EFFICIENT NOTATION ASSEMBLY IN OPTICAL MUSIC RECOGNITION

Carlos Penarrubia[1] **Carlos Garrido-Munoz**[1]
Jose J. Valero-Mas[2] **Jorge Calvo-Zaragoza**[1]
[1] U. I. for Computing Research, University of Alicante, Spain
`{carlos.penarrubia, carlos.garrido, jorge.calvo}@ua.es`
[2] Music Technology Group, Universitat Pompeu Fabra, Spain
`josejavier.valero@upf.edu`

ABSTRACT

Optical Music Recognition (OMR) is the field of research that studies how to computationally read music notation from written documents. Thanks to recent advances in computer vision and deep learning, there are successful approaches that can locate the music-notation elements from a given music score image. Once detected, these elements must be related to each other to reconstruct the musical notation itself, in the so-called *notation assembly* stage. However, despite its relevance in the eventual success of the OMR, this stage has been barely addressed in the literature. This work presents a set of neural approaches to perform this assembly stage. Taking into account the number of possible syntactic relationships in a music score, we give special importance to the efficiency of the process in order to obtain useful models in practice. Our experiments, using the MUSCIMA++ handwritten sheet music dataset, show that the considered approaches are capable of outperforming the existing state of the art in terms of efficiency with limited (or no) performance degradation. We believe that the conclusions of this work provide novel insights into the notation assembly step, while indicating clues on how to approach the previous stages of the OMR and improve the overall performance.

1. INTRODUCTION

Optical Music Recognition (OMR) is the field of research that enables the automatic reading of music notation from scanned documents [1]. OMR has become increasingly important due to its potential for a better preservation of music archives, while also facilitating new data to the wealth of Music Information Retrieval algorithms that rely on symbolic formats [2, 3].

As in many other fields, deep learning brought about a drastic change in the performance of the proposed approaches for OMR [4]. As we will mention in the next section, tasks that used to be a difficult barrier are now feasible and successful models are known, e.g., staff detection [5] or the identification of musical symbols in the image [6]. However, although these tasks are the first obstacles of an OMR system, they are not enough to complete the process. Once the graphic elements have been identified, it is necessary to reconstruct the musical notation itself by inferring the syntactic relationships that exist between such elements, namely *notation assembly*.

To account for all existing relations, the retrieval is usually performed in a pairwise fashion among all the identified graphic units. On this note, given the (typically large) density of elements within music score images, the task exhibits a high computational complexity that complicates its integration in an end-user application. Therefore, in addition to accuracy, one must carefully take into account the efficiency of this type of schemes.

This work addresses the efficient estimation of all the syntactic relations among the elements of a music score using neural network. More precisely, we propose and assess two approaches to address this task in an efficient manner: one that is based on classifying each pair of elements employing a series of numerical features, while the other uses asymmetric kernels [7], which can be computed with high parallelization and provide results very fast. In our experiments, using the well-known MUSCIMA++ corpus, we will compare the trade-off between effectiveness and efficiency that these methods provide and discuss the experimental outcomes. In addition, assuming that the previous stages of the process may contain errors, we also assess the robustness of the assembly proposals by intentionally degrading the estimations of these precedent phases. This analysis is expected to provide useful insights for the adequate design of notation assembly methods in OMR.

The remainder of the paper is as follows: in Section 2, we provide some background on the field of OMR; in Section 3, we present the problem and the proposed approaches; in Section 4, the complete experimental setup is described; in Section 5, results are reported and discussed; and, finally, the main conclusions of the work are summarized in Section 6.

2. RELATED WORK

Traditionally, OMR has been considered a multi-stage process [8]. The legacy pipeline distinguishes four stages:

(i) *image preprocessing*, including tasks such as binarization [9], distortion correction, or stave separation [10]; (ii) *music symbol detection*, including steps such as staff-line removal [11], connected-component search, and classification [12]; (iii) *notation assembly*, where the independent components are related to each other to reconstruct the musical notation [13]; and (iv) *encoding*, in which the recognized notation is exported to a specific language that can be stored and further processed by computational means [14].

With the rise of deep learning, many of these steps have been reformulated as machine learning problems to be solved by neural networks [15, 16]. Also, many stages have been merged, giving rise to models that are capable of locating and categorizing the musical elements of the given image in a single step. This task has been the subject of extensive recent research [6, 17–19]. Alternatively, the so-called *holistic* or *end-to-end* approaches that seek to perform the entire pipeline in a single step have also been proposed, often with some prior pre-processing such as staff segmentation [14, 20, 21].

Although end-to-end approaches seem promising, so far they have only been successfully implemented for monodic music collections, where there is a clear left-to-right reading order. This is useful in many of the historical music heritage, such as plainchant or mensural music, where the different voices (if any) typically appear on different pages or sections, and staves are therefore monodic in the graphical sense. However, to deal with the common western modern notation, multi-stage OMR approaches seem to be the only ones capable of dealing with such complexity [4].

However, despite the aforementioned recent advances in the detection of music symbols with deep learning, there are hardly any proposals that complete the notation assembly stage employing machine learning techniques. To our knowledge, the only existing work that focuses on the retrieval of relationships using learning techniques is that of Pacha et al. [22]. In such work, for each pair of nodes, a single image is built with different channels: one that depicts the area of the image that contains both nodes, another that depicts the same region but only shows the first node, and a last one that depicts also the region of interest but only with the second node. A Convolutional Neural Network (CNN) is then trained to recognize whether or not there is a relationship between the nodes involved in this three-channel image. Despite the reported good results, the approach is tremendously inefficient, since it requires the independent construction and classification of an image for each pair of nodes. As we will see below, this scheme entails a huge computational complexity that makes it infeasible to use in practice.

In this paper, we especially focus on providing a solution to the notation assembly stage with a level of efficiency that enables its use in a real system, while keeping good accuracy figures.

3. METHODOLOGY

This paper follows the formulation proposed in previous works [19, 22, 23], where it is assumed that the computational reading of a music score, in the context of OMR, can be described by retrieving a graph structure from the image. In this graph, the atomic notation elements (referred to as "primitives") represent nodes, while edges denote the relationships between them. Here, we are particularly interested in the retrieval of the edges, once the nodes have been detected somehow (for instance, with the existing approaches mentioned in the previous section). Note that, instead of relying on case-specific heuristics, we frame the task within a learning-based formulation due to its inherent capability of modeling any relationship among the primitives as far as there exists a set of annotated reference data. Therefore, the formulation is general and can be used as long as there is a training set consistent with the envisioned model for the music-notation graph.

3.1 Formulation

A graph is a mathematical structure that models pairwise relationships between elements—referred to as nodes or vertices—through its edges. Here, we aim to retrieve the edges (relationships) between each pair of nodes in music scores, where each node represents a music primitive—e.g., a notehead, a stem, or an accidental.[1] The formal definition of the problem is as follows.

We assume that for a given music score s there exists a graph g_{s} that represents its symbolic music notation. The graph is defined as a pair (V, E), where V denotes the set of nodes and E denotes the set of edges. Two nodes $v_i, v_j \in V$ are connected if there exists an edge $e_{i,j} = (v_i, v_j) \in E$.

In the context of OMR, information about the set of symbols V corresponds to the *music symbol detection* stage of the OMR pipeline. Although they are still far from perfect, there are approaches in the literature that address this stage (cf. Sect. 2). Therefore, we here assume that there exists a function that maps s onto set V. Typically, each symbol $v_i \in V$ is further represented as a set of features with, at least, the following information: primitive class and coordinates within the image score. The problem we address from now on is how to get the set E given V, which corresponds to the *notation assembly* stage of the OMR pipeline.

The problem can be considered as a binary classification task in which a model predicts the class between each pair of nodes v_i, v_j present in the score. In this regard, $e_{i,j}$ is labeled as a 1 if there is a relationship between v_i and v_j, and 0 otherwise. The prediction of the relationship—henceforth, $\hat{e}_{i,j}$—can be represented as a function $\varphi(v_i, v_j)$ that takes the two nodes' features as input and computes the probability of connection, i.e., $\mathrm{P}(e_{i,j} = 1)$. Figure 1 depicts a general outline of the methodology adopted in this work.

[1] Hereafter, we use the terms "node", "symbol", and "primitive" interchangeably: a graphical element placed in the music score with certain attributes.

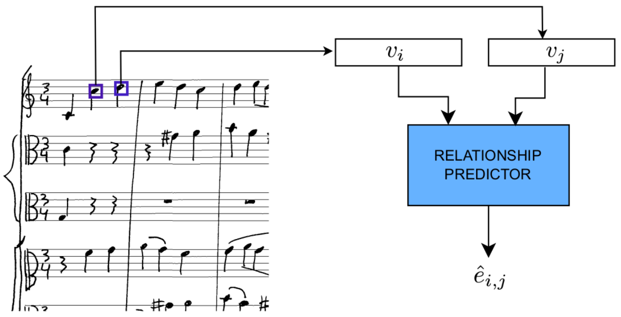


Figure 1: General schema of the methodology for retrieving the edges of the music notation graph.

3.2 Approaches

From the formulation given above, it is important to emphasize that the complexity of predicting each possible edge belongs to $O(|V|^2)$. Therefore, the approaches to φ must take into account the computational cost to make the task feasible in practice. This is a driving criterion for our approaches below since, with a sufficient number of primitives in a score, no system depicting the aforementioned complexity would be practical in a real-world scenario.

We here propose two shallow neural architectures that take a pair of nodes and predict the class of the relationship. These two neural architectures are: (i) a Multilayer Perceptron (MLP) architecture that takes the input of each node's features concatenated; and (ii) an asymmetric kernel model.

3.2.1 MLP architecture.

In this method, the features—attributes—of the nodes are first concatenated, forming a single feature vector that contains the entire information from the pair of nodes. Then, this vector is passed through a series of layers of an MLP. The final layer implements a function σ that models the probability of the two input nodes being connected:

$$\hat{e}_{i,j} = \sigma(\varphi_{\text{MLP}}([v_i, v_j]))$$

3.2.2 Asymmetric kernels.

In this second scheme, our proposed neural architecture learns an asymmetric kernel (AsymK) function [7]. This function is defined by $k(v_1, v_2) = (\langle \phi_{k_1}(v_1), \phi_{k_2}(v_2) \rangle)$, where $\langle \cdot, \cdot \rangle$ is the dot product of two N-dimensional points in two Hilbert spaces—features spaces. In this work, we use this asymmetric kernel as a similarity function between the two mapped features to distinct Hilbert spaces as:

$$\hat{e}_{i,j} = \sigma(\langle \phi_{k_1}(v_i), \phi_{k_2}(v_j) \rangle)$$

In this approach, $\phi_{k_1}(v_1), \phi_{k_2}(v_2)$ are kernels implemented as dense neural layers that map the initial node features onto two different (asymmetric) spaces that suit the task at hand. After computing the similarity score, a σ function is applied to obtain probabilities between 0 and 1.

Note that, since the embeddings are calculated only once per node, the scheme is remarkably efficiency. For each possible relationship, it is only necessary to compute the dot product between node embeddings and apply the σ function. That is why the complexity is much lower than the previous approach.

3.2.3 Loss function.

In both neural architectures proposed, the objective is to minimize the binary cross-entropy (BCE) loss function

$$\mathcal{L}_{\text{BCE}} = \sum_{e_{i,j} \in E} e_{i,j} \log(\hat{e}_{i,j}) + (1 - e_{i,j}) \log(1 - \hat{e}_{i,j}) \tag{1}$$

where $\hat{e}_{i,j}$ corresponds to the probability predicted by the model and $e_{i,j}$ is the ground-truth data for the edge (1 for a positive relationship, 0 otherwise).

4. EXPERIMENTS

In this section, we describe the experimental setup for evaluating the neural architectures proposed. More precisely, the rest of the section presents the corpus considered for the experiments, the contemplated figures of evaluation, the implementation details of the two neural proposals, and the feature descriptions used.

4.1 Data

The experiments were carried out using the MUSCIMA++ dataset [23]. This corpus provides 140 handwritten music scores with manual annotations of the different musical symbols—primitives defined by the symbol bounding box and the corresponding class label—and existing relationships among them. The dataset provides the direction of the edges; in our work, however, an undirected edge is assumed between two nodes that are connected regardless of the specific direction (undirected graph). Figure 2 depicts an example from this corpus.

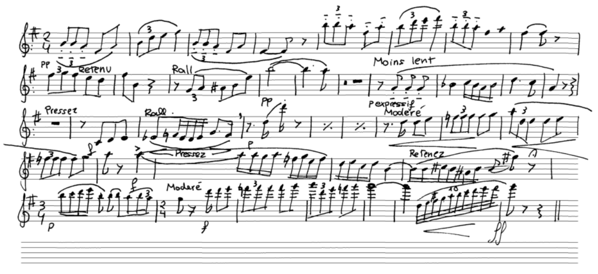


Figure 2: Example of a music score extracted from the MUSCIMA++ dataset.

Concerning the data partitioning, we follow a 5-fold cross-validation scheme. At each iteration, 60% of the dataset is used for training, 20% is used for validation, and 20% is used as test.

Finally, it must be highlighted that each music sheet depicts an average value of 734 primitives, which constitutes a large number of relations to be modeled. Due to this, and as aforementioned, efficiency must be considered in the design of practical notation assembly strategies.

4.2 Figures of merit

We consider a two-fold assessment of the proposed approaches, *i.e.*, we evaluate their recognition capabilities as well as efficiency rates. These criteria are now detailed.

In terms of recognition performance, as in previous works considering the same evaluation corpus [22, 23], we resort to the F-measure (F_1) metric. Note that, instead of providing the average scores for the two classes, the figures reported exclusively refer to the positive relationships with the aim of measuring the quality of the retrieval.

Concerning the efficiency assessment, we measure the computation time in the prediction phase of the methods. Since this metric depends on the computational capabilities of the device used, all methods are run over the same machine to avoid any possible bias. [2] Moreover, each experiment is repeated 10 times, being the average processing time the one reported as the efficiency score.

4.3 Neural architectures

Regarding the MLP architectures, we consider two implementations with a varying number of layers and weights to balance the trade-off between efficiency and representational power:

- $\mathrm{MLP}_{64,512}$: A three-layered fully-connected network comprising two hidden layers with 64 and 512, respectively, with *Rectifier Linear Unit* (ReLU) activations and a single output unit to compute the score of the binary classification.
- MLP_{32}: A two-layered fully-connected network comprising a 32-unit hidden layer and ReLu activation and a single unit as output.

Concerning the AsymK, ϕ_{k_1}, ϕ_{k_2} are implemented as two different 4-layered MLP comprising 512, 1024, 512, and 256 units, respectively, with ReLU activation. The idea is to generate two 256-dimensional embeddings—two points in different Hilbert spaces—to then compute the similarity through the dot product.

In all cases, the last operation is implemented as a *sigmoid* activation function to understand the output as a probability of a positive relationship. This probability is eventually thresholded considering a value of 0.5 to convert it in an actual decision.

Regarding optimization, all models were trained for 200 epochs using the Adam optimizer [24] with a learning rate of 10^{-3}.

4.4 Feature description

As aforementioned, the music-object detection stage of an OMR process retrieves, at least, the position (coordinates) of the detected object in the image sheet together with its estimated class label. For our relationship prediction approaches, we consider that each vertex ($v_i \in V$) is represented only by these features, being the inclusion of additional information left as future work.

Delving on the features considered, the spatial (position) information is directly encoded using four normalized values that denote the top-left and right-bottom corners of the bounding box. Conversely, the class information is processed by a 16-dimension learnable embedding layer to obtain an adequate representation for the task. Therefore, every single node is finally represented as a 20-dimensional feature vector.

5. RESULTS

Having introduced the different neural proposals as well as the experimental procedures, this section presents and discusses the results obtained. To establish a reference in the effectiveness that can be obtained for this task, we include the results of Pacha et al. [22], measured under the same experimental conditions as the rest of the methods in the work. [3]

The rest of the section separately studies and analyzes the two individual aspects considered, *i.e.*, performance efficiency and the ability to retrieve syntactic relationships between primitives.

5.1 Performance efficiency

Focusing first on the temporal aspect of the strategies, Table 1 shows the per-page average execution time of the contemplated notation assembly strategies. Note that, since this evaluation disregards the correctness of the estimation but simply assesses its temporal cost, all experiments are performed considering the ground-truth annotations.

Table 1: Efficiency results in terms of the per-page absolute execution time (in milliseconds) on the MUSCIMA++ corpus for the different notation assembly methods assessed. Each value corresponds to the average execution time obtained with 10 different iterations over all test samples.

	AsymK	$\mathbf{MLP_{32}}$	$\mathbf{MLP_{64,512}}$	**CNN** [22]
Execution time (ms)	< 0.5	55	176	$> 1.5 \cdot 10^6$

As can be observed, the existing CNN method [22] proves to be the least efficient among the considered strategies due to the large execution time it exhibits (roughly, 25 minutes per page). Such a point directly disables its possible integration in any practical system that comprises user interaction.

[2] The experiment was run over 8 cores of i7-7700K CPU at 4.20GHz with 16 GB of RAM memory, with no explicit parallelization or GPU speed-up.

[3] All experiments have been run considering the Python language (version 3.8), being the PyTorch (version 1.12.1) and PyTorch-lightning frameworks (1.9.1) particularly contemplated for reproducing the architecture proposed in Pacha et al. [22].

Oppositely, the different neural proposals presented in the work remarkably outperform these low-efficiency figures, achieving execution times in the order of a few milliseconds per page. More in detail, the AsymK stands as the most efficient strategy of the proposed ones as it reports figures several orders of magnitude faster than the MLP_{32} and $\text{MLP}_{64,512}$. Note that this is because AsymK exploits the parallelization of the dot product operation and the independent node processing while the two other proposals require more computation because of the classification framework they are based on.

It must be pointed out that the presented neural architectures depict execution times several orders of magnitude faster than the reference state-of-the-art method. In this regard, while the CNN strategy may be further optimized, the difference with the AsymK case—the most efficient strategy—must be considered as insurmountable.

5.2 Recognition capability

Let us now move to compare the estimation goodness of the different notation assembly strategies. As aforementioned, these methods take as input the result from a given framework that detects the primitives in the music score, *i.e.*, an object detection strategy.

Taking this into consideration, we will not consider any existing object detection approach as starting point, since other additional issues should be taken into account which are outside the scope of this work—*e.g.*, which object detection strategy to use, what confidence level to actually retrieve an object, or how to evaluate cases where a node is missing or has been predicted with the wrong label. Note that, since these questions are not related to the retrieval of the relationships themselves, any decision in this regard might bias the analysis of the models for the targeted notation assembly stage.

Nevertheless, to cover a greater number of possibilities in an agnostic way to the considered music-object detection step, we will simulate inaccuracies in the location process of the nodes. Specifically, we will consider a set of ranges for the Intersection over Union (IoU) metric between the original objects—the ground-truth annotations—and those generated for this experiment.[4] For that, assuming that the MUSCIMA++ corpus depict an IoU $= 1$ (ground-truth annotation), we will progressively perturb the location of the objects—*i.e.*, altering the coordinates of the bounding boxes—so that the overall IoU metric degrades to the range IoU $\in [0.85, 0.95]$. This range will decrease in steps of 0.1 (i.e., [0.75–0.85], [0.65–0.75], ..., [0.05–0.15]) to simulate scenarios depicting more limited symbol detection methods. In this way, our study focuses on general advantages and limitations of the notation assembly models, which can be then considered for developing more adequate pipelines for the previous stages. Figure 3 shows examples of how node locations are perturbed at some of the IoU levels considering the proposed strategy.

Considering this experimental set-up, Figure 4 shows the recognition rates in terms of F_1 achieved by the different neural schemes contemplated with respect to increasing IoU conditions (x-axis).

For the ideal scenario of perfect object-detection retrieval (IoU $= 1$), the reference CNN method [22] reports the highest recognition rate among all the schemes, with a value of $\text{F}_1 = 93.0\%$. However, the $\text{MLP}_{64,512}$ proposal shows slightly lower figures to the reference strategy—$\text{F}_1 = 91.9\%$—thereby proving itself as a competitive alternative to the CNN-based method in terms of accuracy. In relation to the AsymK and MLP_{32} proposals, these two strategies depict the least competitive results among the ones studied. However, since the MLP_{32} case shows a more competitive performance than the AsymK method, the former may be deemed as an intermediate case among the best-performing strategies—CNN and $\text{MLP}_{64,512}$—and the AsymK approach.

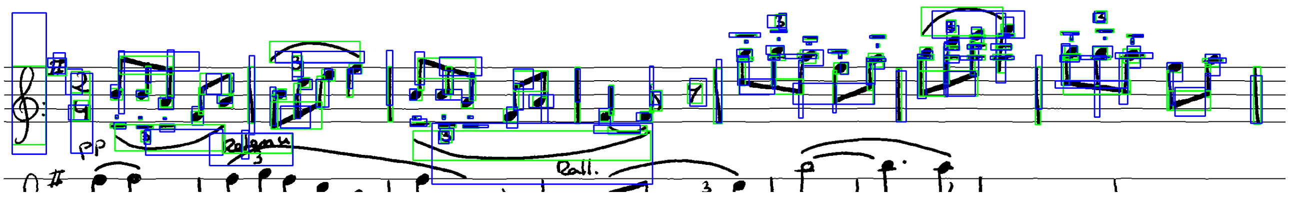
As the music-object detection becomes more realistic (IoU < 1), the neural models (except for CNN, which will be discussed below) do not degrade ostensibly but exhibit certain robustness up to reasonable IoU cases (above 0.5).[5] Digging deeper into the curves, the most relevant phenomenon is that, although at the higher ranges the CNN approach maintains the best accuracy, it decays much faster than the MLPs. Specifically, from IoU $\in$ [0.75–0.85], the $\text{MLP}_{64,512}$ outperforms it in terms of F_1, while maintaining the clear advantage in efficiency reported in the previous section. Furthermore, the rather shallow MLP_{32} approach also outperforms the CNN from IoU $\in$ [0.55–0.65], which are still likely values for music-object detection. These results reflect the adequacy of the efficient approaches proposed in this work, which are not only efficient enough to be used in practice but also keep greater robustness against very common distortions in previous stages to that of notation assembly in the OMR pipeline. In contrast, the AsymK shows a similar trend to the other efficient approaches, so from the perspective of this experiment it maintains the same advantages and disadvantages already discussed above (even higher efficiency but very poor retrieval).

As a last point, it must be noted that the results reported in this work may be considered as a turning point for the development of novel approaches to music-object detection in a complete OMR workflow. For example, using the efficient approaches of this work, one can prioritize retrieving most of the objects at the cost of slightly losing some location accuracy. An object that is not detected is impossible to relate correctly, but if it is detected, even if inaccurately it might connect successfully with other nodes (see Fig. 4).

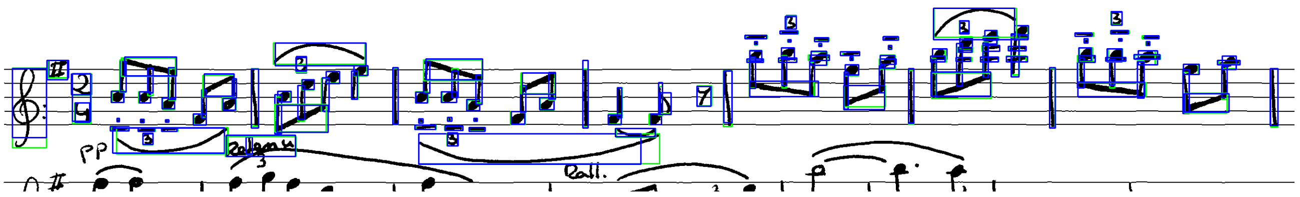
As a final note, the whole experiments clearly prove that there is no single strategy capable of optimizing both contemplated criteria at the same time: high recognition rates imply large execution times (*e.g.*, the CNN method [22],

[4] The IoU estimates the degree of overlap between two sets (in this case, areas of two music-notation objects) as the ratio of their intersection and their union.

[5] While it is true that the models obtain very poor results for the lowest ranges, this is not relevant in practice because 0.5 is the minimum IoU threshold for most object detectors to consider a correct retrieval.



(a) Simulated symbol detection performance of IoU $\in [0.45 - 0.55]$.



(b) Simulated symbol detection performance of IoU $\in [0.85 - 0.95]$.

Figure 3: Examples obtained with our proposal under different simulated symbol-detection scenarios based on the overall IoU. The green and blue bounding boxes respectively denote the ground truth and the modified ones.

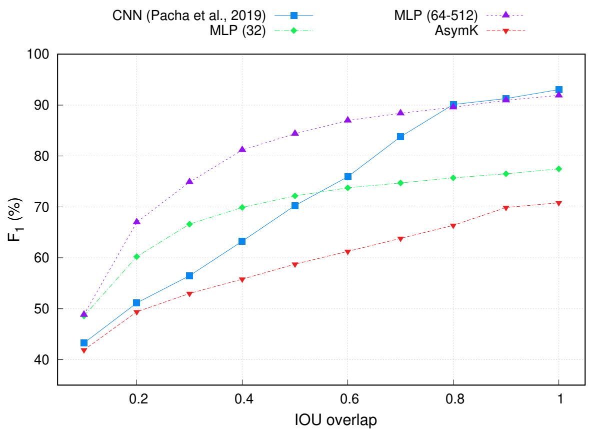


Figure 4: Results in terms of the F_1 metric for the compared note assembly strategies for different object detection performance rates based on the IoU score.

which results impractical in real-world applications) while faster strategies show more limited recognition rates (for instance, the AsymK case). In this regard, the proposed MLP-based architectures seem to provide an adequate balance between the two evaluation criteria, being particularly relevant to the $\text{MLP}_{64,512}$ one as it shows a remarkable temporal efficiency with a slightly worse performance than the highest attainable recognition results by the CNN case.

6. CONCLUSION

Optical Music Recognition (OMR) represents the research field that studies how to computationally read music notation from written documents. Generally, these strategies comprise an initial phase in which the music-notation elements from a given image are located—symbol detection—followed by a notation assembly stage that estimates the relations among these elements to reconstruct the musical notation itself. However, while there exist a large number of approaches that address the former process, the latter one has been scarcely addressed in the related literature.

This work frames in this particular assembly stage. Considering the high number of possible relationships in a music score, this work proposes two neural architectures to address this task in an efficient manner: (i) a strategy based on a Multilayer Perceptron (MLP) scheme; and (ii) a model based on asymmetric kernels. The results obtained with the MUSCIMA++ benchmark corpus [23] show that the MLP-based approach achieves recognition rates comparable to those of the reference strategy by Pacha et al. [22] with considerably less computational cost. Moreover, the asymmetric kernel approach, while proven to be extremely fast, exhibits a noticeable loss of accuracy with respect to the highest attainable one. In addition, these results also prove MLP-based schemes as remarkably robust when facing adverse symbol detection scenarios compared to the state-of-the-art method.

Several avenues of future research are opened: on the one hand, it would be important to estimate the relevance of each error produced since it has not been yet studied what errors—missing positive relationships or predicting non-existing relationships—and what type of elements involved cause the most impact on the eventual OMR system. On the other hand, this work has considered the given labeling of the MUSCIMA++; however, it has not been explored in depth whether this annotation scheme is actually adequate for these learning algorithms. More consistent or easy-to-learn annotations may be possible, as long as the goal of correctly encoding music notation is still met. Besides, just as the music-object detection step has been integrated into a single process, it would be beneficial to train end-to-end models that take into account both object detection and notation assembly. In this way, the model could leverage contextual and semantic information, provided by the notation assembly stage, when detecting objects that would be otherwise difficult or impossible. Finally, it would be also relevant to carry out user studies to assess the usefulness of these efficient approaches in real-world OMR scenarios.

7. ACKNOWLEDGMENT

Work produced with the support of a 2021 Leonardo Grant for Researchers and Cultural Creators, BBVA Foundation. The Foundation takes no responsibility for the opinions, statements and contents of this project, which are entirely the responsibility of its authors.

8. REFERENCES

[1] D. Bainbridge and T. Bell, "The challenge of optical music recognition," *Computers and the Humanities*, vol. 35, pp. 95–121, 2001.

[2] C. Müller, *Between Digital Transformation in Libraries and the Digital Humanities: New Perspectives on Librarianship*. De Gruyter, 2020, pp. 379–384.

[3] E. R. Miranda, *Handbook of artificial intelligence for music*. Springer, 2021.

[4] J. Calvo-Zaragoza, J. Hajic Jr., and A. Pacha, "Understanding Optical Music Recognition," *ACM Comput. Surv.*, vol. 53, no. 4, pp. 77:1–77:35, 2020.

[5] F. J. Castellanos, C. Garrido-Munoz, A. Ríos-Vila, and J. Calvo-Zaragoza, "Region-based layout analysis of music score images," *Expert Systems with Applications*, vol. 209, p. 118211, 2022.

[6] A. Pacha, J. Hajič, and J. Calvo-Zaragoza, "A baseline for general music object detection with deep learning," *Applied Sciences*, vol. 8, no. 9, p. 1488, 2018.

[7] W. Wu, J. Xu, H. Li, and S. Oyama, "Asymmetric kernel learning," *Technical Report, Microsoft Research*, 2010.

[8] A. Rebelo, I. Fujinaga, F. Paszkiewicz, A. R. Marcal, C. Guedes, and J. S. Cardoso, "Optical music recognition: state-of-the-art and open issues," *International Journal of Multimedia Information Retrieval*, vol. 1, no. 3, pp. 173–190, 2012.

[9] J. A. Burgoyne, L. Pugin, G. Eustace, and I. Fujinaga, "A comparative survey of image binarisation algorithms for optical recognition on degraded musical sources," in *Proceedings of the 8th International Conference on Music Information Retrieval*, 2007, pp. 509–512.

[10] M. Kletz and A. Pacha, "Detecting Staves and Measures in Music Scores with Deep Learning," in *Proceedings of the 3rd International Workshop on Reading Music Systems*, Alicante, Spain, 2021, pp. 8–12.

[11] J. dos Santos Cardoso, A. Capela, A. Rebelo, C. Guedes, and J. P. da Costa, "Staff Detection with Stable Paths," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 6, pp. 1134–1139, 2009.

[12] A. Rebelo, G. A. Capela, and J. S. Cardoso, "Optical recognition of music symbols - A comparative study," *Int. J. Document Anal. Recognit.*, vol. 13, no. 1, pp. 19–31, 2010.

[13] F. Rossant and I. Bloch, "Robust and Adaptive OMR System Including Fuzzy Modeling, Fusion of Musical Rules, and Possible Error Detection," *EURASIP Journal on Advances in Signal Processing*, vol. 2007, no. 1, p. 081541, 2006.

[14] A. Ríos-Vila, J. Calvo-Zaragoza, and D. Rizo, "Evaluating simultaneous recognition and encoding for optical music recognition," in *Proceedings of the 7th International Conference on Digital Libraries for Musicology*. ACM, 2020, pp. 10–17.

[15] A. Pacha and H. Eidenberger, "Towards a universal music symbol classifier," in *Proceedings of the 12th International Workshop on Graphics Recognition*. IEEE, 2017, pp. 35–36.

[16] F. J. Castellanos, A. Gallego, and J. Calvo-Zaragoza, "Automatic scale estimation for music score images," *Expert Syst. Appl.*, vol. 158, p. 113590, 2020.

[17] A. Pacha, K. Choi, B. Coüasnon, Y. Ricquebourg, R. Zanibbi, and H. Eidenberger, "Handwritten music object detection: Open issues and baseline results," in *Proceedings of the 13th IAPR International Workshop on Document Analysis Systems*. IEEE Computer Society, 2018, pp. 163–168.

[18] L. Tuggener, Y. P. Satyawan, A. Pacha, J. Schmidhuber, and T. Stadelmann, "The DeepScoresV2 Dataset and Benchmark for Music Object Detection," in *Proceedings of the 25th International Conference on Pattern Recognition*. IEEE, 2020, pp. 9188–9195.

[19] C. Garrido-Munoz, A. Ríos-Vila, and J. Calvo-Zaragoza, "Retrieval of music-notation primitives via image-to-sequence approaches," in *Proceedings of the 10th Iberian Conference on Pattern Recognition*, ser. Lecture Notes in Computer Science, vol. 13256. Springer, 2022, pp. 482–492.

[20] J. Calvo-Zaragoza, A. H. Toselli, and E. Vidal, "Hybrid hidden markov models and artificial neural networks for handwritten music recognition in mensural notation," *Pattern Anal. Appl.*, vol. 22, no. 4, pp. 1573–1584, 2019.

[21] C. Garrido-Munoz, A. Ríos-Vila, and J. Calvo-Zaragoza, "A holistic approach for image-to-graph: application to optical music recognition," *Int. J. Document Anal. Recognit.*, vol. 25, no. 4, pp. 293–303, 2022.

[22] A. Pacha, J. Calvo-Zaragoza, and J. Hajic Jr., "Learning notation graph construction for full-pipeline optical music recognition," in *Proceedings of the 20th International Society for Music Information Retrieval Conference*, 2019, pp. 75–82.

[23] J. Hajic Jr. and P. Pecina, "The MUSCIMA++ dataset for handwritten optical music recognition," in *Proceedings of the 14th IAPR International Conference on Document Analysis and Recognition*. IEEE, 2017, pp. 39–46.

[24] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations*, 2015.