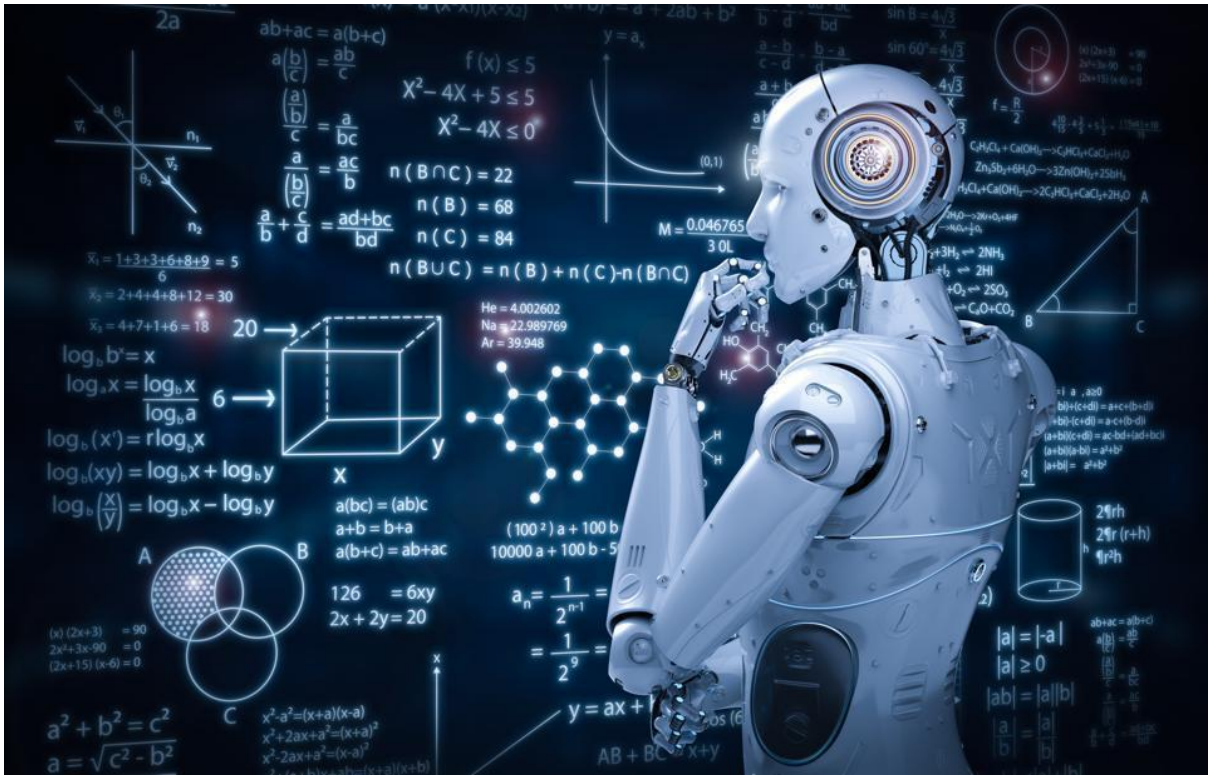

Image Detection and Classification using OpenCV and DNN

Report By: Aniruddh Bhandarkar

Table of Contents

Executive Summary

This project uses computer vision for object detection and classification. Computer vision is a field of artificial intelligence (AI) that enables computers and systems to derive meaningful information from digital images, videos and other visual inputs — and take actions or make recommendations based on that information. If AI enables computers to think, computer vision enables them to see, observe and understand. It is a subset of Machine Learning. Here we use OpenCV along with Tensor flow to enable detection and classification of images per input configuration either into a specific category per the classification file or an undetected file. OpenCV is a module in Python used for the purpose of computer vision and Tensor flow is used to leverage an underlying machine learning module to classify images.

Background

Machine Learning is a field in Artificial Intelligence that allows software applications to become predict outcomes more accurately using underlying reference models. Machine Learning is important because it helps give an insight into future probabilities on the basis of the existing data. This technology is used extensively by today's leading tech companies, such as Meta(Facebook),Google, Netflix, Uber.

Machine Learning is a growing field and has multiple applications in fields such as: Dynamic Parking, Translation Software, Autonomous Cars, Search Engines, Virtual Personal Assistants, Healthcare, Marketing, Sales and Finance.

Classes of Machine Learning:

1. Supervised Learning :In this type of learning, the users have to supply Algorithms with labeled training data and define the variables they want the algorithm to assess for correlations.
2. Unsupervised Learning: In this type of learning, the Algorithms trained on unlabeled data where the algorithms scan through the data to look for any meaningful connection.

3. Semi-supervised Learning: In this type of learning, there is a mix of the former two types of learning where the user may label the data and may explore data on its own.
4. Reinforcement Learning: In this type of learning, to teach a machine about multi-step processes and thus program on the algorithm. The algorithm decides on what steps it is going to take on its own.

In this project we have used semi supervised learning to process and classify images.

Model & Algorithm

OpenCV has a module called DNN(Deep Neural Network) which is used to develop a Neural Network. Open CV cannot be used for training hence in this project we have used TensorFlow framework to train the model. Other frameworks researched included frame Caffe, Pytorch, ONXX etc. The underlying learning model was built using the following steps:

1. Importing the necessary libraries(CV2,NumPy,MatplotLib)
2. Defining the various classes, required for classification
3. Initializing the DNN Module
4. Pre-processing the image
5. Use multi-label classification to initially train the model and then classify data

Processing Flow

- Preprocess the image by compressing the image to a standard HD size
- Train the underlying tensor flow model to classify the image as per the common nouns file
- Set a threshold for classification of the image to 0.45
- Upon classification verify if the image is dark or light i.e is there an object or not
- Categorize this image(if it exists) into one or many of the common nouns
- Compare using Matplotlib the most likely classification with the threshold

- If the threshold is above 0.45 and the highest value for a given noun then the image is classified in that category

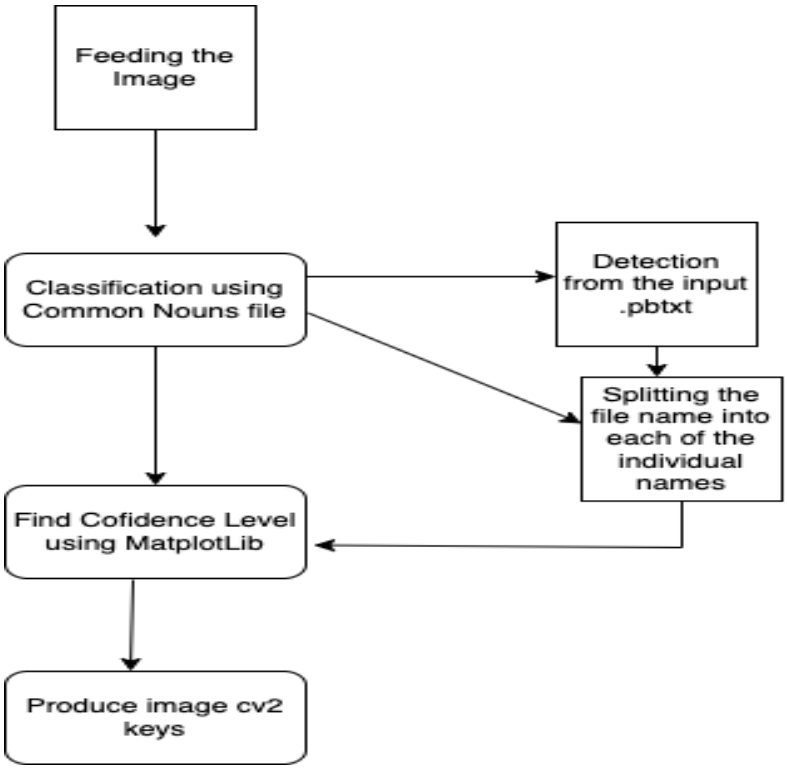

Figure 1: Algorithm for Classification

Figure 1 outlines the algorithm and processing flow of the machine learning model to classify images with a high confidence interval

Code Review

The image classification confidence interval is set at 0.45. Furthermore parameters like ConfigPath, WeightPath are used to define the text. Next leverage and expand pre-defined TensorFlow engine by adding class name as dark and light.

Write code to create a green box/ green text for the image. Flatten the box using Matplotlib. Use the confidence level in the image using the machine learning framework to classify the image based on configuration and weight path.

cv2.waitKey(5000) allows display to capture live image from the webcam feed. This is translated for multiple images using the following code shown in the image attached below.

Results

The code allows output for the images shown below. This is an array for cv2.waitKey

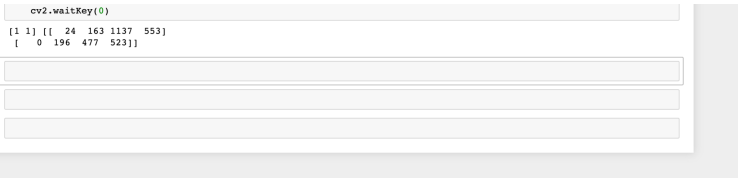

Figure 2: Array from the output of the code

Input	Output
Image from the Database	Array given adobe as well as the output in Figure 3

This is output for a image that is frozen because of the cv2.waitkey wait time which is 0 and detects the class of the object

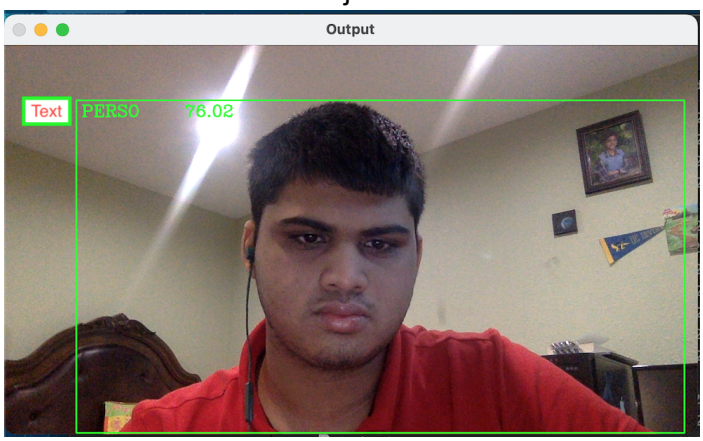

Figure 3

This is a sample output that was there for this code

Input	Output
Machine Learning framework	Image and box using Matplotlib

Figure 4: Sample Output for Dark/Light classification

Input	Output
Code using OpenCV	Image Classification for multiple objects

Figure 5: Output after fixing the code

Input	Output
Machine Learning(Tensor Flow)	Classification as Dark

Light Code

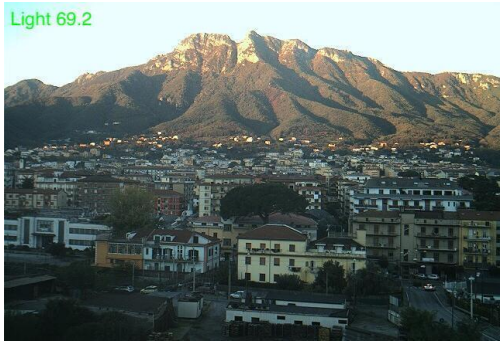

Figure 6: Light Output code

Input	Output
Code for MatplotLib and TensorFlow	Light Classification using Image

The code can be scaled for multiple images as the model will work for the images as shown in the output

1. The DNN module is used for Language Translation
2. The module is also used for Image Detection using Functions and Classification Learning
3. This is also used for supervised learning in this Project

Improvements

1. Running the code on better processing machines
2. Reducing the WaitKey time
3. Fixing the DNIM issue using better processing machines

Conclusions

The following was achieved from this Project:

1. Creation of image processing system
2. Using Matplotlib to plot the WebCam coordinates
3. Using RGB functions to draw a green box around the image captured
4. Successful using Machine Learning Tools to interpret the Confidence Level
5. Using Waitkey Function to display for an image

Bibliography

1. Day-night Dataset | Kaggle
2. What Is Machine Learning and Why Is It Important? (techtarget.com)
3. Deep Learning in OpenCV · opencv/opencv Wiki · GitHub
4. Deep dive into multi-label classification..! (With detailed Case Study) | by Kartik Nooney | Towards Data Science
5. Getting paths of each file of a directory into an Array in python - Stack Overflow
6. OpenCV: cv::dnn::Model Class Reference
7. OpenCV: Conversion of TensorFlow Classification Models and Launch with OpenCV Python

Appendix

```
import cv2
thres = 0.45 # Threshold to detect object

cap = cv2.VideoCapture(0)
cap.set(3,1280)
cap.set(4,720)
cap.set(10,70)

classNames= ['dark','light']
classFile = ['/Users/aniruddh/Desktop/Object_Detection_Files/coco.names']
with open(classFile,'rt') as f:
    classNames = f.read().rstrip('n').split('n')

configPath = 'ssd_mobilenet_v3_large_coco_2020_01_14.pbtxt'
weightsPath = 'frozen_inference_graph.pb'

net = cv2.dnn_DetectionModel(weightsPath,configPath)
net.setInputSize(320,320)
```

```
net.setInputScale(1.0/ 127.5)
net.setInputMean((127.5, 127.5, 127.5))
net.setInputSwapRB(True)

while True:
    success,img = cap.read()
    classIds, confs, bbox = net.detect(img,confThreshold=thres)
    print(classIds,bbox)

    if len(classIds) != 0:
        for classId, confidence,box in zip(classIds.flatten(),confs.flatten(),bbox):


            cv2.rectangle(img,box,color=(0,255,0),thickness=2)
            cv2.putText(img,classNames[classId-1].upper(),(box[0]+10,box[1]+30),
                        cv2.FONT_HERSHEY_COMPLEX,1,(0,255,0),2)

    cv2.imshow("Output",img)
    cv2.waitKey(5000)
```