Predicting vehicles' position and angles from single image of road using deep learning

A PROJECT WRITEUP
BY

Abdul Samad

April 2020

Abstract

This paper presents an approach that has been tested to find rotational (yaw, pitch, roll) and translational angles (x, y, z) from collection of single images; frames of video recorded from camera mounted on a vehicle during drive through urban environment. Using deep learning application in two distinct stages, a pretrained YOLOv3 network is applied to find and extract cars present in the picture and then extracted images alongwith their coordinates in original image are fed into Convolutional Neural Network to learn and later predict angles in different settings. During sensitivity analysis, the algorithm achieves 57.3% CSI (Critical Success Index) (Schaefer 1990) during extraction stage, results yet need to be combined for MAP (Mean Absolute Precision).

Contents

List of Figures

1 Introduction

The ubiquity of self-driving cars has a long way to go, considering criticality of fail-safe systems that are also not unreasonably expensive. Compared to LIDAR, cameras currently present a very affordable workaround because of their existing prevalence. However, there still needs to be enough computer vision algorithms, hence many problems to solve, to make a sound argument against a LIDAR data augmentation requirement. One of these problems is determining the distance to each vehicle, its direction and the way it is oriented in space at a given time. Robustness of this task requires predicting these values from a single image (a single frame of a continuous video). Later on, this can be also used to validate with the recurrent feed, for which few well performing algorithms already exist where predicting direction and orientation is inferred from previous states.

2 Background

YOLO (Redmon and Farhadi 2018) has been a popular algorithm to perform fast object detection which has its weights pretrained on MS COCO dataset (Lin, Maire, et al. 2014), containing 1.5 million instances and 80 categories. The algorithm can be limited to certain categories for an application such as vehicles in our case. Similarly, RetinaNet (Lin, Goyal, et al. 2017) also trained on MS COCO dataset has shown promising results in quick bounding-box based object detection algorithms.

3 Implementation

3.1 Data pre-processing

1. Mask far away vehicles using provided binary masks dataset,
2. Mask objects that can cause issues with the first stage (extraction/detection), such as own car's bonnet;
3. Remove images from dataset that have no possibility of any detection at all, yet have come labelled. These can interfere with CSI (Critical Success Index) calculation later when FNs (False Negatives) will always exist.

3.2 Detection / Extraction

1. Identify number of cars present in image using pretrained models such as YOLO and RetinaNet over randomly sampled data from complete images dataset (100 out of 4262 full images),
2. Plot CSI for both models with different DPTs (Detection Probability Thresholds) (Schaefer 1990),
3. Choose the best network and DPT for future extractions based on CSI score,
4. Extract images of objects (vehicles) detected as bounding-boxes in complete image for all images.

3.3 Prediction

For our extraction stage, CNN with two convolutional layers followed by two pooling (max) layers respectively are combined with MLP (MultiLayerPerceptron) hidden layers. The input sizes are matched with lowest of squared extracted images, and output units for each layer are gradually decreased until the output layer which matches with the number of classes (six DOF angles, three translational + three rotational).

1. Start with basic CNN (Convolutional Neural Network) model 3.1, map fixed square sized normalized image to 6 angle outputs,
2. Train on detection/extraction database from previous stage with shuffling,
3. Validate on 10% dataset (without replacement).

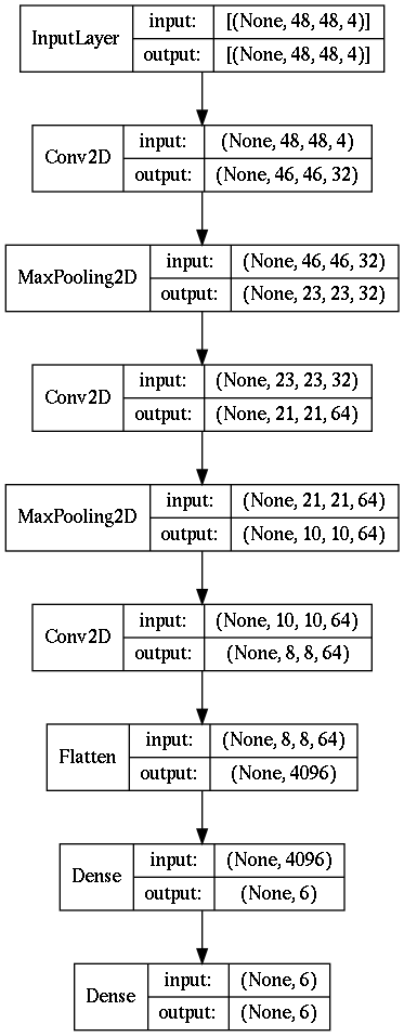

Figure 3.1: Basic model for prediction stage

4 Results

The learning from basic CNN model at the end of second stage are shown in Fig 4.1 and 4.2. To improve on this basic CNN model, 81 variants have been tried on less number of iterations (epochs) being under resource constraints 4.3. As examples in Fig 4.5 and 4.6, the results from detection stage that classify an object as a car are shown as orange line bounding boxes whereas red dot reveals the ground truth. For reference, A red dot inside bounding box means TP (True Positive), red dot without bounding box means FN (False Negative) and bounding box without any red dot inside means FP (False Positive). There are few cars that have been marked white, these are the masks that were applied to ignore objects that are irrelevant to the problem because of much distance and obstructions in view. CSI Analysis is shown in Figure 4.4 for the detection stage for YOLO and ResNet (RetinaNet) with different DPT over 100 randomly sampled rows from database.

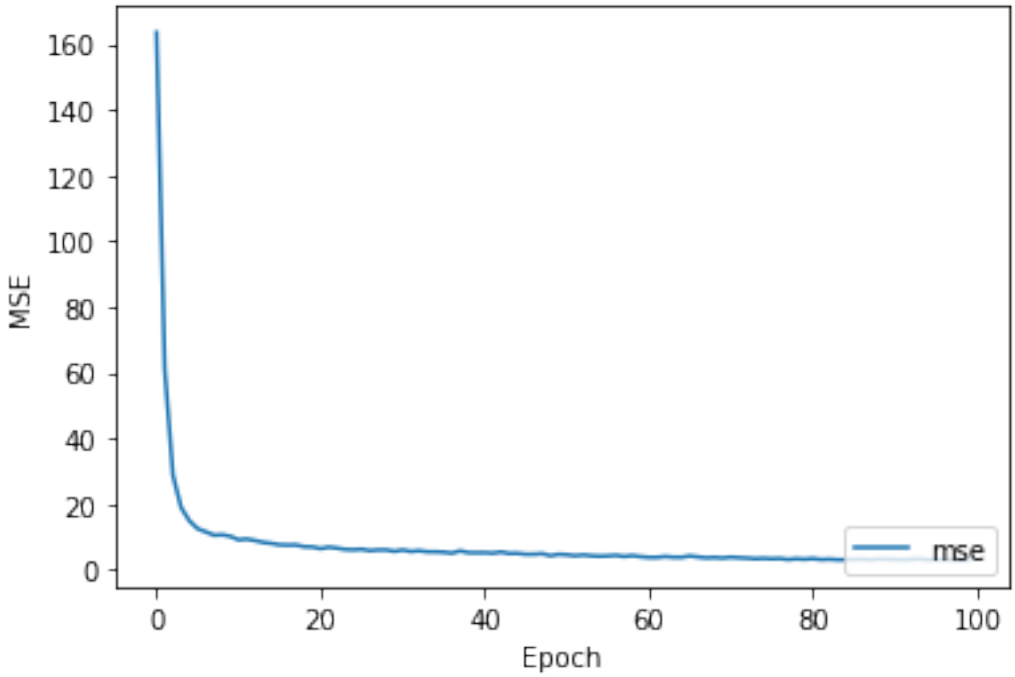

Figure 4.1: Training learning curve with mean squared error over 1000 epochs

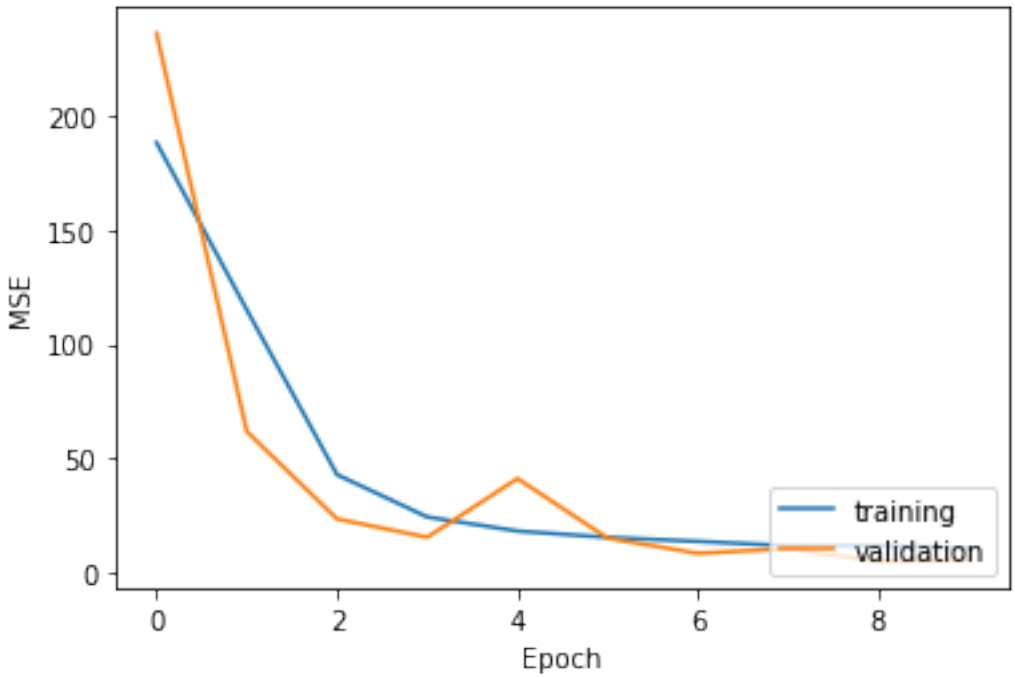

Figure 4.2: Training with validation, mean square error over 10 epochs

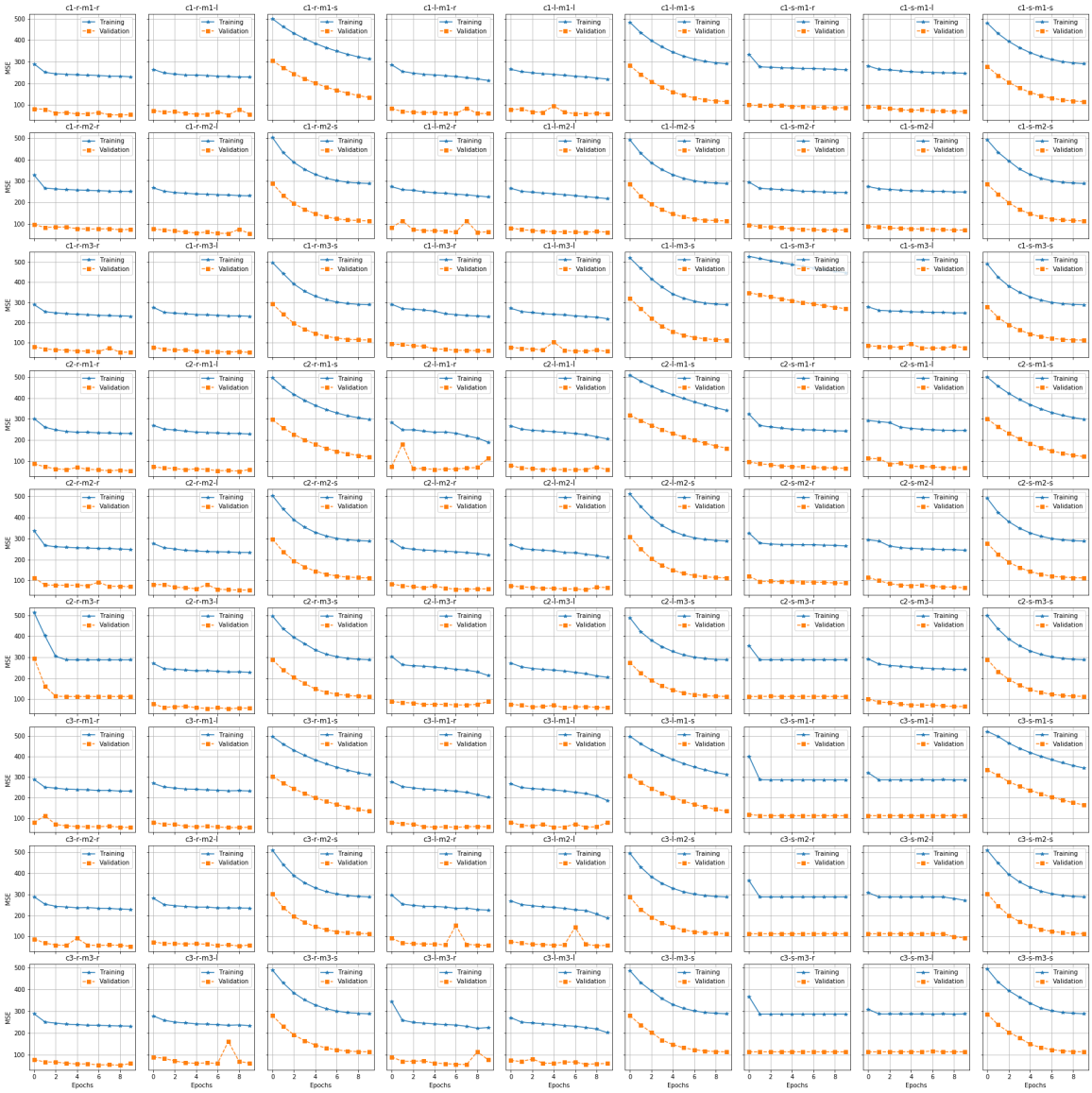

Figure 4.3: CNN model grid search for prediction stage

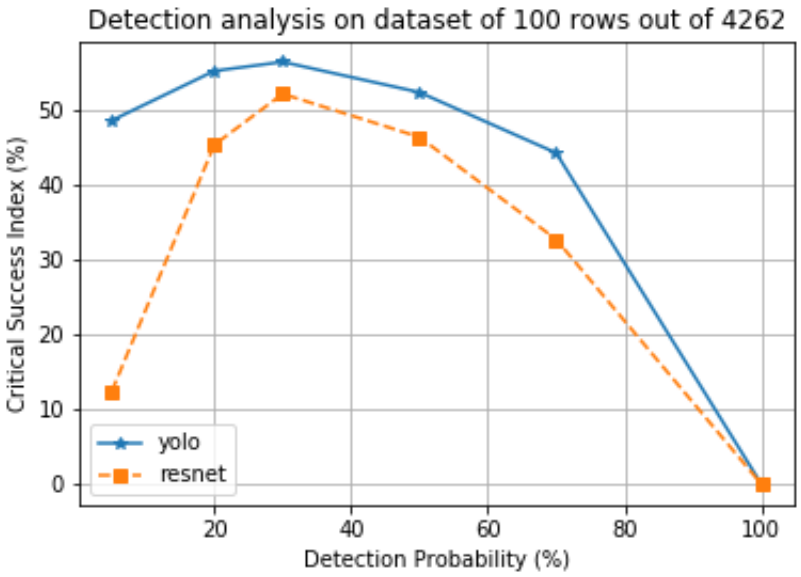

Figure 4.4: CSI analysis for both detection algorithms

Figure 4.5: Detection example 1 - image has 6 TP, 0 FP, 5 FN

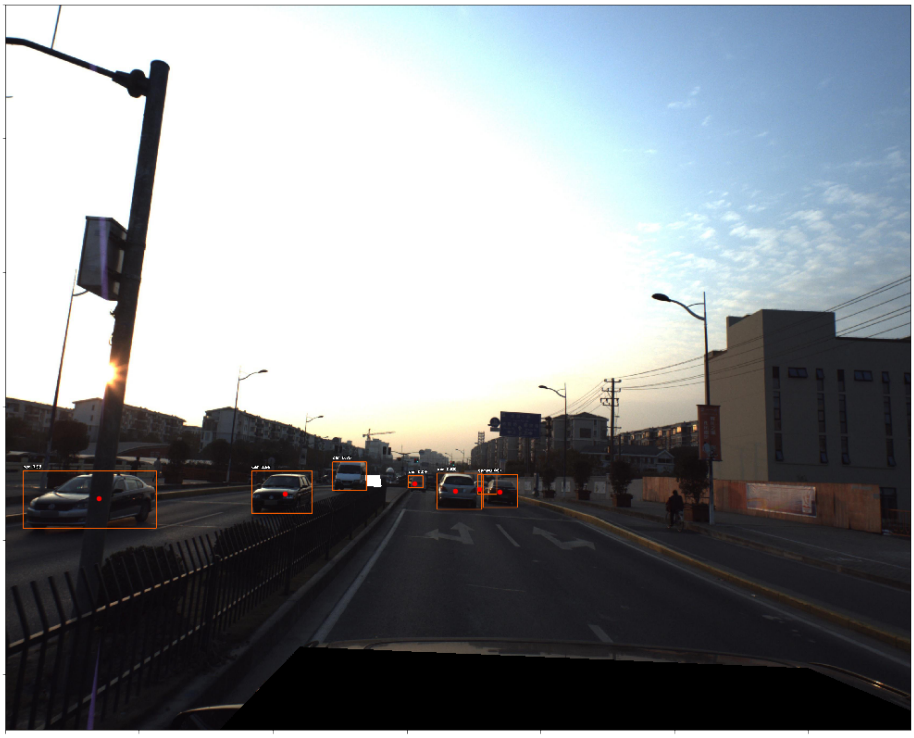

Figure 4.6: Detection example 2 - image has 1 FP

5 Conclusions and Future Work

The results need to be combined in a unified metric such as MAP (Mean Average Precision). During training, The predicted angles information can be evaluated against ground truth angles within certain thresholds sorted according to percentage probability feature, to be considered TP (True Positives) and other predictions being FP (False Positives). Precision calculation will then show combined result of two stages and hence degree of relevance of solution to the actual goal.

Second stage also requires many hyper-parameters optimization, such as changing number of hidden layers, number of units in each layer, loss function, activation functions, batch-sizes, longer number of epochs with early stopping etc.

There is also a possibility of reusing other pre-trained networks for each of the stage.

References

Lin, Tsung-Yi, Priya Goyal, et al. (2017). *Focal Loss for Dense Object Detection*. arXiv: 1708.02002 [cs.CV] (cit. on p. 2).

Lin, Tsung-Yi, Michael Maire, et al. (2014). *Microsoft COCO: Common Objects in Context*. arXiv: 1405.0312 [cs.CV] (cit. on p. 2).

Redmon, Joseph and Ali Farhadi (2018). *YOLOv3: An Incremental Improvement*. arXiv: 1804.02767 [cs.CV] (cit. on p. 2).

Schaefer, Joseph T (1990). "The critical success index as an indicator of warning skill". In: *Weather and forecasting* 5.4, pp. 570–575 (cit. on pp. i, 3).