# Watershed 알고리즘을 이용한 딥러닝 기반 악보 내 음악기호 탐지 방법*

홍석균[01], 이정렬[1], 김경민[1], 오준영[2], 정지훈[1*]
[1]충북대학교 소프트웨어학부, [2]충북대학교 컴퓨터과학전공
{goob5748, dlwjdfuf99, kkmlouis, jy.oh, jh.jeong}@chungbuk.ac.kr

# Method of Deep Learning-Based Music Symbol Detection Using Watershed Algorithm

Suk-Kyun Hong[1], Jeoung-Ryeol Lee[1], Kyeong-Min Kim[1], Jun-Young Oh[2], Ji-Hoon jeong[1*]
[1]School of Computer Science, Chungbuk National University
[2]Department of Computer Science, Chungbuk National University

**Abstract**

Musical score performers used to play through paper scores, but recently, they make and play music through electronic scores that are easy to produce and edit. However, not all musical scores have been converted into electronic scores, and scores without electronic scores should be produced by the performer based on paper scores. In this study, deep watershed detector using the watershed algorithm is used to recognize music symbols and reduce the effort of performers to read and input music symbols manually. The dataset is an image of a score including several music symbols. By changing the hyperparameters that can be changed in the model and comparing the performance of the experiments. This technology for detecting music symbols in sheet music can be used to convert paper scores into electronic scores in the future and contribute to digitalization of the music field.

## 1. Introduction

Even with the same music, scores can be arranged in various ways by the arranger based on factors such as difficulty and the number of performers, and accordingly, various versions of music may exist for the same music. As an example, Pachelbel's Canon In D Major has various versions of music through arrangement by many people so far. However, among these scores, only a few have already been converted into electronic scores and can be played as electronic scores without separate work. If a performer wants to convert a score without electronic score into electronic score, a person must directly read the score in the image or PDF format and enter music symbols (tone, note, rest, etc.) one by one through the electronic score editing program, and hundreds of types of music symbols can be entered. The average number of music symbols per page of the used score dataset is 243 [1], and it takes a lot of time and effort to enter each music symbol one by one.

In this study, we propose a method to reduce the labor required for a performer to directly convert into an electronic score through a deep learning-based music symbol recognition method using the watershed algorithm [2].

## 2. Materials & Method

### 2.1. Dataset

We used DeepScoresV2 [1] as the dataset for research. An example of the DeepScoresV2 dataset is shown in Fig. 1. DeepScoresV2 contains a total of 135 music symbol classes, and in addition to the note class, various music symbols such as a four-minute rest and a treble clef are included. The Dataset of DeepScoresV2 follows a format similar to the COCO Dataset. It is largely composed of musical score images
and JSON-type annotation file. The size of the score image varies, such as 1960 x 2772 and 2426 x 3432, but the width and length ratio are constant at 7:10. The annotation file contains the coordinates of the bounding box, which is expressed as axis-aligned bounding box (AABB), where all sides of the bounding box are aligned with the axis, and oriented bounding box (OBB), a directional box. AABB is in the form (x0, y0, x1, x1, y1, x2, x3, y3) and OBB is in the form (x0, y1, x2, x3, y3).

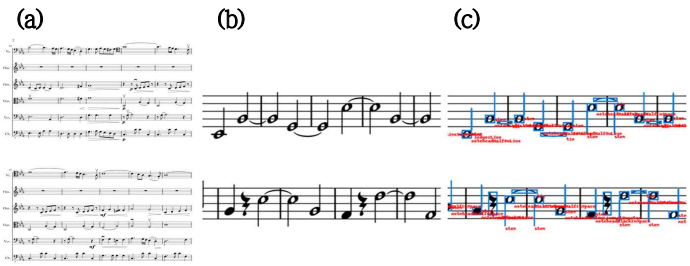


Fig. 1. Example of a DeepScoreV2 dataset (a) Original image of musical score. (b) An image of a part of the score (c) A ground truth with bounding box for a picture (b).

### 2.2. Model Architecture

The watershed algorithm [2] is one of the region-based image segmentation methods, which distinguishes objects by comparing them with neighboring pixel values based on pixel values. The watershed algorithm is known to perform well when dividing images that are touched or overlapped by objects, and since the

This research was supported by the MSP(Ministry of Science, ICT & Future Planning), Korea, under the National Program for Excellence in SW (2019-0-01183) supervised by the IITP(Institute of Information & communications Technology Planing & Evaluation) (2019-0-01183) and supported by the Institute of Information & communications Technology Planning & Evaluation (IITP) grant, funded by the Korea government (MSIT) (RS-2021-II212068, Artificial Intelligence Innovation Hub) and supported by Korea Institute of Planning and Evaluation for Technology in Food, Agriculture and Forestry(IPET) through Agriculture and Food Convergence Technologies Program for Research Manpower Development Program funded by Ministry of Agriculture, Food and Rural Affairs(MAFRA)(RS-2024-00398561),

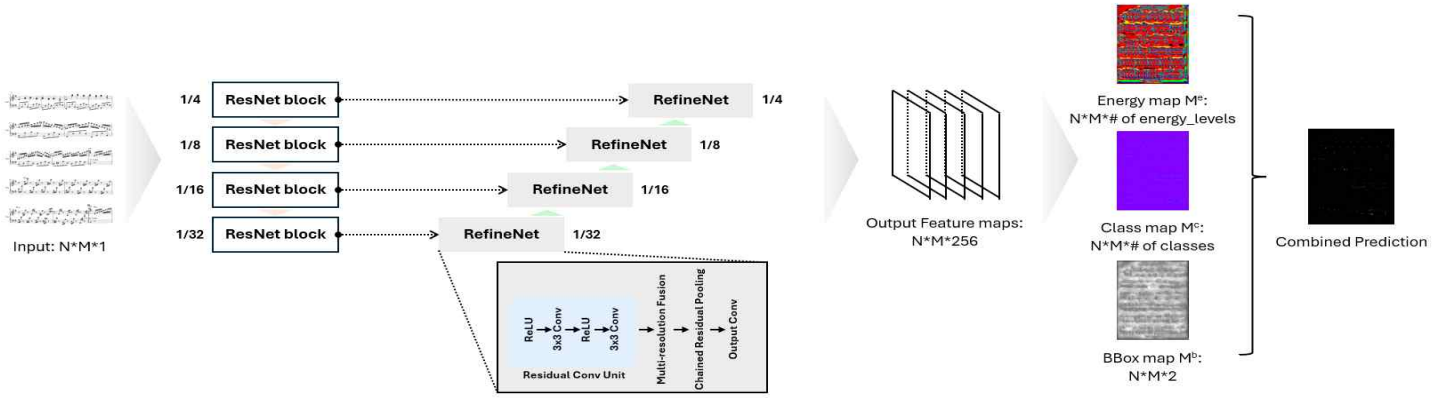


Fig. 2. The overall architecture of model. Using ResNet blocks for feature extraction and using RefineNet block during the up-sampling process.

music symbols in the score are located closely with the stave, their application in the field is expected to be useful. The deep watershed detector [3] consists of RefineNet [4], which uses ResNet [5] as a backbone inside and has three 1x1 convolution layers as the ouput layer. The energy map, class map, and bounding box (BBox) map are extracted as the output of the feature vector extracted from the image through the RefineNet through the 1x1 convolution layer. The energy map divides the image along the boundary line according to the height of each pixel value by the watershed algorithm and calculates the object center of the divided objects. The class map classifies objects according to the centers of gravity of the objects estimated earlier. If the object has the same center of gravity, it is considered the same class and has the same color. The BBox map predicts the bounding box for objects to determine the boundary of the bounding box. Finally, we learn by considering all the losses from the energy, class, and BBox processes. The overall architecture is shown in Fig. 1.

### 2.3. Experiment Method

In this study, 958 score data randomly selected from the entire dataset were studied with Train:Validation:Test = 8:2:2 without using all data from DeepScoresV2, with 684 Train data, 137 Validation data, and 137 Test data. Training is largely composed of four tasks. Each tasks consists of Energy, Class, BBox, and finally, Combine that learns by considering the previous tasks. 500 Iterations were given in the previous energy, class, and BBox. And in the combine task, 600 iterations were given to a total of 2200 Iterations.

## 3. Experimental Result & Discussion

As the model performance metrics, average precision (AP) [7] and F1 score, which are representative performance metrics of object detection, were used. Intersection over unit (IoU) is a metric that evaluates the accuracy of object detection and is an indicator of the ratio of overlapping areas between the predicted bounding box and the actual object bounding box. In this study, AP at 0.5, the ratio of predictions with IoU values of 0.5 or more per class, was used as the performance metric.

Precision is the ratio of what the model classifies as true to be true, and Recall is the ratio of what the model predicts to be true to be true. F1 score is calculated as the harmonized average of precision and recall.

Table Ⅰ. Three comparison groups that were trained by adjusting Hyperparameter

| | Backbone | Optimizer | Learning Rate | Iteration |
|---|---|---|---|---|
| A | ResNet-101 | RMSProp | 0.0001 | 2200 |
| B | **ResNet-152** | RMSProp | 0.0001 | 2200 |
| C | ResNet-152 | **Adam** | 0.0001 | 2200 |

Table Ⅱ. Detecting performance of three comparison groups: precision, recall, and f1-score

| | Precision | Recall | F1 Score |
|---|---|---|---|
| A | 0.000028 | **0.000004823** | 0.0000082288 |
| B | 0.0001 | 0.00001642 | **0.000028526** |
| C | **0.0002** | 0.000003766 | 0.0000073972 |

We will analyze the results through the three comparison groups that conducted the learning by adjusting the hyperparameter that can be changed. The settings for each case are shown in Table Ⅰ. In case of experiment A, ResNet-101 was used as the backbone and root mean square propagation (RMSProp) [6] was used as the optimizer. In case of experiment B, the backbone was changed to ResNet-152 under the condition of experiment A. In case of experiment C, the optimizer was changed to Adam [6] under the condition of experiment B. In addition, the learning rate and iteration were set the same for all case. Comparing F1 Score, precision, and recall by experimental group is shown in Table Ⅱ.

The experiment with the highest precision is C, and the experiment with the highest recall is A. However, in the case of F1 score, it can be seen that experiment B is the highest. In general, the lower the Precision value, the more cases of erroneous detection, and the lower the Recall value, the more cases of undetected. In the field of detecting musical symbols within a score, there is no significant difference between false and non-detections. In music, there is a beat score such as a quarter beat, and in the case of a quarter beat, four beats are included in a bar, for example. In the case of erroneous detection, one quarter note may be mistakenly recognized as a two-quarter note, and the beat may be exceeded within a bar. In this case, the beat within the bar does not match, so it can be easily corrected. In the case of undetected, it is easy to find a case where only three beats exist within the bar because one quarter note is not recognized. Therefore, in the

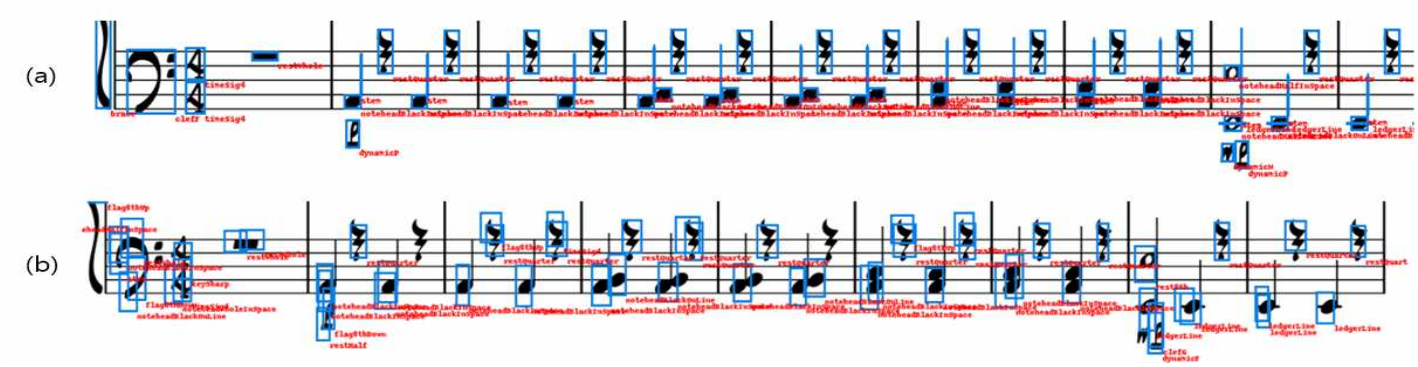


Fig. 3. Images visualizing the ground truth with bounding box and the bounding box predicted by the model

Table Ⅲ. Occurrences number of classes and AP at 0.5

| Class | No. Occurrences | AP at 0.5 |
|---|---|---|
| timeSig1 | 35 | 0.143 |
| timeSig8 | 127 | 0.067 |
| keySharp | 1120 | 0.061 |
| rest16th | 234 | 0.048 |
| timeSig4 | 413 | 0.043 |
| rest8th | 818 | 0.043 |
| timeSig3 | 103 | 0.042 |
| noteheadHalfOnLine | 957 | 0.032 |
| noteheadHalfInSpace | 1045 | 0.022 |
| noteheadBlackOnLine | 13845 | 0.014 |
| slur | 956 | 0 |

field of detecting musical symbols within a score, it is not necessary to think that either precision or recall is important. Therefore, among the three experiments, the experiment B with the highest F1 score is the optimal model for music symbol detection. Fig. 3 (a) is the ground truth bounding box, and Fig. 3 (b) is the model's prediction picture for experiment B.

The Table Ⅲ shows the AP value when the IoU for each music symbol class is 0.5 or higher in the prediction process of experiment B and the frequency of music symbols included in the test data. Only some class data were extracted and arranged in descending order of AP values. The class with the highest AP is the beat signal. In particular, timeSig1, the value of 0.143, the highest AP number, was measured, and the note class, the most important and common symbol in the score, was also measured as 0.01 to 0.03, which was relatively lower than other rare classes. And slur class, it could not accurately detect even one as zero.

## 4. Conclusion

In this study, we proposed a deep learning-based model for automatically recognizing and classifying music symbols within a score image. Through the above model, performers can contribute to the digitalization of the music field by reducing the effort required to convert a score image into an electronic score and enabling automated work in order to play a song.

Currently, our research is facing a challenge with inadequate performance that needs to be addressed. Unlike conventional object detection methods, recognizing individual objects in sheet music poses significant difficulty due to its composition of black music notes and staff lines against a white background. Additionally, sheet music often contains overlapping symbols, complicating accurate recognition. While the performance in detecting music symbols outside the staff lines showed promising results, accuracy significantly dropped for symbols overlapping with the staff lines. Therefore, to address these issues, it is imperative to improve the performance through adjustments such as expanding the network or tuning other hyperparameters, based on the model that achieved the highest F1 score during experiments.

Following the enhancement of performance, our future work is to develop a framework for generating electronic scores by accurately classifying and organizing recognized music symbols using deep learning techniques, and saving them in text file format. Furthermore, we aim to conduct research on automatically converting these scores into the standard musical instrument digital interface (MIDI) format.

## Reference

[1] L. Tuggener, Y.P. Satyawan, A. Pacha, J.Schmidhuber and T.Stadelmann, "The DeepScoresV2 Dataset and Benchmark for Music Object Detection" International Conference on Pattern Recognition, pp. 9198-9195, 2021.

[2] Kornilov, Anton S., I.V. Safonov, "An Overview of Watershed Algorithm Implementations in Open Source Libraries" Journal of Imaging, 2018.

[3] Bai, Min, R. Urtasun, "Deep Watershed Transform for Instance Segmentation" Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5221-5229, 2017.

[4] G. Lin, A. Milan, C. Shen, I. Reid, "RefineNet: Multi-Path Refinement Networks for High-Resolution Semantic Segmentation" Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1925-1934, 2017.

[5] K. He, X. Zhang, "Deep Residual Learning for Image Recognition" Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770-778, 2016

[6] R. Zaheer, H. Shaziya, "A Study of the Optimization Algorithms in Deep Learning" International Conference on Inventive Systems and Control, pp. 536-539, 2019.

[7] M. Everingham, L. Gool, "The pascal visual object classes (voc) challenge" International Journal of Computer Vision, 2014