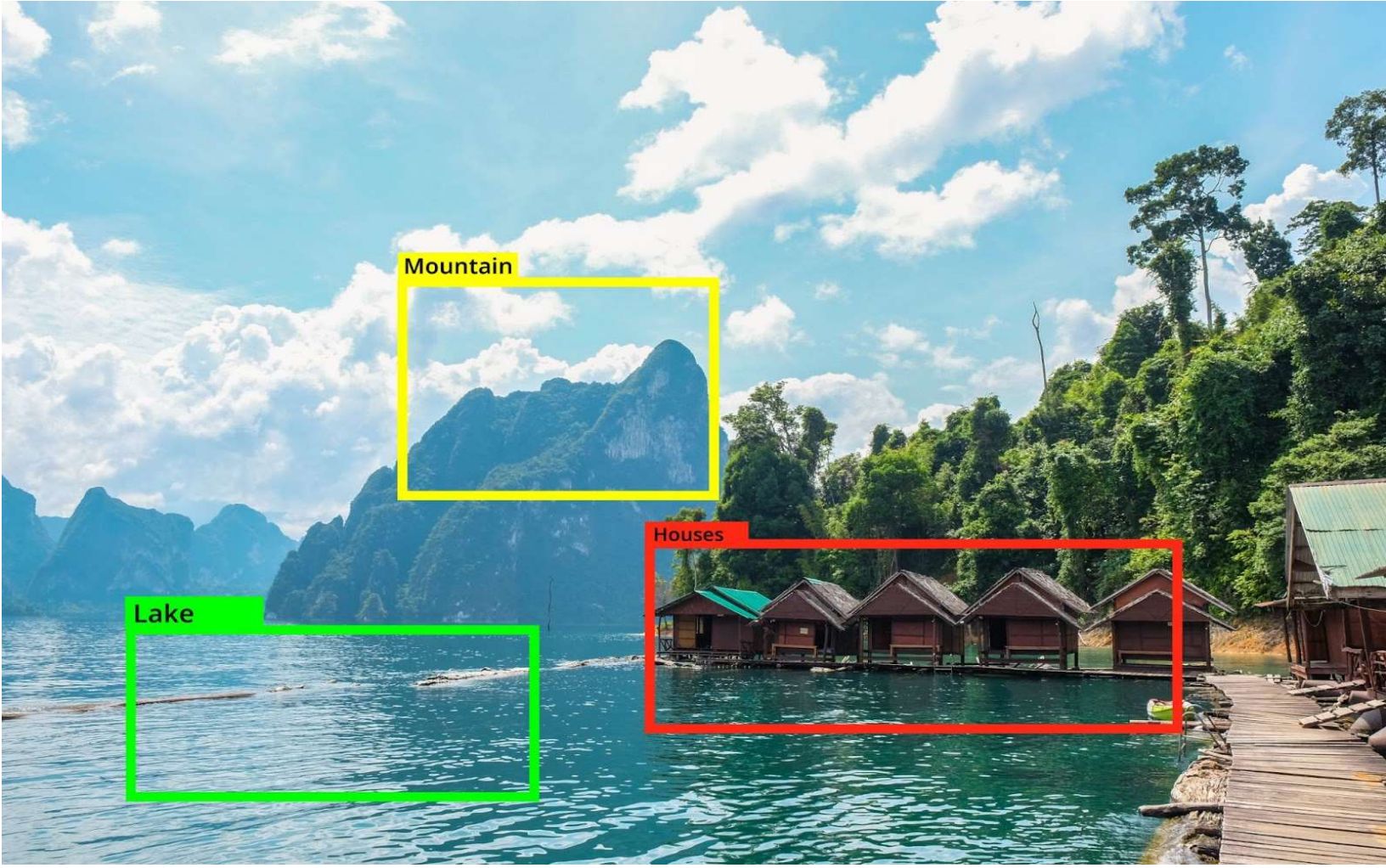


JOUN TECHNOLOGIES

Object Detection: Overview, Methods, and Applications

Garo Keuchkarian

Abstract

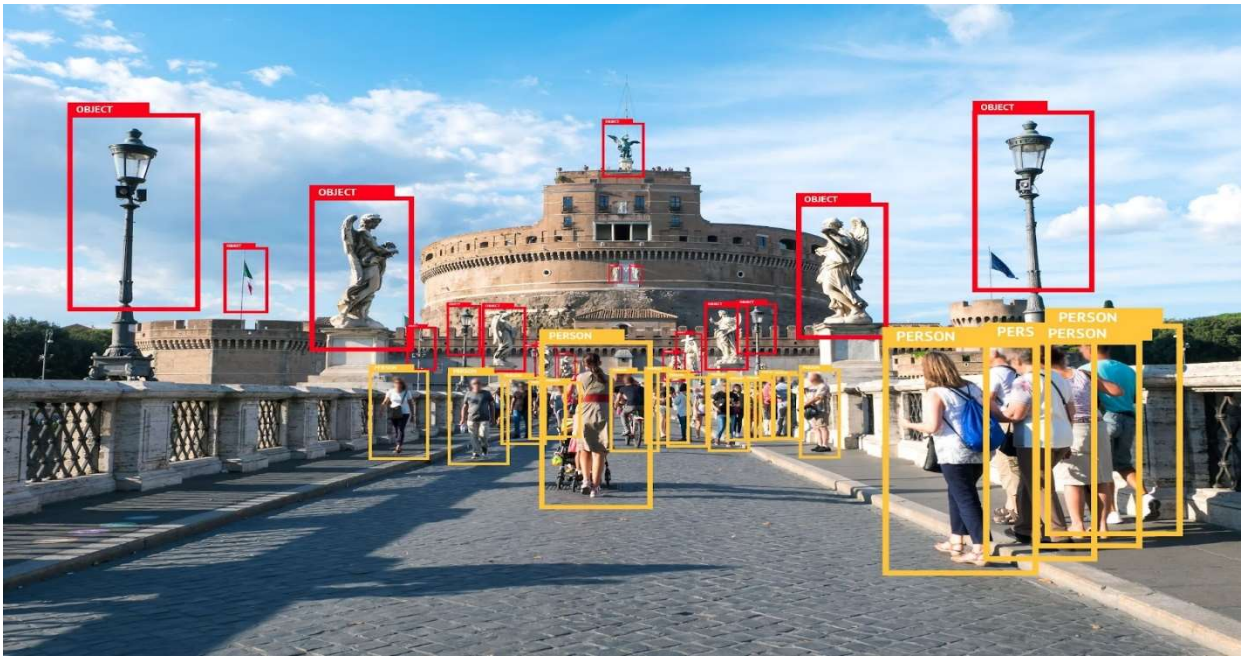
The advancement of artificial intelligence, machine learning and computer vision have further driven the integration of powerful reinforcement learning tools and algorithms in the process of object detection. Traditional object detection methods have now evolved using image processing and visual analysis. Modern object detection and tracking have become an essential part of current visual surveillance systems. This paper will introduce the importance of object detection across several applications and industries. In addition, it will provide a general overview on machine learning and deep learning. Moreover, it will discuss the different steps of object detection and compare the different reinforcement learning models and neural networks used. Finally, it will discuss how to combine all the above through the implementation of the techniques and the training of the model for object detection and human face detection.

Table of Contents:

Introduction:

The purpose of object detection is to determine the location of objects in a given image and to properly categorize the object. Object detection and tracking, as well as analyzing their movement, has been an area of significant research and development. Modern methods such as computer vision techniques allow for precisely locating, identifying, and tracking objects. Using advanced algorithms, even objects that are in motion can be detected and classified. [1]

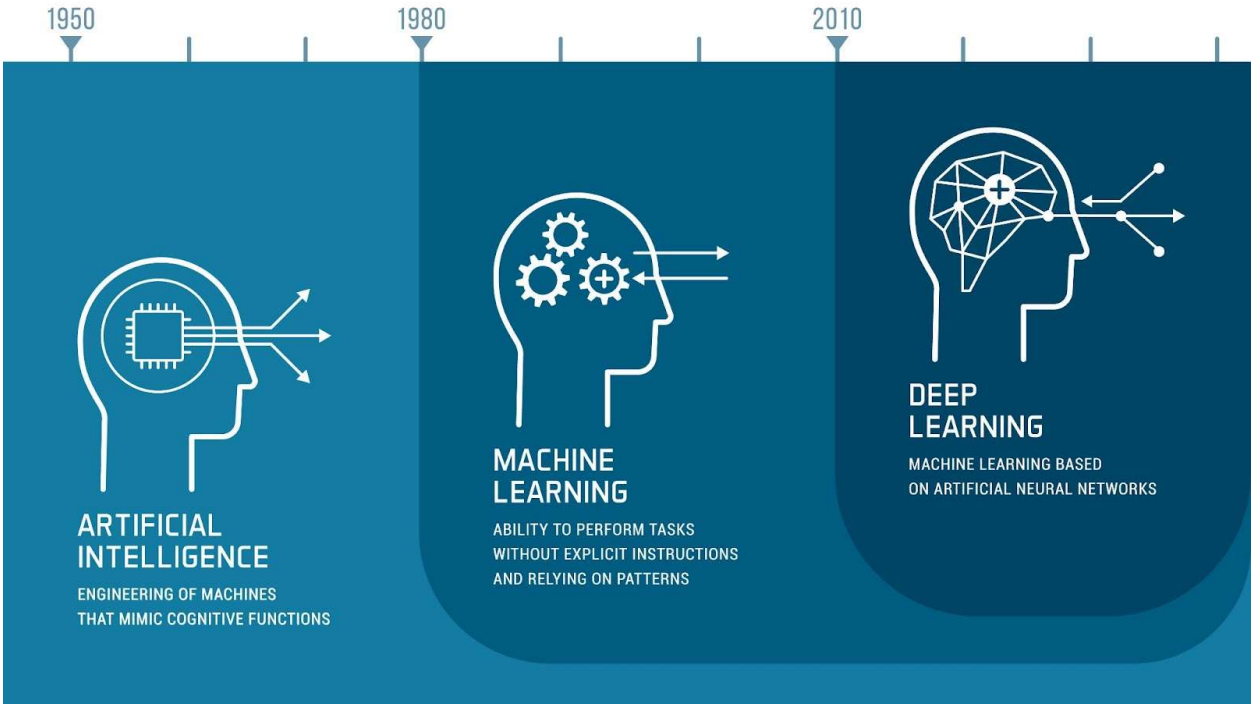
Object detection frameworks that are based on deep learning have been one of the most attractive research topics, mainly due to advanced learning abilities. The implementation of deep learning in object detection is valuable for a wide variety of industries. Human face detection is one such application. Automated surveillance, motion-based recognition, human-computer interaction, crowd counting, detection of anomaly (especially in healthcare and agriculture), vehicle navigation and traffic monitoring are among the top applications and uses of object detection.[2]



Classical and generic object detection models have been shown to be inefficient in handling the large data analysis and reconfigurations needed for these leading application implementations. With recent advancements of artificial intelligence, reinforcement learning techniques have further improved the performance of object detection and solved the inefficiency problem. These practices necessitate large number of iterations for training the generic data labeling methods. Let's look at reinforcement techniques more closely.

Reinforcement Learning Techniques:

Reinforcement learning agents can perceive and interpret their environment. Accordingly, they decide their upcoming actions by learning through experience and using a trial and error approach.



Machine Learning

Machine Learning is one of the most efficient and practical AI technologies, with applications ranging from self-driving cars to speech recognition. Machine Learning can be described as the science of enabling reinforcement learning for the computers. The goal is finding a mapping function *f*, for a given data *x*, that would predict the output $y=f(x)$ through automated training with data samples. [2] Machine Learning can be classified into 3 different categories of training methods: Supervised learning, unsupervised learning, and reinforcement learning. This object detection project will focus on reinforcement learning techniques.

Deep Learning

Deep Learning, also called deep structured learning, is a branch of machine learning that uses artificial neural networks to recognize complex patterns and develop modeling predictions. It uses a multi-layer approach to extract high level features from the data provided. These neural networks, which are composed of an input, an output, and several hidden layers, are constantly modified and improved. Deep Learning techniques have proven to be more efficient in image classification and computer vision, as well as in predictive analytics. Deep Learning algorithms are preferred for large datasets to allow for larger reinforcement learning capacity and avoiding overfit or underfit. The latter occurs when there is a mismatch between the size of the data training samples and the learning capacity of the agent. Hence, deep learning uses optimized neural networks to solve this mismatching problem.

Each layer of a Convolutional Neural Network (CNN) can be thought of as a feature map that is connected to its previous and succeeding layers. This hierarchical feature representation grants CNN the advantage of disentangling hidden patterns automatically from input data through the

non-linear multilevel structure. Another important advantage is its high learning capability which can be key to solving complex computer vision tasks. R-CNN is one of the most significant improvements of the CNN, since it allows for more complex and high-level feature extraction with enhanced precision and accuracy.[4] This has further modified into Fast R-CNN which optimizes classification and bounding box regressions, and then into Faster R-CNN which provides an additional layer of networks that generates region proposals. The most recent development has been into the different version of the YOLO (You Only Look Once) algorithm, which accomplishes object detection with image grids of a sing neuron layer.

Overview on Object Detection:

Object detection from a complex background is a challenging application in image processing. Each object in a set of images or videos is characterized by its own special features that are essential for classifying the object. Through computer vision, object detection provides relevant information and data on images and videos that is applicable to face recognition, autonomous driving, and image classification. Detection is performed in the image domain, then mapped to real-world coordinates.

Procedure

To achieve detection, objects that could be a potential target need to be localized and then classified into their specific categories through informative region selection, feature extraction, and classification. The first step is to scan the environment or the image and search for information that may be considered relevant. One risk is the generation of undesired or inefficient regions. The second step is to extract external visual features to represent robust and semantic representation of the objects scanned. The final step is to classify the targeted objects and categorize them

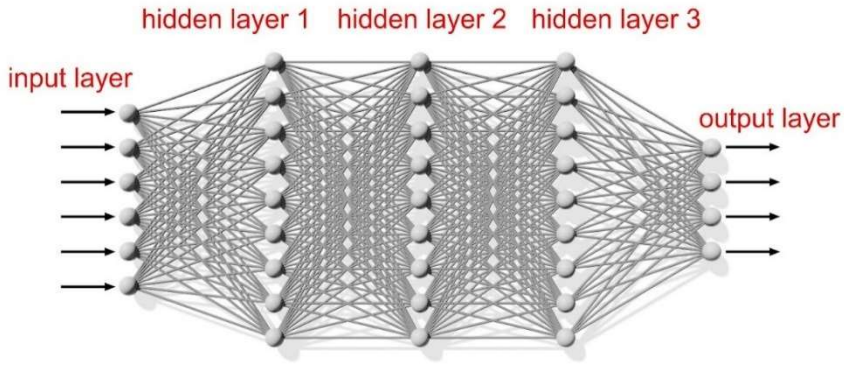
accordingly. Object classification is based on shape, motion, color and texture. Furthermore, with the significant improvements in the efficiency of the algorithms, object detection can be performed even in environments where the object is partially obstructed from the direct view.[5]

Object Detection with Deep Learning:

Object detection based on deep learning can be divided into two parts: encoder and decoder. The encoder scans the image through a series of layers and blocks. This is necessary to locate and label the targeted objects by extracting the statistical features and generate outputs. The decoder then takes the outputs and predicts the bounding boxes and labels for each of the objects.

Convolution Neural Networks

Deep Learning techniques implement CNN to generate end-to-end unsupervised object detection, where features are defined and extracted automatically. Let's explore the architecture of the CNN models and various algorithms. [6] CNN models are made up of neurons with learnable weights and biases. Each neuron is connected to several inputs, takes a weighted sum over the inputs and passes it through an activation function to generate the output into the next layer.



deep neural network

Region with CNN (R-CNN) avoids the slow and exhaustive search by using high computation performance to propose a selective search that can cover up to 2000 region proposal extraction from a given image. Although it solves the problem of localization of generic CNN, it is still considered a time-consuming search.

Fast R-CNN feeds the entire input image to the CNN, instead of the region proposals, and the feature map is generated. From the feature map, we identify the region proposals and feed them into a fully connected layer, and finally predict the classes and the bounding boxes of the objects detected. The advantage of Fast R-CNN is that each image undergoes convolution only once.

Faster R-CNN is faster than the previous algorithms since it uses a Region Proposed Network (RPN) rather than the usual selective search. After feeding the input image and obtaining the feature map, RPN generates a set of rectangular object proposals that scores the objects and computes the output. While the image test by R-CNN takes 50 seconds, it takes only 2 seconds per image by Fast R-CNN and 0.2 seconds by Faster R-CNN.

Single Shot Detector (SSD) is a single shot multi-box detector that is faster than the previous algorithms and improves accuracy during real-time processing. It involves first extracting feature maps followed by applying convolution filters to predict object classes and fixed size bounding boxes.

YOLO

The YOLO model performs as the name would suggest - “You Only Look Once” - to determine the classes and the location of the objects. It has a single neuron network based on regression and instead of image selection, it divides the input image into grids that predict the bounding boxes with their confidence scores. The benefit of YOLO is its ability to use any image of any size, in

which the YOLO model will output all the detectable objects in the database which it has been trained on. YOLO has multiple updated versions. To improve the performance and the accuracy of the detection, the four different variants have been developed. The first version uses a general architecture, whereas the second version refines the design and improves the bounding box proposals by manipulating the anchor boxes. Version 3 further refines the model architecture and training process and the fourth version has made significant advancements in the training speed of the datasets. The only drawback of using YOLO is that it faces difficulties in detecting very small objects.

Implementation:

Libraries Used

In my practical implementation, we decided to choose Python through Google Colab as the software to test my algorithms. This is not only because of its ease in practicality and syntax, but also because of its libraries that are essential for the object detection implementation using YOLO and CNN. Two of the most useful libraries in this project are OpenCV and NumPy. OpenCV (Open-Source Computer Vision) is an open-source computer vision and machine learning software library that provides a common infrastructure for applications and increases the use of machine perception in the commercial products that are used for object detection. [7] It provides the facility to the machine to recognize faces or objects. NumPy is a Python package for scientific computing and contains useful data structures such as arrays and matrices.

Algorithm

Based on accuracy in detection, distance measurement and desired processing speed, we decided to use both YOLO and CNN to train the datasets. YOLO was used to detect people or specific

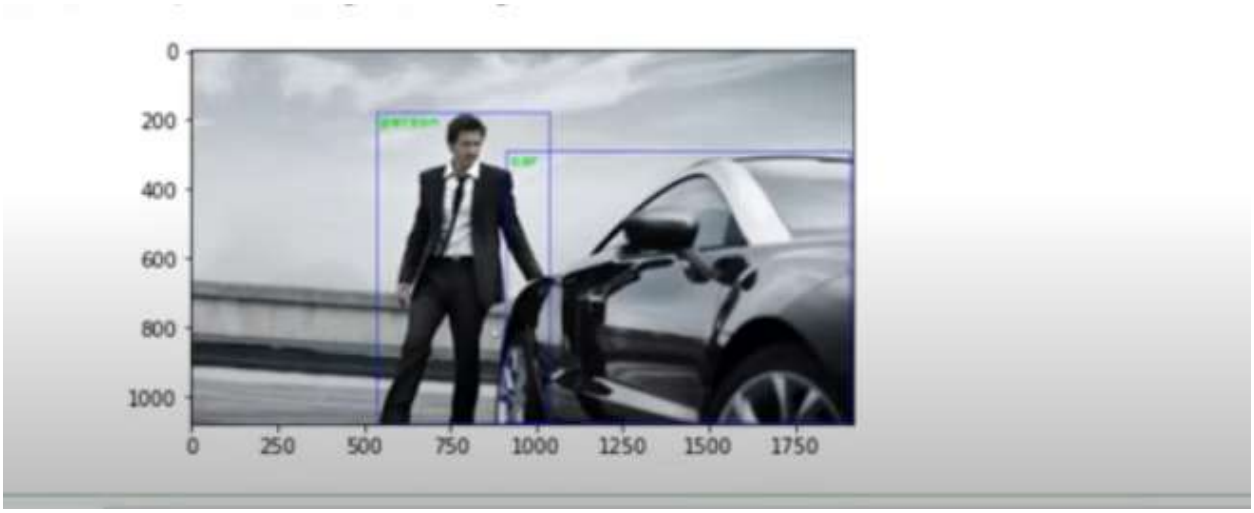
objects and animals, while CNN was used to define and model the input, hidden and output layers to meet certain specifications of the pretrained data. With respect to the modeled datasets, there are many available sources that are useful for the training algorithms and sets of images. Detection of people was achieved by inputting the feed of one of the cameras into a previously trained Yolov3-tiny dataset of images. The center of the resulting bounding boxes was used for Euclidean distance calculations and violations were detected by mapping real distances to pixel lengths. YOLO divides the input image of the objects into an S × S grid and then each grid cell is responsible for predicting the object centered in that grid cell. Each grid cell predicts several bounding boxes and their corresponding confidence scores. The confidence score of a bounding box, can be determined by the following equation: *Conf Score= P(Object)*PrConf* , which is the probability that the particular object exists multiplied by the confidence of the prediction made.

```
In [11]: confidence_scores={}
         class_id={}
         for out in outputlayer:
             for detect in out:
                 score= detect[x:]
                 class_id=np.argmax(score)
                 confidence=score[class_id]
                 if confidence>0.7:
                     center_x=int(detect[0]*width)
                     center_y=int(detect[0]*height)
                     wid=int(detect[0]*width)
                     hei=int(detect[0]*height)
                     x=int(center_x-wid/2)
                     y=int(center_y-hei/2)
```

Model Training

Every reinforcement learning software needs a dataset to predict output with known data. To complete the implementation, training and customizing the object detection datasets of the specific model was required. The first step of the training was the image collection and labeling. This was easily found in online resources in the case of images of common objects such as vehicles, buildings, household objects. Labeling was also be done through online tools such as MAKESENSE.AI. The next part of the training was data augmentation. This step generated multiple images with small variations and changes to improve the model. The changes can be easily customized by either resizing or flipping the object.

While training the model, files containing the weights and the biases were generated. When the training was complete, the algorithm saved the most desirable file based on the percentage of accuracy of the iteration. Through successive iterations in the reinforcement tools, the neural network improved the accuracy of its weight and biases, until it reached the output stage. This was key in generating the model, which was the final step.



In the above image, the man and car were detected and classified based on their corresponding confidence scores and accuracy percentage. This same concept could be implemented in other object detection and classification scenarios.

Human Face Detection:

With the spread of the COVID-19, we see many countries using drones to monitor if people are abiding by social distancing and face masks regulations (or recommendations) and aiding in contact tracing. Object Detection is at the heart of computer vision implementations in the industry today. To complete the detection, we used all the previously discussed algorithms and model training on the datasets.

Outcome

The algorithm we used included both CNN and YOLO. The CNN allowed the sequential neural networks with customized hidden layers such that its output layer determined if a certain required condition was met or not. In our implementation, the output showed the safety score, which is the probability that the person detected was socially distant (or not) from other people in the image. The YOLO algorithm was used to detect whether the people in the image were wearing masks. The input did not need any prior image processing, as they were passed to the model as is. The pretrained model was customized so that it detected only the people in the image. The output of this algorithm was similar to the input image with additional boxes surrounding the detected people and an array of the coordinates of these boxes. The model was tested and represented training accuracy of more than 90%. Here is an example of an output image, where the bad distances were represented by red lines and accepted distances were represented by blue lines.

Conclusion:

After using both YOLO and CNN algorithms and analyzing the results, several observations were made. The model relying on YOLO was not favorable in terms of depth perception. For the 2-dimensional images tested, it frequently produced false negatives on social distancing due to depth perception issues. The CNN model was more accurate in terms of depth perception, as it was trained on human-labelled data when determining the distance between people. However, in terms of reliability, the YOLO model exceeded the CNN model since it estimated the number of people

in the image and the distances between them and then determined how safe the environment was, making the results much more interpretable. The CNN model used a safety ratio calculated as an estimate of the probability the people in the image were socially distant from each other, rather than a direct safety score used by YOLO.

It is clear how these reinforcement techniques, with their massive improvements, have revolutionized generic object detection techniques. CNN and YOLO are two examples of how accurate, fast and efficient object detection has become in real-life applications. As we make further advancements in computing and artificial intelligence, object detection will have even more applications across many industries.

References:

[1] Nath, V., & Levinson, S. E. (2014). Machine learning. Autonomous Robotics and Deep Learning, 39–45. https://doi.org/10.1007/978-3-319-05603-6_6

[2] Singh, H. (2019). *Practical machine learning and image processing: For facial recognition, object detection, and pattern recognition using python*. Apress.

[3] *Object detection guide*. Fritz. (n.d.). https://www.fritz.ai/object-detection/.

[4] Vahab, A., Naik, M. S., Raikar, P. G. R. G., & R, P. S. (2019). Applications of Object Detection System. *International Research Journal of Engineering and Technology (IRJET)*, *06*(04).

[5] Akköse, O. (2020, November 24). *A review of object detection models*. Medium. https://medium.com/analytics-vidhya/a-review-of-object-detection-models-f575c515655c.

[6] Z. Zhao, P. Zheng, S. Xu and X. Wu, "Object Detection With Deep Learning: A Review," in *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3212-3232, Nov. 2019, doi: 10.1109/TNNLS.2018.2876865.

[7] Gupta, B., Chaube, A., Negi, A., & Goel, U. (2017). Study on object detection using open cv - python. *International Journal of Computer Applications*, *162*(8), 17–21. https://doi.org/10.5120/ijca2017913391

About the Author:

Garo Keuchkarian

Garo Keuchkarian is a senior Electrical and Computer Engineering student at the American University of Beirut with a minor in Economics. His interests include machine learning, data science and business intelligence.