Proceedings of CONF-MLA 2024 Workshop: Securing the Future: Empowering Cyber Defense with Machine Learning and Deep Learning
DOI: 10.54254/2755-2721/110/2024MELB0054

Image-based Efficient Music Note Detection by Advanced Lightweighted YOLO

Yuhan Shi

Department of Computer Science, University College London, London, WC1E 6BT, United Kingdom

zcabysh@ucl.ac.uk

Abstract. In the realm of music note detection, deep learning models have demonstrated exceptional potential, enabling a wide range of applications, including automated music transcription and real-time music analysis. Despite their effectiveness, the resource-intensive nature of these models often renders them impractical for deployment in resource-limited environments. To tackle this issue, this research focuses on adapting an existing deep learning model, You Only Look Once (YOLO), to explore lightweight alternatives specifically tailored for music note detection. This work assessed the performance of the adapted model using a comprehensive dataset that includes 2,136 meticulously labeled music sheet images. Preliminary results suggest that the streamlined version of the model not only matches the accuracy of its predecessor but also boasts a substantially lower parameter count and reduced Floating Point Operations (FLOPs). These enhancements make the model an ideal candidate for music note detection across a broader spectrum of devices, including those with limited computational capabilities. This advancement opens up new possibilities for real-time music analysis and transcription in various settings, such as mobile devices and low-power embedded systems.

Keywords: Music note detection, Deep learning, YOLO.

1. Introduction

With the development of desktop computers, scorewriters or music notation programs were developed as powerful tools for musicians. These programs offer user interfaces that allow composers to edit, playback, and analyze music sheets in digital format, which improves the efficiency of music creation. Incomplete music sheets are often stored as project files in score writers, which can be exported as image files of the music sheets. However, most existing programs are unable to convert an image back into a project file that can be edited in scorewriters. As a result, composers must manually transcribe each note from the existing music sheet into the scorewriter, a process that can be extremely time-consuming especially for complex compositions. Therefore, a program with the ability to automatically detect and transcribe music notes from sheet music images can enhance the efficiency of music composition.

With the rise of deep learning in recent years, the capabilities of automated music note detection systems have advanced considerably [1]. Deep learning models, particularly convolutional neural networks (CNNs), have dominant power in computer vision tasks [2,3]. This includes object detection, aiming to identify the target objects and mark its location in the image [4]. These models are trained with large amounts of labeled data to learn intricate patterns, enabling them to achieve high accuracy in

DOI: 10.54254/2755-2721/110/2024MELB0054

detecting and classifying musical symbols. The models with such good performance are often complex, requiring substantial memory and processing power to train and deploy. This high demand for computational resources can lead to slow performance and increased energy consumption, making it impractical for applications on mobile devices or other resource-constrained environments.

To address this challenge, it is crucial to explore ways to reduce the complexity of deep learning models. One approach is to develop lightweight models, which require less computational resources without huge loss in their performance [5]. In this paper, lightweight models are developed based on You Only Look Once (YOLO), a popular computer vision model known for its balance between speed and accuracy [6]. The modified model is evaluated based on metrics such as mean Average Precision (mAP), the number of trainable parameters and Floating Point Operations (FLOPs). The goal is to discover a lightweight approach can achieve comparable accuracy to the original YOLO model in the realm of music note detection.

2. Methods

2.1. Preliminaries of YOLO

YOLO is a state-of-the-art object detection model that has gained popularity due to its efficiency and accuracy. Unlike traditional models, which first generate region proposals and then classify them, YOLO makes predictions in a single pass through its neural network, resulting in faster prediction speeds.

The model divides the input image into a grid, with each grid cell predicting a fixed number of bounding boxes and their corresponding confidence scores. A bounding box is defined by four parameters: the “x” and “y” coordinates of the box's center relative to the grid cell, and the width and height of the box relative to the entire image. The confidence score of each bounding box indicates the likelihood that the box contains the target object. This score is calculated as the product of the probability that the box contains any target object, and the probability that the detected object belongs to a specific class. Finally, YOLO compares confidence scores of the bounding boxes to a predefined threshold value, and the boxes with the score higher than the threshold value are selected as the final detections.

To complete the detection process, the model comprises several modules responsible for different tasks: the backbone, neck, and head. The backbone is a deep convolutional neural network responsible for extracting features from the input image. The neck combines these features in a way that helps the model in detecting objects at different scales, using layers that merge or upsample feature maps from different stages of the backbone. The head consists of convolutional layers responsible for making the final detections. It maps the multi-scale features to a fixed number of predictions per grid cell, predicting bounding boxes, confidence scores, and class probabilities. By modifying the layer structures of the backbone, neck, and head, YOLO can be optimized for greater efficiency.

2.2. Lightweighted approach: LeYOLO

LeYOLO is developed based on YOLO model, specifically designed to enhance efficiency and performance of the model in resource-constrained environments [7]. It applies advanced layer structures in the backbone, neck, and head of the original YOLO model to reduce the number of parameters and FLOPs.

The structure of the backbone is modified using a Neural Architecture Search algorithm, which is trained to identify the optimal model structure. The layers that contribute most to predictions was identified, and a new structure was designed to reduce the repetition of other layers. This lowers the number of parameters without significantly compromising the model's accuracy. Additionally, the backbone utilizes an inverted bottleneck structure inspired by MobileNetV2 [8]. In this structure, each block in the network begins by expanding the number of channels, processes the expanded channels, and then reduces the channels back to a smaller number. This structure allows the network to learn richer features without a significant increase in the computational cost.

The neck of the YOLO model is optimized by employing the Fast Pyramidal Architecture Network (FPANet), which draws inspiration from Pyramidal Architecture Network (PANet) and Feature Pyramid

DOI: 10.54254/2755-2721/110/2024MELB0054

Network (FPN) [9,10]. PAN or FPN are frequently utilized in object detection by researchers to allow neural networks to detect objects of varying sizes relative to the image. FPANet was developed with a reduced number of convolution layers and computational flow between certain feature map levels, further reducing the model size.

Finally, the Decoupled Network-in-Network (DNiN) head is designed and implemented in the LeYOLO model. DNiN separates the head into two parts, each responsible for classification and regression (bounding boxes). LeYOLO then executes pointwise convolution operations specifically designed for each part, further improves the accuracy without requiring additional computational cost.

Through applying these modifications, the experiment shows that LeYOLO achieves accuracy on the COCO dataset comparable to the original YOLO [7]. In this paper, YOLOv8n, one of the most advanced and compact versions of the YOLO model, is compared to two sizes of the LeYOLO model: nano and small. Each model is trained using a dataset consisting of labeled music sheets. To assess the models' capabilities, the number of parameters and FLOPs are used as metrics to evaluate the computational cost of the models. The best mAP50, defined as Mean Average Precision at 50% Intersection over Union (IoU), is used as the metric to evaluate the performance of the models. IoU is calculated by dividing the area of intersection between the predicted box and the labeled box by the area of their union, with a value greater than 0.5 considered a successful prediction. The expected results should show that the LeYOLO models have fewer parameters and FLOPs, while their mAP50 values remain close to those of YOLOv8n.

3. Results

The experiments utilize a public dataset comprising 2,136 labeled images of sheet music, which were augmented using basic techniques such as flipping the sheet music horizontally or vertically. In this dataset, 2106 images were allocated for training and the remaining 30 images were evenly split between validation and testing. The labels of each image are stored in a corresponding text file, which records coordinates of the bounding box vertices and the type of music notes within the box. The bounding box vertices indicate the size and location of the box surrounding the music notes. After training each model for 60 epochs using the dataset, their mAP50 on the test dataset is presented in Figure 1.

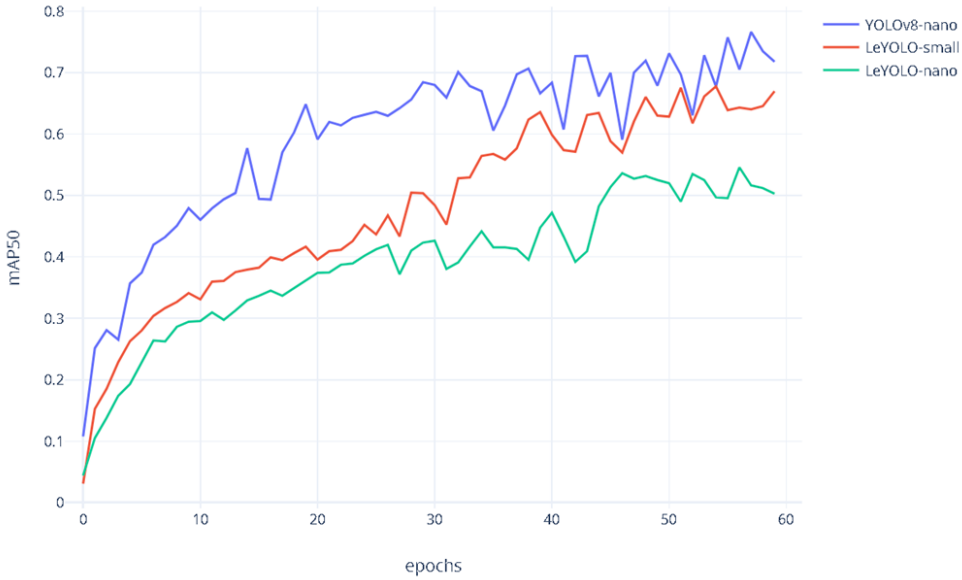

Figure 1. Changes of mAP50 during training (Figure Credits: Original).

The accuracy of LeYOLO-small and YOLOv8n converged after 40 epochs, with YOLOv8n performing slightly better than LeYOLO-small. In contrast, LeYOLO-nano showed a noticeable decline

DOI: 10.54254/2755-2721/110/2024MELB0054

in performance compared to the other two models, and its mAP50 among epochs showed a tendency to flatten. The FLOPs and the number of parameters is recorded from the trained model in Table 1.

Table 1. Performance comparison of different models.

	YOLOv8-nano	LeYOLO-small	LeYOLO-nano
Number of parameters	3,007,013	2,290,901	1,368,997
GFLOPs	8.1	6.2	3.8
Best mAP50	0.73504	0.67787	0.54591

Compared to the original YOLO model (YOLOv8-nano), both LeYOLO models demonstrate significant reductions in parameter count and FLOPs. LeYOLO-nano achieves a 54% reduction in parameters and a 53% reduction in FLOPs, while LeYOLO-small achieves a 24% reduction in parameters and a 23% reduction in FLOPs. These reductions in complexity of the YOLO model are accompanied by a loss in their performance. The LeYOLO-small model experiences a 7.78% decrease in performance, while the LeYOLO-nano model exhibits a more substantial 25.7% decrease, measured using their best mAP50 as the metric.

4. Discussion

In terms of computational resource efficiency, both LeYOLO models are smaller than one of the smallest and most advanced versions of the YOLO model, demonstrating effective optimization. Among these, LeYOLO-nano is the most efficient, with a significant reduction in size and computational demands. However, the 25.7% reduction in mAP significantly impacts the model's performance, leading to reduced reliability for music note detection tasks. In contrast, LeYOLO-small, a version requires more computational resource than LeYOLO-nano, provides more accurate detection of music notes. In the practical application of the model, it is important to balancing performance and computational demands. Despite the small size of LeYOLO-nano, it might not be as suitable for music note detection as LeYOLO-small.

The observed decline in performance for both LeYOLO models can be attributed to the unique challenges associated with the music note detection task. Unlike the COCO dataset, where target objects typically occupy a larger portion of the image and appear at various angles, music notes in the music sheet are numerous, uniformly oriented and occupy small section of the sheet. These differences highlight the effect of task-specific characteristics on the performance of lightweight models.

5. Conclusion

The demand for automated music note detection has increased with advancements in deep learning, especially for tasks like real-time music analysis and automated transcription. However, the high computational requirements of deep learning models limit their use on devices with limited resources. This study focused on finding a lightweight approach specifically optimized for music note detection, based on object detection model YOLO.

An existing model named 'LeYOLO' was developed using several lightweight techniques. An experiment was conducted with two versions of LeYOLO, LeYOLO-small and LeYOLO-nano, with a version of YOLO called YOLOv8n. After training the three models on a public music sheets dataset, their mAP50, number of parameters, and FLOPs were compared to evaluate the success of the lightweight approach. The results show that both LeYOLO models significantly reduced computational complexity compared to the original YOLO. LeYOLO-nano reduced the number of parameters and FLOPs by more than 50%, while LeYOLO-small achieved about a 24% reduction in both metrics. However, both LeYOLO models show a decline in performance, with LeYOLO-nano and LeYOLO-small experiencing decreases of 25.7% and 7.78% respectively.

The findings suggest that the LeYOLO models, especially LeYOLO-nano, are more efficient in terms of resource usage and less accurate in predicting results. LeYOLO-small provides a better balance between performance and efficiency, making it more suitable for music note detection in environments

DOI: 10.54254/2755-2721/110/2024MELB0054

with limited computational resources. The study also highlights the challenge of adapting YOLO to specialized tasks such as music note detection, where the target objects differ significantly from those in typical datasets like COCO.

Future research could focus on enhancing the performance of models by developing techniques specifically tailored to the challenges of music note detection. This may involve designing new architectures that more effectively capture the unique characteristics of musical notation, or simplifying model structures to eliminate components that are less relevant, such as those related to color detection or varied orientations, which are not critical in the context of music sheets.

References

[1] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. nature, 521(7553), 436-444.

[2] Li, Z., Liu, F., Yang, W., Peng, S., & Zhou, J. (2021). A survey of convolutional neural networks: analysis, applications, and prospects. IEEE transactions on neural networks and learning systems, 33(12), 6999-7019.

[3] Zhiqiang, W., & Jun, L. (2017). A review of object detection based on convolutional neural network. In 2017 36th Chinese control conference, 11104-11109.

[4] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... & Fei-Fei, L. (2015). Imagenet large scale visual recognition challenge. International journal of computer vision, 115, 211-252.

[5] Mittal, P. (2024). A comprehensive survey of deep learning-based lightweight object detection models for edge devices. Artificial Intelligence Review, 57(9), 242.

[6] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition, 779-788.

[7] Hollard, L., Mohimont, L., Gaveau, N., & Steffenel, L. A. (2024). LeYOLO, New Scalable and Efficient CNN Architecture for Object Detection. arXiv preprint arXiv:2406.14239.

[8] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE conference on computer vision and pattern recognition, 4510-4520.

[9] Zhao, H., Shi, J., Qi, X., Wang, X., & Jia, J. (2017). Pyramid scene parsing network. In Proceedings of the IEEE conference on computer vision and pattern recognition, 2881-2890.

[10] Lin, T. Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition, 2117-2125.