Handwritten Music Object Detection: Open Issues and Baseline Results

Alexander Pacha, Horst Eidenberger
Institute of Visual Computing and Human-Centered Technology, TU Wien, Vienna, Austria
{*first name*}.{*last name*}@*tuwien.ac.at*

Kwon-Young Choi, Bertrand Coüasnon, Yann Ricquebourg
Univ Rennes, CNRS, IRISA, F-35000 Rennes, France
{*first name*}.{*last name*}@*irisa.fr*

Richard Zanibbi
Rochester Institute of Technology, Rochester, USA
rlaz@cs.rit.edu

***Abstract*—Optical Music Recognition (OMR) is the challenge of understanding the content of musical scores. Accurate detection of individual music objects is a critical step in processing musical documents because a failure at this stage corrupts any further processing. So far, all proposed methods were either limited to typeset music scores or were built to detect only a subset of the available classes of music symbols. In this work, we propose an end-to-end trainable object detector for music symbols that is capable of detecting almost the full vocabulary of modern music notation in handwritten music scores. By training deep convolutional neural networks on the recently released MUSCIMA++ dataset which has symbol-level annotations, we show that a machine learning approach can be used to accurately detect music objects with a mean average precision of over 80%.**

***Keywords*-Optical Music Recognition; Object Detection; Handwritten Scores; Deep Learning**

I. Introduction

Optical Music Recognition (OMR) attempts to understand the musical content of documents containing printed or handwritten music scores by recognizing the visual structure and the objects within a music sheet. Once, all objects are recognized, a semantic reconstruction step attempts to understand the relations of objects to each other and recover the musical semantics. With recent advances in computer vision, accelerated by the popularity of deep convolutional neural networks (CNN), OMR received a number of groundbreaking contributions that generate very accurate results for particular sub-problems, such as staff line removal [1] or symbol classification [2]. In this work, we investigate the challenge of music object detection which aims at accurately detecting music objects in music scores. Music objects can be both primitive glyphs (e.g. note-head, stem, beam) or compound symbols (e.g. notes, key-signatures, time-signatures) used in music notation. A music object detector takes an image and outputs the bounding-box and class-label for each found object. Traditionally, this was solved by first removing the staff lines, followed by symbol segmentation and classification [3] (see Figure 1).

In this work, we present the first attempt to establish a baseline for music object detection of handwritten scores with the full vocabulary of modern music notation. By following a machine learning approach and using an end-to-end trainable object detector on the recently published MUSCIMA++ dataset, we demonstrate how to build a generalizable and accurate music object detector and investigate the effects of various technical choices like the use of a particular detector or feature extractor.

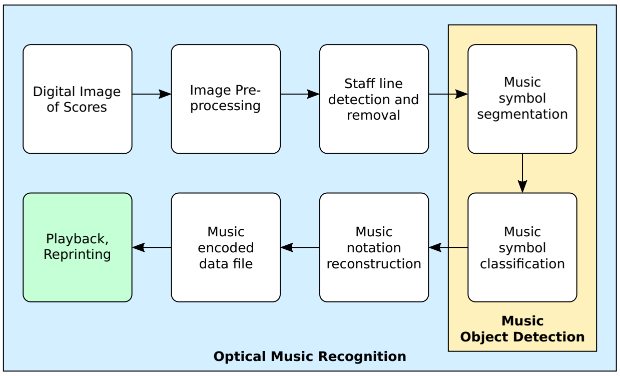
Digital Image of Scores
Image Pre-processing
Staff line detection and removal
Music symbol segmentation
Music symbol classification
Music notation reconstruction
Music encoded data file
Playback, Reprinting
Music Object Detection
Optical Music Recognition

Figure 1. The traditional pipeline for Optical Music Recognition. Music object detection subsumes segmentation and classification of music symbols.

II. Related Work

Visual object detection is a very active field of research with remarkable results on detecting objects in natural images with a variety of active competitions. Many competing approaches have been proposed in the last few years such as Faster R-CNN [4], R-FCN [5] and Single shot detectors [6], [7]. While some optimize for accuracy, others strive for high-performance [8]. However, all of them share the fact, that they heavily make use of deep convolutional neural networks.

The traditional pipeline of segmenting and classifying symbols has been shown to work well on simple typeset music scores with a known music font [9]. But when considering low-quality images, complex scores or even handwritten ones [10], these systems tend to fail, mainly because errors propagate from one step to subsequent steps [11], e.g. a segmentation error could cause incorrectly detected objects. Initial attempts to overcome this limitation by directly detecting music objects with CNNs were made by Hajič and colleagues, who suggest an adaptation of Faster R-CNN with a custom region proposal mechanism based on the morphological skeleton to

978-1-5386-3346-5/18 $31.00 © 2018 IEEE
DOI 10.1109/DAS.2018.51

accurately detect noteheads [12] and Choi and colleagues, who are able to detect accidentals in dense piano scores with high accuracy, given previously detected noteheads, that are being used as input-feature to the network [13]. However, both of them are limited to experimentations on a tiny subset of the full vocabulary used in modern music notation. Although both approaches can be extended to other classes, it remains an open question, whether a general purpose detector that can learn a large vocabulary is superior to multiple class-specific detectors.

A very interesting alternative to the traditional OMR pipeline is the attempt of solving OMR in a holistic fashion. The first notable attempt at doing so was by Pugin [14], who used Hidden Markov Models to read typographic prints of early music. More recently, the combination of using CNNs jointly with Recurrent Neural Networks to build an end-to-end trainable OMR system [15] was adapted and extended in [16] and [17]. Both train very similar models on a very large set of monophonic music scores containing a single staff per image. Although the reported results on the given datasets are very good, the two systems mentioned lastly, currently exhibit the following limitations:

- They operate only on very primitive, printed, monophonic scores. Extending their pipeline to more complex music scores with multiple voices requires a different formulation of the output data to at least include onset and offset of each note and not only the pitch and duration.
- By using pooling operations during the feature extraction, the network gains location invariance that conflicts with the interest of precise location information, which is needed to correctly infer the pitch of a note.
- By omitting the positional information of individual symbols and only considering the audible information of music symbols as output, such systems restrict themselves to replayability, as reprinting of music scores requires precise positional information [18].

While in theory semantic segmentation of the scores would go one step further and extract considerable more information – basically a classification of each pixel – two things should be noted: classifying pixels assumes that the class of each pixel is unique and mutually exclusive [19], an assumption that might not hold for overlapping symbols but can probably be ignored for practical applications; and most traditional systems that attempt to perform semantic reconstruction operate on detected objects, not on individual pixels, thus requiring a clustering step after the semantic segmentation. Therefore we argue, that detecting bounding boxes of musical objects directly is preferable for OMR.

III. The Challenge of Detecting Music Symbols

When comparing music object detection to detection of objects in natural scenes or optical character recognition, two unique challenges are worth noting: firstly, music scores often have a very high density of objects with more than 1000 objects printed on a single page. Secondly, the relative position between a symbol and its staff lines is crucial. Already a tiny error along the y-axis may have a significant impact on recovering the correct pitch of a note.

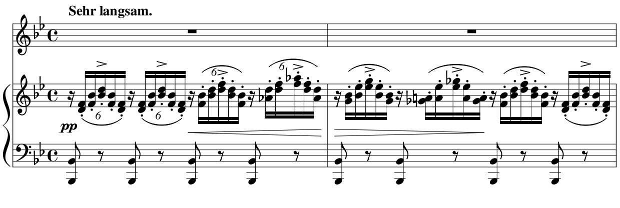

Figure 2. Beginning of Franz Schubert's Ave Maria D. 839, with simplifications in the second bar that intentionally violate the syntactic rules of common music notation.

The detection of music objects is of paramount importance to the overall OMR process because once all symbols were detected accurately, a set of rules can be applied to infer the semantics of the objects and perform music notation reconstruction as demonstrated by [20]. We also suggest that the point right after individual objects were detected and classified, is probably the best moment for putting the user into the loop, if that is intended. Fixing errors at this stage can be performed locally without dealing with complicated semantic rules or affecting neighboring symbols (changing the duration of a single note in a music notation program often entails side effects on other notes within the same of subsequent bars). Highlighting uncertain detections and suggesting likely alternatives could improve the usability and reduce editing costs even further.

Note that even with all symbols being correctly detected and classified, recovering the musical semantics still remains a very challenging problem, as demonstrated in Figure 2. Here, the second staff in the first bar contains a small 6 for each tuplet, indicating that the first rest and the following five chords sum up to a quarter note. This small number is intentionally omitted in the second bar for simplification but would now result in an invalid meter if interpreted in isolation. Only with the preceding information and prior knowledge about common simplifications, a musician can interpret such scores correctly.

To be able to introduce such semantics into an OMR system, it is necessary to formalize and use musical notation knowledge. Rule-based systems can perform such formalization. For example, with the DMOS system [20] it has been possible to formalize the musical notation, graphically and syntactically, for full polyphonic scores, and produce a system which allows to assign notes to multiple voices and use the vertical alignments of synchronized notes in orchestral scores as well as the number of beats in a bar to detect and correct recognition errors. This grammatical formalization is built on terminals which correspond to the musical objects we propose to recognize with deep convolutional neural networks.

IV. Building a Music Object Detector

For building a robust and extensible music object detector, we propose a machine-learning approach with deep convolutional neural networks, which operate directly on the input image. This simplifies the OMR process to the following steps: preprocessing, music object detection, and semantic reconstruction. Steps such as removing the staff lines and segmenting symbols do not need to be addressed explicitly. Existing state-of-the-art object detectors such as Faster R-CNN or R-FCN were designed to detect objects in natural scenes and have been shown to work well on challenging datasets such as COCO [21] or ImageNet [22]. But applying them out-of-the-box on sheets of music can lead to a suboptimal performance, due to the dense nature of music scores with many tiny objects. Therefore, we suggest applying a certain amount of preprocessing to the data and tailor these detectors to perform well on the task at hand.

A. Dataset and Preprocessing Steps

For training a music object detector, we use the MUSCIMA++ dataset [23], as it contains 140 high-quality images with over 90000 symbol-level annotations, made by human annotators across 105 different classes of music symbols for the underlying CVC-MUSCIMA dataset [24]. The images have a high resolution of about 3500x2000 pixel, are binarized and optionally come with staff lines removed. For consistency, all white-on-black images are first inverted and then converted to RGB, as the evaluated implementations take colored images as input[1]. To efficiently train an object detector on such images, the image size has to be reduced. We propose to crop the images in a context-sensitive way, by cutting images first vertically and then horizontally, such that each image contains exactly one staff and has a width-to-height-ratio of no more than 2:1, with about 15% horizontal overlap to adjacent slices (see Figure 3). Basically, each horizontal slice extends from the bottom of the staff above to the top of the staff below. This cropping can also be done by automatically detecting staffs and then applying the same slicing rules leading to image crops that partially overlap both horizontally and vertically. For splitting the cropped images into a train and test set, we follow the recommendations from [23] to ensure that the test set contains scores of all complexities and that there is no overlap of writers between the training and the test set. We furthermore used 10% of the remaining training set for validation during the training. In total, we obtained 6181 samples, that were divided into a training, validation and test set, containing 4794, 533 and 854 images respectively.

One limitation of this approach is, that all objects significantly exceeding the size of such a cropped region, will not appear in the data, as only annotations that have an intersection-over-area of 0.8 or higher between the object and the cropped region are considered part of the ground truth.

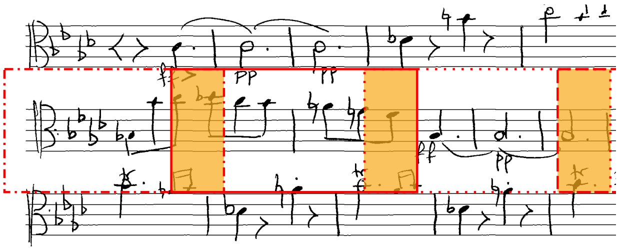

Figure 3. Illustration of the sliding window approach, used to crop music scores into meaningful subimages (red) with horizontally overlapping areas (orange) between adjacent crops.

As music objects, we consider the full vocabulary of all 105 classes contained in the MUSCIMA++ dataset, containing both primitives such as noteheads as well as compound objects such as key-signatures that consist of one or multiple accidentals.

B. Experimental Design

For evaluating our suggested approach, we conducted several experiments to study the performance of various object detectors and feature extractors, as well as the effects of staff line removal, transfer-learning and removing classes with rare symbols. Using the deep learning library TensorFlow[2], we adapted the work from [8] to detect music objects by training on the data described in Section IV-A. The entire source code, including training protocols and detailed instructions to reproduce our results, can be found at http://github.com/apacha/MusicObjectDetector-TF. We considered:

- the three meta-architectures Faster R-CNN, R-FCN, and SSD as object detectors. Faster R-CNN and R-FCN are both two-stage detectors with a region proposal network and a region classifier. The difference is that Faster R-CNN uses a sliding window for classification, whereas R-FCN uses position sensitive score maps and per-RoI pooling, which is more efficient at the cost of a slightly reduced precision. SSD is a generalized region proposal network for one stage detection on multiple feature maps
- ResNet50, Inception-ResNet-v2, MobileNet-v1 and Inception-v2 as feature extractors, explicitly excluding custom-made networks that cannot benefit from transfer-learning
- images with and without staff lines (based on the images provided along the CVC-MUSCIMA dataset)
- the full vocabulary of all 105 classes included in the MUSCIMA++ dataset, as well as a reduced set of only 71 classes, removing 34 classes that appear less than 50 times in the ground truth and are only of minor importance such as uncommon numerals and letters. Exceptions were only made for the classes double sharp and the numerals 5, 6, 7 and 8: although they appear less than 50 times in the dataset, we consider them essential to recover music semantics such as pitch and time signature.

[1] The overhead created by this conversion is only minimal, as the duplicated information gets merged again in the first layer of the CNN.

[2] https://www.tensorflow.org, last seen 9th February 2018

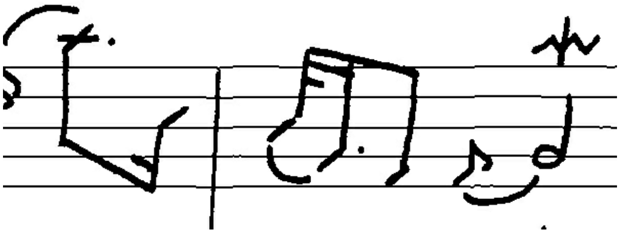

Figure 4. Typical sample of a cropped image that serves as input for the music object detector.

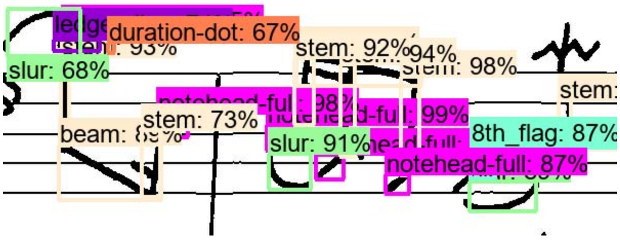

Figure 5. Typical detection results with most symbols recognized correctly.

All of the above-mentioned object detectors have a certain set of hyperparameters that need to be fine-tuned for the particular dataset. For example, [7] shows that using statistical analysis to obtain a sensitive number of anchor boxes, anchor box sizes, and anchor box ratios can improve the results significantly compared to handpicked priors. When running similar analysis on the cropped images, we obtain the following characteristics: For a typical input image of 600 pixels width and 300 pixels height (see Figure 4), we found the average square box size is about 37 pixels with a standard deviation of 48 pixels. Note, that the dataset also contains extreme cases of small objects like dots with only a few pixels and large objects that spans hundreds of pixels. The mean ratio from width to height of boxes is 0.7 which means that the majority of boxes are higher than they are wide. Furthermore, cropped images that are to be fed to the detector contain 19 symbols on average, with a standard deviation of 11. Concluding the analysis, we decided to use a grid of 32x32 pixels with a stride of 8 pixels and aspect ratios of 0.06, 0.29, 0.48, and 2.2 with the scales 0.25, 0.5, 0.75, 1.0, 1.75, and 4.0 to reflect the wide range of object shapes in the dataset.

C. *Evaluation and Results*

Following the evaluation protocols of the Pascal VOC challenge [25], we report the mean average precision (mAP) for each completed training in Table I and the detailed average precision per class for the combination that yielded the best results in Table II. Figure 5 shows a typical detection within a single image.

We find that the best performing detector with regards to precision is the Faster R-CNN using the Inception-Resnet V2 feature extractor, pre-trained on the COCO dataset. This model produces a mAP of over 80%. The training on a GeForce GTX 1080 Ti takes approximately one day per configuration before results become stable. Validating ˜500 images takes about 2-4 minutes, so inference should take less than half a second per (cropped) image. When comparing the results of training on images with and without staff lines, the impact is no longer significant, supporting the claim of [14], that staff line removal might no longer be necessary. However, readers should also note that the staff lines in the CVC-MUSCIMA dataset are synthetic and do not experience the usual distortions that apply to scans or pictures of real music scores.

Other detectors like the R-FCN or SSD produce good results as well, with a mAP of 75% and 71% respectively. Our results, therefore, comply with the findings of [8], where in particular the SSD model trades smaller accuracy for higher processing speed. Using pre-trained weights, instead of random initialization and the RMSprop optimizer as opposed to Stochastic Gradient Descent, improved the results significantly, speeded up convergence and was therefore used throughout the experiments. Modifying the set of classes by removing underrepresented classes as described in Section IV-B, boosted the mAP by up to 6% in some cases. Note, that Table II is missing six classes, that did not have any instances in the test set because they exceeded the size of the image crops and were thus discarded during the preprocessing.

V. Discussion and Conclusion

In this work, we show that state-of-the-art deep learning detectors like Faster R-CNN, R-FCN and SSD can produce accurate detection results on a wide range of music symbols. After optimizing different hyperparameters, we achieve a mAP of over 80%, which is a solid baseline.

However, there are still a couple of open issues, that need to be addressed in future work, like how to process a whole page of a score. In this work, we used a simple overlapping sliding window approach. This method, although simple to use, has many well-known downsides like the poor performance of processing empty images or cutting up large symbols as well as a non-trivial merging step that has to fuse information from multiple overlapping sections.

Another problem, specific to OMR, is the inherent imbalance of symbol classes: some symbols like noteheads are extremely frequent whereas others like double sharps are rare and often tied to a specific type of score. Having experimented with state-of-the-art deep learning object detectors, we found that classes do not interact with each other: simplifying the task by removing line-shaped classes did not improve the overall precision. There also seems to be a minimum threshold of about 20 samples per class, in order to be meaningful during the training. Currently, there is no guarantee, that the model does not overfit, but with recently published work like the RetinaNet and its focus loss [26] the effects of this class-imbalance could be mitigated to improve the training, especially on hard to detect classes.

Table I
DETAILED RESULTS FOR VARIOUS HYPERPARAMETER COMBINATIONS OF THE MUSIC OBJECT DETECTOR.

Meta-Architecture	Feature Extractor	Number of classes	Images have staff lines	Mean Average Precision on Test Set (%)	Weighted Mean Average Precision on Test Set (%)
Faster R-CNN	Inception-ResNet-v2	105	✓	81.56	94.22
Faster R-CNN	Inception-ResNet-v2	105	✗	81.23	94.56
Faster R-CNN	Inception-ResNet-v2	71	✓	85.12†	94.68
Faster R-CNN	Inception-ResNet-v2	71	✗	87.80‡	95.05
Faster R-CNN	ResNet50	105	✓	76.39	93.07
Faster R-CNN	ResNet50	105	✗	78.45	93.10
Faster R-CNN	ResNet50	71	✓	82.30	93.47
Faster R-CNN	ResNet50	71	✗	84.85	93.63
R-FCN	Inception-ResNet-v2	105	✓	69.75	89.12
R-FCN	Inception-ResNet-v2	105	✗	70.88	89.42
R-FCN	ResNet50	105	✓	75.53	92.59
R-FCN	ResNet50	105	✗	74.29	92.33
SSD	Inception-v2	105	✓	71.52	82.44
SSD	Inception-v2	105	✗	70.40	81.75
SSD	MobileNet-v1	105	✓	62.30	74.97
SSD	MobileNet-v1	105	✗	61.56	76.74

Although we used the test set, proposed by the MUSCIMA++ authors, where writers in the test set do not appear in the training set, we are still not certain whether this system is truly writer independent or not. One way to confirm this would be to perform a cross-validation, where each writer in the dataset is evaluated independently.

Finally, we have shown that removing staff lines can be omitted for music object detection, when using CNNs. Future experiments that apply data-augmentation using noise models and deformed images, as proposed for the staff removal challenge [27], can give even more insights into the robustness of our approach.

ACKNOWLEDGMENT

The authors would like to thank all creators of public OMR datasets for collecting them and making them freely available to other researchers.

REFERENCES

[1] A.-J. Gallego and J. Calvo-Zaragoza, "Staff-line removal with selectional auto-encoders," *Expert Systems with Applications*, vol. 89, pp. 138 – 148, 2017.

[2] A. Pacha and H. Eidenberger, "Towards a universal music symbol classifier," in *Proceedings of the 12th IAPR International Workshop on Graphics Recognition*, IAPR TC10 (Technical Committee on Graphics Recognition). New York, USA: IEEE Computer Society, 2017, pp. 35–36.

[3] 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.

[4] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99.

[5] Y. Li, K. He, J. Sun *et al.*, "R-FCN: Object detection via region-based fully convolutional networks," in *Advances in Neural Information Processing Systems*, 2016, pp. 379–387.

[6] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *European conference on computer vision*. Springer, 2016, pp. 21–37.

[7] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," *arXiv preprint arXiv:1612.08242*, 2016.

[8] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, and K. Murphy, "Speed/accuracy trade-offs for modern convolutional object detectors," *CoRR*, vol. abs/1611.10012, 2016.

[9] 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.

[10] Arnau Baró, Pau Riba, and Alicia Fornés, "Towards the recognition of compound music notes in handwritten music scores," in *2016 15th International Conference on Frontiers in Handwriting Recognition (ICFHR)*. Institute of Electrical and Electronics Engineers Inc., 2016, pp. 465–470.

[11] A. Pacha and H. Eidenberger, "Towards self-learning optical music recognition," in *Proceedings of the 16th IEEE International Conference On Machine Learning and Applications*, 2017, in print.

[12] J. j. Hajič and P. Pecina, "Detecting noteheads in handwritten scores with convnets and bounding box regression," *arXiv preprint arXiv:1708.01806*, 2017.

[13] K.-Y. Choi, B. Coüasnon, Y. Ricquebourg, and R. Zanibbi, "Bootstrapping samples of accidentals in dense piano scores for cnn-based detection," in *Proceedings of the 12th IAPR International Workshop on Graphics Recognition*, IAPR TC10 (Technical Committee on Graphics Recognition). New York, USA: IEEE Computer Society, 2017.

[14] L. Pugin, "Optical music recognitoin of early typographic prints using hidden markov models." in *ISMIR*, 2006, pp. 53–56.

Table II
Detailed precision results per class for the best obtained music object detector on the reduced set of classes (see Table I, line 3† and 4‡).

Class name	Total number of instances	Average precision on the test set (%)	
		with staff lines†	w/o staff lines‡
notehead-full	31084	99.85	99.64
stem	27108	98.82	98.71
ledger_line	14500	97.89	97.40
beam	8677	93.86	94.57
slur	3859	90.34	88.54
duration-dot	3195	95.12	94.21
thin_barline	3071	99.49	99.64
8th_flag	2744	93.46	93.37
measure_separator	2649	43.64	52.09
staccato-dot	2507	94.23	94.97
sharp	2420	99.42	99.46
notehead-empty	2385	99.31	99.11
flat	1467	96.97	97.98
natural	1427	96.90	97.61
dynamics_text	1374	85.25	87.12
8th_rest	1339	98.86	99.36
tie	1085	82.39	81.85
quarter_rest	1060	96.05	96.78
letter_p	1038	89.70	89.84
letter_f	1035	93.10	92.77
letter_e	926	82.12	85.29
letter_r	750	51.64	62.25
key_signature	697	79.31	77.80
letter_o	655	94.47	93.82
16th_flag	652	36.62	40.19
letter_s	649	71.89	74.30
grace-notehead-full	576	85.75	85.37
numeral_3	548	98.73	98.04
16th_rest	531	96.17	99.93
letter_t	513	92.10	94.42
other_text	508	83.99	89.30
letter_c	469	89.82	88.57
tuple	459	30.41	77.11
accent	421	99.08	95.75
g-clef	403	100.00	100.00
other-dot	402	94.40	95.19
repeat-dot	359	99.75	100.00
trill	315	100.00	99.74
letter_d	313	93.49	89.36
letter_m	293	74.19	74.43
f-clef	285	100.00	98.21
half_rest	241	95.53	91.16
time_signature	221	96.33	95.02
tenuto	216	88.45	74.79
letter_l	192	78.75	86.00
c-clef	190	97.68	98.68
whole_rest	183	90.73	84.66
letter_P	177	45.83	45.80
tempo_text	174	69.40	78.32
letter_i	171	66.48	81.08
letter_n	164	79.51	80.26
numeral_4	155	99.60	99.47
letter_a	134	90.36	83.81
multiple-note_tremolo	126	81.01	82.42
ornament(s)	123	85.22	83.90
letter_M	115	65.83	71.47
grace_strikethrough	110	98.14	97.96
letter_u	106	65.98	62.69
repeat	73	84.42	88.87
double_sharp	44	100.00	100.00
numeral_2	40	100.00	92.50
numeral_6	36	100.00	100.00
numeral_8	36	100.00	91.67
numeral_7	24	28.32	62.59
numeral_5	11	26.67	100.00

[15] B. Shi, X. Bai, and C. Yao, “An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PP, no. 99, pp. 1–1, 2016.

[16] J. Calvo-Zaragoza, J. J. Valero-Mas, and A. Pertusa, “End-to-end optical music recognition using neural networks,” in *18th International Society for Music Information Retrieval Conference*, 2017.

[17] E. van der Wel and K. Ullrich, “Optical music recognition with convolutional sequence-to-sequence models,” in *Proceedings of the 18th International Society for Music Information Retrieval Conference, ISMIR 2017, Suzhou, China*, 2017.

[18] H. Miyao and R. M. Haralick, “Format of ground truth data used in the evaluation of the results of an optical music recognition system,” in *IAPR workshop on document analysis systems*, 2000, pp. 497–506.

[19] J. Calvo-Zaragoza, G. Vigliensoni, and I. Fujinaga, “A machine learning framework for the categorization of elements in images of musical documents,” in *Third International Conference on Technologies for Music Notation and Representation. A Coruna: University of A Coruna*, 2017.

[20] B. Coüasnon, “Dmos: a generic document recognition method, application to an automatic generator of musical scores, mathematical formulae and table structures recognition systems,” in *Proceedings of Sixth International Conference on Document Analysis and Recognition*, 2001, pp. 215–220.

[21] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, *Microsoft COCO: Common Objects in Context*. Cham: Springer International Publishing, 2014, pp. 740–755.

[22] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, June 2009, pp. 248–255.

[23] J. j. Hajič and P. Pecina, “The MUSCIMA++ dataset for handwritten optical music recognition.” *Proceedings of the 14th IAPR International Conference on Document Analysis and Recognition*, 2017.

[24] A. Fornés, A. Dutta, A. Gordo, and J. Lladós, “CVC-MUSCIMA: a ground truth of handwritten music score images for writer identification and staff removal,” *International Journal on Document Analysis and Recognition (IJDAR)*, vol. 15, no. 3, pp. 243–251, 2012.

[25] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes challenge: A retrospective,” *International Journal of Computer Vision*, vol. 111, no. 1, pp. 98–136, 2015.

[26] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár, “Focal loss for dense object detection,” *CoRR*, vol. abs/1708.02002, 2017.

[27] A. Fornés, A. Dutta, A. Gordo, and J. Lladós, *The 2012 Music Scores Competitions: Staff Removal and Writer Identification*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 173–186.