Azerbaijan Journal of High Performance Computing, Vol 4, Issue 1, 2021, pp. 15-28
https://doi.org/10.32010/26166127.2021.4.1.15.28

Object Recognition for Augmented Reality Applications

Vladislav Li[1], Georgios Amponis[2], Jean-Christophe Nebel[1], Vasileios Argyriou[1], Thomas Lagkas[2] and Panagiotis Sarigiannidis[3]

[1] *Department of Networks and Digital Media, Kingston University, London, UK*
[2] *Department of Computer Science, International Hellenic University, Greece*
[3] *Department of Electrical and Computer Engineering, University of Western Macedonia, Kozani, Greece, psarigiannidis@uowm.gr*

*Correspondence: Panagiotis Sarigiannidis, Department of Electrical and Computer Engineering, University of Western Macedonia, Kozani 50131, Greece, psarigiannidis@uowm.gr

Abstract

Developments in the field of neural networks, deep learning, and increases in computing systems' capacity have allowed for a significant performance boost in scene semantic information extraction algorithms and their respective mechanisms. The work presented in this paper investigates the performance of various object classification- recognition frameworks and proposes a novel framework, which incorporates Super-Resolution as a preprocessing method, along with YOLO/Retina as the deep neural network component. The resulting scene analysis framework was fine-tuned and benchmarked using the COCO dataset, with the results being encouraging. The presented framework can potentially be utilized, not only in still image recognition scenarios but also in video processing.

Keyword: Object Recognition, Scene Analysis, Super Resolution, Machine Learning, High-Performance Computing, Feature Extraction.

1. Introduction

Modern improvements and advances in the fields of computer vision, and deep learning, along with the introduction of smart devices and powerful computing equipment, have stimulated research interest in real-time scene analysis. Scene analysis describes the contents of an environment given as an input by using information computationally extracted from that environment. The field of surveillance, security, and the increasingly popular autonomous vehicles all use scene analysis as a key component for providing their respective services. Thus, all those disciplines can benefit from a performance increase to allow their more extensive deployment and adoption as the industry standard. This study aims to research scene analysis methods that could be widely adopted in the modern high-performance computing landscape for the modem industry and propose a novel preprocessing mechanism for scene analysis using super-resolution.

High performance computing strongly relates to image processing and object recognition mechanisms due to said method's architectures and training procedures significantly benefiting from low processing times. More specifically, GPU-based

high-performance computing allows for deep learning networks to be trained, tested, and evolved en masse. High-performance computing allows for the most prominent challenges and issues with deep learning to be effectively tackled. The impact and applications of GPU-enabled high-performance computing in deep learning are numerous, with the most prominent ones being the potential to solve the issues of (a) training models with complex topologies, (b) configuring a network to an optimal topology, (c) implementing deep learning models by utilizing simulated hardware-layer synapse architectures, and (d) constructing an efficient network from non-examined data examined.

Apart from proposing a novel preprocessing mechanism, the presented work constitutes a review of the currently utilized scene recognition methodologies. More specifically, this paper engages in a comparative study of object classification and recognition solutions and models for high-performance computing systems and a review of the current state-of-the-art approaches for scene analysis in terms of computational requirements, speed, and accuracy.

The rest of this paper is structured in four main sections. The following section, "Related work" analyzes the previously researched domain of image recognition using various models. Accordingly, the following section, "Proposed Framework for Scene Analysis," is dedicated to explaining this paper's proposal of a novel scene analysis framework and is composed of two subsections: one concerning preprocessing with super-resolution and one respectively concerning scene analysis using deep models. Next, the section titled "Performance evaluation" constitutes an assessment of various image recognition models, for which the COCO dataset was utilized. Lastly, the final section concludes the paper while assessing its contribution to the domain of high-performance computing enabled image processing.

2. Related work

This section analyzes existing related work and research conducted by other experts in the field. In short, this section analyzes the R-CNN model, the Mask R-CNN model (which is based on R-CNN), the Cascade R-CNN model, and the Optimized SSD, CenterNet, ONCE, MAL, D2Det, and Faster R-CNN models alike. Background information for each model is provided, along with a short description of their basic functionality and improvements compared to their predecessors.

Girshick et al. (2014) introduced the R-CNN model, which constitutes a region proposal-based framework, relying on a two-step process. In said region proposal-based frameworks, the input image's region, which may contain recognizable objects, is proposed. Continuing, the discernable features of said objects are extracted from the proposed regions. To extract the class of proposed objects, the model resorts to the usage of classifiers.

He et al. (2017) proposed the Mask R-CNN method, which relies on the R-CNN principle. The introduced method is capable of discerning bounding boxes and parallel segmentation masks. The process of generation of the candidate regions is followed by an attempt to discern the object's class while refining the bounding box containing the possibly discerned object and calculating the object's mask.

Cai and Vasconcelos (2019) attempt to address the issue of performance degrada-

tion upon the increase of the intersection-over-union threshold observed in the previously mentioned Mask R-CNN by resorting to the new object recognition architecture

Cascade R-CNN. Cascade R-CNN's central idea is to use different detection head networks at each stage, where each one of them is designed for one intersection-over-union threshold. All detectors comprising the architecture are trained successively, and each one's output is handed over to the next one. Figure 1 visualizes Cascade R-CNN's architecture, with the input being denoted as "I", the backbone network as "conv", the region extraction layer as "pool", the detection area of interest as, and lastly, the bounding boxes and classification scores as "B" and "C" respectively.

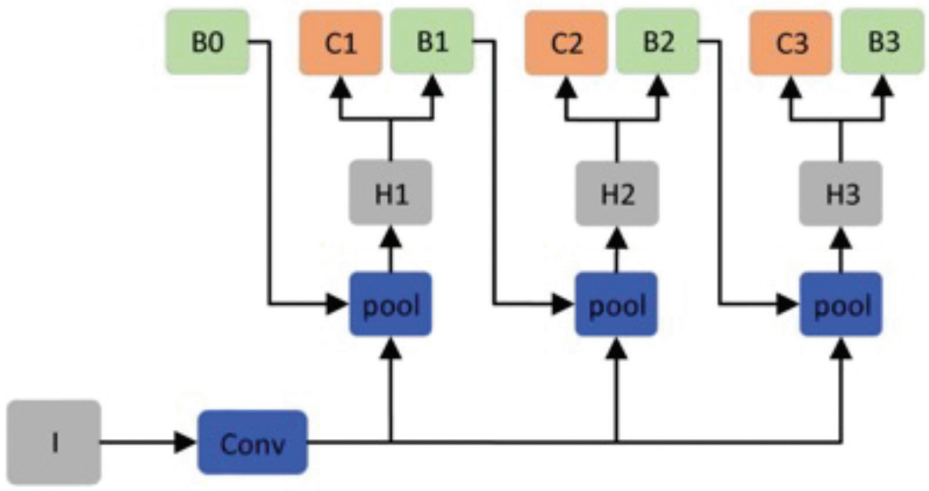

Fig.1. The architecture of Cascade R-CNN.

One-step object detectors directly predict objects' bounding boxes and class labels alike by feeding a full image to a convolutional network that does not require any region proposals. Because those frameworks' pipeline is a single network, we can observe a faster solution than the one provided by two-stage detectors, where their first step is instead time-consuming. We can thus conclude that those one-step frameworks are particularly suitable for real-time applications.

In the context of increasing detectors' efficiency, Kong et al. (2019) proposed the Optimized SSD, which has the potential of significantly increasing performance. The main idea of Optimized SSD is the utilization of the refined anchors during training for both classification and regression. This is implemented by extending the single-stage detector's optimization method. Figure 2 illustrates the network architecture used in single-shot detectors and the one proposed in (Kong et al., 2019), with the input image being denoted as "I", the backbone network as conv, the convolutional network head

as "H", the classification score as "C" and the original and refined anchor boxes as AO and A1, respectively.

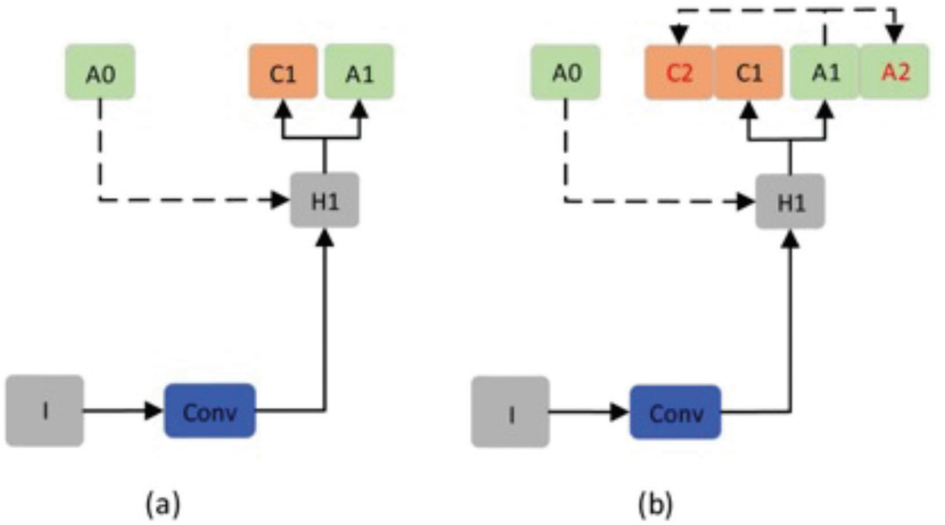

Fig. 2. (a) Network architecture of single-shot object detectors, (b) Network architecture of Optimized SSD (Kong et 2019)

Anchor-free techniques are the current trend in object detection methods. The main idea behind anchor-free object detectors is to compute the bounding box corners around an object instead of predicting fixed bounding boxes.

Researchers in (Duan et al., 2019) proposed CenterNet, which essentially is a state-of-the-art Lidar-based 3D detection and tracking framework. It combines the usage of center and corner points alike. The backbone network receives an image as an input. By applying cascade, comer, and center pooling, CenterNet extracts the corner and the center heat-maps. This framework also supports the extraction of embedding vectors and offsets for the corner points by utilizing the architectural elements of CornerNet (Duan et al., 2019). Furthermore, CenterNet predicts the offsets of the center keypoints; detected corners and embeddings are used to compute and spatially define the bounding boxes of the recognized objects. The detected center points are used to increase the accuracy of the final bounding boxes' coordinates, effectively refining the framework's output.

Researchers in (Perez-Rua et al., 2020) proposed OpeN-ended Centre nEt (ONCE), which is based on CenterNet (Duan et al., 2019), with the added functionality that

can detect objects from classes with a small number of examples inside its training dataset. The ONCE network is based on Incremental Few-Shot Detection learning, where a predictive model can be trained on large datasets and be deployed afterward in real-world scenarios with only a small number of samples inside its dataset. At the same time, the training procedure, which comprises two stages, receives a test image from the training dataset as an input and extracts the 3D feature maps.

In addition to the above, Ke et al. (2020) introduced Multiple Anchor Learning (MAL). The researchers based their work on RetinaNet (Lin et al., 2017). MAL was created to address the inherent limitations of CNN-based detectors regarding the jointly optimized classification and localization of a detected object. This limitation stems from the classifi-

cation and localization modules in detectors being optimized under a static set of possible bounding boxes. The researchers propose a module that selects anchors and jointly optimizes both the classification and localization modules of a CNN-based detector by finding an optimal anchor selection. The main idea of MAL is that an "anchor bag" is created for each existing object detected on an input image. Said anchor bags include the top k anchors (depending on the intersection over union threshold between the anchors which MAL predicted and the bounding box). At the next stage, MAL evaluates said anchors; this evaluation is implemented using the confidence of their joint classification and localization. Lastly, MAL selects the most representative anchor from each respective bag.

Finally, Cao et al. (2020) proposed a two-step object detection method called D2Det based on the Faster R-CNN framework (Ren et al., 2015). The Region of Interest (RoI) features which the Region Proposal network (RPN) network generates for each possible object proposal are passed and processed through two different modules: a) high-density local regression and b) discriminant RoI pooling. In particular, a dense local regression block replaces the FasterR-CNN offset regression.

Faster R-CNN extracts RoI features which are traversed through numerous fully connected layers to predict a single global offset. In turn, dense local regression predicts multiple position-sensitive local offsets by utilizing a convolutional network. Faster R- CNN's strategy introduces a measurable increase in accuracy in terms of box localization. Furthermore, when it comes to the classification task, the discriminative RoI pooling block first uses a lightweight offset prediction for each RoI and then performs adaptive weight pooling. This process is followed to assign higher weights to RoI's discriminative sampling points.

3. Proposed Framework for Scene Analysis

In this work, we propose a framework for scene analysis combining super-resolution with object detection architectures. An overview of the proposed framework is shown in Figure 3, and it contains two major components: (a) the super-resolution module as part of the preprocessing component and (b) the object detection and classification models.

This section constitutes the presented work's main contribution to the field; with the proposed framework being composed of a Pre-processing and a Deep Neural Network module, a respective subsection analyzes both components and explains their relevance to the extraction of semantic information from a given input. A comparative study and thorough investigation of different deep neural network models for high-performance computers were conducted, with the results being presented in the respective section. The significance of this paper's outputs stems from the results indicating which model (and thus overall framework architecture) offers lower processing times and higher overall efficiency.

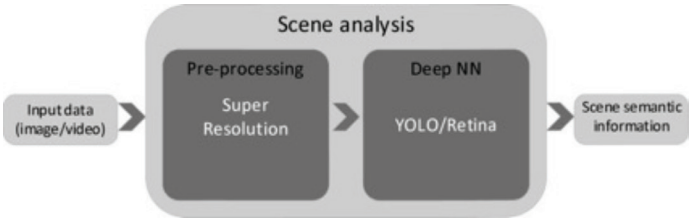

Fig. 3. The architecture of proposed scene analysis framework

3.1. Preprocessing with super-resolution

Super-resolution can be defined as creating high-resolution (HR) images by using corresponding low resolution (LR) images. High-resolution images provide high-quality scene reconstruction that can be useful for many real-world high-performance computing applications. Such applications include but are not limited to satellite and medical imaging (Huang et al., 2017), multimedia content, face recognition (Mudunuri and Biswas, 2015), and security.

A substantial amount of research is centered around obtaining Convolutional Neural Networks (CNNs) architectures and different loss functions (which are used to boost said networks' performance). Researchers have introduced Generative Adversarial Networks (GANs) to tackle the super-resolution (SR) problem. In most strategies, the loss function utilized to train deep neural networks is the Mean Square Error (MSE) between the reconstructed and the actual image. Nevertheless, usage of this metric usually makes the generated high-resolution pictures seem oversmoothed and somewhat blurred. Although those methods have superb results using the peak signal-to-noise ratio (PSNR), they are unable to create "good" (i.e., realistic, not over-smoothed) looking images. An answer to this issue is using perceptual loss as a loss function (Johnson et al., 2016). Specifically, this loss function is computed on high-level options extracted from pre-trained convolutional neural networks, like VGG (Simonyan and Zisserman, 2014).

Goodfellow et al. (2014) introduced Generative Adversarial Networks (GANs) to tackle the image quality issue mentioned above. As a general rule, such adversarial networks are heavily influenced by game theory. Two distinct networks (generator and discriminator) are conjointly trained, with each one's purpose is to manage to trick the other party. What happens in more detail regarding SR is that the generator receives the LR image as an input and then tries to generate a super-resolution (SR) image (as seen in Figure 4 The discriminator module (D) receives the real and the generated HR images as input, alike. The discriminator then tries to distinguish between them and deduct which is which. The discriminator's output is fed to the generator as an input to enable the future generation of high-fidelity and realistic high-resolution images; conjointly training those two networks (the generator and the discriminator) lets the generator produce an output high quality and indistinguishable from the real HR images.

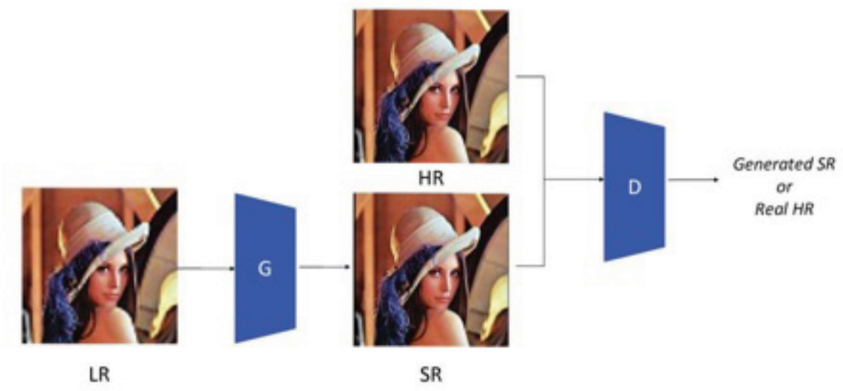

Fig. 4. Function of GANs

Ledig et al. (2017) proposed a GAN-based SR framework named "SRGAN". How SRGAN functions on a high level are: the generator receives the LR image as input and outputs an SR image. The generator also contains residual blocks with skip connections instead of convolutional layers. Such connections have proven to be more suitable in training deep networks. In turn, the discriminator receives both the ground truth and the generated HR images as input and then tries to distinguish between them. Usage of the discriminator network and the adversarial nature of the "fight" between generator and discriminator encourages the latter to produce high fidelity results, which translates to images that are more likely to trick the discriminator.

3.2. Scene analysis using deep models

This section is dedicated to analyzing the YOLOv3 and the RetinaNet machine learning (ML) methods; more specifically, this section aims to provide sufficient background information and details regarding said methods' operation. Those two ML methods were selected because they have displayed excellent performance on different datasets in the existing literature. Additionally, YOLOv3 and RetinaNet fit representatives for different object detectors, e.g., two-step detectors or pyramid architectures. Regression/Classification-Based Frameworks are composed of several integrated stages. Those stages may be:

a) region proposal generation
b) CNN-based feature extraction
c) classification
d) bounding box regression

The stages mentioned above are generally trained separately, and as a consequence, they are inherently computationally demanding, which renders them non-suitable for real-time applications. Nevertheless, some strategies (such as mapping pixels to bounding box coordinates and object class probabilities) manage to reduce time complexity. Amidst the various one-step detection architectures, there exist two which distinguish themselves; namely, only look once (YOLO) (Redmon et al., 2016) (with the name indicating that this is a single-step method), and Single Shot MultiBox Detector (SSD), (Liu et al., 2016).

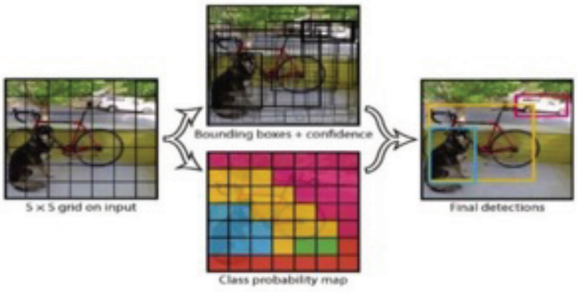

Fig. 5. The YOLO concept

The YOLO framework utilizes the whole feature map to predict both confidences for multiple categories and bounding boxes. YOLO's basic function is illustrated in Figure

5. The YOLO concept. The input image is initially divided into an S x S grid. Each grid cell is tasked with predicting the object-centered inside of it. Continuing, each cell predicts B bounding boxes, along with their corresponding confidence scores.

3.2.1. YOLOv3

In contrast to its predecessors (YOLO9000 and YOLO), YOLOv3 is more complex due to skip-connections, residual blocks, and upsampling alike being integrated. Error! Reference source not found illustrates YOLOv3's architecture on a high level. After being processed by the first set of approximately 80 convolutional layers (which is represented by the first "Convolutional layers" block), the input image is downsampled by a factor of 32. For example, an input image of 1216x1216 after the mentioned downsampling stage would be represented by a 38x38 feature map. This downsampled 38x38 feature map is at that stage converted into a prediction map

through a lxl convolutional layer. Inside the prediction map obtained at this stage, each cell is responsible for detecting a fixed number of bounding boxes. Since three anchors per prediction map are used, the number of predicted bounding boxes at this stage will be equal to 38x38x3=4332. Each bounding box can be represented by 4+1+C (where C is the number of classes), which stands for the 4 box's angular coordinates, 1 objectness score, and C class scores. Regarding the objectness and the class scores,

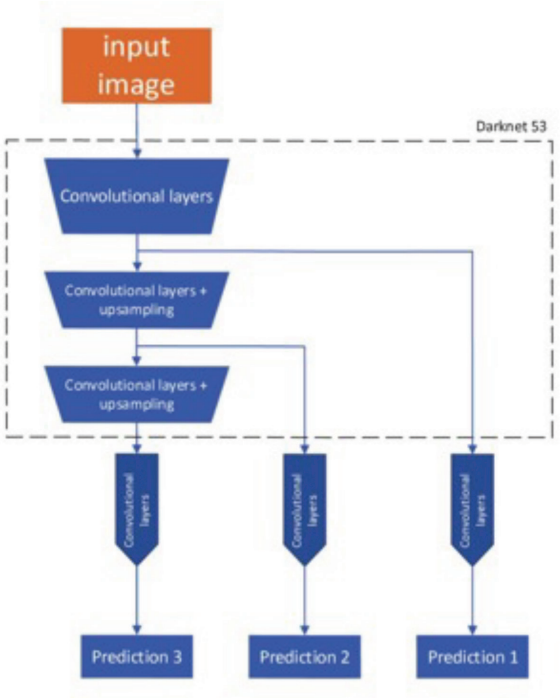

Fig. 6. Abstract representation of the fulYOLOv3 architecture. Predictions are made at three different scales.

their values were extracted using softmax functions in the previous versions (YOLO9000 and YOLO), while their values are obtained via a sigmoid function in the current version (YOLOv3). Additional convolutional layers are included, at which stages the input is also upsampled by a factor of two. As a result, at the second stage, the prediction map's size will grow into 76x76, thus resulting in additional 76x76x3 = 17328 bounding box predictions. The upsampling process is again repeated - now having a map size of 152x152, for the third round of predictions, the resulting number of bounding boxes now being 152x152x3 = 69312. Given the size of this number and the difficulty of working with it, the predictions undergo filtering; said filtering is done using the objectness score as a metric; additionally, non-max-suppression is applied to discard duplicate predictions.

3.2.2. RetinaNet

RetinaNet is comprised of three main elements that work together to make bounding box and class predictions. Figure 7 shows the architecture as it is presented by Lin et al. (2018). At the left of the figure shown below, one can see the mechanism's backbone: a feed-forward convolutional network.

Note that even though only three distinct stages are shown, the network consists of a significantly greater number of layers (ResNet 50 has 50 convolutional layers). As the "Stage 1", "Stage 2", and "Stage 3" modules indicate, the input's features are extracted thrice (one for every key point) and are used to build up the "feature pyramid" net, as seen in the second module of the figure below, resembling a pyramid comprised of five levels. The first three levels in the feature pyramid are directly obtained from the backbone features through a lxl convolutional layer and a combination with an upsampled

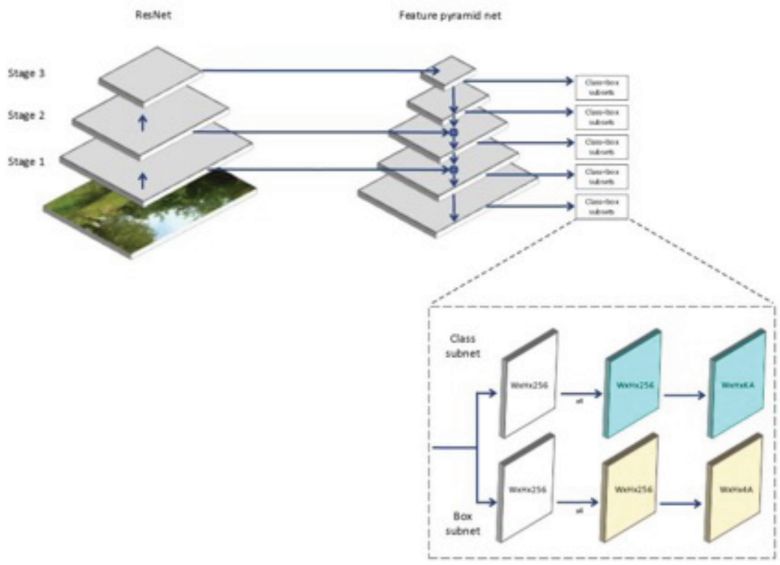

Fig. 7. Abstract representation of the RetinaNet architecture. Note that through the feature pyramid network, features from all scales are shared.

version of the directly above layer. The last two pyramid levels are obtained from the last backbone features through 3x3 convolutions with stride 2. At each feature pyramid's level, a subnet that classes and bounding boxes (as illustrated in the figure). Each class and box subset is comprised of two respective subnets. Lastly, a 9-anchor set is defined for every pyramid network level, covering a range between 16 and 406 pixels concerning the input image.

4. Performance evaluation

This section is dedicated to evaluating various methods that the proposed framework can potentially embed either RetinaNet or YOLOv3 to facilitate object recognition and scene analysis. The evaluation considers the mean average precision and processing time required for each method. As mentioned below, the COCO dataset was used to evaluate the performance of the models.

The COCO (Common Objects in Context) dataset is the most commonly utilized benchmarking tool for evaluating the performance of computer vision models. COCO is designed to represent a sizeable array of common objects with which we commonly

Azerbaijan Journal of High-Performance Computing

interact. The dataset is labelled, providing data to train supervised computer vision models. The model's goal is to identify the common objects in the given dataset. Based on the model's outputs, COCO evaluates the possible improvement of these models. The COCO dataset contains 121408 images, 80 classes, and 883331 object annotations.

Table 1 constitutes a presentation and examination between advanced calculations. The most well-known assessment measurements are the Average Precision (AP) and Mean Average Precision (mAP) to assess object identification techniques. MAL and D2Det (Cao et al., 2020) both achieve significant performance improvements compared to the other methods. They demonstrably outperform previous state-of-the-art detection algorithms.

TABLE 1 Performance comparison (AP %) with the state-of-the-art methods on the MS-COCO test-dev dataset (Lin et al., 2014).

Method	Backbone	AP (%)
Mask R-CNN (He et al., 2017)	ResNetXt-101	39.8
Cascade R-CNN (Cai and Yasconcelos, 2019)	ResNet-101	42.8
D2Det (Cao et al., 2020)	ResNetl 01 -deform v2	47.4
CornerNet (Law and Deng, 2018)	Hourglass-104	40.5
ExtremeNet (Zhou et al., 2019)	Hourglass-104	40.2
CenterNet (Duan et al., 2019)	Hourglass-104	44.9
MAL (Ke et al., 2020)	ResNeXt-101	47.0

4.1. The Mean Average Precision

Mean average precision was first introduced by Lin et al. (2014) to represent object detection performance according to a user-defined set of criteria. This metric is now widely applied, e.g. (Zhou et al., 2019). The metric mAP can be defined as the mean value of the average precision of the individual classes:

$$mAP = \frac{1}{n}\sum_{k=1}^{k=n} AP_k$$

where APk is the AP of class k and n is the number of classes

A key element to understanding the average precision is the precision-recall graph. More specifically, for the average precision values to be computed, all the predictions for a specific class for all the input images are collected and sorted by considering their respective confidence levels. At that point, the average precision can be defined as an approximation of the area under the precision-recall curve, where first, the "envelope" of the precision-recall curve is computed. In addition to averaging over all the relevant classes for calculating the metric mAP, researchers in Lin et al. (2018) suggested that the metric mAP can be averaged over different IoU thresholds alike. Averaging over the different thresholds can offer information into how accurately the predicted boxes match the ground truth annotation; however, this depends on the quality of the ground truth annotation itself.

4.2. Processing Time

The average inference time per image is used to calculate processing time. As a result, it excludes the time spent loading the architecture and/or weights. To arrive at an expected value, the processing time is averaged through the entire batch of test data and then divided by the number of images evaluated. An averaging effect can be achieved with this method for different numbers of instances per image and different input sizes. Both the absolute processing time per frame (in milliseconds) and the average number of frames per second are commonly used units to express this KPI in the literature. The results of the experiments are displayed below in Figure 8.

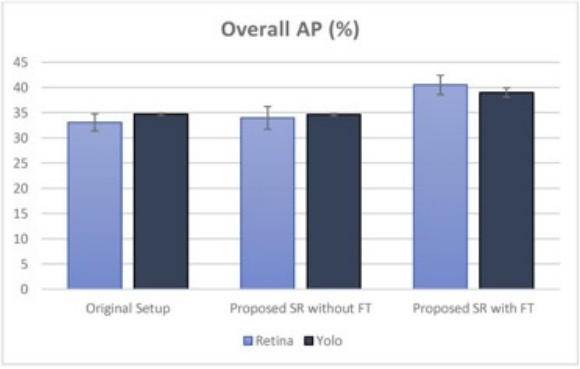

Fig. 8. Models' performance evaluation and comparison

TABLE 2: Average Time required per image over the two models and the obtained AP% for each model.

Method	Retina AP% (std)	Yolo AP% (std)	Average Time (sec per image)
Original Setup	33.0925 (1.7023)	34.6983 (0.2153)	1.591
Proposed SR without FT	33.9675 (2.2398)	34.6164 (0.2019)	2.1242
Proposed SR with FT	40.5125 (1.9109)	38.9844 (0.9117)	2.2102

The utilized processing time as seen in
Table 2 was computed based on the time required to process 128 images and then divided by the number of images to get an average time needed to process a single image since measuring the time required to process a single image by itself is non-realistic. The utilized hardware had the following specifications, a CPU: i7-11700F, a GPU: 2080ti lOgb, RAM: 32 GB, and Ubuntu 20.04.

5. Conclusions

This paper has presented a number of object recognition methods and algorithms. Furthermore, a novel image recognition framework aimed at scene analysis comprising two submodules (super-resolution and deep neural network) was proposed. A review and thorough analysis of the relevant existing work was conducted, along with the proposed framework's relation to it. The framework proposed in this paper may either use RetinaNet or YOLO to facilitate scene analysis at a neural network level; as such, both Retina and YOLO undergo review. The paper is concluded with a performance investigation of various state-of-the-art methods. The proposed analysis framework is benchmarked against standard scene recognition frameworks using COCO. The framework was benchmarked using both YOLO and Retina. As seen in the respective section, the proposed framework significantly outperforms the standard setups while using fine-tuning (FT), while performance shows no significant improvement otherwise. That being said, Retina shows a measurable higher deviation in all cases. High-performance computing has the potential of enabling object and scene recognition models to be trained more efficiently. It is model training and highly demanding deployed model architectures that can significantly benefit from low processing times, especially when it comes to real-time video processing.

References

Cai, Z., & Vasconcelos, N. (2019). Cascade R-CNN: high quality object detection and instance segmentation. *IEEE transactions on pattern analysis and machine intelligence.*

Cao, J., Cholakkal, H., Anwer, R. M., Khan, F. S., Pang, Y., & Shao, L. (2020). D2det: Towards high quality object detection and instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 11485-11494).

Duan, K., Bai, S., Xie, L., Qi, H., Huang, Q., & Tian, Q. (2019). Centernet: Keypoint

triplets for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 6569-6578).

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 580-587).

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014). Generative adversarial networks. arXiv preprint arXiv: 1406.2661.

He, K., Gkioxari, G., Dollar, P., & Girshick, R. (2017). Mask r-cnn. *Proceedings of the IEEE international conference on computer vision* (pp. 2961-2969).

Huang, Y., Shao, L., & Frangi, A. F. (2017). Simultaneous super-resolution and cross-modality synthesis of 3D medical images using weakly-supervised joint convolutional sparse coding. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 6070-6079).

Johnson, J., Alahi, A., & Fei-Fei, L. (2016, October). Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision* (pp. 694-711). Springer, Cham.

Ke, W., Zhang, T., Huang, Z., Ye, Q., Liu, J., & Huang, D. (2020). Multiple anchor learning for visual object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 10206-10215).

Kong, T., Sun, F., Liu, H., Jiang, Y., & Shi, J. (2019). Consistent optimization for single-shot object detection. *arXiv preprint* arXiv: 1901.06563.

Law, H., & Deng, J. (2018). Cornernet: Detecting objects as paired keypoints. In *Proceedings of the European conference on computer vision (ECCV)* (pp. 734-750).

Ledig, C., Theis, L., Huszar, F., Caballero, J., Cunningham, A., Acosta, A., ... & Shi, W. (2017). Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 4681-4690).

Lin, T. Y., Goyal, P., Girshick, R., He, K., & Dollar, P. (2017). Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision* (pp. 2980-2988).

Lin, T. Y., Goyal, P., Girshick, R., He, K., & Dollar, P. (2018). Focal Loss for Dense Object Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 42(2), 318-327.

Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., ... & Zitnick, C. L. (2014, September). Microsoft coco: Common objects in context. *European conference on computer vision* (pp. 740-755). Springer, Cham.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016, October). Ssd: Single shot multibox detector. In *European conference on computer vision* (pp. 21-37). Springer, Cham.

Mudunuri, S. P., & Biswas, S. (2015). Low resolution face recognition across variations in pose and illumination. *IEEE transactions on pattern analysis and machine intelligence,* 38(5), 1034-1040.

Perez-Rua, J. M., Zhu, X., Hospedales, T. M., & Xiang, T. (2020). Incremental few-shot object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 13846-13855).

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* (pp. 779-788).

Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *arXivpreprint* arXiv: 1506.01497.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint* arXiv: 1409.1556.

Zhou, X., Zhuo, J., & Krahenbuhl, P. (2019). Bottom-up object detection by grouping extreme and center points. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 850-859).

Submitted: 10.07.2020
Accepted: 30.04.2021