Optical Music Recognition by Long Short-Term Memory Networks

Arnau Baró[1], Pau Riba[1], Jorge Calvo-Zaragoza[2] and Alicia Fornés[1]

[1] Computer Vision Center - Computer Science Department
Universitat Autònoma de Barcelona, Bellaterra, Catalonia, Spain
Email: {abaro,priba,afornes}@cvc.uab.cat
[2] Schulich School of Music
McGill University, Montreal, Canada
Email: jorge.calvozaragoza@mcgill.ca

Abstract. Optical Music Recognition refers to the task of transcribing the image of a music score into a machine-readable format. Many music scores are written in a single staff, and therefore, they could be treated as a sequence. Therefore, this work explores the use of Long Short-Term Memory (LSTM) Recurrent Neural Networks for reading the music score sequentially, where the LSTM helps in keeping the context. For training, we have used a synthetic dataset of more than 40000 images, labeled at primitive level. The experimental results are promising, showing the benefits of our approach.

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

1 Introduction

Sheet music uses music notation to encode information on how to interpret a piece. It is one of the most considered means for the transmission of music. With the advent of the digital era, there is a number of computational tools for working with musical scores. However, to take advantage of these benefits, it is necessary to transcribe sheet music into a digital format that can be processed by a computer.

The transcription process can be carried out manually. However, the wealth of music notation inevitably leads to burdensome software for music score editing, which makes the whole process very time-consuming and prone to errors. Consequently, automatic transcription systems for musical documents represent interesting tools. The field devoted to address this task is known as Optical Music Recognition (OMR) [1–3]. Nowadays, there exist many commercial OMR tools, like PhotoScore[3] or SharpEye[4].

[3] http://www.neuratron.com/photoscore.htm
[4] http://www.visiv.co.uk/

Typically, an OMR system takes an image of a music score and automatically exports its content into some structured format such as MEI or MusicXML. In addition to the automatic transcription of sheet music, the OMR field comprises many other applications such as writer identification, graphic reconstruction of old music scores, generation of audio files from images, or retrieval of the same piece from different authors.

The process of recognizing the content of a music score from its image is complex because it has to deal with many music-specific difficulties [2], such as the two-dimensional nature of the notation, the double component of music symbols,[5] the presence of the staff lines, and so on.

Traditionally, OMR has been approached considering multi-stage systems [1]. The different stages comprise several small sub-tasks such as image binarization [4], staff-line removal [5], or music symbol classification [6]. Our work, however, focuses on directly recognizing the music content appearing on an image.

We do assume that the image depicts a single staff section (e.g. scores for violin, flute, etc.), much in the same way as most text recognition research focuses on recognizing words appearing in a given line image [7]. Note that this is not a strong assumption, as there exist algorithms that achieve good performance for both isolating staff sections [8] and separating music and lyrics (accompanying text) [9]. For this reason, one can assume that staves are already segmented and, therefore, can be processed as a sequence.

To address this specific task, the proposed architecture is based on Recurrent Neural Networks (RNN), since they have been applied with great success to many sequential recognition tasks such as speech [10] or handwriting [7] recognition. Specifically, to avoid the vanishing gradient problem, Long Short-Term Memory (LSTM) units are considered. Moreover, Bidirectional LSTMs are used to benefit from context information.

As a scientific novelty, we address the OMR by separating the two components of the musical symbols: duration and pitch, both in terms of training and evaluation. This provides the RNN with greater robustness, as they can focus on the specific aspects that concern each component. Our experimental results demonstrate the viability of this approach, obtaining results that are close to optimal on an exhaustive set of musical staves.

The rest of the paper is organized as follows. Section 2 explores the state of the art. Section 3 describes the method, whereas Section 4 analyzes the results. Finally, conclusions and future work are drawn in Section 5

2 Related Work

This section overviews the key approaches in Optical Music Recognition, and also, overviews the Deep Learning architectures applied to music research that are relevant to the present work.

[5] Symbols appear with specific duration (rhythm) and pitch (melody).

2.1 Optical Music Recognition

An OMR system aims to recognize each element located in the music score. Figure 1 illustrates the usual pipeline from a scanned music score to a machine-readable format. The steps are the following. First, the image is preprocessed to reduce problems in segmentation. Usually, before segmenting the musical symbols and/or primitives, the staff lines are removed. Hence, the segmentation task is simplified. Afterwards, the primitives are merged to form symbols. Some methodologies use rules or grammars in order to be able to validate and solve some ambiguities from the previous step. Finally, the musical description file (e.g. MusicXML, MEI) is created with the information of the previous steps. These steps are described in more detail next.

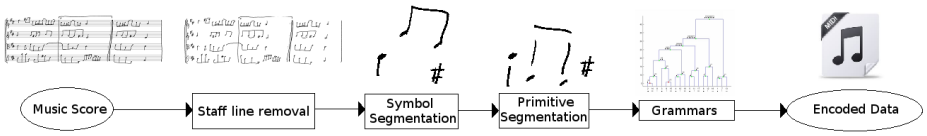
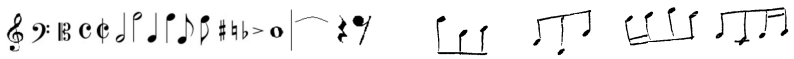


Fig. 1. Typical OMR pipeline.

The first stage is devoted to preprocessing and layout analysis. The most common techniques are binarization, noise removal and blur correction. However, other techniques as enhancement, skew correction or deskewing, among others have also been proposed. In music scores documents it is important to segment the document into regions. Authors in [11] propose a new algorithm to segment the regions that include text and regions containing music scores. Normally, the staff removal algorithm are based on projections and run-length analysis, contour-line tracking, or graphs.

The recognition of music symbols consists in the recognition of isolated and compound music symbols. Figure 2 shows examples of isolated and compound music symbols. This classification is done because the techniques are usually different. For example, isolated music symbols are usually detected by symbol recognition methods [12–15], grammar/rules [16], sequence analysis (e.g. Hidden Markov Models) [17,18], graphs [19], Multilayer perceptron [14] and deep neural networks [6, 20, 21]. In the case of compound music symbols, most methods are based on grammars or rules [22]. It must be noted that the combination of compound music symbols is large, so it is impossible to have examples of all these possible combinations. This limitation must be taken into account when developing learning-based approaches.

The validation stage is related to the previous one. Usually grammars or rules are defined to make more robust the recognition step in front of ambiguities. Some works [16, 22, 23] propose the use of grammars to correct any mistakes as repeating or missing symbols. Another aspect that could be verified is whether the number of beats match the time signature.



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

Fig. 2. Examples of music symbols.

OMR systems typically provide an output file at the end of the process. The most common output files are MIDI[6], MusicXML[7] or MEI[8]. MIDI (Musical Instrument Digital interface) is a communication technical standard used in electronic music devices. MusicXML is an open musical notation format based on Extensible Markup Language (XML). MEI is an open-source effort to define a system for encoding musical documents in a machine-readable structure.

2.2 Deep Learning in Music

According to Goodfellow, Bengio and Courvill [24], Deep Learning appeared between 1940s-1960s. However, it has become popular quite recently due to, among others, the technological advances in Hardware. In the last decade, several deep learning techniques have been applied to music or audio processing. For example, CNNs have been applied to audio processing in order to detect and recognize the sound of certain objects and scenes in videos. In [25] they use CNNs to compensate differences between video and audio sampling rates, whereas, the authors in [26] use CNN in order to process sounds. The authors of [27, 28] have applied RNNs to MIDI generation, specifically LSTM in order to produce new music files taking advantage of sequence model of the music input files. In [29] RNNs are used in polyphonic music in order to make predictions.

As far as we know, there are very few deep learning-based systems that cover the whole stages of Optical Music Recognition. For example, a very recent OMR work which uses a Convolutional Sequence-to-Sequence network for recognizing printed scores has been published [30]. Then, Calvo-Zaragoza *et al.* [31] use an end-to-end architecture based on a Recurrent Convolutional Neural Network in order to recognize monophonic music scores. Finally, Pacha *et al.* [32] proposes a first step towards OMR developing a handwritten music symbol classifier. However, it is still far from a complete OMR system.

3 Methodology

Single staff sheet music can be seen as a sequence. In this way, a music score is read from left to right. In order to automatically process the music score and take into account the sequence of music symbols, a *Recurrent Neural Network* (RNN) seems an appropriate tool. In this work, we propose to make use of *Long*

[6] `https://www.midi.org/`
[7] `http://www.musicxml.com/`
[8] `http://music-encoding.org/`

Short-Term Memory (LSTM) [33] networks. LSTMs have the ability to decide which information has to be kept as context and which information has to be removed, *i.e.* forgotten.

Figure 3 shows the different stages of our proposed pipeline. Firstly, the input music scores are preprocessed (Subsection 3.1). Afterwards, each column of the image is processed by an LSTM network (Subsection 3.2). The output of the LSTM is passed by two fully connected layers in order to distinguish between rhythm and pitch (Subsection 3.3). Finally, the output of the system is the recognition of symbols, including rhythm and pitch (Subsection 3.4). The different steps of this pipeline are explained in the following sections.

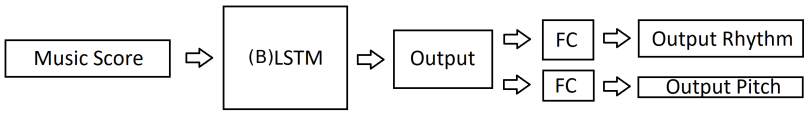


Fig. 3. Architecture of the network

3.1 Input

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

3.2 Long Short-Term Memory

A LSTM network has been used in order to recognize the elements of the sheet music. In this work, we use a bidirectional network to increase the performance and reduce ambiguities when recognizing some symbols. The combination of forward and backward pass allows to recognize symbols that may be confused if only one direction is used.

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

3.3 Fully Connected Layers

At the end of the LSTM network, we propose to use two heads in order to separately predict rhythm and melody. Therefore, after the LSTM output, two fully connected layers (FC) are used to obtain two different outputs. The reason to split the output in two parts is that the number of combinations between melody and rhythm is very high. In case of using a single output, all possible combinations of rhythm-melody must be created as possible classes. Therefore, we propose to exploit the idea that rhythm and pitch can be independent. Thus, rhythm is decided by the symbol whereas the pitch depends on the position with respect the staff lines. Following this idea, many more examples are available to train our system.

3.4 Output

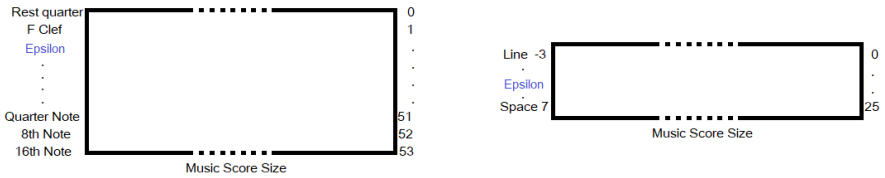
After the FC layers, the next step is to calculate the loss and backpropagate the errors. Even though two heads are used to separate between rhythm and pitch, the output of our system should be able to deal with multiple classes per time step. Therefore, in validation and test, a threshold is applied to both outputs (Rhythm and Pitch) in order to obtain the corresponding classes. The outputs and the ground-truth of each music score is represented by two binary matrices, one for the rhythm and another for the melody or pitch. Horizontally, it corresponds to the width of the input image whereas the vertical axis tells the different classes for symbols and pitch, 54 and 26 respectively. Pitch corresponds to locations in the staff. Figure 4 shows the structure of the matrix for both melody 4a and rhythm 4b and Fig. 5 shows a real example with its corresponding groundtruth. The corresponding pixels where the symbol is located in the music score will be activated in both matrices indicating which symbols are activated in each time step *i.e.* pixels. The following symbols have been manually added to ease the recognition task:

- Epsilon(ε) is used to know where each symbol starts and ends, as it is used in text recognition. This symbol can be seen as a separator. Wherever this symbol is activated, it means that it is not possible to have any other symbol activated as well (see Figure 4c blue marks). This symbol appears in both the rhythm and pitch ground-truths.
- *No note* is used to indicate that a symbol has not any pitch. This symbol only appears in the pitch ground-truth.

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

3.5 Loss Function

In music, we can find one or more symbols in one instance of time, for example, chords or time signature (see Figure 4c red marks). Therefore, a multilabel loss



(a) (b)

Fig. 4. Output representation for Rythm (a) and Pitch (b).

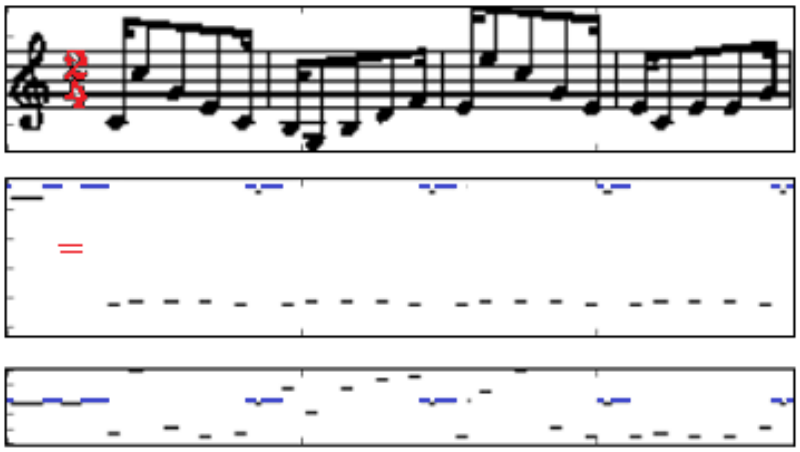


Fig. 5. Example of Music Score and the corresponding Ground-truth in a Binary Matrix. The first row is the music score. The second row is the Rhythm Ground-truth. The third row is the Pitch Ground-truth.

function has to be chosen to deal with the before-mentioned problem. In other words, the loss function must allow more than one activation per time step. Thus, the softmax activation function cannot be used because it is thought for single-label classification problems. In this work, two different loss functions have been used: On the one hand, SmoothL1Loss creates a criterion that uses a squared term if the absolute element-wise error falls below 1 and an L1 term otherwise (Equation 1). On the other hand, MultiLabelSoftMarginLoss creates a criterion that optimizes a multi-label one-versus-all loss based on max-entropy (Equation 2). The loss is calculated independently for rhythm and melody. Once both losses are calculated, they are summed and backpropagated.

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

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

4 Experiments and Results

This Section presents and discusses the experimental results.

4.1 Dataset

A Synthetic dataset has been used to train the network. This collection is composed of more than 50000 music scores with 3 different typographies. The dataset corresponds to incipts from the RISM catalog [9]. It is composed of almost 50000 music scores with 3 different typographies. The staffs are divided in 60% (29815) for training, 20% (9939) for validation and 20% (9939) for test.

4.2 Evaluation

The evaluation of a complete OMR system is not well defined in the literature. Thus, we propose to follow the evaluation described in [30]. The authors proposed to evaluate three aspects of the framework; pitch, rhythm and their combination. Note that the combination of pitch and rhythm corresponds to the performance of the whole system. The chosen evaluation metric is the *Symbol Error Rate* (SER) applied to an array produced by the system. Note that a threshold is applied to convert the output of the FC layers to an array of symbols.

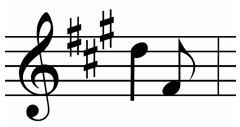


Fig. 6. Example of music score.

An example of the format of the three output arrays, corresponding to Figure 6, is the following.

– Rhythm: [gClef, accidental sharp, accidental sharp, accidental sharp, quarter note, eight note, bar line].

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

– Pitch: [noNote, L5, S3, S5, L4, S1, noNote][10].
– Rhythm+Pitch: [[gClef, noNote], [accidental sharp, L5], [accidental sharp, S3], [accidental sharp, S5], [quarter note, L4], [eight note, S1], [bar line, noNote]].

Symbol Error Rate (SER) This metric is based on the well-known Word Error Rate (WER) metric [34] used in speech and text recognition. SER also uses the Levenshtein distance. The main difference between them is that the Levenshtein distance computes the differences at character level, WER does it at the word level and SER does it at symbol level. In the case of music scores, given a prediction and a reference ground-truth, the SER is defined as the minimum number of edit operation *i.e.* insertions, substitutions and deletions, to convert one array into the other.

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

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

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

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

Output's Threshold Evaluation A threshold is applied to decide which symbols are activated at each time step. Note that this threshold is needed because we have no knowledge about the number of symbols appearing at each time step. This threshold has been experimentally set using a grid search from 0 to 1 and step of 0.1. We have selected the combination of rhythm and pitch Error Rate as a metric to choose the best threshold. Figure 7 shows the evolution of the error rate depending on the threshold. As we can see, the best threshold is 0.5 even though 0.4 achieves similar results.

4.3 Results

All the results that are shown in this section are obtained using *Adam* as optimizer with a learning rate of 10^{-4}. In this work, the PyTorch [11] library has been used in order to build the proposed framework.

Table 1 shows an error rate comparison in terms of the average and standard deviation among 5 runs. In this comparison, single directional and bidirectional

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

[11] http://pytorch.org/

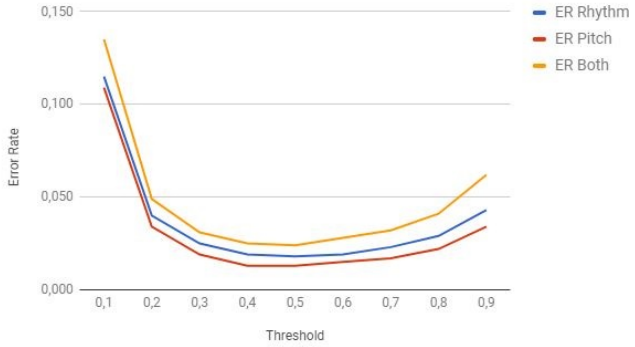


Fig. 7. Evaluation of the best threshold in terms of Error Rate: Rhythm, Pitch and Rhythm+Pitch.

LSTM are analyzed with the two described loss functions. The first column shows the loss function and the network that has been used, the second one shows the error rate of the rhythm, the third one the results concerning the pitch; and the last column shows the results when considering the rhythm and pitch jointly. Note that the BLSTM produces better results. Moreover, regarding the loss function, the Smooth L1 function obtains better results with 1.5% SER when recognizing the pitch, 2% SER when recognizing the rhythm and 2.8% SER when recognizing the pitch an rhythm jointly. Using a bidirectional network, the input is processed in both directions. Thus, it obtains information of the whole symbol, and becomes more accurate. For example, if one direction recognizes a note-head, the other direction can discard that the vertical line that it is reading is a bar line, but instead a note stem (both stems and bar lines are straight vertical lines).

In Figures 8 and 9 we can see some qualitative results. First subfigure shows the input of the system. The second, third and fourth subfigures correspond to the Rhythm ground-truth, output and thresholded output respectively. In the fifth, sizth and seventh subfigures we can see the Melody ground-truth, output and thresholded output respectively.

4.4 Comparison with a commercial OMR software

The proposed method has been compared with PhotoScore [12], a commercial OMR software able to recognize printed and also handwritten music scores. Figure 10 show qualitative results. Note that this comparison might not be completely fair. PhotoScore has some features to improve its performance that are not considered in our method. PhotoScore probably uses syntactic rules for validation. For instance, the commercial system can recognize the time signature and then validate the amount of music notes at each bar unit (which is used to

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

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

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

solve ambiguities). Contrary, in our work, no syntactic rules have been applied. This is an important difference because we do not correct any miss-classification using music notation rules.

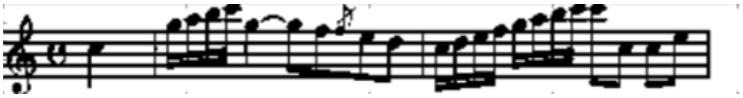
Figure 10 shows that, even with a very simple music score, PhotoScore has produced two errors. First, it has confused the time of silence (5-10), and second it has added a duration dot at the end. It must be said that the method proposed in this work has correctly recognized all the music symbols.

5 Conclusion and Future Work

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

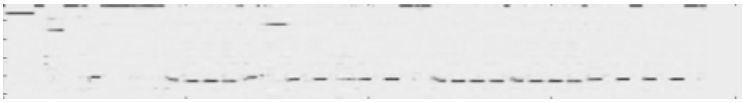
The obtained results show that single staff music scores could be recognized by means of RNN. We have also shown the benefits of using a BLSTM instead of an LSTM applied to musical images. However, the recognition of scores as sequences has some limitations. For instance, more complex music scores (e.g.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

Future work will be focused on investigating transfer learning methods to recognize handwritten music scores. Moreover, we would like to incorporate musical rules or semantics as in our previous work [22] in order to solve ambiguities. In addition, we plan to investigate more suitable techniques for recognizing complex polyphonic music scores such as CNNs and attention mechanisms. In addition, we would like to convert the output of the architecture into a MIDI file, able to be listened, or to convert it into a sheet format as PhotoScore does.



(a) Input image

(b) Rhythm ground truth



(c) Rhythm Output

(d) Rhythm Output with threshold higher than 50%

(e) Melody ground truth

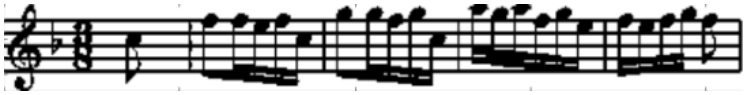
(f) Melody Output

(g) Melody Output with threshold higher than 50%

Fig. 8. Qualitative Results Example using LSTM.

Acknowledgment

This work has been partially supported by the Spanish project TIN2015-70924-C2-2-R, the Ramon y Cajal Fellowship RYC-2014-16831, the CERCA Program/ Generalitat de Catalunya, FPU fellowship FPU15/06264 from the Spanish Ministerio de Educación, Cultura y Deporte, the social Sciences and Humanities Research Council of Canada and the FI fellowship AGAUR 2018 FI_B 00546 of the Generalitat de Catalunya. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan Xp GPU used for this research.



(a) Input image

(b) Rhythm ground truth

(c) Rhythm Output

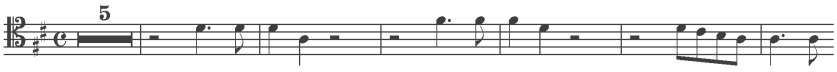
(d) Rhythm Output with threshold higher than 50%

(e) Melody ground truth

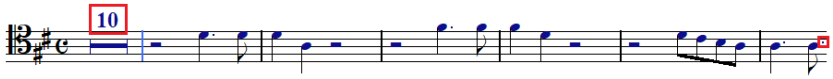
(f) Melody Output

(g) Melody Output with threshold higher than 50%

Fig. 9. Qualitative Results Example using BLSTM.



(a) Original Image



(b) Visual Result of PhotoScore

Fig. 10. Recognition of a music score using the PhotoScore Commercial OMR software. The errors are shown in red color.

References

1. A. Rebelo, I. Fujinaga, F. Paszkiewicz, A. R. S. Marçal, C. Guedes, and J. S. Cardoso, "Optical music recognition: state-of-the-art and open issues.," *IJMIR*,

vol. 1, no. 3, pp. 173–190, 2012.
2. D. Bainbridge and T. Bell, "The challenge of optical music recognition," *Computers and the Humanities*, vol. 35, no. 2, pp. 95 – 121, 2001.
3. A. Fornés and G. Sánchez, "Analysis and recognition of music scores," in *Handbook of Document Image Processing and Recognition*, pp. 749–774, Springer London, 2014.
4. T. Pinto, A. Rebelo, G. A. Giraldi, and J. S. Cardoso, "Music score binarization based on domain knowledge," in *Pattern Recognition and Image Analysis, 2011*, pp. 700–708, 2011.
5. A. Gallego and J. Calvo-Zaragoza, "Staff-line removal with selectional auto-encoders," *Expert Systems with Applications*, vol. 89, pp. 138–48, 2017.
6. A. Pacha and H. Eidenberger, "Towards a universal music symbol classifier," in *12th International Workshop on Graphics Recognition (GREC)*, pp. 35–36, 2017.
7. A. Graves and J. Schmidhuber, "Offline handwriting recognition with multidimensional recurrent neural networks," in *NIPS*, pp. 545–552, 2009.
8. V. B. Campos, J. Calvo-Zaragoza, A. H. Toselli, and E. Vidal-Ruiz, "Sheet music statistical layout analysis," in *ICFHR*, pp. 313–318, 2016.
9. J. A. Burgoyne, Y. Ouyang, T. Himmelman, J. Devaney, L. Pugin, and I. Fujinaga, "Lyric extraction and recognition on digital images of early music sources," in *ISMIR*, pp. 723–727, 2009.
10. A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 6645–6649, IEEE, 2013.
11. F. Pedersoli and G. Tzanetakis, "Document segmentation and classification into musical scores and text," *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 19, no. 4, pp. 289–304, 2016.
12. A. Fornés, J. Lladós, G. Sánchez, and D. Karatzas, "Rotation invariant hand drawn symbol recognition based on a dynamic time warping model," *IJDAR*, vol. 13, no. 3, pp. 229–241, 2010.
13. S. Escalera, A. Fornés, O. Pujol, P. Radeva, G. Sánchez, and J. Lladós, "Blurred Shape Model for binary and grey-level symbol recognition," *Pattern Recognition Letters*, vol. 30, no. 15, pp. 1424–1433, 2009.
14. A. Rebelo, G. Capela, and J. S. Cardoso, "Optical recognition of music symbols: A comparative study," *IJDAR*, vol. 13, no. 1, pp. 19–31, 2010.
15. A. Rebelo, J. Tkaczuk, R. Sousa, and J. S. Cardoso, "Metric learning for music symbol recognition," in *2011 10th International Conference on Machine Learning and Applications and Workshops*, vol. 2, pp. 106–111, Dec 2011.
16. B. Coüasnon and B. Rétif, "Using a grammar for a reliable full score recognition system," in *ICMC*, 1995.
17. L. Pugin, "Optical music recognitoin of early typographic prints using hidden markov models," in *ISMIR*, 2006.
18. L. Pugin, J. A. Burgoyne, and I. Fujinaga, "Map adaptation to improve optical music recognition of early music documents using hidden markov models," in *ISMIR*, 2007.
19. J. C. Pinto, P. Vieira, and J. M. Sousa, "A new graph-like classification method applied to ancient handwritten musical symbols," *Document Analysis and Recognition*, vol. 6, no. 1, pp. 10–22, 2003.
20. K.-Y. Choi, B. Coüasnon, Y. Ricquebourg, and R. Zanibbi, "Bootstrapping samples of accidentals in dense piano scores for cnn-based detection," in *12th International Workshop on Graphics Recognition (GREC)*, pp. 19–20, 2017.

21. M. Dorfer, J. Hajič, and G. Widmer, "On the potential of fully convolutional neural networks for musical symbol detection," in *12th International Workshop on Graphics Recognition (GREC)*, pp. 53–54, 2017.
22. A. Baró, P. Riba, and A. Fornés, "Towards the recognition of compound music notes in handwritten music scores," in *ICFHR*, pp. 465–470, Oct 2016.
23. T. Matsushima, S. Ohteru, and S. Hashimoto, "An integrated music information processing system: Psb-er," *Proceedings of the international computer music conference*, pp. 191–198, 1989.
24. I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.
25. A. Owens, P. Isola, J. H. McDermott, A. Torralba, E. H. Adelson, and W. T. Freeman, "Visually indicated sounds," *CoRR*, vol. abs/1512.08512, 2015.
26. Y. Aytar, C. Vondrick, and A. Torralba, "Soundnet: Learning sound representations from unlabeled video," *CoRR*, vol. abs/1610.09001, 2016.
27. Y. C. Sübakan and P. Smaragdis, "Diagonal rnns in symbolic music modeling," *CoRR*, vol. abs/1704.05420, 2017.
28. V. Kalingeri and S. Grandhe, "Music generation with deep learning," *CoRR*, vol. abs/1612.04928, 2016.
29. R. Pascanu, Ç. Gülçehre, K. Cho, and Y. Bengio, "How to construct deep recurrent neural networks," *CoRR*, vol. abs/1312.6026, 2013.
30. E. van der Wel and K. Ullrich, "Optical music recognition with convolutional sequence-to-sequence models," *CoRR*, vol. abs/1707.04877, 2017.
31. J. Calvo-Zaragoza, J. J. Valero-Mas, and A. Pertusa, "End-to-end optical music recognition using neural networks," in *ISMIR*, 2017.
32. A. Pacha and H. M. Eidenberger, "Towards self-learning optical music recognition," *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp. 795–800, 2017.
33. S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, pp. 1735–1780, Nov. 1997.
34. V. Frinken and H. Bunke, "Continuous handwritten script recognition," in *Handbook of Document Image Processing and Recognition*, pp. 391–425, Springer, 2014.