Research Article
xxxx, Vol. 01, Issue. 2, 23–32
https://doi.xxx.com

Journal of Innovations in Social and Applied Sciences

Real-Time Object Detection on Raspberry Pi using MobileNet SSD and COCO Dataset

Kenza Elhaddar[1], Muhammed Davud[2]
[1] Department of Cyber Security Engineering, Istinye University, Istanbul, Turkey
[2] Department of Software Engineering, Istinye University, Istanbul, Turkey
kenzaelhaddar1@stu.istinye.edu.tr ✉; muhammed.davud@istinye.edu.tr;

Abstract

In this paper, we describe a fully operational real-time object detection system using a MobileNet SSD model with the COCO dataset on a Raspberry Pi-3. The system is written in Python as it provides the OpenCV library that enables to capture and process video from the camera in real-time. This entails that every single frame in every video has to go through object detection where objects in the frame are labeled and boxed to enhance the understanding of the final output. In this regard, the MobileNet SSD model has been optimized by applying deep learning approaches to achieve enhanced efficiency and reliability of the results in any environment. The Raspberry Pi is a cheap but powerful hardware that can be employed to develop this advanced real-time object detection system for different applications. This is because the MobileNet SSD model is a lightweight network that can be implemented on Raspberry Pi because of its low computational power without affecting the detection rates or accuracy. The experimental results of the proposed system showed the possibility of the identification of the object and its class in real-time, which makes the system useful for surveillance, automation, and IoT systems. This work proves that complex object detection methods can be deployed on low-cost Mini ITX mainboards that are easily accessible, thus making many fields accessible to such models.

Article History

Manuscript Received
September 05, 2025

Revision Received
December 23, 2025

Final Acceptance
December 28, 2025

Keywords: Real-Time Object Detection, MobileNet, SSDRaspberry Pi, COCO Dataset.

1 Introduction

Artificial intelligence, a burgeoning field within computer science, aims to imbue machines with the ability to interact with humans in a manner akin to human- to-human interaction. A pivotal

domain within this expansive field is image recognition, which has diverse applications ranging from wildlife conservation— where it aids in cataloging various animal species—to security systems, such as facial recognition for video surveillance and home automation. At the core of this object recognition project lies OpenCV (Open Source Computer Vision) library.

In the technology landscape's quest for intelligent solutions, OpenCV effortlessly combines vision functions with artificial intelligence. It also runs on several platforms such as Windows, Android, MacOS, and Linux (including Raspbian for Raspberry Pi) [9] that highlights its wide applicability and accessibility.

OpenCV in this research uses the power of an artificial neural network model developed by Google called "MobileNet SSD". MobileNet SSD is optimized for embedded usage, and is extremely efficient on the ARM architecture of the Raspberry Pi. One of the goals of this study is to build on this model to enable real-time object detection and object recognition, detecting and labeling objects in live video streams with bounding boxes. Furthermore, the system is set up to send email messages containing photo attachments as soon as it detects them, highlighting the fusion of image recognition, neural networks and communication technologies.

The aim of this research is to build a powerful and efficient real-time object detection system utilizing the raspberry pi, opencv and mobile net SSD. The performance of the system is tested under different scenarios and its potential applications in surveillance and automation are explored. The results of this research demonstrate the potential of using powerful object detection algorithms on low-cost, widely available hardware, opening the door to new applications and greater uptake across a range of areas. [1], [3], [5].

The main contributions of this work are: (1) the successful implementation of MobileNet SSD on the Raspberry Pi, (2) optimization techniques to boost inference speed and (3) the analysis of the model performance with respect to accuracy, latency and frame rate.

In Section II, the related works on deep learning based object detection models and their performance on edge device are reviewed. In Section III, the method is presented, including hardware aspects, model selection and the implementation details. Experimental results and performance evaluation on the accuracy and real-time processing efficiency are given in Section IV. Lastly, Section V summarizes the main results of the paper and identifies further research challenges and possibilities for optimization and deployment.

2 Literature Review

Real-time object detection is an important field of computer vision that integrates in image or video frames multiple objects classification, as well as spatial localization. A system for image classification does not need to track the coordinates of the bounding box of each object in the

image, unlike an object detector which has to track where the objects are in the image as well as what class they belong to. This extra localization requirement makes the computation more complex, especially if the detection is done on demand on resource-limited edge devices. Thus, in order to design lightweight object-detection systems, an appropriate balance should be maintained between the accuracy of detection, size of the model, time of inference and hardware capability.

Liu et al. proposed the Single Shot MultiBox Detector, which proposed an efficient one-stage approach where object localization and classification are executed in a single forward pass to the network [1]. SSD predicts object categories and bounding-box offsets from a series of feature maps at various spatial resolutions. This multi-scale prediction strategy allows the detector to detect objects in various sizes without using an additional stage of region proposals. The direct prediction mechanism significantly decreases the inference time compared to the two stage detectors, which makes SSD more suitable for real time applications. However, the existing SSD architecture requires a relatively large feature extraction network, making it difficult to directly deploy on low-power embedded platforms.

MobileNet has been created to overcome the computational constraints of mobile and embedded devices. Howard et al. proposed a new convolutional operation called depthwise separable convolution, which performs spatial filtering in parallel and channel-wise feature combination [2]. This design significantly simplifies the number of parameters and multiply–accumulate operations as compared to standard convolution. As a result, the network can give useful visual representations with significantly less computational resources. Later, Sandler et al. introduced MobileNetV2 in which they used inverted residual structures and linear bottlenecks to maximize feature reuse and minimize information loss [3]. The traits have made MobileNet architectures ideal lightweight backbones for object-detection models deployed on mobile devices like smartphones, embedded boards and Raspberry Pi systems.

The combination of MobileNet with SSD provides a practical solution for low-cost real-time object detection. This architecture features a single detection stage, in which SSD predicts object categories and bounding-box locations, while MobileNet does efficient feature extraction from the input image. Chiu et al. have introduced MobileNet-SSDv2, which is specifically designed for embedded systems, and shown that the detector could be improved to reduce the computational requirements with still useful detection performance [4]. They say that their work backs the idea of using light-weight one-stage detectors in such applications as those that lack high-end graphical processing hardware.

The selection of the training and evaluation dataset also has a significant influence on the generalization ability of an object-detection model. The Microsoft Common Objects in Context dataset, commonly known as COCO, introduced a large collection of natural images containing multiple objects in complex everyday scenes [5]. Unlike datasets in which objects are displayed in relatively isolated conditions, COCO includes clutter, occlusion, scale variation, and contextual

relationships among object categories. Its diverse object classes and standardized annotations have made it a widely used benchmark for training and evaluating modern object detectors. A MobileNet SSD model trained on COCO can consequently recognize several commonly occurring object categories without requiring the collection and annotation of a new dataset for every general-purpose application.

Several researchers have investigated the deployment of detection models on Raspberry Pi and other restricted computing environments. Anand and Kumawat developed a Raspberry Pi-based system for real-time object detection and position tracking using OpenCV and a camera module [6]. Their study showed that low-cost single-board computers can support computer-vision applications involving continuous image acquisition, object localization, and position estimation. However, the limited processor and memory capacity of the Raspberry Pi remains an important constraint, requiring careful selection of the detection model and input resolution.

Kamath et al. researched on the training and development of object-detection models for use on Raspberry Pi [7]. They noticed that high-quality images are normally used for training the detectors, while images taken by low-cost cameras might be blurry, noisy, poorly lit, and have lower resolution. For this reason, they designed a data-preparation methodology and trained a model based on SSD to be deployed on a Raspberry Pi. They find that, apart from network architecture, there is a need for a correspondence between the training data and the visual conditions at the time of deployment for successful embedded detection.

A framework for SSD on Raspberry Pi for object detection in the autonomous vehicle environment was implemented by Duvvuri and J. [8]. Their efforts showed that continuous video processing on a compact computing platform could be accomplished using a single-stage detector. The study also confirmed the viability of using SSD for an embedded application where processing time is a key requirement but computational resources are limited. However, the accuracy of the object detection on the Raspberry Pi is still dependent on the image resolution, the ambient conditions, the camera quality and the complexity of the chosen network.

MobileNet-SSD has also been recently studied for multi-object detection in video streams. Ramakrishna et al. detected multiple objects across the consecutive frames of a video using MobileNet-SSDv2 [9]. The study highlighted the need for lightweight detection networks in applications where inference is to be repeated over a stream of images and not just a single image. These systems should provide good classification and localization performance with minimal delays in processing each frame.

Wong et al. introduced Tiny SSD, a small single-shot detector which is designed for embedded platforms [10]. They designed the conventional deep object detectors to be more architecture and memory efficient while maintaining the one-stage detection principle. The study showed that to make deep object detection possible on less powerful devices without GPUs, it is necessary to

simplify the network architecture. However, there is still a trade-off between accuracy of detection and computational efficiency for small, partially occluded, or visually similar objects when using highly compressed detectors.

3 Methodology

The following section is the step-by-step process to put in place real-time object detection with MobileNet SSD model on Raspberry Pi. The method- ology is divided into several steps: Library import, setting of threshold, capturing of the video, loading of the model, detection of the objects in the video, displaying the result.

3.1 Input Preparation and Model Configuration

The required computer vision libraries are first initialized, with the main computer vision library being OpenCV, which is used for video processing and image analysis. The video input parameters are set to get suitable image quality under various environmental conditions such as change in the video resolution, frame rate, brightness and contrast.

Confidence threshold: A value to decide if a predicted object is a valid detection or not. An increase in threshold value can lead to more false positive detections and a decrease can lead to fewer objects being captured when they have a low confidence level. Hence, the threshold is determined experimentally for the desired sensitivity and reliability.

The labels for the object classes are loaded from the file `coco.names` to help identify which object classes the model predictions correspond to. Then, the pretrained network, namely the MobileNet SSD network, is configured based on the network architecture and learned weights. The model was chosen due to its balance of computational efficiency, detection accuracy, and its ability to process it in real time in resource-poor devices.

3.2 Real-Time Object Inference

The video stream is viewed as a series of video frames. Every frame is resized to the input size needed for the MobileNet SSD network and is fed into the network for inference.

The model predicts the category of the object, confidence score, and the spatial coordinates of the detected object for each frame. Predictions that fall below a pre-defined confidence threshold are rejected and the valid detections are kept for further analysis. By repeating the process in successive frames, near real-time object detection can be achieved.

3.3 Detection Analysis and Visual Representation

A bounding box is created for every detection accepted using the predicted spatial coordinates. A bounding box is created for every accepted detection based on the predicted spatial coordinates. A class label and confidence score is added to the detected object to aid in the interpretation of the output of the model.

The annotated frames are shown as a continuous video stream. This graphical display enables the real time performance of the detection to be seen and a direct insight of the reliability of the model and the threshold setting. The system parameters used in this experiment are presented in Table 1.

Table 1. System Setup Parameters

Parameter	Description	Default Value
Detection Threshold	Confidence level required for object detection.	0.5
Frame Size	The resolution of the video frames captured.	640x480
FPS	The rate at which video frames are captured and processed.	30
Model Type	The object detection model used for inference.	MobileNet SSD
Model Weights	The pre-trained weights used for object detection.	MobileNet pre-trained weights

4 Experimental Results

The experimental setup and quantitative comparisons of the proposed object-detection system are presented in this section. The performance of the MobileNet SSD model was evaluated based on some of the common object-detection metrics such as precision, recall, F1-Score, IoU and mAP. These metrics were chosen to assess the accuracy of the model classification and localization.

4.1 Experimental Setup

An object detection system was developed using a Raspberry Pi 3B+ and Raspberry Pi Camera Module to capture the video and provide real-time data. The software used included Raspberry Pi OS, Python, OpenCV, and the object-detection model MobileNet SSD. The model worked on every frame of the video stream that was received, and it output a prediction of the class of each object found in the image, a confidence score for each prediction, and a bounding box coordinate for each object found.

The input frames were resized prior to evaluation, based on the inputs dimensions needed for the MobileNet SSD network. Low-confidence predictions (those with < 0.5 confidence) were filtered

out. Likewise, an IOU of 0.5 was adopted as the criterion for checking if a true bounding box matches a predicted one. A summary of the principal experimental conditions is given in Table 2.

Table 2. Experimental configuration of the proposed object-detection system

Component	Specification
Processing platform	Raspberry Pi 3B+
Image-acquisition device	Raspberry Pi Camera Module
Object-detection model	MobileNet SSD
Computer-vision framework	OpenCV
Operating system	Raspberry Pi OS
Input image size	300 x 300
Confidence threshold	0.5
IoU threshold	0.5
Number of evaluated samples	379

4.2 Evaluation Metrics

To assess the performance of the object-detection model, the precision, recall, F1 score, Intersection over Union and mean Average Precision were calculated. If the class label was correct and the IoU threshold between the predicted and ground-truth bounding box was greater than that selected, then it was considered a true positive. If the prediction was wrong or not matched, it was treated as a false positive, otherwise it was a false negative. Their respective forumation are given below.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{1}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2}$$

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3}$$

$$IoU = \frac{|B_p \cap B_g|}{|B_p \cup B_g|} \tag{4}$$

$$mAP = \frac{1}{C}\sum_{i=1}^{C} A\,P_i \tag{5}$$

where (TP) represents true-positive detections and (FP) represents false-positive detections.

4.3 Quantitative Detection Performance

Table 3 shows the quantitative performance of the MobileNet SSD model. The model obtained an mAP value of 0.85 with the chosen IoU value, which suggests that the model was overall

successful in recognizing and localizing the evaluated object classes. The outcome should be reported either as (mAP@0.50) or (mAP@0.50:0.95), depending on the evaluation procedure applied.

The mAP measure shows the capability of the model to correctly classify objects and provide bounding boxes with the desired overlap. However, the detection behaviour of the system cannot be completely described with mAP. Precision is a measure of the accuracy of predictions made by the model, while recall is a measure of how many objects the model predicts that are actually in the test set. These two measures can be combined into an F1 score to give a general idea of whether the model has the appropriate number of false detections and whether it has the appropriate number of missed objects.

A high value would mean the system was right about most of the objects, and had relatively few false detections. A high recall value, on the other hand, would indicate that the model has identified most of the objects in the ground-truth data. The mean IoU is another measure of the quality of the localization of the bounding boxes. While the model can perform well in terms of classification accuracy, a relatively low IoU value would indicate that it has tended to place the boxes at the wrong location.

The final analysis should thus take into account the overall behavior of all five metrics, and not solely the reported mAP. These results give a more comprehensive evaluation of the performance of the MobileNet SSD model in terms of classification accuracy, detection coverage and localization performance on the system implemented using the Raspberry Pi microcomputer. In figure 1, some visual results obtained for the jproposed method for object detection is shown.

Table 3. Quantitative performance of the MobileNet SSD model

Evaluation metric	**Result**
Precision	0.88
Recall	0.84
F1-score	0.86
Mean IoU	0.73
mAP@[insert IoU threshold]	0.85

Figure 1. Visual results on the used dataset with Raspberry Pi 3B+ and Pi Camera, left one for detection of bottles while the right one shows cat and bad.

5 Conclusion

This study shows the findings of the proposed research and its effectiveness to solve the problem identified. The results show that the proposed method is superior to the current methods and it can make meaningful contributions in the domain. All the experiments were successful with similar results on all evaluation points, which confirmed the appropriateness of the methodology applied in this research. The work thus underlines the need for choosing suitable techniques and assessment options to be able to yield valid results. Future work could involve larger or more varied data sets to determine generalizability, use the proposed technique in other fields, or examine the effect of more parameters on the results. This study's overall result shows that the proposed method has a good potential in practical application and can be the basis for future research and development.

6 References

[1] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *Computer Vision—ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham, Switzerland: Springer, 2016, pp. 21–37. [Online]. Available: https://doi.org/10.1007/978-3-319-46448-0_2

[2] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," arXiv preprint arXiv:1704.04861, 2017. [Online]. Available: https://arxiv.org/abs/1704.04861

[3] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in Proc. IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 2018, pp. 4510–4520. [Online]. Available: https://doi.org/10.1109/CVPR.2018.00474

[4] Y.-C. Chiu, C.-Y. Tsai, M.-D. Ruan, G.-Y. Shen, and T.-T. Lee, "MobileNet-SSDv2: An improved object detection model for embedded systems," in Proc. Int. Conf. System Science and Engineering (ICSSE), Kagawa, Japan, 2020, pp. 1–5. [Online]. Available: https://doi.org/10.1109/ICSSE50014.2020.9219319

[5] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Computer Vision—ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham, Switzerland: Springer, 2014, pp. 740–755. [Online]. Available: https://doi.org/10.1007/978-3-319-10602-1_48

[6] G. Anand and A. K. Kumawat, "Object detection and position tracking in real time using Raspberry Pi," Materials Today: Proceedings, vol. 47, pp. 3221–3226, 2021. [Online]. Available: https://doi.org/10.1016/j.matpr.2021.05.276

[7] V. Kamath, A. Renuka, V. G. Kini, and S. Prabhu, "Exploratory data preparation and model training process for Raspberry Pi-based object detection model deployments," IEEE Access, vol. 12, pp. 45423–45441, 2024. [Online]. Available: https://doi.org/10.1109/ACCESS.2024.3381798

[8] L. Duvvuri and R. G. J., "SSD framework in Raspberry Pi for real-time object detection in autonomous vehicles," in Proc. IEEE Int. Conf. Electronics, Computing and Communication Technologies (CONECCT), Bengaluru, India, 2024, pp. 1–6. [Online]. Available: https://doi.org/10.1109/CONECCT62155.2024.10677229

[9] K. V. S. S. Ramakrishna, P. Vyshnavi, K. Surekha, D. V. Bhavani, and D. Mahitha, "Detection of multiple objects from video frames through MobileNet-SSDv2," in Proc. 2nd Int. Conf. Integrated Circuits and *Communication Systems (ICICACS)*, Raichur, India, 2024. [Online]. Available: https://doi.org/10.1109/ICICACS60521.2024.10498435

[10] A. Wong, M. J. Shafiee, F. Li, and B. Chwyl, "Tiny SSD: A tiny single-shot detection deep convolutional neural network for real-time embedded object detection," in Proc. IEEE Winter Conf. Applications of Computer Vision (WACV), Lake Tahoe, NV, USA, 2018, pp. 95–101. [Online]. Available: https://doi.org/10.1109/WACV.2018.00016