Modifying the neck of YOLO architectures with input partitioning for improving small object detection

Hamza Görgülü [*1] **Onur Çakı** [*2]

Abstract

The research on object detection keep their importance since it has many real-world applications. When it comes to the detection of small objects, the current methods often face limitations such as lack of focus on small objects, scarcity of image samples with such objects, very small relative size in high-resolution images, limited context information, vanishing gradient, and vanishing spatial information in deep layers. This project aims to solve these problems with a real-time small object detector based on the YOLO algorithm for use on the edge and mobile devices. To that end, we propose modifying the YOLO architectures to improve its performance and accuracy in small object detection. These modifications involve adding an upsampled layer to the neck and head of the architecture and additional residual connections between the backbone and neck. Additionally, we slice the input into smaller parts using a fixed-size window before feeding the network. Furthermore, random cropping is applied to annotations in order to increase the generalization of our model. The modified YOLO architecture is trained and evaluated using the VisDrone dataset, which consists of a diverse collection of images and video clips captured by drone-mounted cameras.

1. Introduction

The goal of object detection is to detect the location of objects along with their class in images and the frames of videos. In recent years, the methods based on convolutional neural networks using deep learning techniques have become mainstream for object detection tasks as the huge developments in the hardware technologies that allow parallel computing and the dramatic rise in the number of digital image data. Especially, the implementation of deep neural networks on Graphics Processing Units (GPUs) and the publication of large-scale publicly available image datasets such as DOTA (Xia et al., 2017), ImageNet (Deng et al., 2009), and COCO (Lin et al., 2014) make object detection with deep learning one of the most popular research areas in computer vision. Thanks to being usable in edge and mobile devices, developing object detection models are vital today for many computer vision applications including robotics (Karaoguz & Jensfelt, 2019), autonomous driving (Gupta et al., 2021), face recognition (Zhang et al., 2018), object tracking from UAV (Zhu et al., 2021b), anomaly and defect detection in industry product (Czimmermann et al., 2020), etc.

To date, various deep learning-based methods have been proposed for object detection. These methods can be categorized into two-stage detectors and one-stage detectors(Zou et al., 2019). Fig. 1 illustrated this categorization. In two-stage detectors (Girshick et al., 2016), region proposals that may belong to an object are first selected using either an algorithm or a convolutional neural network (CNN), and the selected regions are then passed through a CNN. On the other hand, one-stage detectors remove the region proposal stages while approaching the localization problem as a regression task. The location of the objects in images along with their labels is identified after the whole image passes a single network. Since the two-stage detectors are not suitable for real-time inference on the edge and mobile devices, the focus of current research for real-world applications is on boosting the one-stage detectors. The most commonly used one-stage algorithm is YOLO which stands for You Only Look Once (Redmon et al., 2015),(Redmon & Farhadi, 2018), (Bochkovskiy et al., 2020), (Glenn Jocher), (Li et al., 2022), (Wang et al., 2022). Therefore, we dwelled on the tiniest version of the YOLOv5 model called YOLOv5 Nano (Glenn Jocher) in this project as we would like to deploy our proposed model on an edge device.

The detection of small objects is a crucial task in many applications, including security, surveillance, and object tracking. Small object detection is important in a variety of contexts, including security and surveillance, where the ability to accurately detect small objects such as weapons or explosives can help prevent potential harm. It is also

[*]Equal contribution [1]Department of Computer Engineering [2]Department of Computer Engineering. Correspondence to: Onur Çakı <ocaki13@ku.edu.tr>.

important in the context of object tracking, where the ability to accurately detect and track small objects such as wildlife or aircraft can have significant practical and scientific implications. Since these applications require real-time detection, one-stage detectors based on YOLO algorithms are the de facto standard selection because of their speed. However, they have accuracy issues, especially for small objects. Although there are many improvements in one-stage object detection algorithms, the detection of small objects still remains a challenge for some reasons. Firstly, current detection models have been developed for general purposes instead of prioritizing small object detection. Many object detection algorithms are designed to detect a wide range of objects from 80 classes, which may give lower accuracy in the detection of small object classes than other classes. Apart from the accuracy problem, this fact brings along unnecessary calculations and computation resource usage. Secondly, such objects suffer from being represented by a small number of pixels in the image, meaning that the features extracted from these objects are not sufficient to distinguish them from the background. It induces serious problems where the image-object ratio is too big and the image has complicated backgrounds. Moreover, the spatial information that belongs to small objects may vanish in very deep layers due to downsampling through pooling operations. Resizing operation in the input of the network which is an obligation for high-resolution images may also cause vanishing spatial information of small objects. Furthermore, the vanishing gradient during backpropagation may cause the loss of the feature information of small objects. The last problem in small object detection is the under-representation of such object samples in datasets against the other class and background samples. To address these mentioned problems, we propose a lightweight object detection model that can achieve high accuracy with low inference time on edge devices in this project. To that end, we embedded input partitioning in the input layer and modified YOLOv5 architecture by adding additional residual connections and a feature fusion layer.
It is obvious that small object detection tasks represent a significant challenge and has the potential to impact a wide range of applications. By designing a real-time small object detector that is specifically tailored to drone images and is more efficient and accurate than existing methods, we hope to make a meaningful contribution to the field and facilitate the use of small object detection in a variety of contexts.

2. Related Work

Architectures based on the YOLO paradigm consist of three main parts. The first one is the backbone with which the feature maps are extracted through convolutional and pooling layers. The second one is the neck in which extracted feature maps are upsampled or downsampled and combined with each other. The last one is the head which carries out the final predictions on features coming from the neck using predefined anchors. An input image is resized to the input dimension of the model. Then the input image is divided into grids with three different scales. Finally, the model regresses the coordinates of the bounding boxes and predicts the class probabilities and the objectness probabilities. The model carries out this prediction for three anchors separately at each downscale level. Finally, the non-maximum suppression algorithm eliminates redundant detections and makes sure that each object is detected only once. The default YOLOv5 architecture is demonstrated in Fig. 3. As seen in the figure, YOLOv5 uses CSPDarknet as a backbone and Path Aggregation Network(PANet) as a feature fusion mechanism in the neck. The main idea of CSPDarknet is based on cross-stage partition connections which are similar to DenseNet blocks. As shown in Fig. 2, CSPDenseNet divides the input feature maps into two parts. The first one goes through the dense block while the other skips the dense block. Lastly, these parts are concatenated with each other. The information from all layers is fused through element-wise max operation in PANet.

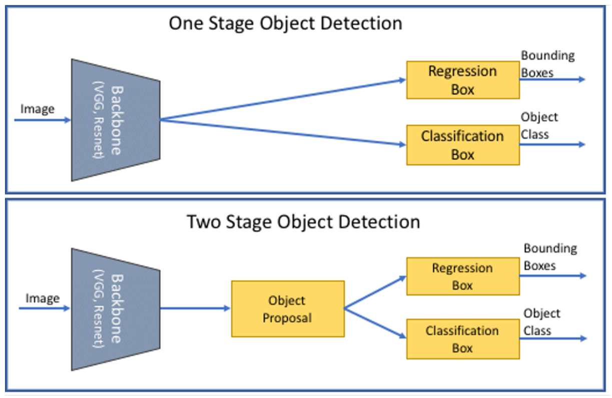


Figure 1. Two stage vs. one stage detectors

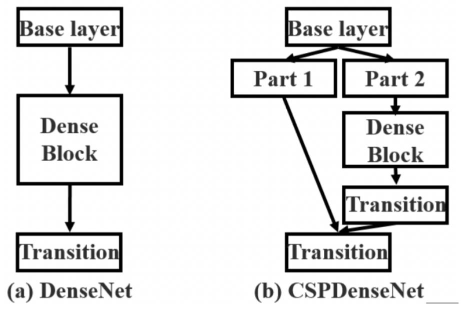


Figure 2. An illustration of DenseNet vs CSPDenseNet

Several recent studies have been proposed to solve the issues associated with small objects. (Kisantal et al., 2019) suggests overcoming the underrepresentation problem by oversampling the images that have small objects using some augmentation designed for such objects. (Akyon et al., 2022) and (Zhang et al., 2020) divide the input images into small parts and inference on each of them separately instead of applying the model to the whole image. By doing so, they resolve the problem of spatial information loss during resizing the input image and the very small relative size in high-resolution images. (Benjumea et al., 2021) used down-scaled densenet as a backbone to enrich context information captured from small objects. Additionally, they modified YOLOv5 architecture such that the resolution of feature maps at the end of the backbone is upsampled one more time and aggregated with other spatial resolution feature maps in the neck. By doing so, they avoid vanishing spatial information of small objects while preserving context information in deeper layers. (Xiao, 2021) has provided exYOLO, a small object detector that has an improved feature fusion algorithm by superimposing the actual network model's 2-fold downsampled feature map. (Liu et al., 2022) presents a study in which a novel feature fusion method based on PANet (Liu et al., 2018) and BiFPN (Tan et al., 2019) is used in the neck for boosting small object detection accuracy. Furthermore, they canceled out the last layer of the backbone whose extracted features mostly belong to larger objects, and they added spatial pyramid pooling(SPP) (He et al., 2015) after the feature fusion network. In (Zhu et al., 2021a), extra residual connections are added between the backbone and the neck to alleviate the vanishing gradient problem. There are also some methods that aim to eliminate the dependencies of the predefined set of anchor boxes by performing prediction on each pixel. Although they achieve good very good results on small objects, they are computationally expensive. (Klasen et al., 2021)

The small object detection improvement methods proposed so far apply their modifications on YOLOv3, and YOLOv5. YOLOv3 is outdated while YOLOv5 does not officially belong to the development process of the YOLO family. Moreover, those bags of freebies methods and those bags of special methods have never been implemented together so far. Lastly, the performance of the modified networks on the edge and mobile devices in other frameworks such as TensorRT, and TVM has not been studied. Thus, we are motivated to address these problems in this study.

3. The Approach

We introduce a slicing mechanism in the input of the network where a sliding window with a certain size extracts overlapped partitions from the input image. The network predicts the locations and classes of objects for each partition. After the network, the inference results on all partitions are merged and mapped to their global locations in the whole image before the non-max suppression. Fig. 4 shows the whole image and 4 overlapped partition instances with 480x480 size. Note that the window size is a new hyperparameter that we tuned in the experiments considering inference time and accuracy. Thanks to this method, we prevent the problem of resizing that causes vanishing small objects. All input partitioning processes are performed in test phase. In the training and validation phase, we create many image samples with this window size by applying random cropping over input images. Note that we oversampled the small objects much more than the relatively larger objects. Thereby, we can alleviate the underrepresentation problem. Fig. 5 presents two different training samples. As you can observe in this figure, random cropping induces an augmentation effect, thereby increasing the generalization of the model.

To increase the accuracy of the small object detections, we introduce a new head and a neck layer. We add one more oversampling layer in the feature fusion pyramids and combine it with the fourth-to-last feature maps in the backbone. Thereby, we aim to resolve the problem of spatial information loss of small objects. Furthermore, we add new residual connections from the backbone to the neck in order to avoid gradient vanishing. Thereby, we propose to reduce the loss of feature information of small objects. The modified architecture is illustrated in 6.

4. Results

4.1. Dataset

We evaluated our proposed method on the VisDrone dataset. The VisDrone dataset (Zhu et al., 2021b) is a collection of data gathered by drones that are equipped with cameras for use in computer vision tasks. The data includes 288 video clips and 10,209 static images, totaling over 261,000 frames, captured by drones in a variety of locations, environments, and conditions. The images contain over 2.6 million annotated bounding boxes around objects of interest such as pedestrians and vehicles, and also include attributes such as scene visibility and object occlusion. We used the object detection task of this dataset which consists of a number of 6471 train, 548 validation, and 1610 test images that are taken by drones.

4.2. Evaluation Metrics

We compare our models using precision and recall. These values are calculated based on the IoU score, which stands for Intersection over Union as shown in Fig. 7. If the IoU score between the ground truth and the predicted box is above a certain threshold(0.2), and their classes are the same, we call it as a True Positive. Unmatched predictions are counted as False Positives while unmatched ground truths

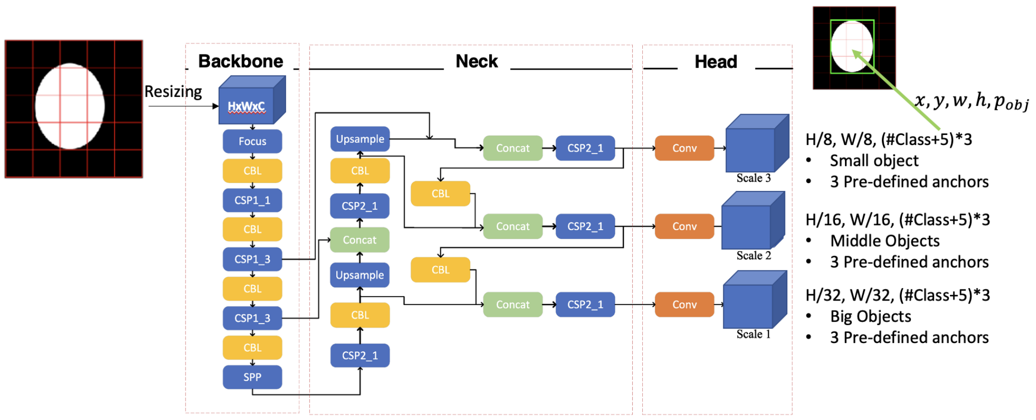


Figure 3. The architecture of the default YOLOv5 model

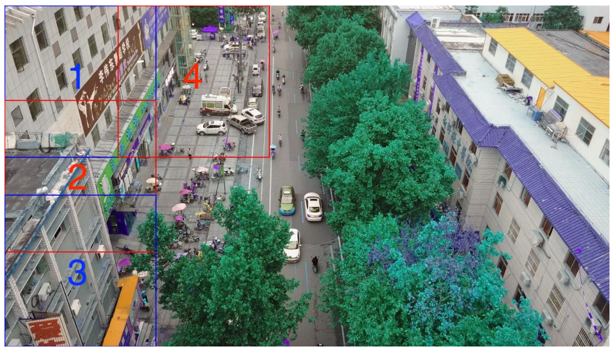


Figure 4. 4 overlapped partition instances with 480x480 size on an input image sample

are counted as False Negatives as demonstrated in Fig. 8.

4.3. Experiments

We used the YOLOv5-nano which is the tiniest version of YOLOv5. In yolov5 nano, the number of layers in the backbone is divided by three and the number of filters in each layer is divided by four. We select the best weights on the validation set after 150 epochs of training. We used a stochastic gradient descent optimizer with a 0.01 learning rate. The network is regularized by applying weight decay with a value of 0.0005 to some weights(57 weights of all). We first trained the default YOLOv5 nano model with various input sizes: 480x480, 640x640(default for YOLOv5 nano), and 1280x1280. We should note that the input size of 1280x1280 may be a problem in edge devices. As you can see in Fig. 9, our proposed methods achieve similar or better performance by only using an input size of 480x480. The orange-colored cells represent relatively smaller objects in the dataset and it can be observed that our model has improved the performance over such objects.

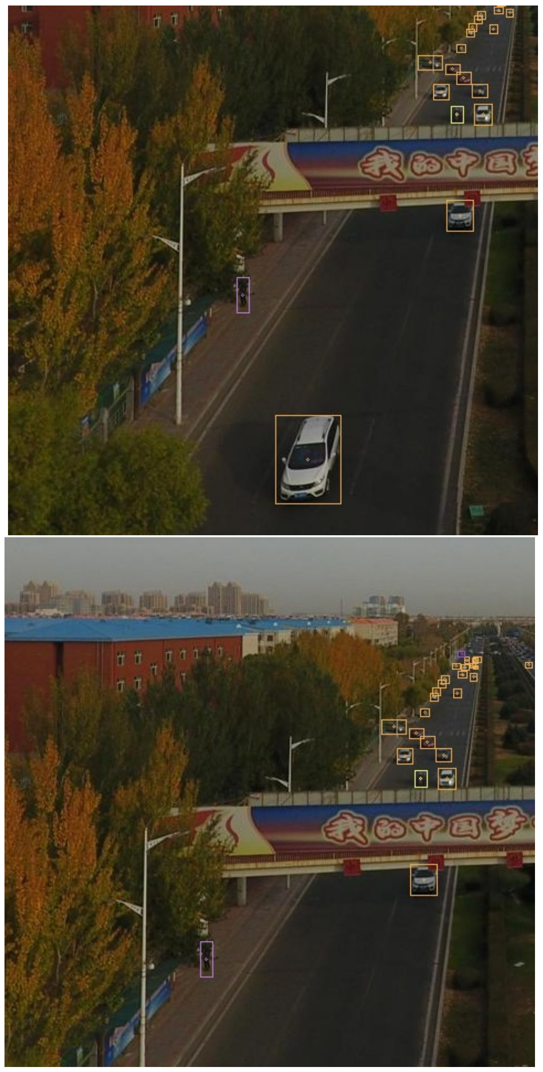


Figure 5. Two different training samples extracted from an input image by random cropping.

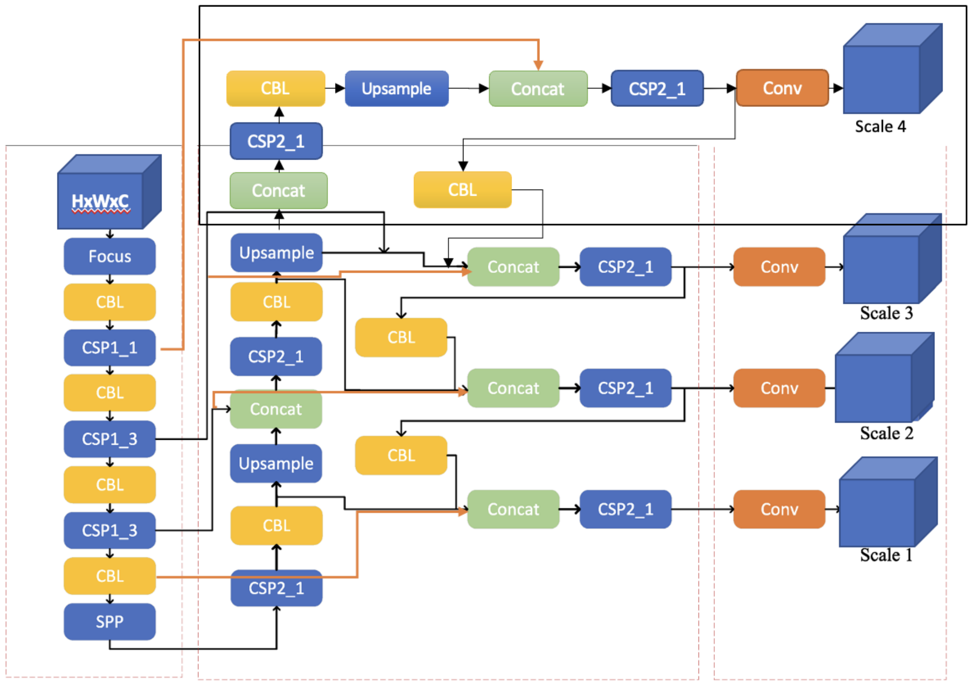


Figure 6. The modified YOLOv5 architecture for small object detection

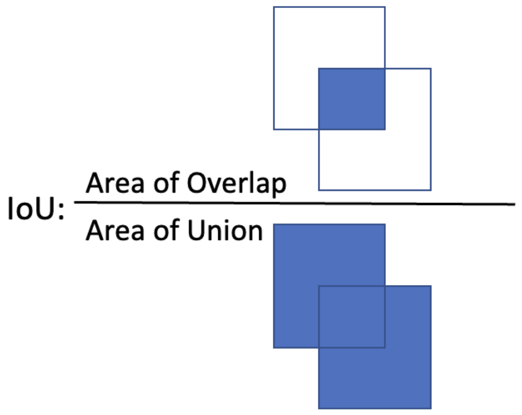


Figure 7. IoU score illustration

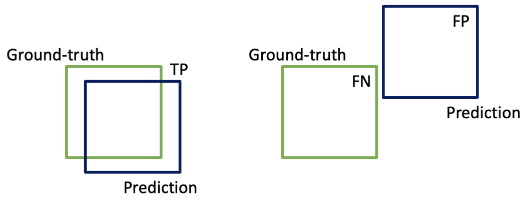


Figure 8. True positive(TP), false positive(FP), and false negative(FN) illustration

Moreover, we filtered out the larger objects that have either a width or height greater than 25 pixels during the inference in order to see the effectiveness of our model on small objects. The results(Fig. 10) show the improvements on the small object in terms of precision, recall, and localization(average IoU) along with the inference times of all models. We measured the inference time on NVIDIA GeForce RTX 3060 Laptop GPU, with a memory of 6144MiB. It is calculated by taking an average of all times measured per image. Note that inference time includes preprocessing, postprocessing, and non-maximum suppression times as well.

Fig. 11 shows the improvement in a visual sense. On the left side of Fig. 11, there are very small car objects and the default YOLOv5 nano model cannot detect them while our proposed model can as seen on the right side of the figure. Moreover, our proposed model also eliminates false positives(FP). As you see in the left figure, the YOLOv5 nano model detects some part of the bridge as a car and we do not observe this mistake in the proposed model.

5. Conclusions

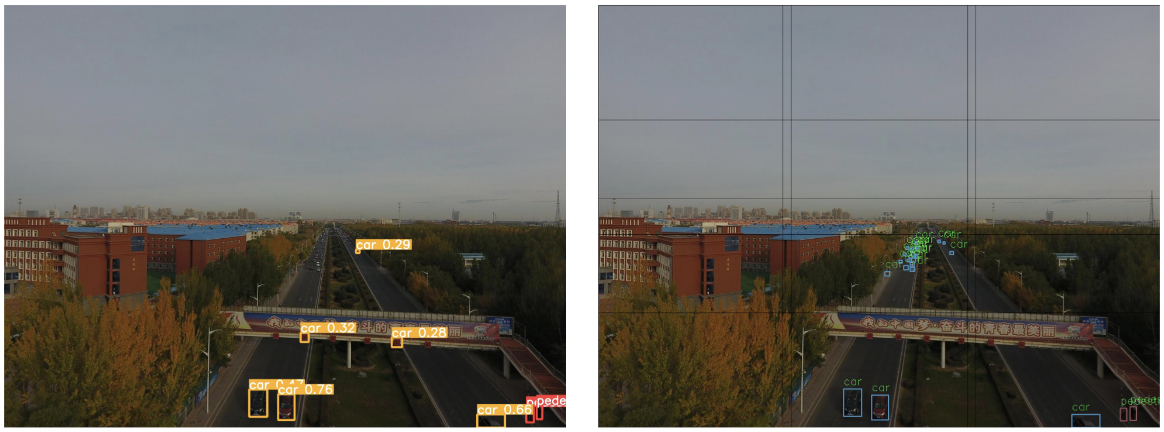
In this study, we propose to improve the small object detection accuracy of one-stage object detection algorithms. We introduced a slicing mechanism in the input of the network and a modified neck in the YOLOv5 architecture to boost the performance. We also applied random cropping and

Model	Input Size	Metrics	Bus	car	van	truck	pedestrian	people	tricycle	awning - tricycle	bicycle	motor
Yolov5 nano	480x480	Recall	0.3424	0.645	0.1192	0.213	0.2138	0.1073	0.0283	0.0067	0.01	0.1184
		Precision	0.7451	0.7501	0.4096	0.4991	0.7968	0.8245	0.3947	0.5	0.3333	0.649
Yolov5 nano	640x640	Recall	0.4161	0.7258	0.1742	0.3009	0.3022	0.1377	0.034	0.01	0.0238	0.1927
		Precision	0.7564	0.7569	0.448	0.5293	0.8091	**0.8484**	0.36	0.5	0.4189	0.6236
Yolov5 nano	1280x1280	Recall	0.5686	0.8302	**0.3843**	**0.5671**	0.5143	0.2041	0.2852	0.192	**0.187**	0.4404
		Precision	0.7723	0.7937	0.5469	0.5796	**0.8125**	0.7906	0.314	**0.506**	0.4583	**0.6661**
Yolov5 nano with partitioning	480x480	Recall	**0.5818**	**0.8462**	0.365	0.5452	0.5675	0.2301	**0.3688**	0.195	0.1563	0.4594
		Precision	**0.7829**	**0.7844**	**0.6207**	**0.6332**	0.7831	0.7811	**0.3668**	0.4776	**0.5**	0.6521
Yolov5 nano with partitioning and modifying	480x480	Recall	0.54	0.8357	0.3569	0.4974	**0.5852**	**0.2347**	0.3315	**0.205**	0.1574	**0.4626**
		Precision	0.7668	0.7926	0.6189	0.6148	0.7721	0.7489	0.3383	0.4862	0.494	0.657

Figure 9. Results using various methods on different classes

Model	Input Size	Precision	Recall	F1 Score	Average IoU	Inference Time
YOLOv5 Nano	480x480	0.1181	0.124	0.121	0.5569	**56.36 ms**
YOLOv5 Nano	640x640(default)	0.1537	0.1945	0.1717	0.51	59.67 ms
YOLOv5 Nano	1280x1280	0.228	0.3896	0.2876	0.6041	74.38 ms
YOLOv5 Nano with input partitioning	480x480	0.2332	0.4211	0.2993	0.5997	93 ms
YOLOv5 Nano with input partitioning and modification	480x480	**0.2473**	**0.4562**	**0.3157**	**0.6123**	99.74 ms

Figure 10. Results using various methods on small objects ($max(width, height) < 25$)



a)Default YOLOv5 nano(640x640)

b)Our proposed method

Figure 11. Inference samples using default YOLOv5 nano(left) and our proposed method(right)

oversampling to increase the generalization of the model and alleviate the underrepresentation problem of small objects. We achieved the following conclusions. Firstly, one can boost the performance of the YOLO model on small objects while sacrificing inference time. There is a trade-off that should be chosen depending on the business problem. Secondly, increasing input resolution rise up both accuracy and inference time. However, it can lead to memory issues on edge devices and input partitioning overperformed the default YOLOv5 Nano with high resolution(1280x1280) input. For future works, the accuracy and inference time can be measured on edge devices as well as after quantization and pruning. We could not do it since we were not able to obtain any edge devices. Lastly, our proposed method brings a new hyperparameter which is partitioning size. Its effect must be examined in more detail from both inference time and accuracy aspects.

References

Akyon, F. C., Altinuc, S. O., and Temizel, A. Slicing aided hyper inference and fine-tuning for small object detection. In *2022 IEEE International Conference on Image Processing (ICIP)*. IEEE, oct 2022. doi: 10.1109/icip46576.2022.9897990. URL https://doi.org/10.1109%2Ficip46576.2022.9897990.

Benjumea, A., Teeti, I., Cuzzolin, F., and Bradley, A. Yolo-z: Improving small object detection in yolov5 for autonomous vehicles, 2021. URL https://arxiv.org/abs/2112.11798.

Bochkovskiy, A., Wang, C.-Y., and Liao, H.-Y. M. Yolov4: Optimal speed and accuracy of object detection, 2020. URL https://arxiv.org/abs/2004.10934.

Czimmermann, T., Ciuti, G., Milazzo, M., Chiurazzi, M., Roccella, S., Oddo, C. M., and Dario, P. Visual-based defect detection and classification approaches for industrial applications—a survey. *Sensors*, 20(5):1459, Mar 2020. ISSN 1424-8220. doi: 10.3390/s20051459. URL http://dx.doi.org/10.3390/s20051459.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009. doi: 10.1109/CVPR.2009.5206848.

Girshick, R., Donahue, J., Darrell, T., and Malik, J. Region-based convolutional networks for accurate object detection and segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(1):142–158, 2016. doi: 10.1109/TPAMI.2015.2437384.

Glenn Jocher, Ayush Chaurasia, A. S. J. B. Yolov5 sota realtime instance segmentations. URL https://zenodo.org/record/7347926#.Y9PINuxBzFo.

Gupta, A., Anpalagan, A., Guan, L., and Khwaja, A. S. Deep learning for object detection and scene perception in self-driving cars: Survey, challenges, and open issues. *Array*, 10:100057, 2021. ISSN 2590-0056. doi: https://doi.org/10.1016/j.array.2021.100057. URL https://www.sciencedirect.com/science/article/pii/S2590005621000059.

He, K., Zhang, X., Ren, S., and Sun, J. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1904–1916, 2015. doi: 10.1109/TPAMI.2015.2389824.

Karaoguz, H. and Jensfelt, P. Object detection approach for robot grasp detection. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 4953–4959, 2019. doi: 10.1109/ICRA.2019.8793751.

Kisantal, M., Wojna, Z., Murawski, J., Naruniec, J., and Cho, K. Augmentation for small object detection, 2019. URL https://arxiv.org/abs/1902.07296.

Klasen, M., Ahrens, D., Eberle, J., and Steinhage, V. Image-based automated species identification: Can virtual data augmentation overcome problems of insufficient sampling? *Systematic Biology*, 71, 06 2021. doi: 10.1093/sysbio/syab048.

Li, C., Li, L., Jiang, H., Weng, K., Geng, Y., Li, L., Ke, Z., Li, Q., Cheng, M., Nie, W., Li, Y., Zhang, B., Liang, Y., Zhou, L., Xu, X., Chu, X., Wei, X., and Wei, X. Yolov6: A single-stage object detection framework for industrial applications, 2022. URL https://arxiv.org/abs/2209.02976.

Lin, T., Maire, M., Belongie, S. J., Bourdev, L. D., Girshick, R. B., Hays, J., Perona, P., Ramanan, D., Doll'a r, P., and Zitnick, C. L. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. URL http://arxiv.org/abs/1405.0312.

Liu, H., Sun, F., Gu, J., and Deng, L. Sf-yolov5: A lightweight small object detection algorithm based on improved feature fusion mode. *Sensors*, 22(15), 2022. ISSN 1424-8220. doi: 10.3390/s22155817. URL https://www.mdpi.com/1424-8220/22/15/5817.

Liu, S., Qi, L., Qin, H., Shi, J., and Jia, J. Path aggregation network for instance segmentation, 2018. URL https://arxiv.org/abs/1803.01534.

Redmon, J. and Farhadi, A. Yolov3: An incremental improvement, 2018. URL https://arxiv.org/abs/1804.02767.

Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. You only look once: Unified, real-time object detection, 2015. URL https://arxiv.org/abs/1506.02640.

Tan, M., Pang, R., and Le, Q. V. Efficientdet: Scalable and efficient object detection. 2019. doi: 10.48550/ARXIV.1911.09070. URL https://arxiv.org/abs/1911.09070.

Wang, C.-Y., Bochkovskiy, A., and Liao, H.-Y. M. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *arXiv preprint arXiv:2207.02696*, 2022.

Xia, G.-S., Bai, X., Ding, J., Zhu, Z., Belongie, S., Luo, J., Datcu, M., Pelillo, M., and Zhang, L. Dota: A large-scale dataset for object detection in aerial images. 2017. doi: 10.48550/ARXIV.1711.10398. URL https://arxiv.org/abs/1711.10398.

Xiao, J. exyolo: A small object detector based on yolov3 object detector. *Procedia Computer Science*, 188:18–25, 2021. ISSN 1877-0509. doi: https://doi.org/10.1016/j.procs.2021.05.048. URL https://www.sciencedirect.com/science/article/pii/S1877050921011339. CQVIP Conference on Data Driven Intelligence and Innovation.

Zhang, T., Zhang, X., Yang, Y., Wang, Z., and Wang, G. Efficient golf ball detection and tracking based on convolutional neural networks and kalman filter, 2020. URL https://arxiv.org/abs/2012.09393.

Zhang, Z., Shen, W., Qiao, S., Wang, Y., Wang, B., and Yuille, A. Robust face detection via learning small faces on hard images, 2018. URL https://arxiv.org/abs/1811.11662.

Zhu, L., Geng, X., Li, Z., and Liu, C. Improving yolov5 with attention mechanism for detecting boulders from planetary images. *Remote Sensing*, 13(18), 2021a. ISSN 2072-4292. doi: 10.3390/rs13183776. URL https://www.mdpi.com/2072-4292/13/18/3776.

Zhu, P., Wen, L., Du, D., Bian, X., Fan, H., Hu, Q., and Ling, H. Detection and tracking meet drones challenge. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2021b. doi: 10.1109/TPAMI.2021.3119563.

Zou, Z., Shi, Z., Guo, Y., and Ye, J. Object detection in 20 years: A survey, 2019. URL https://arxiv.org/abs/1905.05055.