be sure not to cut a part of the accidental.

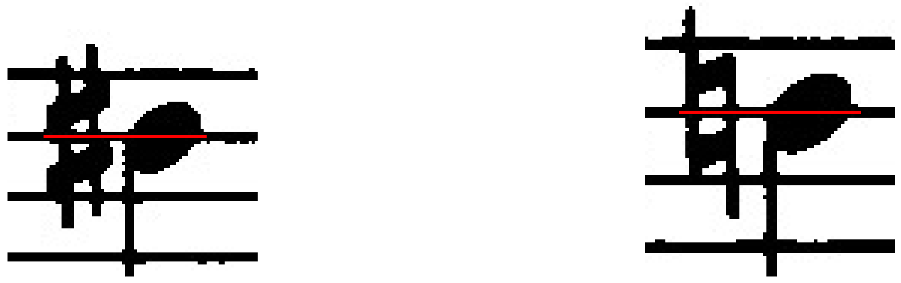


Figure 11: The vertical position of an accidental is directly dependent of the vertical position of the associated note head.

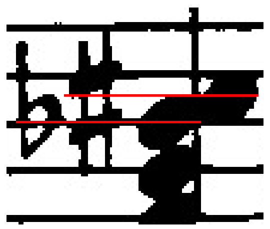


Figure 12: The horizontal position of the accidental is dependent of the context.

The generation of the dataset was done from five different music sheets, for in total of 70 pages of scores. A particular style of music score were chosen in order to find many accidentals like in figure 13.

This dataset contains examples of well segmented symbols as well as examples of not well segmented symbols. We also added a reject class that will permit the neural network to learn whether an image contains or not an accidental. The figure 14 presents examples of images contains in this dataset.

At the end of this work, during the experimentation part, we will need to evaluate the localization accuracy of the network. However, we only have the ground truth data of the localization of the accidental only on images of well segmented symbols. That's why we will use different part of this dataset for different task:

- When we will have to explicitly use in the experiment the ground truth localization of the accidental, we will only use images of well segmented accidental;
- On other cases, we will use the whole dataset.

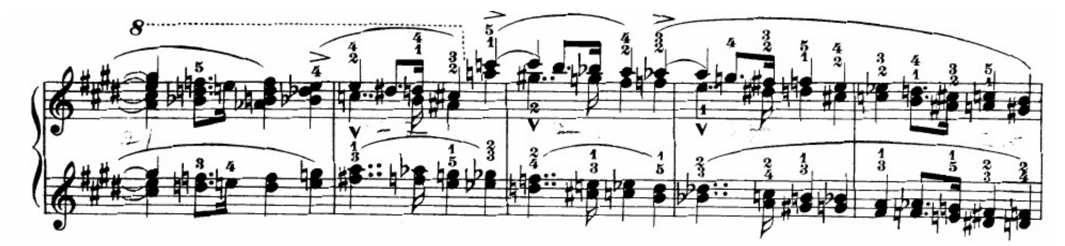
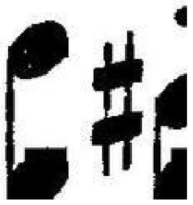
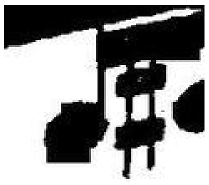
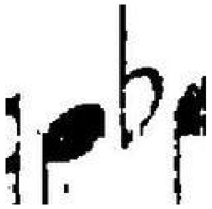
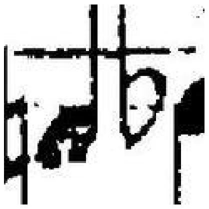


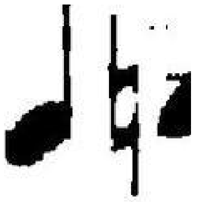
Figure 13: Score containing many accidentals.

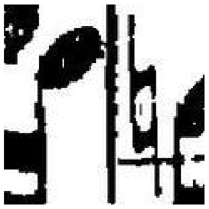

(a) Well segmented sharp example.


(b) Not well segmented sharp example.


(c) Well segmented flat example.


(d) Not well segmented flat example.


(e) Well segmented natural example.


(f) Not well segmented natural example.

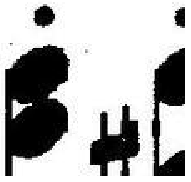

(g) Example from the reject class.

Figure 14: Example of images in the dataset generated by the grammar.

This dataset is described in table 2

3.3 Neural network architecture proposition

Our neural network architecture proposition is strongly inspired from the one proposed by Jaderberg [Jaderberg et al., 2015]. We will first describe our localization network used to make segmentation prediction. Then, we will expose our classification network that was used in the grammar to recognize well segmented symbols and in conjunction with the localization network. Finally, the full architecture composed of the localization network and the classification network is presented in section 3.3.3.

3.3.1 Localization network

This localization network is used to produce an segmentation hypothesis of an accidental. In order to do this regression task, we sued a convolutional neural network composed of a two dimensional

Label	WS accidental	Not WS accidental	Total
JUNK	0	1965	1965
BECARRE	697	256	953
DIESE	621	150	771
BEMOL	143	98	241
Total	3426	2469	3930

Table 2: Images distribution by class for the dataset generated from the grammar. The first column contains the number of Well Segmented (WS) images. The second column contains the number of not Well Segmented images. The last column contains the total of images in the whole dataset. We only have the ground truth localization on well segmented symbols. Examples of images are presented in figure 14

input layer of size 120x120 neurons. Then we used two consecutive layer of pooling and convolution with pooling size of 2x2, 20 convolutional filters of size 5x5. The end of the network is composed of two fully connected layers of 50 neurons and finally 6 outputs neurons. We used the mean square error as the objective function and trained by using the adam optimizer.

The output of this network is composed of 6 neurons, where each output represents a geometrical transformation matrix parameter T_n: $\begin{bmatrix} T_1 & T_2 & T_3 \\ T_4 & T_5 & T_6 \end{bmatrix}$
T_1 and T_5 parameters determine the zoom, T_3 and T_6 parameters determine the translation, T_2 and T_4 parameters determine skewness. In order to simplify the task of the neural network, we will ignore the skewness parameter in the transformation.

3.3.2 Classification network

In order to classify the music symbols and reject the image if it doesn't contain any symbols, we used a convolutional neural network with the following architecture:

- A two dimension input layer. The size of the input layer is dependent of the experiment. When this network is used in conjunction of the localization network, the size is calculated from the input size of the localization network and the downsampling factor of the spatial transformer network. For example, if the input size of the localization network is 120x120 neurons and the downsampling factor is 2, the input of this network will be 60x60 neurons;
- Two consecutive convolutional and pooling layer with 32 convolutional filters of size 3x3 and a pooling size of 2x2.
- A dense layer containing 256 fully connected neurons.
- The final layer contains as much as neurons as class to classify. For example, in the first experiment 4.1, this layer contained 18 neurons in order to classify all the 18 classes of music symbols present in the dataset 3. For our final example 4.2.2, we only use four neurons in order to classify three kinds of accidentals or reject if no accidentals is present in the image.

Label	Number of images
MEZZO	38
CLE_FA	258
CLE_UT	20
FORTE	191
SOUPIR	656
PIANO	236
POINT	295
BECARRE	1394
BEMOL	2092
PAUSE	126
CLE_SOL	904
DMSOUPIR	719
CHEVRONV	33
CHEVRONH	356
DIESE	1886
FINPEDALE	273
TALON	52
PEDALE	230
Total	9759

Table 3: Images distribution by class for the dataset of well segmented symbols.

- When doing a classification task, it is very interesting to combine our final layer with the softmax activation function in order to produce a probability distribution over the output of the network.
- Finally, the loss used to measure the accuracy of this network was the categorical cross entropy function. The training update was the adam optimizer, the same as the localizer network.

3.3.3 Full architecture

In order to do a joint localization and classification of music symbols, we merged the two previous architecture into a single, end to end trainable, neural network. The junction between the localization network and the classification network was done using the spatial transformer network introduced in section 2.3.3. Once the localization network has produced a set of transformation parameters that localize the symbols, the spatial transformer network is used to transform the input image. A down sample factor is used in order to reduce the input size of the classifier network. This factor creates a way to make a visual attention mechanism and make use of multiple resolution inside the same network. This transformed image is then fed to the classifier in order to recognize a symbol or reject the image. We used this network in different ways during our different experiments. In the experiment 4.2.1, this network was used directly. However, in the experiment 4.2.2, we first directly trained the localization network to regress the transformation parameters. We then loaded and fixed the learned parameters of the localizations network and learned the classification task only.

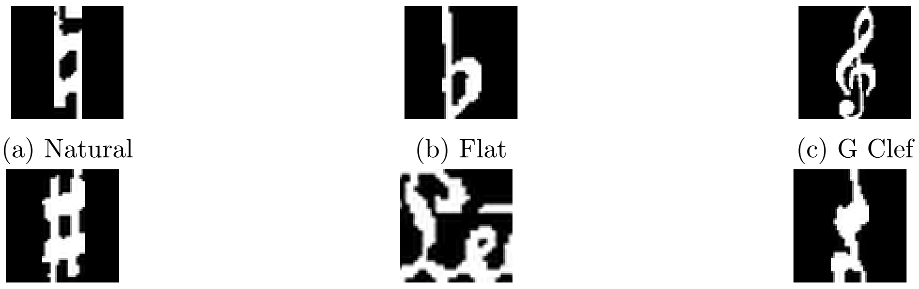


Figure 15: Example of images in the well segmented dataset

4 Experiments

In order to validate the model, we have conducted multiple experiments that will be presented in the following section.

4.1 Classification of well segmented music symbols

The goal of the first experiment was to make a classifier of well segmented music symbols. This experiment informed us about the level of difficulty for neural networks to classify music symbols. The dataset used was generated from older data used in a precedent internship and from data already present in the intuidoc team. One difficulty in making a dataset of music symbols is the fact that there is a large variation of size between class of music symbols. For example, a G clef will be around 15 times taller than a simple point. Here we chose the simple solution to resize every symbols to the same size. However, the output size cannot be chosen arbitrarily because if the output size is to far from the original size of the symbols, an aliasing effects will be created and the symbols will be too damaged to be recognized. To choose a good output size, we reasoned in term of staff interline height. Indeed, almost every symbols' height in a score is relative to the staff interline height of the score. For example, we can quantify that a G clef is 8 times the size of an interline, an accidental is three time the size of an interline and a point half the size of an interline. We chose here to take the middle of these two extreme and set the output size to three time the size of an interline. We didn't constrain the width of symbols and kept the original ratio between the height and width of the images. As a consequence, the symbols was always placed at the center of the frame. We considered here that the resolution of an interline was 20 pixels, which is a reasonable value to have. Therefore, we have squared images of size 60x60 pixels with centered symbols of height 60 pixels and variable width. The table 3 resume the image distribution of the dataset over all the class of the dataset.

The architecture we took was the classifier network presented in the section 3.3.2 which is a simple convolutional neural network composed of two layers of convolution and pooling followed by a dense layer. The dataset has a total of 9759 images with a very irregular distribution of images between classes. For a supervised training of convolutional neural network, it is considered as a small dataset. We therefore need to test thoroughly the training on the whole dataset. We used the cross validation technique by splitting the dataset into five part. For each of the five runs, three splits of the dataset was used for training, one split for validation and one split for testing. The validation part was used to validate the accuracy of the network at each epoch of the training.

Label	Exact	Error	Accuracy
MEZZO	38	0	100%
CLE_FA	258	0	100%
CLE_UT	20	0	100%
FORTE	191	0	100%
SOUPIR	655	1	99.8%
PIANO	236	0	100%
POINT	295	0	100%
BECARRE	1388	6	99.6%
BEMOL	2092	0	100%
PAUSE	126	0	100%
CLE_SOL	903	1	99.9%
DMSOUPIR	718	1	99.9%
CHEVRONV	32	1	97%
CHEVRONH	354	2	99.4%
DIESE	1877	9	99.5%
FINPEDALE	272	1	99.6%
TALON	52	0	100%
PEDALE	230	0	100%
Total	9737	22	99.8%

Table 4: Result of testing the whole dataset with the music symbol classifier by using cross validation.

An early stopping mechanism was put in place to automatically detect over-fitting and stop the training. At the end of the training, a split was used to definitely test the training. For each run, the testing split was different which result in testing the whole dataset. The results presented in table 4 shows that with 99.8% of accuracy, this is an easy task for a convolutional neural network. It is this network was used within the grammar in order to make classification hypothesis on whether the connected component was an accidental or not.

4.2 Localization and classification of accidental

We will now introduce experiments done in order to make a neural network architecture able to localize and classify accidental symbols.

4.2.1 Joint localization and classification training

This experiment was done in order to find how the spatial transformer will react when working with music symbols with background noise coming from real scores. We used in this experiment the whole architecture introduced in 3.3.3. The dataset used is the whole dataset presented in section 3.2. The training was done with the same protocol presented in section 4.1. We divided the dataset into five splits for doing a cross validation on the entire dataset. Three splits was used to train the network, one split for validation and early stopping and finally one split for testing. The results are

Label	Exact	Error	Accuracy
BEMOL	218	23	90.5%
JUNK	1946	19	99%
BECARRE	927	26	97.3%
DIESE	758	13	98.3%
Total	9737	22	99.8%

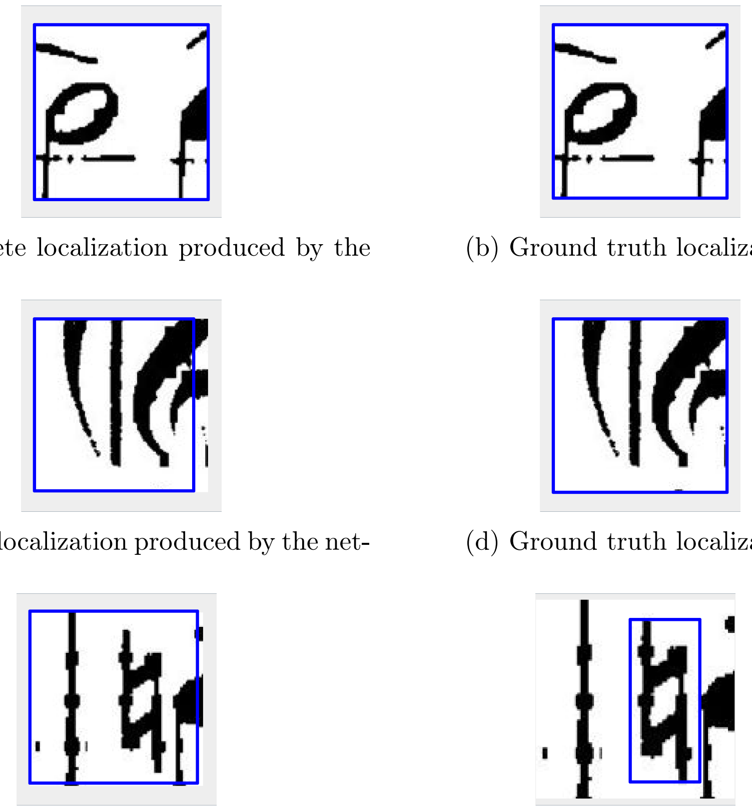
Table 5: Result of testing the whole dataset generated by the grammar using the whole architecture, localization and classification, by using cross validation.

presented in the table 5. We can see that we achieve a good classification results with a slightly more error for the flat class but this is not surprising as this class is the less present in the dataset.

The methodology used to evaluate the localization accuracy is the following. We distinguish three class of localization: a complete localization, a partial localization and an error localization. Different minimum and maximum thresholds on the width and height of the ground truth bounding box was used to decide if a bounding box produced by the network was a complete, partial localization or an error. We compared the recognized bounding box and the ground truth bounding box using these thresholds in order to verify that the recognized bounding box width and height was approximately the same size as the width and height of the ground truth bounding box. We also computed the intersection bounding box between the recognized bounding and the ground truth bounding box and compared the width and height of this new box with the ground truth to check the position of the recognized box. The complete localization is decided when the height and width of the recognized box and the intersection box is within a centered interval of 10% around the ground truth value. The partial localization is decided when the height of the recognized box and the intersection box is within a centered interval of 30% around the ground truth value, the same is computed for the height, however we took a centered interval of 20%. We chose to be more flexible on the width parameter unlike the height parameter because the vertical localization task is easier than the horizontal localization task. Indeed, we chose earlier to generate images for the dataset vertically centered on the note head associated to the accidentals 3.2. This leads to the fact that the vertical position of accidentals is very stable. If any of those values are outside these thresholds, the localization is considered as a wrong localization. We can see that for all accidental classes, the network didn't try to make any segmentation. For this experiment, the localization results are really disappointing as shown in the table 6. We can see that only the junk class is giving a good localization, but it's a deceptive result. It's because we evaluate that a good junk localization is the identity transform, and as we see in the localization examples in figure 16, the localization network is always doing a transformation that is really close to the identity transform. Here we can emit the hypothesis that we don't have enough data to make the localization layer find implicitly a good transformation to localize the symbol. To resolve this problem, we will see now how to explicitly regress the localization network to do a good segmentation of the input data by applying the principle of transfer learning.

Label	Complete Rec		Partial Rec		Error	
	Number	rate	Number	rate	Number	rate
BEMOL	0	0%	0	0%	143	100%
BECARRE	0	0%	0	0%	621	100%
DIESE	0	0%	0	0%	697	100%
Total	0	0%	0	0%	1461	100%

Table 6: Results of the localization task on the whole dataset generated by the grammar using the whole architecture, localization and classification, using cross validation. Although the recognition rate is good 5, the localization is not working. This probably because there not enough data to infer the localization of the symbol.



(a) Complete localization produced by the network.

(b) Ground truth localization.

(c) Partial localization produced by the network.

(d) Ground truth localization.

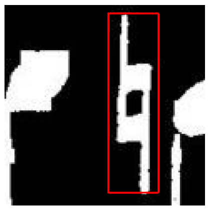
(e) Localization error produced by the network.

(f) Ground truth localization.

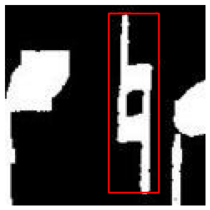
Figure 16: Example of localization (blue bounding box) produced by the whole network. There are no real localization learned by the network.

4.2.2 Joint localization and classification with transfer learning

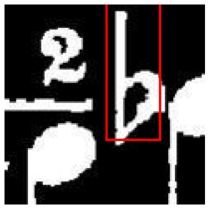
During this experiment, we will try to improve the results obtained in section 4.2.1 by doing transfer learning. Transfer learning is a technique commonly used in neural network to explicitly guide what the neural network should learn. The principle is to do separate training of subparts of the whole



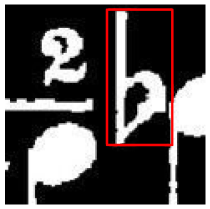
(a) Complete localization produced by the network.



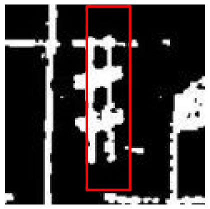
(b) Ground truth localization.



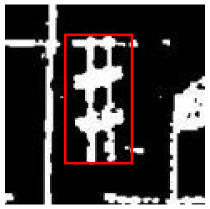
(c) Partial localization produced by the network.



(d) Validation localization.



(e) Localization error produced by the network.



(f) Validation localization.

Figure 17: Example of localization produced when explicitly training the localization network.

Label	Complete Rec		Partial Rec		Error	
	Number	rate	Number	rate	Number	rate
BEMOL	61	42.7%	68	47.6%	14	9.8%
BECARRE	313	45.0%	310	44.5%	73	10.5%
DIESE	455	73.3%	156	25.1%	10	1.6%
Total	829	56.8%	534	36.6%	97	6.7%

Table 7: Measure of the localization task on the well segmented symbol dataset generated by the grammar produced the localization network, using cross validation. We can see a clear improvement on the localization rate 93.3% from the 0% from before 6.

architecture and then load all the prelearned parameters into the whole network and redo a whole training of the whole architecture. This way, we can guide some subpart of the network to explicitly learn what we want. In our context, we saw precedent section that the localization network didn't do any progress during the training of the whole network. In this experimentation, we tried to explicitly learn the localization parameters produced by the localization network. For this, we only used a subpart of the dataset generated by the grammar, for which we know the ground truth localization of the symbols. Also, we removed the junk class as there is nothing to localize in these

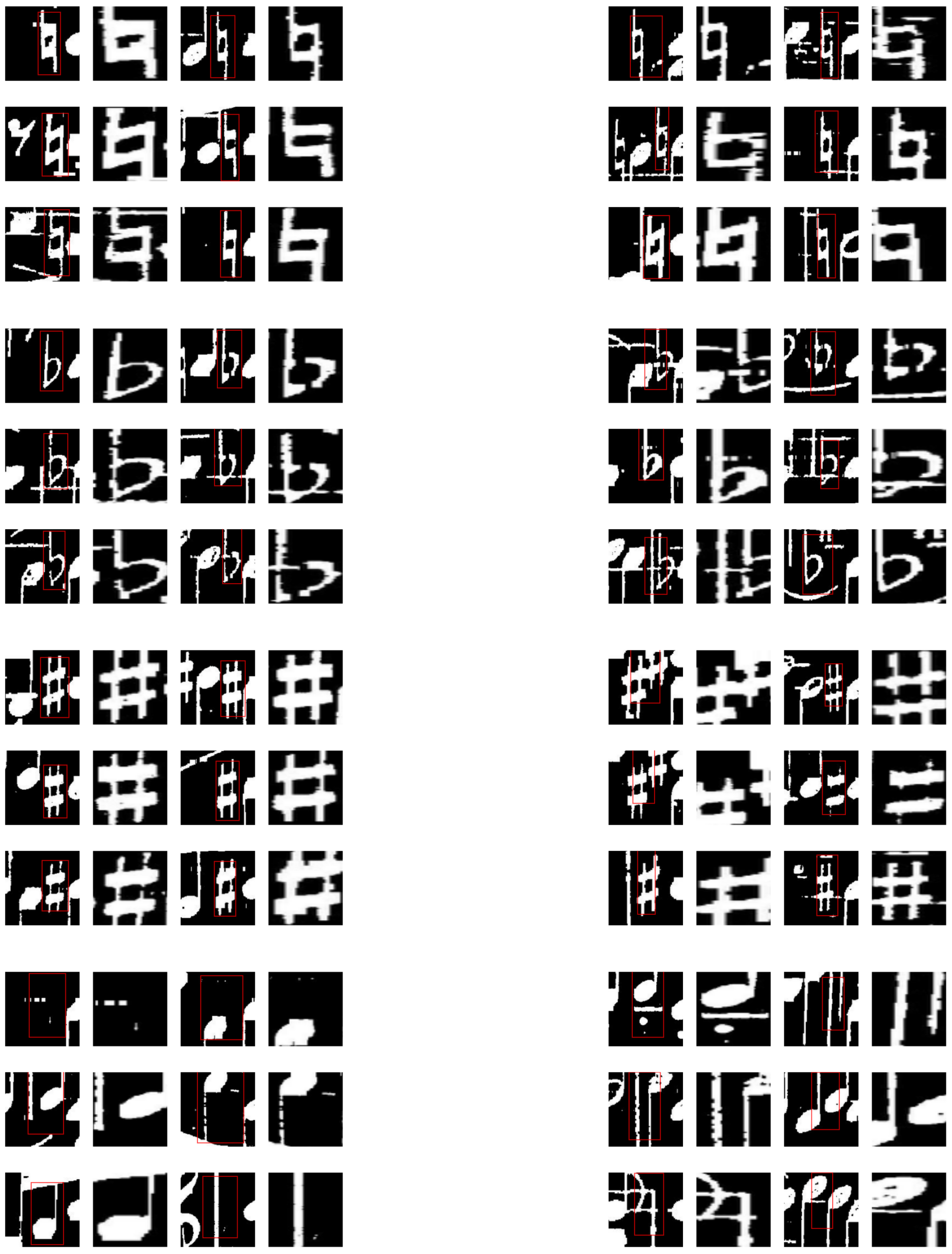


Figure 18: Examples of localization done by the final experiments. The examples are shown in pair of images with at the left the original image with the red bounding box localization and at the right, the image resulting of this transformation that was then fed to the classifier. The first column shows localization of symbols that led to a good classification. The last column shows localization of symbols that led to a bad classification.

Label	Exact	Error	Accuracy
BEMOL	198	43	82.2%
JUNK	1924	41	97.9%
BECARRE	909	44	95.4%
DIESE	750	21	97.3%
Total	3781	149	96.2%

Table 8: Result of testing the whole dataset generated by the grammar using the whole architecture, localization (fixed pre-learned parameters) and classification, by using cross validation.

images. The results we obtain in table 7 show that we obtain a big improvement from the last experiment. These results are generated using the same techniques and threshold as earlier. We obtained 93.4% of good localization with 56.8% of really accurate localization and 36.6% of partial localization. A large window of improvement is still possible as the dataset may still contains ground truth errors as the manual checking wasn't done perfectly.

The following experiment was to trained the classification network on the results of the localization network. In order to do this, we used the whole architecture presented in section 3.3.3. However, we fixed the localization network parameters during the training phase to the one learned by the precedent experimentation. The accuracy we obtained are slightly lower than the one we had during the first joint localization and classification experiment in table 5 with 96.8% instead of 99.8%. When we see at the segmentation produced by the localization layer in figure 18 sorted by the fact that if their classification was good or not, we can see that when the localization layer is doing an accurate segmentation the classification is going well and when the localization is less accurate the classification end up wrong. However, we can see that multiple badly segmented symbols are accurately localized and classified by the network. One big improvement we could do to our system would be to produce the ground truth localization on badly segmented symbols and add those images to the dataset used to train the localization layer. This would lead to a better localization and consequently to a better classification.

4.3 Experiment summary

The table 9 is a summary of all experiments presented before in order to localize and recognize or reject a music symbol.

	1st Experiment 4.2.1		2nd Experiment 4.2.2	
Label	Loc accuracy	Classif accuracy	Loc accuracy	Classif accuracy
BEMOL	0%	90.5%	90.3%	82.2%
JUNK		99%		97.9%
BECARRE	0%	97.3%	89.5%	95.4%
DIESE	0%	98.3%	98.4%	97.3%
Total	0%	99.8%	93.4%	96.2%

Table 9: Summary of joint segmentation and classification experiment. We can see a clear improvement on localization rate caused by explicitly learning the localization task. However, the classification is slightly decreased because of the instability of the localization network.

5 Conclusion

A renewed interest is showing toward OMR in the computer vision and pattern recognition research field because it has still many challenges to overcome. Many solutions have been proposed for every step of OMR: pre-processing, staff recognition and removal, music symbols segmentation and recognition, music notation reconstruction. However, traditional OMR system that implement all these steps have an ascending architecture and imply many limitations. By applying all these steps sequentially, it is not possible to use the context during the segmentation and recognition phase. The DMOS system is different because it is the grammar that guide the segmentation and recognition of the score. The starting point of this internship is to improve the segmentation and recognition of broken and overlapping musical symbols of this DMOS method. However, this method should be generically applicable to any types of structured document. For this internship, the DMOS system will be used as a base for supporting this joint segmentation and recognition method but it should be usable in another recognition system.

In this work, we proposed a system combining a music grammar by using the DMOS system and an architecture of neural network capable of localizing and recognizing music symbols. The music grammar can guide the neural network to localize and recognize symbols on small zone of the music score. Moreover, the generation of these by the music grammar can be used to bootstrap the generation of a dataset in order to train neural network on the task of localizing and recognizing music symbols. Our neural network architecture is composed of two part: a localization network capable of producing geometrical transformation parameters that will localize the symbols and a classification network capable of recognizing music symbols and reject when there is no symbol. We demonstrate the use of such an architecture on the specific task of recognizing accidentals in music scores. We first train the whole network without explicit ground truth localization data. The classification rate we obtain was of 99.8%, however no localization was done by the network. On the second experiment, we obtained 93.4% of localization accuracy by explicitly training the localization network and 96.2% of accuracy for the classification network. Lots of improvement can still be make in order to increase these scores, especially by producing ground truth localization on badly segmented symbols and expand the dataset used to train the localization network. Furthermore, this method should be generalizable to other music symbols like note heads, silence, clefs... The interesting property of this system is that it could be applied on handwritten music score with very few modifications because both the grammar and neural network can be used on handwritten data. Finally, other kinds of document types could get benefit from this method by using neural networks adapted to character recognition like recurrent neural networks.

References

[Ba et al., 2014] Ba, J., Mnih, V., and Kavukcuoglu, K. (2014). Multiple Object Recognition with Visual Attention. *arXiv:1412.7755 [cs]*. arXiv: 1412.7755.

[Couasnon, 2001] Couasnon, B. (2001). DMOS : a generic document recognition method, application to an automatic generator of musical scores, mathematical formulae and table structures recognition systems. In *Sixth International Conference on Document Analysis and Recognition, 2001. Proceedings*, pages 215–220.

[Couasnon and Camillerapp, 1995] Couasnon, B. and Camillerapp, J. (1995). A way to separate knowledge from program in structured document analysis: application to optical music recognition. In *, Proceedings of the Third International Conference on Document Analysis and Recognition, 1995*, volume 2, pages 1092–1097 vol.2.

[Coüasnon et al., 1995] Coüasnon, B., Brisset, P., and Stéphan, I. (1995). Using Logic Programming Languages For Optical Music Recognition. In *In Proceedings of the Third International Conference on The Practical Application of Prolog*, pages 115–134.

[Coüasnon and Camillerapp, 1994] Coüasnon, B. and Camillerapp, J. (1994). Using grammars to segment and recognize music scores. *International Association for Pattern Recognition Workshop on Document Analysis Systems*, pages 15–27.

[d'Andecy et al., 1994] d'Andecy, V. P., Camillerapp, J., and Leplumey, I. (1994). Kalman filtering for segment detection: application to music scores analysis. In *, Proceedings of the 12th IAPR International Conference on Pattern Recognition, 1994. Vol. 1 - Conference A: Computer Vision amp; Image Processing*, volume 1, pages 301–305 vol.1.

[Erhan et al., 2014] Erhan, D., Szegedy, C., Toshev, A., and Anguelov, D. (2014). Scalable Object Detection Using Deep Neural Networks. In *2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2155–2162.

[Everingham et al., 2009] Everingham, M., Gool, L. V., Williams, C. K. I., Winn, J., and Zisserman, A. (2009). The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*, 88(2):303–338.

[Fornés and Sánchez, 2014] Fornés, A. and Sánchez, G. (2014). Analysis and Recognition of Music Scores. In Doermann, D. and Tombre, K., editors, *Handbook of Document Image Processing and Recognition*, pages 749–774. Springer London. DOI: 10.1007/978-0-85729-859-1_24.

[Jaderberg et al., 2015] Jaderberg, M., Simonyan, K., Zisserman, A., and Kavukcuoglu, K. (2015). Spatial Transformer Networks. *arXiv:1506.02025 [cs]*. arXiv: 1506.02025.

[Lecun et al., 1998] Lecun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

[Moysset et al., 2014] Moysset, B., Bluche, T., Knibbe, M., Benzeghiba, M., Messina, R., Louradour, J., and Kermorvant, C. (2014). The A2ia Multi-lingual Text Recognition System at the Second Maurdor Evaluation. In *2014 14th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 297–302.

[Netzer et al., 2011] Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. (2011). Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, page 5. Granada, Spain.

[Pruslin, 1966] Pruslin, D. (1966). *Automatic recognition of sheet music*. PhD thesis, Massachusetts Institute of Technology.

[Pugin, 2006] Pugin, L. (2006). Optical Music Recognitoin of Early Typographic Prints using Hidden Markov Models. In *ISMIR*, pages 53–56.

[Rebelo et al., 2009] Rebelo, A., Capela, G., and Cardoso, J. S. (2009). Optical recognition of music symbols. *International Journal on Document Analysis and Recognition (IJDAR)*, 13(1):19–31.

[Rebelo et al., 2012] Rebelo, A., Fujinaga, I., Paszkiewicz, F., Marcal, A. R. S., Guedes, C., and Cardoso, J. S. (2012). Optical music recognition: state-of-the-art and open issues. *International Journal of Multimedia Information Retrieval*, 1(3):173–190.