e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:05/Issue:06/June-2023 Impact Factor- 7.868 www.irjmets.com

OBJECT DETECTION USING OPENCV WITH PYTHON

P. Nikitha[*1], T. Nikitha[*2], M. Nikshitha[*3], T. Nikhil[*4], T. Nithin[*5],
K. Nithish Reddy[*6], Prof. Sanjaykumar J Hamilpure[*7]

[*1,2,3,4,5,6]School Of Engineering Mallareddy University Hyderabad, Telangana, India.

[*7]Guide, Assistant Professor, Department Of AIML School Of Engineering Mallareddy University Hyderabad, Telangana, India.

DOI : https://www.doi.org/10.56726/IRJMETS41586

ABSTRACT

Our project is to Detect an object with OpenCV- Python . OpenCV has a bunch of pre-trained classifiers that can be used to identify objects such as trees, number plates, faces, eyes, etc. Object detection is a well-known computer technology connected with computer vision and image processing that focuses on detecting objects or its instances of a certain class (such as humans, flowers, animals) in digital images and videos. Object detection can be used for various purposes including retrieval and surveillance. The purpose of "object detection" is to properly locate an object in a photograph and label it with the relevant category. To be more specific, object detection attempts to handle the difficulty of detecting where and what an item is. However, resolving this issue is not simple. A computer, unlike the human eye, analyses pictures in two dimensions. Furthermore, the object's size, direction in space, attitude, and placement in the picture might all be somewhat different.

I. INTRODUCTION

To Object detection is a computer vision task that involves identifying and localizing objects in images or videos. It has various applications, ranging from autonomous driving and surveillance systems to augmented reality and image recognition.

Python, along with the OpenCV (Open Source Computer Vision) library, provides powerful tools for object detection. OpenCV is an open-source computer vision and machine learning software library that offers various algorithms and functions for image and video processing.

To perform object detection using Python and OpenCV, you'll typically follow these steps:

Install OpenCV: Begin by installing the OpenCV library on your system. You can use package managers like pip or conda to install the necessary dependencies.

Load the image/video: Use OpenCV to load the input image or video you want to perform object detection on. OpenCV provides functions for reading images from files or capturing frames from a video stream.

Preprocess the input: Preprocessing steps such as resizing, normalization, or color conversion may be required before applying object detection algorithms. OpenCV provides a wide range of image manipulation functions to help with these tasks.

Select an object detection algorithm: There are various object detection algorithms available, such as Haar cascades, HOG (Histogram of Oriented Gradients), and deep learning-based approaches like YOLO (You Only Look Once) and SSD (Single Shot MultiBox Detector). Choose an algorithm that best suits your requirements.

Implement the object detection algorithm: Depending on the chosen algorithm, you'll need to implement the necessary code to apply the object detection technique to your input image or video. OpenCV provides functions and classes to facilitate the implementation of various object detection algorithms.

Detect objects: Run the object detection algorithm on your input data. The algorithm will analyze the image or video frames and identify the presence and location of objects within them.

Visualize the results: After detecting objects, you can draw bounding boxes or labels around them to visualize the results. OpenCV offers functions for drawing shapes, text, and annotations on images or video frames.

Post-process the results (optional): Depending on your specific application, you might need to perform additional post-processing on the detected objects. This could involve filtering or refining the detections to improve accuracy or removing false positives.

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:05/Issue:06/June-2023 Impact Factor- 7.868 www.irjmets.com

II. EXISTING SYSTEM

- Haar Cascade Classifier: This is a popular algorithm used for object detection in OpenCV. It uses a set of features to detect objects, and can be trained on any object using positive and negative samples. It was introduced by Viola and Jones in 2001 and has since been widely adopted for tasks such as face detection. The algorithm is based on the concept of Haar-like features, which are simple rectangular filters that can be applied to an image to extract features such as edges, lines, or corners. These features are computed at different scales and positions in the image to capture variations in object appearance. The training process of the Haar Cascade Classifier involves two main steps: feature selection and classifier training. During feature selection, a large set of Haar-like features is evaluated on a training dataset to determine the most discriminative ones. This helps to reduce the computational complexity of the classifier. In the classifier training step, a cascade of weak classifiers is trained using a boosting algorithm, such as AdaBoost or GentleBoost. Each weak classifier focuses on a specific Haar-like feature and learns to classify whether that feature is present in an image region or not.
- HOG (Histogram of Oriented Gradients) Detector: The Histogram of Oriented Gradients (HOG) detector is a feature extraction and object detection algorithm widely used in computer vision tasks. It was introduced by Dalal and Triggs in 2005 and has since become a popular method for detecting pedestrians and other objects in images or videos. The HOG detector works by capturing local gradient information from an image and representing it as a feature vector. The HOG detector has been widely implemented in computer vision libraries, such as OpenCV, making it accessible for developers to use in their applications. It is often used in combination with other techniques, such as sliding window detection and non-maximum suppression, to achieve accurate and efficient object detection.
- Deep Neural Networks: Deep Neural Networks (DNNs), also known as deep learning models, are a class of artificial neural networks that are composed of multiple layers of interconnected nodes, called artificial neurons or units. These networks are designed to learn and represent complex patterns and relationships in data, making them particularly effective for tasks such as image recognition, natural language processing, and speech recognition. Deep neural networks have achieved remarkable success in various domains, including computer vision, natural language processing, and speech recognition. Their ability to automatically learn and extract complex features from large amounts of data has revolutionized many fields and led to significant advancements in artificial intelligence.
- Template Matching: composed of multiple layers of interconnected nodes, called artificial neurons or units. These networks are designed to learn and represent complex patterns and relationships in data, making them particularly effective for tasks such as image recognition, natural language processing, and speech recognition. Deep neural networks have achieved remarkable success in various domains, including computer vision, natural language processing, and speech recognition. Their ability to automatically learn and extract complex features from large amounts of data has revolutionized many fields and led to significant advancements in artificial intelligence.

III. PROPOSED SYSTEM

A proposed system for object detection using OpenCV with Python would typically involve the following steps:

1. Import Libraries: Begin by importing the necessary libraries, including OpenCV and any additional libraries required for image processing or machine learning.
2. Load the Image: Load the target image or video frame on which you want to perform object detection.
3. Preprocessing (if needed): Preprocess the image or frame as per the requirements of your specific application. This might include resizing, converting to grayscale, or applying any necessary filters.
4. Load the Pretrained Model: Choose a pretrained object detection model that suits your needs (e.g., Single Shot MultiBox Detector (SSD), You Only Look Once (YOLO), or Faster R-CNN) and load the model using the appropriate library or framework.
5. Object Detection: Apply the loaded model to perform object detection on the preprocessed image or frame. This typically involves passing the image through the network and obtaining the bounding boxes and class labels of the detected objects.
6. Postprocessing: Process the detected objects, such as filtering out low-confidence detections, applying non-

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:05/Issue:06/June-2023 Impact Factor- 7.868 www.irjmets.com

maximum suppression to remove overlapping bounding boxes, or performing any additional custom processing based on your application requirements.

7. Visualization: Draw the bounding boxes or other visual indicators on the image to show the detected objects. You can use OpenCV functions to draw rectangles, labels, and other annotations.

8. Display or Save Results: Show the final image with the detected objects or save the processed image/video with the annotations.

9. Repeat for Video (if applicable): If you are working with video, loop through each frame and apply the object detection process to achieve real-time or batch processing of the video frames.

10. Clean Up: Release any resources, such as memory or file handles, once the processing is complete.

ARCHITECTURE

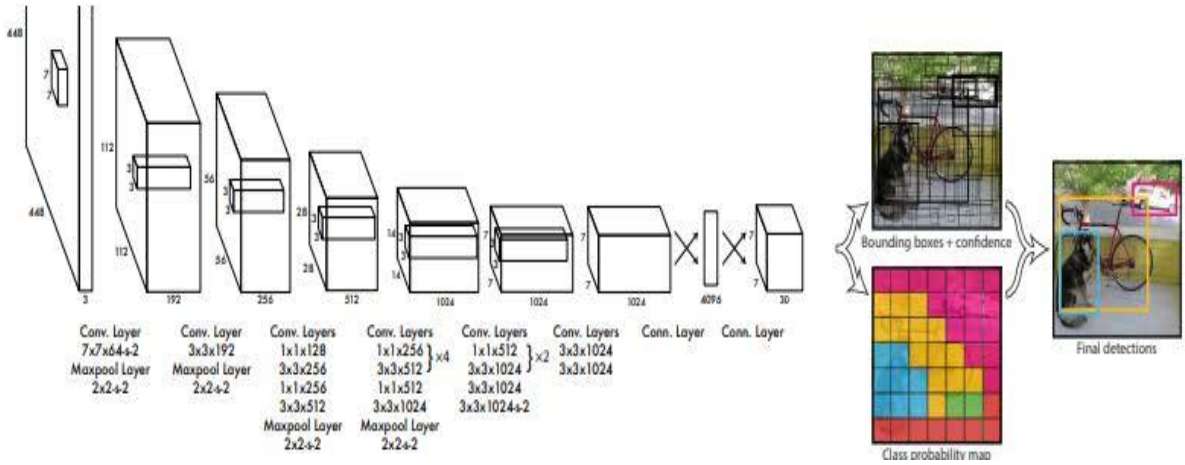

Figure 1: Object Detection Architecture

The architecture tells about the image processing and that will be divided in to small compartments because the algorithm make sure that every single part of the picture or the object should be analysed and detected properly.That goes through different levels and it displayes every feature of the object and provides the correct information about the object.

Compared to other systems

Object detection is a core problem in computer vision.

Detection pipelines generally start by extracting a set of robust features from input images (Haar , SIFT , HOG , convolutional features). Then, classifiers or localizers [1, 32] are used to identify objects in the feature space. These classifiers or localizers are run either in sliding window fashion over the whole image or on some subset of regions in the image [35, 15, 39].

We compare the Object detection system to several top detection frameworks, highlighting key similarities and differences.

Deformable parts models. Deformable parts models (DPM) use a sliding window approach to object detection. DPM uses a disjoint pipeline to extract static features, classify regions, predict bounding boxes for high scoring regions, etc. Our system replaces all of these disparate parts with a single convolutional neural network.

The network performs feature extraction, bounding box prediction, nonmaximal suppression, and contextual reasoning all concurrently. Instead of static features, the network trains the features in-line and optimizes them for the detection task. Our unified architecture leads to a faster, more accurate model than DPM.

R-CNN and its variants use region proposals instead of sliding windows to find objects in images. Selective Search generates potential bounding boxes, a convolutional network extracts features, an SVM scores the boxes, a linear model adjusts the bounding boxes, and non-max suppression eliminates duplicate detections. Each stage of this complex pipeline must be precisely tuned independently and the resulting system is very slow, taking more than 40 seconds per image at test time .

Object Detection shares some similarities with R-CNN.

However, our system puts spatial constraints on the grid cell proposals which helps mitigate multiple

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:05/Issue:06/June-2023 Impact Factor- 7.868 www.irjmets.com

detections of the same object. Our system also proposes far fewer bounding boxes, only 98 per image compared to about 2000 from Selective Search. Finally, our system combines these individual components into a single, jointly optimized model.

Other Fast Detectors Fast and Faster R-CNN focus on speeding up the R-CNN framework by sharing computation and using neural networks to propose regions instead of Selective Search . While they offer speed and accuracy improvements over R-CNN, both still fall short of real-time performance.

Many research efforts focus on speeding up the DPM

pipeline. They speed up HOG computation, use cascades, and push computation to GPUs. However, only 30Hz DPM actually runs in real-time. Instead of trying to optimize individual components of a large detection pipeline, Object Detection throws out the pipeline entirely and is fast by design. Detectors for single classes like faces or people can be highly optimized since they have to deal with much less

variation. Object Detection is a general purpose detector that learns to detect a variety of objects simultaneously.

Deep MultiBox. Unlike R-CNN, Szegedy et al. train aconvolutional neural network to predict regions of interest

instead of using Selective Search. MultiBox can also perform single object detection by replacing the confidence

prediction with a single class prediction. However, MultiBox cannot perform general object detection and is still just a piece in a larger detection pipeline, requiring further image patch classification. Both Object Detection and MultiBox use a convolutional network to predict bounding boxes in an image but Object detection is a complete detection system.OverFeat. Sermanet et al. train a convolutional neural

network to perform localization and adapt that localizer to perform detection [32]. OverFeat efficiently performs sliding window detection but it is still a disjoint system. OverFeat optimizes for localization, not detection performance. Like DPM, the localizer only sees local information when making a prediction. OverFeat cannot reason about global context and thus requires significant post-processing to produce coherent detections. MultiGrasp. Our work is similar in design to work on grasp detection by Redmon et al [27]. Our grid approach to bounding box prediction is based on the MultiGrasp system for regression to grasps. However, grasp detection is a much simpler task than object detection. MultiGrasp only needs to predict a single graspable region for an image containing one object. It doesn't have to estimate the size, location,or boundaries of the object or predict it's class, only find a region suitable for grasping. YOLO predicts both bounding boxes and class probabilities for multiple objects of multiple classes in an image.

4. Experiments

First we compare YOLO with other real-time detection systems on PASCAL VOC 2007. To understand the differences between YOLO and R-CNN variants we explore the errors on VOC 2007 made by YOLO and Fast R-CNN, one of the highest performing versions of R-CNN [14]. Based on the different error profiles we show that YOLO can be used to rescore Fast R-CNN detections and reduce the errors from background false positives, giving a significant performance boost. We also present VOC 2012 results and compare mAP to current state-of-the-art methods. Finally, we show that YOLO generalizes to new domains better than other detectors on two artwork datasets.

Data Flow Diagram

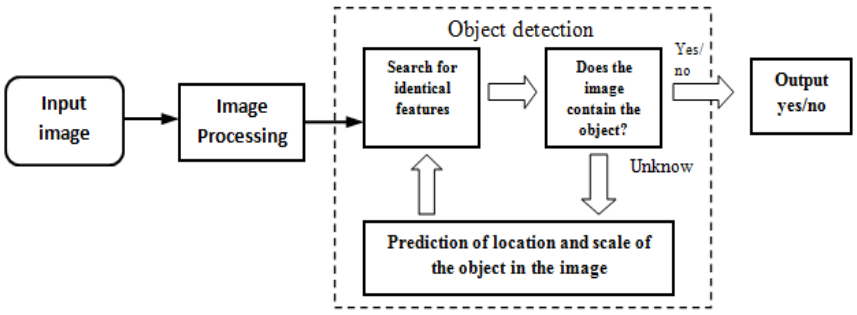

Figure 2: Object Detection DATA FLOW

IV. RESULTS

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:05/Issue:06/June-2023 Impact Factor- 7.868 www.irjmets.com

Qualitative Results.

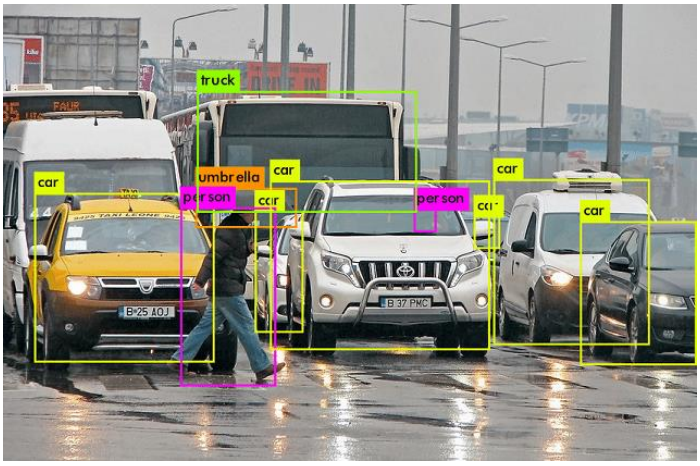

Figure 3: Qualitative Results of traffic

Figure 4: Qualitative Results of Electronics

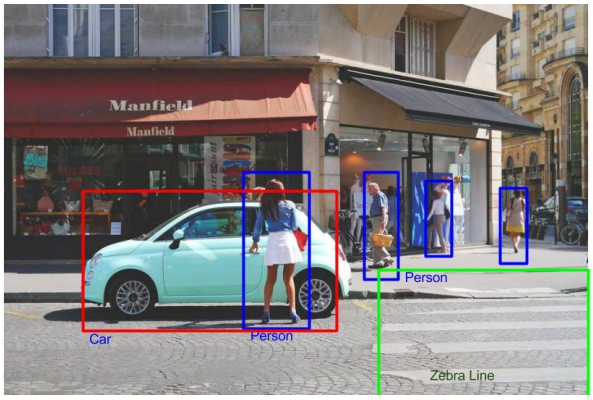

Figure 5: Qualitative Results of Street Road

V. CONCLUSION

Object detection using python and Open cv is the state of the art real-time object detection algorithm as it is much faster compared to other algorithms while being able to maintain a good accuracy. object detection is a crucial computer vision task that involves identifying and localizing objects of interest in images or videos.It ambitions to create an environment this is accommodating to a blind person's desires to make sure most consolation and performance for them in imminent instances. This model ambitions to reform how matters are accomplished and paintings on making better effects for coming generations. With the increasing availability of large-scale datasets and advances in deep learning techniques, object detection has seen significant progress in recent years and has many practical applications in fields such as autonomous driving, surveillance, and

robotics.

VI. REFERENCES

[1] M. B. Blaschko and C. H. Lampert. Learning to localize objects with structured output regression. In Computer Vision– ECCV 2008, pages 2–15. Springer, 2008.

[2] L. Bourdev and J. Malik. Poselets: Body part detectors trained using 3d human pose annotations. In International Conference on Computer Vision (ICCV), 2009.

[3] H. Cai, Q. Wu, T. Corradi, and P. Hall. The crossdepiction problem: Computer vision algorithms for recognising objects in artwork and in photographs. arXiv preprint arXiv:1505.00110, 2015.

[4] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on, volume 1, pages 886–893. IEEE, 2005.

[5] T. Dean, M. Ruzon, M. Segal, J. Shlens, S. Vijayanarasimhan, J. Yagnik, et al. Fast, accurate detection of 100,000 object classes on a single machine. In Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on, pages 1814–1821. IEEE, 2013.

[6] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. arXiv preprint arXiv:1310.1531, 2013.

[7] J. Dong, Q. Chen, S. Yan, and A. Yuille. Towards unified object detection and semantic segmentation. In Computer Vision–ECCV 2014, pages 299–314. Springer, 2014.

[8] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov. Scalable object detection using deep neural networks. In Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on, pages 2155–2162. IEEE, 2014.

[9] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. International Journal of Computer Vision, 111(1):98–136, Jan. 2015.

[10] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. IEEE Transactions on Pattern Analysis and Machine Intelligence, 32(9):1627–1645, 2010.