Image processing and object detection with Recommender System for Marketing Analytics

Amrit Madhav (TA17012)
Project Report

Abstract:

Enabling intelligent devices/cameras to accurately detect and identify the objects inside an image/video and recommending the customer to buy similar products which other customers have bought. This kind of recommendation will help retailer to boost their profit margins.

Introduction:

In this report, I present an object detection approach for low quality images/videos or images from the streaming videos and recommending the customer to buy other products too. The models are integrated with still images and webcam videos and evaluated on the accuracy of the identified image and then recommending user with most associated products.

Computer vision and recommender system has recently been greatly integrated into marketing, robotics and automation.

Locating and identifying multiple items in an image is something that is still difficult for machines to accomplish. However, significant work has been made in the last few years on object detection with convolutional neural networks (CNNs).

In this project, I have applied CNNs for object detection in Images/streaming videos/video, with the aim of creating a detector that can identify the most important objects in an image/video.

The input to my detector is an unprocessed low-quality image or video. Depending on the application i have used YOLOv3 to detect and classify/identify the objects in each image so that i can then rank each object in terms of importance and apply image processing filters to make the most important objects stand out. The output of my detector is the processed photo or video.

After this detection my Recommender Models invokes using inputs from the Detector model in which it takes sequential input from the Detector Model and applies Appriori Algorithm then provide recommendation to the customer.

Lots of research has been done on recommendation using Text data, this application will provide recommendation on the basis of Objects detected from the Image/ streaming videos/Video. This application will have a major use in Retail Industry.

As an Instance of the utility of this detector and recommender system, detector would be able to process the clicked image from the camera/sensor and detect the image and provide recommendation to customer. Moreover, this application can also identify streaming objects.

So, to accomplish this goal I have to address several challenges. First challenge I faced is in Identifying the objects from any image/streaming video/video. Second challenge creating a detector which can pass detected objects value to a recommender system. Third challenge which I faced is my recommender should be smart enough to take sequentially all objects from detector model and recommend the customer to buy other products also.

Related Work:

Most of the Early Deep Learning based object detection algorithms like the R-CNN and Fast R-CNN used a method called Selective Search to narrow down the number of bounding boxes that the algorithm had to test whose performance is not good. Moreover, another approach called Overfeat involved scanning the image at multiple scales using sliding windows-like mechanisms done convolutionally. This model creates many CNN, which leads to delay in prediction of the object.

So, to allow my detector to work for variety of different image or videos applications. I looked at YOLOv3 and Faster R-CNN.As YOLO is an improved model over Faster R-CNN with respect to faster processing speed.

YOLO runs more than 150 frames per second which makes it the best choice for processing Video streams too.

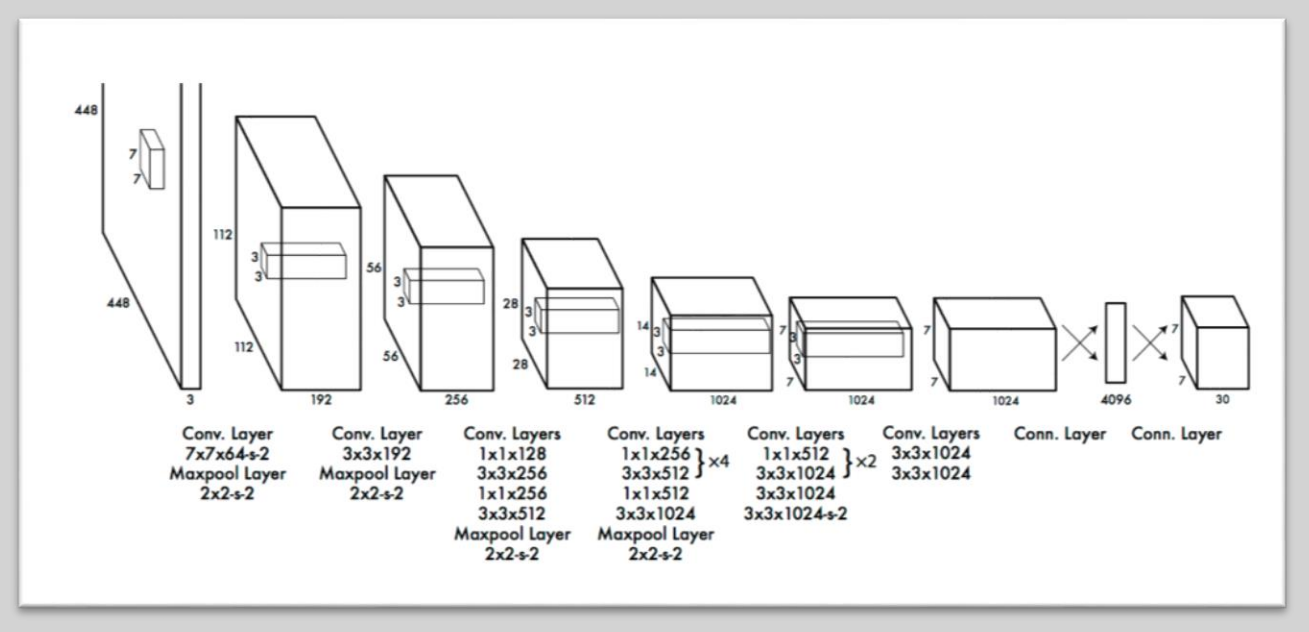
YOLO Algorithm:

YOLO being an improved model over Faster R-CNN and it is as accurate as SSD and three times faster.

YOLO Design and Architecture:

YOLO uses unified detection which unites the separate components of object detection into a single neural network. This Neural network uses features from the entire image to predict each bounding box. This design differentiates YOLO from other methods and enables end-to-end training with realtime speeds while maintaining high average precision.

YOLOv3 Architecture:



Yolo's Unified detection system uses S x S grid cells , each grid cell predicts only one object it predicts 'B' bounding boxes for each cells.

Each bounding boxes produces '5' predictions: x, y, w, h, and confidence. The (x, y) is the coordinates of the center of box relative to the bounds of the grid cell. The (w, h) is the width and height of the bounding box. The confidence score is calculated by the probability of the object multiplied by the intersection of the union of ground truth and prediction labels. Then each grid cell produce the conditional class probabilities. These probabilities are used at test time to produce the final predictions. The final class-specific confidence scores are calculated by multiplying the conditional class probabilities with the individual box confidence predictions.

The neural network architectural design includes two main components: feature extraction and prediction. The feature extraction consists of 24 convolutional layers to obtain the final 7x7x1024 activation map. This image information is passed to prediction section that has 2 fully connected layers.

Furthermore, on YOLOv3 uses 13×13 grids (S×S), 2 boundary boxes (B) and 80 classes (C).YOLOv3 makes 3 predictions per location. Each prediction composes of a boundary box, a objectness and 80 class scores, i.e. N × N × [3 × (4 + 1 + 80)] predictions.

Mathematics:

Confidence here refers to the probability an object exists in each bounding box and is defined as:

C = Pr(Object) * $IOU \begin{cases} truth, \\ pred, \end{cases}$

The loss function is used to correct the center and the bounding box of each prediction.

Apriori Algorithm:

Association analysis using the Apriori algorithm to derive rules of the form {A} -> {B}. Here in the project, I have leveraged the Apriori algorithm to generate simple {A} -> {B,C,D} association rules.

Apriori is an algorithm used to identify frequent item sets. It does so use a "bottom up" approach, first identifying individual items that satisfy a minimum occurence threshold. It then extends the item set, adding one item at a time and checking if the resulting item set still satisfies the specified threshold. The algorithm stops when there are no more items to add that meet the minimum occurrence requirement. By using Apriori I prepared the item set.

Association Rules Mining: Once the item sets have been generated using Apriori ,I started mining association rules . There are 3 key metrics to consider while evaluating association rules.

1. **Support:** This is the percentage of orders that contains the item set. It means , number of transactions containing a particular item divided by the total number of transactions: Ex as per my dataset refrigerator is in 40 transactions out of 369,therefore support = 0.1084 = 10.84% . So, the minimum support threshold required by Apriori can be set based on requirement and domain knowledge. As my data is of retail so I have kept it as very low.

2. **Confidence:** Confidence refers to the likelihood that an item 'B' is also bought if item 'A' is bought. It can be calculated by finding the number of transactions where 'A' and 'B' are bought together, divided by the total number of transactions where 'A' is bought.
   Mathematically, it can be represented as:
   Confidence(A -> B) = (Transactions containing both (A and B))/(Transactions containing A)

3. **Lift:** Lift refers to the increase in the ratio of the sale of B when A is sold. Lift(A -> B) can be calculated by dividing Confidence(A -> B) divided by Support(B).
   Mathematically it can be represented as:
   Lift(A->B) = (Confidence (A->B))/(Support (B))

So, in my project for developing Recommender system I have used below steps:

As my Datasets is large I have filtered out my larger dataset, by putting some minimum support and confidence on the Appriori output by taking only those rules which has support greater than min_support. This means that I am only interested in finding rules for the items that have certain default existence. Moreover, selecting those items which have a minimum co-occurrence with other items(Confidence).

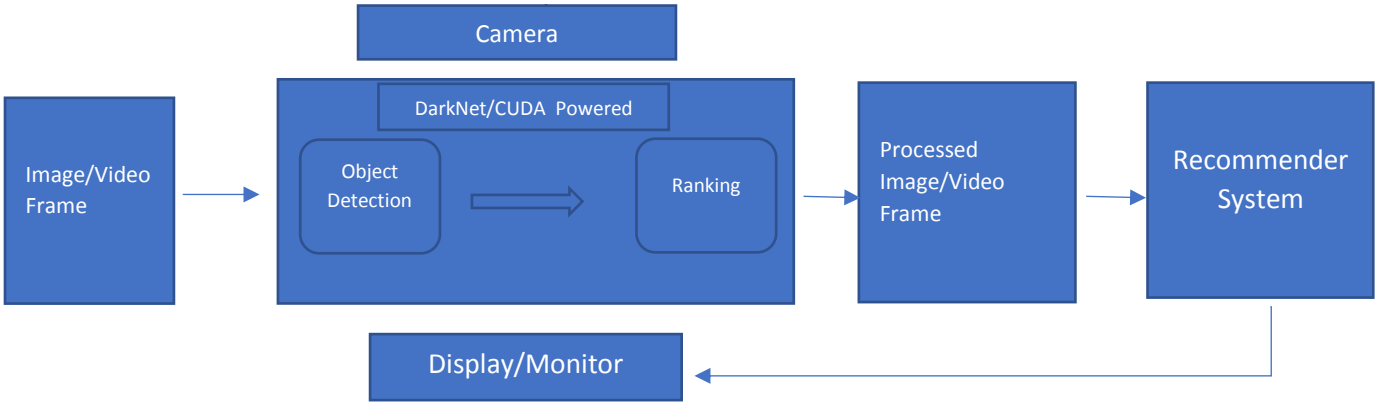
I have also calculated minimum length of the Rule through which I only pick those rules which is greater than the specified value, I have taken minimum length as 2.

Along with other constraint I have put in a separate constraint which filter out all the rules which has the identified object name in the rule set. By doing this I selected all the rules which has object name in the

set. Later on, at the end I give customer only those recommended products which are unique across all the set of Rules.

I have used this datasets for creating rules from the Apriori algorithm. [“retail_store_data.csv”]

System Diagram:



RESULTS: Conclusion and Discussion:

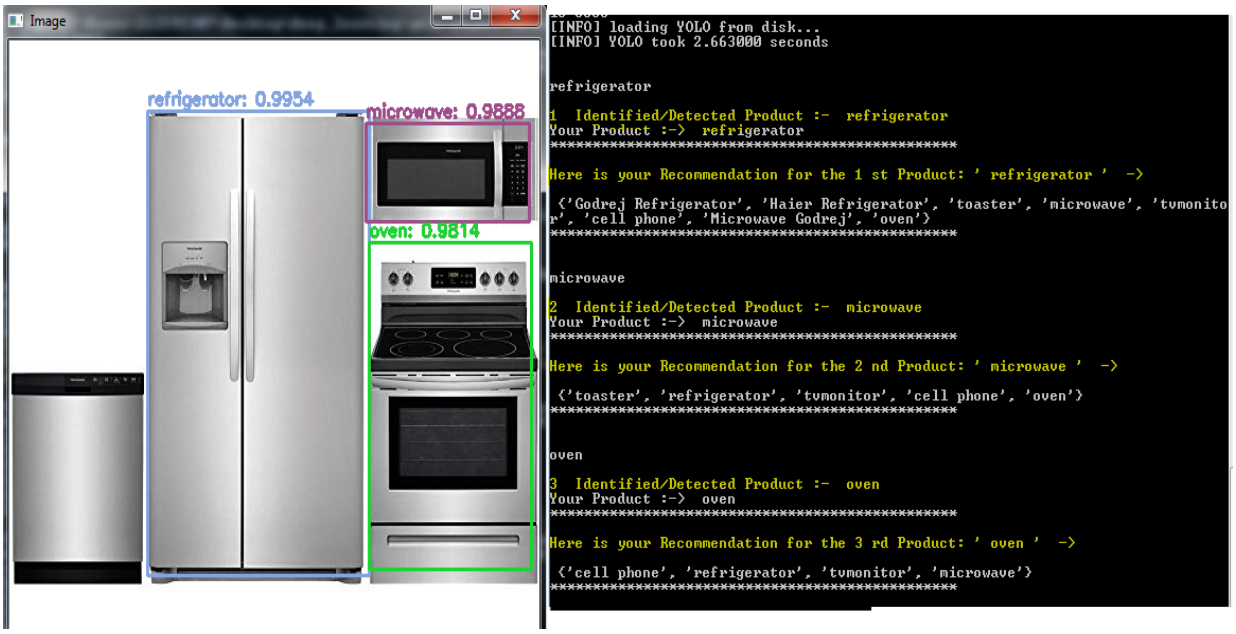
In all the below Scenario, you will observe that the Model has identified the objects inside the image and has the confidence also displayed. Once all the Objects identified the Recommender picks each object and provides the Recommendation to the Customer.

Scenario 1:

Passed in the Image having Refrigerator,Microwave,Oven to the YOLOv3 Model -> Recommender took the Object name and provided the recommendation for all the identified products.

Recommender took all identified objects from the image sequentially and iteratively and Recommended the product for the same.

Left hand side is the output from YOLOv3 Detector Model and Right-hand side is the Recommender Model recommendations for each identified products(3 products identified-refrigerator,oven,microwave).



Scenario 2:

Passed in the Image having sofa,tvmonitor,vase,chair,pottedplant to the YOLOv3 Model -> Recommender took the Object name and provided the recommendation for all the identified products.

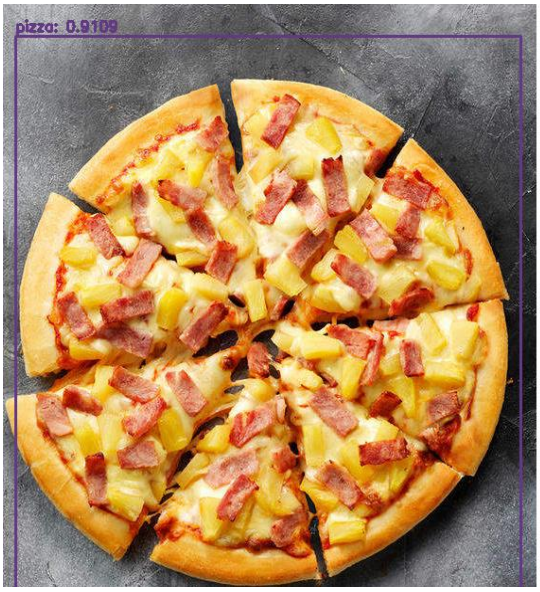
Recommender runs sequentially and iteratively on the identified objects and provided the Recommendation for all the products.

Scenario 3:

Passed in the Image having Hot dog to the YOLOv3 Model -> Recommender took the Object name and provided the recommendation for the identified product.

Scenario 4:

Passed in the Image having Pizza to the YOLOv3 Model -> Recommender took the Object name and provided the recommendation for identified products.



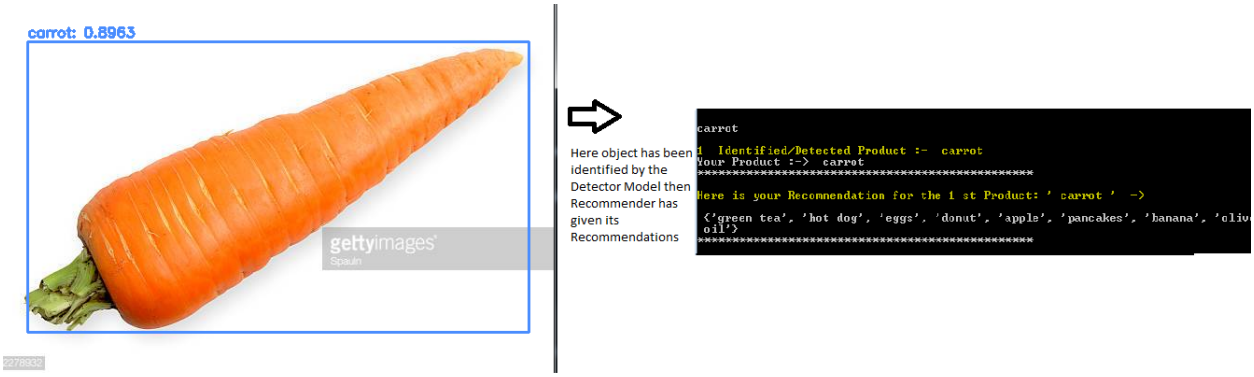
Recommendation for the Object detected (Pizza)

```
pizza
1  Identified/Detected Product :-  pizza
Your Product :->  pizza
**************************************************

Here is your Recommendation for the 1 st Product: ' pizza '  ->
 {'eggs', 'sandwich', 'green tea'}
**************************************************
```

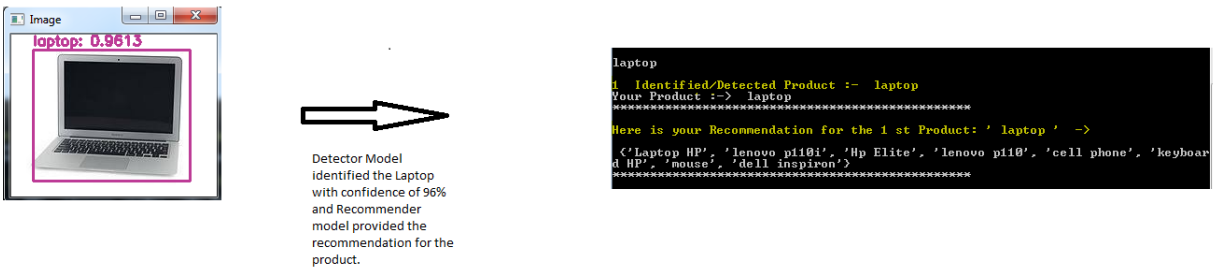
Scenario 5:

Passed in the Image having Carrot to the YOLOv3 Model -> Recommender took the Object name and provided the recommendation for identified products.



Scenario 6:

Passed in the Image having Laptop to the YOLOv3 Model -> Recommender took the Object name and provided the recommendation for identified products.

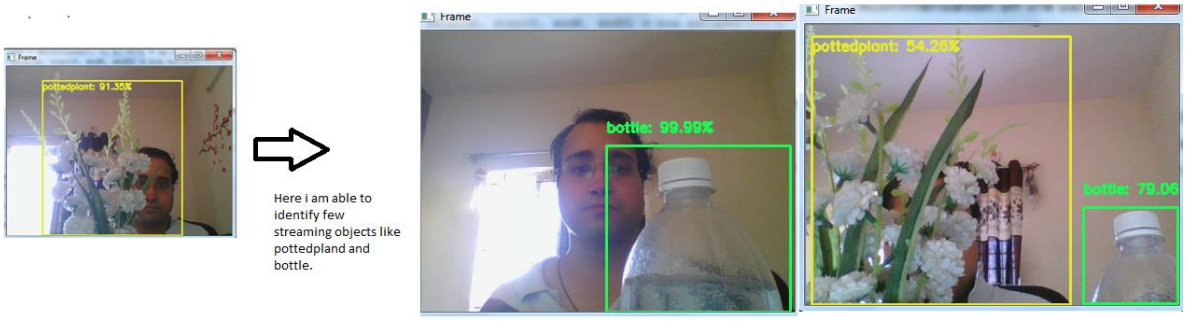


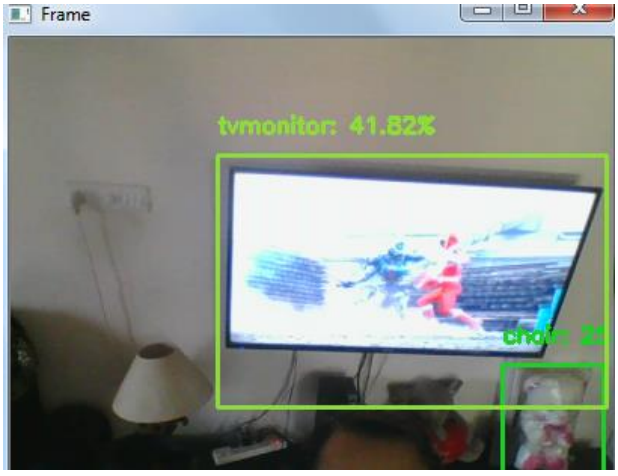
Scenario 7: Streaming Objects Detection

Here in this Model I am able to identify few streaming objects too, using my Webcam.

Streaming objects like Pottedplant and bottle and TV monitor.

As streaming objects detection model as of now is not must robust. So, I am not using this for Recommendation purpose. I have kept this scenario for more research and development





References:

- http://cs231n.stanford.edu/reports/2016/pdfs/287_Report.pdf
- http://cs231n.stanford.edu/slides/2016/winter1516_lecture8.pdf

- http://cs231n.stanford.edu/slides/2017/cs231n_2017_lecture11.pdf

- J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei- Fei. Imagenet: A large-scale hierarchical image database. In Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, pages 248–255. IEEE, 2009