Real Time Object Detection for Self Driving Cars using Convolutional Networks

Kunal Roy
DePaul University, College of Computing
CSC 578

1. Introduction

Self driving cars will increasingly become a very important aspect in our lives, lots of major car companies and well known engineers and businessmen are placing big bets on it to be the technology of the future. One of the reason it is being touted as something that could be the future is that once the algorithms behind it have matured, the system will be safer than a human driver and therefore self driving cars can reduce the amount of road accidents. Many people are tired due to lack of sleep, have had a few drinks at a party and therefore can often endanger the lives of other people and themselves. Self driving cars greatly reduce the chances of that happening. Another reason for its current popularity and future promise is that self driving cars will usually be electric and therefore that will greatly reduce the emissions from vehicles into our atmosphere. This will attempt to reduce the effect of green gas emissions on our atmosphere and climate change.

Self driving cars work in 4 main stages, they first have to get good at perceiving the world around them, then they have to localize objects in images, then the trained model has to make predictions on new images and then finally they have to make decisions through motion control of the car. In this paper we will be talking about perception, localization and prediction. Tools that are required for the car to be good at perceiving the world around us are cameras and sensors. Cameras provide us with the images we need to classify them and the objects that are in them so that the car can maneuver accordingly. We need sensors so we not only identify objects in our surrounding world but we also need to know the distance between our car and surrounding objects. In the next few sections of this paper we will discuss how exactly the perception works through object localization and detection in images which are fed through a convolutional neural network (CNN), a network which is highly optimized for processing and recognizing images.

2. Architecture

Before we get into object localization and detection, I want to overview the architecture of a CNN (Figure 1) which serves as the basis of our ability to detect objects in an image. A CNN architecture starts with an input image that is fed into a convolutional layer, in this layer a sliding filter is run over the image to perform convolution operations on it, and then it maps the results from that on to a feature map. The convolution operation is a mathematical operation that amounts to combining the affects from two entities. The feature map is then fed into a max pooling layer, where the maximum value pixels are selected in sliding windows to again reduce the size of the feature map to hold only the pixels with the highest value to capture the main information. There can be a number of filters used on the image to capture different aspects of

the image such as horizontal lines , vertical lines etc. After the max pool layer, the feature map is flattened into a one dimensional vector and inputted into a fully connected layer which is the same as a feed forward neural network with activation functions. This fully connected layer does the job of learning the target feature map from the input feature map by back propagation through adjusting the wrights after each epoch until the cost function is minimized. For a regular classification task the output layer may just have one unit which basically holds a 1 or 0 to indicate if the object it was trained to spot is there or not. For the task of object localization and detection, the output layer is changed so that there are more number of outputs that include the coordinates of the bounding box around the object that was spotted and the class of the object.

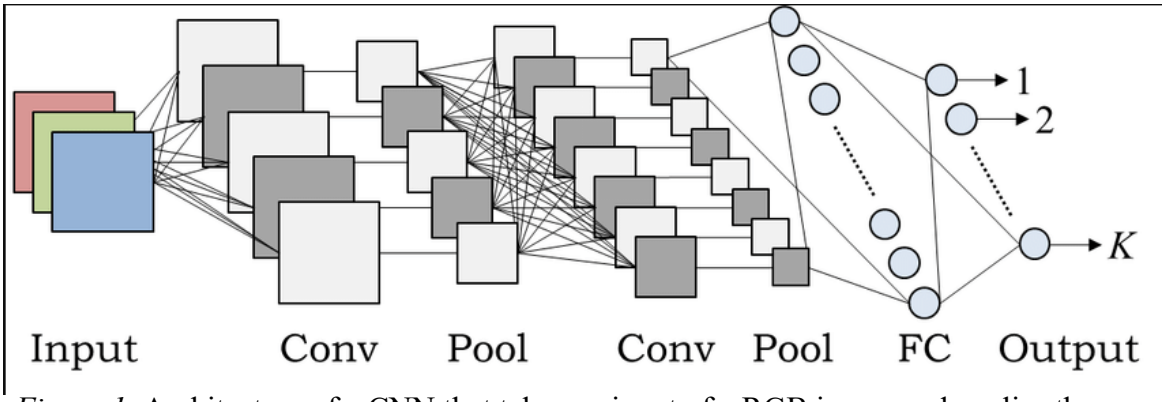

Figure 1: Architecture of a CNN that takes an input of a RGB image and applies the convolutional layer and pooling layers to it twice before sending it to fully connected layer

3. Object Detection

Before I get into object detection, I want to start off by talking about object localization. As I mentioned before, the output of a neural network can be changed to include the outputs b_x, b_y, b_h, b_w where b_x and b_y are the midpoint of the box, b_h and b_w are the height and width of the bounding box. The output target vector y includes these bounding box dimensions and an output to indicate whether the object is present or not with p_c and an output for the class of the object with c as shown in figure 2. One can specify the ideal dimensions beforehand so that one can use supervised learning to learn these ideal dimensions from the target output y. The feature map must output these bonding box coordinates so that the fully connected layers can learn them. One can use the squared error cost function on the output vector and the target vector and then this cost function will be minimized in order to learn the box dimension and localize the object.

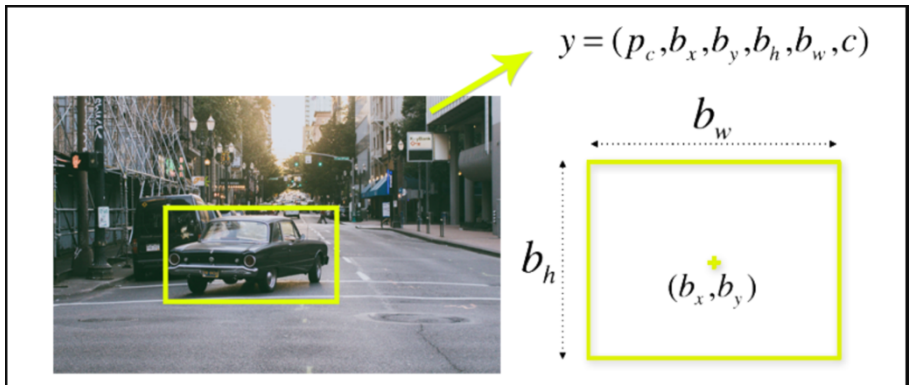

Figure 2: A picture with a bounding box around the object to be localized, with its coordinates and midpoint put into a target vector y to be learnt

Now I can get into object detection and the approach that we can use for this to start off is the sliding window approach. In this approach, we train a CNN on a labeled training set of cropped objects that we want to find, so it should just be the object itself, no background. Then you can use that trained CNN on the labeled cropped images to feed into it sliding windows over an image with a background and everything including the object itself. You keep sliding that window and inputting that into the CNN to see if it detects the object that it was trained to identify. The step sizes have to be small so the number of images fed into the CNN sequentially end up being a lot. On top of that, larger and larger windows have to be tried in order to capture any size of object. There is a huge computational cost associated with this approach as you have to feed in a lot of windows into the CNN sequentially until you find the objects that you are looking for. This will not be very helpful when you are trying to predict objects in images in real time while in a self driving car that is trying to navigate through the streets. The solution to this slow computation is that the fully connected layer in the CNN architecture must be turned into convolutional layers. In this approach you can input an image into the CNN and in one pass it will convolutionally do the sliding window approach and combine the results of it in the output. Instead of doing the sliding window approach by sequentially inputting the windows into the CNN every time, this is approach makes it possible to do it one pass, greatly reducing the compute power needed. Another approach to object detection can be found in the YOLO algorithm that stands for you only look once. To start with this approach, you lay down a grid on an image, say a 3 x 3 grid with 9 cells and for each cell you will have a label vector y as show in figure 3. The CNN architecture is modified so that the output target vector is a 3 x 3 x 8 for the 8 values in the target vector y and for each of the cell in the grid, this target vector is used. Then like before, you would use back propagation to learn this modified target. When you feed in a new image and propagate it forward, the output will be the 8-element long output vector for each of the cells in the grid. You can also use finer grids to make detection is more accurate to say a 19 x 19 grid. This approach also has the convolution implementation and therefore executes this algorithm in one pass. Therefore this algorithm runs very fast and is used for real time prediction.

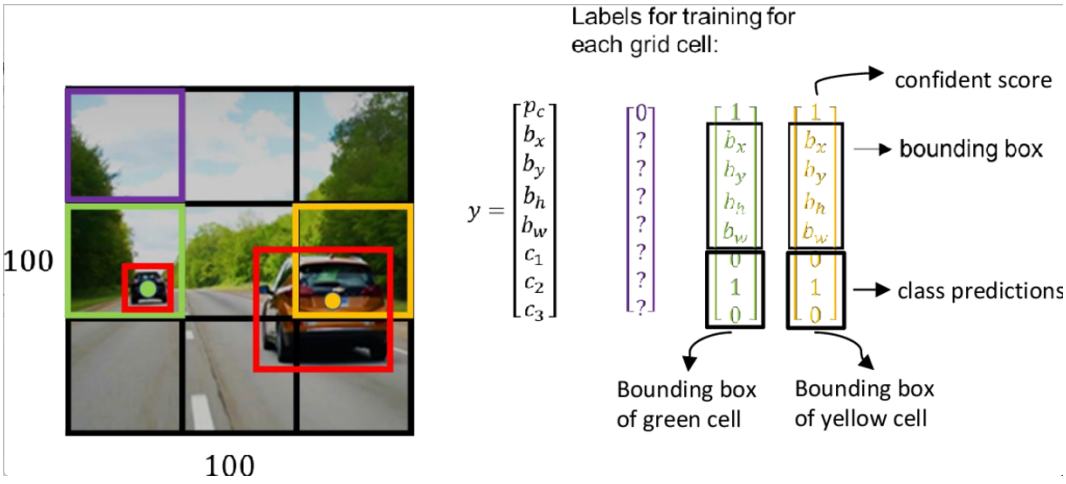

Figure 3: A 3 x 3 grid laid down on an image and each cell of the grid is assigned an output target vector y

4. Conclusion

Overall, as you can see the current state of object detection has come a long way. Back in the day a lot of it was very slow and inaccurate. Now however, new algorithms like YOLO have made it possible that they are implemented quickly. Also, with the advancement in chips and computational power, it has made it possible that larger and larger training data sets be used to train these models. Larger training data sets means the models can learn patterns from more and more cases that will eventually lead to more accurate models as well. Recently, the automotive company Tesla launched their new AI chip called the Dojo chip which is specifically optimized to train neural networks and is touted to have very high compute power. This will only enable self driving to get more accurate as the training data and the networks can get more complex and bigger. This will enable us as a society to have more trust in self driving cars and therefore help humanity transition from petrol cars to electric and self driving cars.

References

"Self-Driving Cars With Convolutional Neural Networks (CNN) - neptune.ai." *Neptune.ai | Metadata Store for MLOps.* Web. <http://neptune.ai/blog/self-driving-cars-with-convolutional-neural-networks-cnn>.

Stanford University. Web. <http://web.stanford.edu/class/cs231a/prev_projects_2016/object-detection-autonomous.pdf>.

"C4W3L01 Object Localization - YouTube." Web. <http://www.youtube.com/watch?v=GSwYGkTfOKk&list=PL_IHmaMAvkVxdDOBRg2CbcJBq9SY7ZUvs&index=2>.