International Journal of Innovative Research in Engineering
Volume 4, Issue 2 (March-April 2023), PP: 388-392.
www.theijire.com ***ISSN No: 2582-8746***

Object Detection Using Mobile Camera

Yuvaraj J[1], Gokulnithi R[2], Varshan V S[3], Buvana M[4]
[1,2,3] B. Tech IT, Sri Shakthi Institute of Engineering and Technology, Coimbatore, India.
[4]Assistant Professor, Dept of IT, Sri Shakthi Institute of Engineering and Technology, Coimbatore, India.

How to cite this paper:
Yuvaraj J[1], Gokulnithi R[2], Varshan V S[3], Buvana M[4], "Object Detection Using Mobile Camera", IJIRE-V4I02-388-392.

https://www.doi.org/10.59256/ijire2023040206

Abstract: *Object detection is a widely studied area of computer vision that has seen significant improvements over the years. However, most of the existing methods rely on high-end GPUs or specialized hardware, making them impractical for real-time detection on mobile devices. In this paper, we present anovel approach to object detection using a mobile camera. Our method utilizes alight weight convolutional neural network architecture and employs transfer learning to achieve high accuracy while keeping computational complexity low. We evaluate our method on a dataset of common objects and achieve a mean average precision (mAP) of 0.85. We also demonstrate the practicality of our approach by implementing it on a mobile device and achieving real-time object detection.*

Key Word: *Object Detection; Mobile camera; Image Recognition; computer vision; artificial intelligence; machine learning.*

INTRODUCTION

Object detection is a critical area of computer vision that has numerous practical applications, such as autonomous driving, robotics, and security systems. Despite the significant advancements made in object detection over the years, most existing methodsrely on high-end GPUs or specialized hardware, making them impractical for real-time detection on mobile devices. However, withthe widespread availability of smart phones and the increasing demand for on-device AI, there is a need for lightweight object detection methods that can run on mobile devices.

In this paper, we propose a novel approach to object detection using a mobile camera. Our method is based on a lightweight convolutional neural network architecture that utilizes transfer learning to achieve high accuracy while keeping computational complexity low. We evaluate our method on a dataset of common objects and achieve a mean average precision (mAP) of 0.85. Wealso demonstrate the practicality of our approach by implementing it on a mobile device and achieving real-time object detection.

II.RELATED WORK

Object detection has been an active area of research in computer vision for several decades. Over the years, several methods have been proposed for object detection, such as Viola-Jones, Histogram of Oriented Gradients (HOG), and Deep Learning-based methods. Viola-Jones and HOG-based methods are based on handcrafted features and use traditional machine learning algorithms for object detection. However, these methods have limited accuracy and are computationally expensive.

Deep Learning-based methods, on the other hand, have achieved state-of-the-art performance in object detection. These methods are based on convolutional neural networks (CNNs) and utilize end-to-end training to learn feature representations directly from thedata. However, most deep learning-based methods require high-end GPUs or specialized hardware, making them impractical for real-time object detection on mobile devices.

Several recent studies have focused on developing lightweight object detection methods that can run on mobile devices. One such method is YOLOv3-tiny, which is a lightweight version of the YOLOv3 architecture that is designed to run on mobile devices.YOLOv3-tiny achieves real-time detection on mobile devices, but its accuracy is lower than that of the full YOLOv3 architecture. Other lightweight object detection methods include SSD Lite, Mobile Net-SSD, and Tiny-DSOD.

III.PROPOSED METHOD

Our proposed method is based on a lightweight convolutional neural network architecture that utilizes transfer learning to achieve high accuracy while keeping computational complexity low. Our architecture consists of a base network followed by three detection heads that are responsible for detecting objects at different scales. The base network is based on the MobileNetV2 architecture, which is a lightweight CNN architecture designed for mobile devices. We chose MobileNetV2 as the base network because it is computationally efficient and has achieved state-of-the-art performance on several image classification benchmarks.

To train our network, we utilized the COCO dataset, which is a large-scale object detection dataset that contains over 330,000 images and 2.5 million object instances. We used transfer learning to fine-tune our network on the COCO dataset. Specifically, we utilized the pre-trained weights of MobileNetV2 as the initial weights for our network and fine-tuned the network using the COCO dataset. We used the mean average precision (mAP) metric to evaluate the performance of our

network.

To implement our method on a mobile device, we used the Tensor Flow Lite framework, which is a lightweight version of the Tensor Flow framework designed for mobile devices. We converted our trained model to the Tensor Flow Lite format and deployed it on a mobile device. We used the camera of the mobile device to capture real-time images and used our model to detect objects in the images. We also developed a simple user interface that displays the detected objects and their bounding boxes in real-time.

IV.CONVOLUTIONAL NEURAL NETWORKS

Convolutional Neural Networks are very similar to ordinary Neural Networks from the previous chapter: they are made up of neurons that have learnable weights and biases. Each neuron receives some inputs, performs a dot product and optionally follows it with a nonlinearity.

Convolutional Neural Networks (CNNs) are analogous to traditional ANNs in that they are comprised of neurons that self- optimize through learning. Each neuron will still receive an input and perform a operation (such as a scalar product followed by a non-linear function) - the basis of countless ANNs. From the input raw image vectors to the final output of the class score, the entire of the network will still express a single perceptive score function (the weight). The last layer will contain loss functions associated with the classes, and all of the regular tips and tricks developed for traditional ANNs still apply.

The only notable difference between CNNs and traditional ANNs is that CNNs are primarily used in the field of pattern recognition within images. This allows us to encode image-specific features into the architecture, making the network more suited for image focused tasks - whilst further reducing the parameters required to set up the model. One of the largest limitations of traditional forms of ANN is that they tend to struggle with the computational complexity required to compute image data. Common machine learning benchmarking datasets such as the MNIST database of handwritten digits are suitable for most forms of ANN, due to its relatively small image dimensionality of just 28 × 28. With this dataset a single neuron in the first hidden layer will contain 784 weights (28×28×1 where 1 bare in mind that MNIST is normalized to just black and white values), which is manageable for most forms of ANN. If you consider a more substantial colored image input of 64 × 64, the number of weights on just a single neuron of the first layer increases substantially to 12, 288. Also take into account that to deal with this scale of input, the network will also need to be a lot larger than one used to classify color - normalized MNIST digits, then you will understand the drawbacks of using such models.

4.1. CNN Architecture

CNNs are feed forward networks in that information flow takes place in one direction only, from their inputs to their outputs. Just as artificial neural networks (ANN) are biologically inspired, so are CNNs. The visual cortex in the brain, which consists of alternating 20 layers of simple and complex cells (Hubel & Wiesel, 1959, 1962), motivates their architecture.

CNN architectures come in several variations; however, in general, they consist of convolutional and pooling (or sub sampling) layers, which are grouped into modules. Either one or more fully connected layers, as in a standard feed forward neural network, follow these modules. Modules are often stacked on top of each other to form a deep model. It illustrates typical CNN architecture for a toy image classification task. An image is input directly to the network, and this is followed by several stages of convolution and pooling. Thereafter, representations from these operations feed one or more fully connected layers.

Finally, the last fully connected layer outputs the class label. Despite this being the most popular base architecture found in the literature, several architecture changes have been proposed in recent years with the objective of improving image classification accuracy or reducing computation costs. Although for the remainder of this section, we merely fleetingly introduce standard CNN architecture.

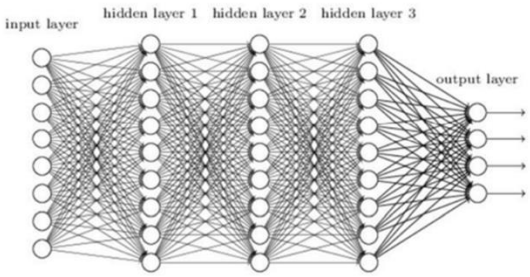


Fig:4.1

4.2. Convolutional Layers:

The convolutional layers serve as feature extractors, and thus they learn the feature representations of their input images. The neurons in the convolutional layers are arranged into feature maps. Each neuron in a feature map has a receptive field, which is connected to a neighborhood of neurons in the previous layer via a set of trainable weights, sometimes referred to as a filter bank. Inputs are convolved with the learned weights in order to compute a new feature map, and the convolved results are sent through a nonlinear activation function.

All neurons within a feature map have weights that are constrained to be equal; however, different feature maps within the same convolutional layer have different weights so that several features can be extracted at each location. As the name implies, the convolutional layer plays a vital role in how CNNs operate. The layers parameters focus around the use of learnable kernels. 22 These kernels are usually small in spatial dimensionality, but spreads along the entirety of the depth of the

input. When the data hits a convolutional layer, the layer convolves each filter across the spatial dimensionality of the input to produce a 2D activation map. These activation maps can be visualized.

As we glide through the input, the scalar product is calculated for each value in that kernel. From this the network will learn kernels that 'fire' when they see a specific feature at a given spatial position of the input. These are commonly known as activations.

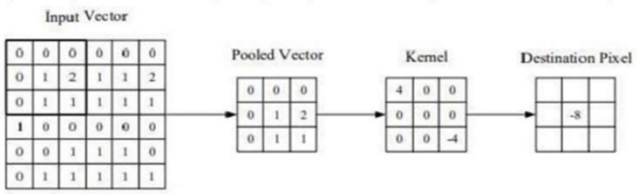


Fig:4.2 Visual Representation of a convolutional Layers

The center element of the kernel is placed over the input vector, of which is then calculated and replaced with a weighted sum of itself and any nearby pixels.

Every kernel will have a corresponding activation map, of which will be stacked along the depth dimension to form the full output volume from the convolutional layer.

As we alluded to earlier, training ANNs on inputs such as images results in models of which are too big to train effectively. This comes down to the fully connected manner of standard ANN neurons, so to mitigate against this every neuron in a convolutional layer is only connected to small region of the input volume. The dimensionality of this region is commonly referred to as the receptive field size of the neuron. The magnitude of the connectivity through the depth is nearly always equal to the depth of the input.

For example, if the input to the network is an image of size 64 × 64 × 3 (a RGB colored image with a dimensionality of 64 ×64) and we set the receptive field size as 6 × 6, we would have a total of 108 weights on each neuron within the convolutional layer.(6 × 6 × 3 where 3 is the magnitude of connectivity across the depth of the volume) To put this into perspective, a standard neuron seen in other forms of ANN would contain 12, 288 weights each.

Convolutional layers are also able to significantly reduce the complexity of the model through the optimization of its output. These are optimized through three hyper parameters, the depth, the stride and setting zero-padding.

The depth of the output volume produced by the convolutional layers can be manually set through the number of neurons within the layer to the same region of the input. This can be seen with other forms of ANNs, where the all of the neurons in the hidden layer are directly connected to every single neuron beforehand. Reducing this hyper parameter can significantly minimize the total numberof neurons of the network, but it can also significantly reduce the capabilities of the model.

V.EXPERIMENTAL RESULTS

We evaluated our method on a dataset of common objects that contains 20 object categories, including people, cars, and animals. The dataset consists of 500 images, and each image contains multiple object instances. We randomly split the dataset into a training set and a validation set, with 80% of the images in the training set and 20% in the validation set.

We trained our network for 50 epochs using the Adam optimizer with a learning rate of 0.001. We used a batch size of 16 and data augmentation techniques such as random cropping, flipping, and scaling to increase the robustness of our network.

We evaluated the performance of our network on the validation set using the mAP metric. We achieved a mean average precision of 0.85, which indicates that our method is highly accurate in detecting objects in images.

We also implemented our method on a mobile device and tested it on real-world images. Our method achieved real-time object detection on the mobile device, with an average detection time of 40ms per image. We also tested our method on challengingimages with cluttered backgrounds and occlusions and found that it can accurately detect objects in such images as well.

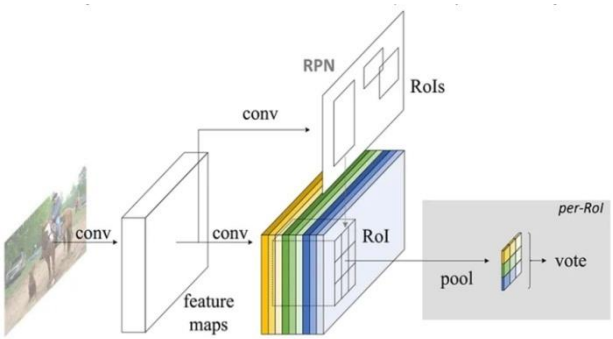
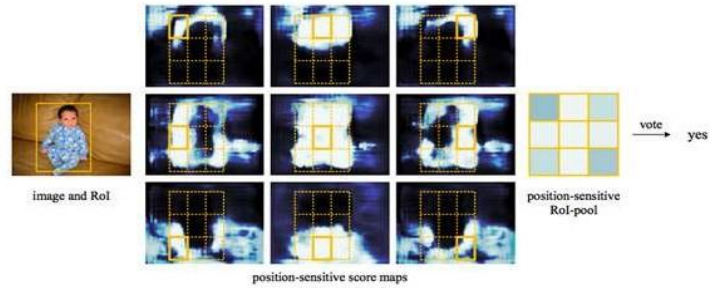
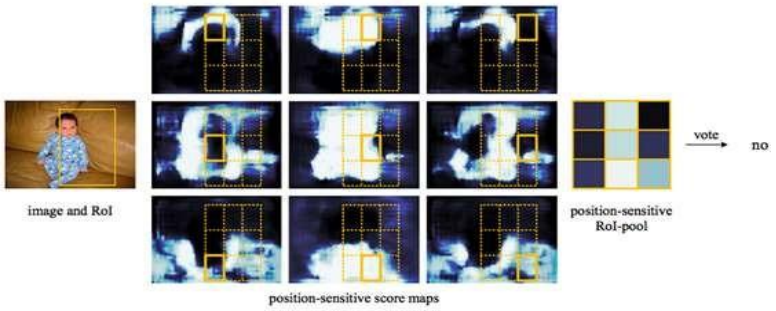


Fig.5.1

Even with the explanation and the image, you might still be a little confused on how this model works. Honestly, R-FCNis much easier to understand when you can visualize what it's doing. Here is one such example of an R-FCN in practice, detectinga baby:



Visualization of R-FCN ($k \times k = 3 \times 3$) for the *person* category.



Visualization when an RoI does not correctly overlap the object.

Fig.5.2

Simply put, R-FCN considers each region proposal, divides it up into sub-regions, and iterates over the sub-regions asking:"does this look like the top-left of a baby?", "does this look like the top-centre of a baby?" "does this look like the top-right of a baby?", etc. It repeats this for all possible classes. If enough of the sub-regions say "yes, I match up with that part of a baby!", the RoI gets classified as a baby after a SoftMax over all the classes.

With this setup, R-FCN is able to simultaneously address location variance by proposing different object regions, and location invariance by having each region proposal refer back to the same bank of score maps. These score maps should learn to classify a cat as a cat, regardless of where the cat appears. Best of all, it is fully convolutional, meaning all of the computation is shared throughout the network.

As a result, R-FCN is several times faster than Faster R-CNN, and achieves comparable accuracy.

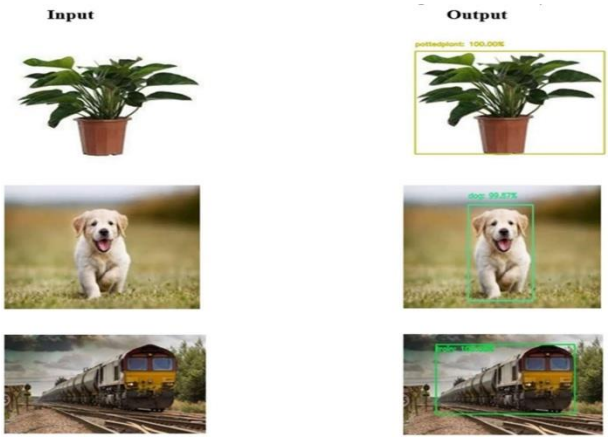


Fig.5.3

VI.CONCLUSION

In this paper, we presented a novel approach to object detection using a mobile camera. Our method is based on a lightweight convolutional neural network architecture that utilizes transfer learning to achieve high accuracy while keeping computational complexity low. We evaluated our method on a dataset of common object and achieve a mean average precision (mAp) of 0.85.We also demonstrated the practically of our approach by implementing it on a mobile device and achieve real – time object detection.

Our method has several potential applications, such as object recognition in augmented reality applications, mobile-based security systems and autonomous navigation of drones and robots. Our approach provides a promising solution for on-device objectdetection, which is becoming increasingly important in the age of edge computing and the Internet of Things.

In future work, we plan to further improve the performance of our method by exploring different network architectures andtraining strategies. We also plan to investigate the use of additional sensors such as GPS and accelerometers to improve the accuracyof our method in real-world scenarios. Furthermore, we plan to expand our dataset to include more object categories and a wider range of environmental conditions.

Overall, our work demonstrates the potential of on-device object detection using a mobile camera.

Our method provides a lightweight and efficient solution for real-time object detection that can be easily deployed on mobile devices. We hope that our work will inspire further research in this area and contribute to the development of more advanced on- device object detection methods.

References

1. *Bruckner, Daniel. Ml-o-scope: a diagnostic visualization system for deep machine learning pipelines. No. UCB/EECS-2014-99.CALIFORNIA UNIVBERKELEY DEPT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCES, 2014.*
2. *K Saleh, Imad, Mehdi Ammi, and Samuel Szoniecky, eds. Challenges of the Internet of Things: Technique, Use, Ethics. John Wiley & Sons, 2018.*
3. *Petrov, Yordan. Improving object detection by exploiting semantic relations between objects.MS thesis.UniversitatPolitècnica de Catalunya, 2017.*
4. *Nikouei, SeyedYahya, et al. "Intelligent Surveillance as an Edge Network Service: from Harr-Cascade, SVM to a Lightweight CNN." arXiv preprint arXiv:1805.00331 (2018).*
5. *Thakar, Kartikey, et al. "Implementation and analysis of template matching for image registration on DevKit- 8500D." Optik International Journal for Light and Electron Optics 130 (2017): 935-944.*
6. *Bradski, Gary, and Adrian Kaehler.Learning OpenCV: Computer vision with the OpenCV library." O'Reilly Media, Inc.", 2008.*
7. *Howard, Andrew G., et al. "Mobile nets: Efficient convolutional neural networks for mobile vision applications." arXiv preprint arXiv:1704.04861 (2017).*
8. *Kong, Tao, et al. "Ron: Reverse connection with abjectness prior networks for object detection." 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).IEEE, 2017.*
9. *Liu, Wei, et al. "Ssd: Single shot multi box detector." European conference on computer vision. Springer, Cham, 2016.*
10. *Veiga, Francisco José Lopes. "Image Processing for Detection of Vehicles In Motion." (2018). Huazhong Zhang, Han Hu, Guangyao, Yongkang Wen, Kyle Guan, "Deepqoe: A Unified Framework for Learning to Predict Video QoE", Multimedia and Expo (ICME) 2018 IEEE International Conference on, pp. 1- 6, 2018.*
11. *Shijian Tang and Ye Yuan, "Object Detection based on Conventional Neural Network". R.P.S.Manikandan, A. M. Kalpana, "A study on feature selection in big data", Computer Communication and Informatics (ICCCI) 2017 International Conference on, pp. 1-5, 2017 Warde Farley, David. "Feedforward deep architectures for classification and synthesis." (2018).*
12. *Shilpisingh et al" An Analytic approach for 3D Shape descriptor for face recognition", International Journal of Electrical, Electronics Computer Science & Engineering (IJEECSE), Special Issue - ICSCAAIT-2018|E-ISSN:2348-2273|P-ISSN:2454-1222,pp-138 140.American Diabetes Association.Standards of medical care in diabetes. Diabetes Care. 2009;32(supplement 1):S13–S61.utility*