Image quality and super resolution effects on object recognition using deep neural networks

Conference Paper · May 2019

DOI: 10.1117/12.2518524

2 authors, including:

Christoph Borel
Army Research Laboratory
126 PUBLICATIONS **1,934** CITATIONS

SEE PROFILE

Image Quality and Super Resolution Effects on Object Recognition Using Deep Neural Networks

Christoph Borel-Donohue and S. Susan Young

U.S. Army Research Laboratory, 2800 Powder Mill Rd, Adelphi, MD, USA 20783

ABSTRACT

Real-time object recognition systems are critical for several UAV applications since they provide fundamental semantic information of the aerial scene. In this study, we describe how image quality limits object detection frame-works such as YOLO which can distinguish 80 different object classes. This paper will focus on vehicles such as cars, trucks and buses. Pristine high-resolution images are degraded using different blurring functions, spatial resolution, reduced image contrast, additive noise and lossy compression. Object recognition results are significantly better after applying an image super-resolution algorithm to realistically simulated under-sampled imagery.

Keywords: Image quality, Deep Convolutional Neural Networks, Object detection.

1. INTRODUCTION

Automatic target recognition (ATR) has been studied for many years and many algorithms have been developed to detect and identify targets [6]. In recent years Deep convolutional neural networks (DCNN) have made significant progress in their effectiveness in recognizing objects in imagery taken on the ground with cellphone, point-and-shoot digital cameras and digital single-lens reflex cameras (DSLR) [3]. Large annotated datasets have been created [1] which are used to train and test the performance of DCNN's which has increased to the point of rivaling human performance [2]. Object recognition is usually performed on images where an object occupies a large fraction of the image. Partially obscured objects can be found as well if enough pixels are present. If an object's apparent size is reduced, DCNN's performance degrades to the point that objects are miss-classified or missed entirely. Very little research has been done to determine the limits of performance of DCNN based object recognition algorithms when the object size is reduced. Other limitations of DCNN based object recognition with image quality are even less known, e.g. the performance with image blurring, additive noise, image contrast and lossy compression. In this paper we discuss how the performance of a very popular and fast DCNN degrades with image quality.

2. DESCRIPTION OF THE YOLO OBJECT RECOGNITION FRAMEWORK

There are many powerful object recognition DCNN's available today, e.g. SSD, DSSD, RetinaNet, R-CNN, F-RCNN, YOLO.... The You Only Look Once (YOLO) [4] is a DCNN with 53 convolutional layers followed by 2 fully connected layers. YOLO is very fast compared to SSD and R-CNN. For this study a CPU-only version YOLO v.3 was available to run within the OpenCV framework using Python [5]. YOLO divides an image of size 416 by 416 pixels into a 13 by 13 grid and each grid cell of 32 by 32 pixels. The DCNN is run for each grid cell predicting the probability of an object if its center falls within the cell. The DCNN predicts bounding boxes (4 values), an "objectness" score (1 value) and score for each object class (80 classes) at 3 spatial scales using a feature pyramid. This makes YOLO significantly faster than other methods which are based on region proposals since it uses single network evaluations to make the predictions in one pass. Furthermore each grid cell's DCNN can run on a different processor or GPU.

It is important to note that the first step YOLO performs is an image scaling which takes an input image at an arbitrary size and scales it to 416 by 416 pixels (or whatever size is used in the YOLO CNN description file). However many imagers provide non-square aspect ratio images and at much higher resolution, e.g. a High

Further author information: Christoph Borel-Donohue, christoph.c.boreldonohue.civ@mail.mil

Definition (HD) video frame has a dimension of 1920 by 1080 pixels. Thus there would be significant loss of image quality when applying YOLO to a scaled HD frame video directly. To avoid this loss of information the image needs to be subdivided into several tiles that cover the frame. During the training phase datasets are often augmented by applying image flipping, random cropping, shearing, rotation and scaling. In some DCNN look-up tables are applied to images, noise is added and images are blurred and sharpened. For YOLO the augmentation is built into the code [4] and performs changes in colors (hue, saturation, exposure) and random cropping to accommodate partial occlusions and scaling of the images over a limited range. YOLO is not specifically trained to tolerate image degradation from blurring, noise and compression.

3. IMAGE DEGRADATION STUDIES

A specific image was selected containing a number of cars which were all detected at the original resolution of 416 by 416 pixels which YOLO uses. To keep everything the same we selected version 3 of YOLO and a weights file that is available to run YOLO inside a Python script. The confidence threshold was fixed to 0.5 and the Non Maximum Suppression (NMS) threshold to 0.4. YOLO provides 80 different object classes. In this study only object detections from 3 vehicle classes (cars, trucks and buses) were counted in the results. The image degradation methods were applied to the pristine image of 416 by 416 pixels in varying amounts, e.g. image scale actor, blur kernel width, signal to noise ratio, bit depth and image compression ratio. The degraded images were then fed into YOLO and the number of vehicle detections was determined. For quality control and visualization purposes animations of the image degradations with overlaid vehicle detections were generated. For each degradation method breaking points where the performance dropped sharply were selected visually from plots of the number of vehicle detections versus the degradation parameter.

3.1 Degradation with Pixels on Target

It is well known that YOLO and other object detection frameworks need a certain number of pixels on the target to detect the object. How many is often not clear for a given target class and will depend on the number of instances of the object class, size distribution and augmentations of the image training dataset used. Since cars, trucks and buses are similar and often confused, it was decided to combine them in the analysis. To compute the minimum number of pixels on target required to detect a vehicle the following procedure was followed:

1. Select an image with desired target present in multiple locations and similar size.
2. Crop a sub-region to the native size DCNN accepts, e.g. for YOLO this is 416 by 416 pixels, and manually count the number of objects, e.g. 41 cars in image on the left of Figure 1.
3. For each scaling factor $fac = \{0.05, 0.1, ..., 1.15, 1.2\}$:
   (a) Scale the image to a size $fac416$ by $fac416$ using a cubic interpolation.
   (b) Interpolate the scaled image back to an image size of 416 by 416 pixels using the nearest neighbor interpolation method.
   (c) Feed the interpolated image into the YOLO v3 network, obtain the bounding box dimensions width w and height h and count the number of vehicles $N_{vehicles} = N_{cars} + N_{trucks} + N_{buses}$.
4. Compute the average box dimensions using:

$$b_i(fac) = \sqrt{\frac{1}{N_i}\sum_{j=1}^{N_i} w_j h_j}, \; i = \{cars, trucks, buses\}.$$

The result of how many vehicles are found as a function of the average box dimension b is shown in Figure 2. If no buses were found the label was omitted. Below a size of 20 by 20 pixels or about 400 pixels on target YOLO fails to detect vehicles reliably. This number probably changes with the view angle of the object and could be changed by retraining YOLO with more scales (3 scales are used in YOLO v3).

Figure 1. Left: Image with size 416 by 416 pixels of a parking lot with 41 cars. Middle: Half size $fac = 0.5$ scaled image shows 40 vehicles with 38 identified as cars and 2 as trucks. Right: Quarter size $fac = 0.25$ scaled image shows 30 vehicles with 25 identified as cars and 5 as trucks.

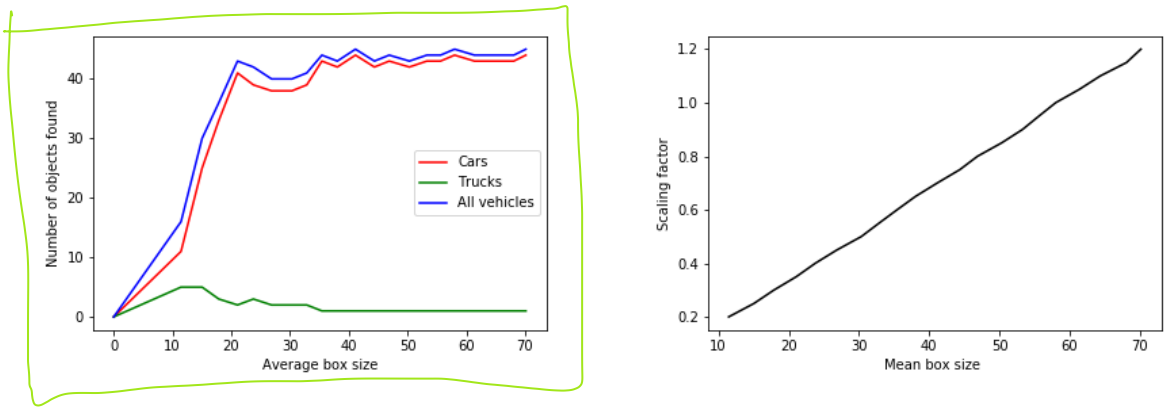

Figure 2. Left: Number of cars detected as a function of the average box dimensions. Right: relationship between scale and average box size.

3.2 Degradation with Image Blurring

Somewhat related to image size is degradation with blurring because a small image where $fac < 1$ is interpolated to a size of 416 by 416 pixels which introduces blurring. However in this study a 2-D separable Gaussian blurring function was used with a width $\sigma = \{0.25, 0.5, 0.75, ..., 3.25, 3.5, 3.75\}$. Typical results for $\sigma = 2.0$ and $\sigma = 3.0$ are shown in Figure 3. Figure 4 shows the number of detected vehicles drops approximately linearly from 45 vehicles for $\sigma = 0.25$ to about 30 for $\sigma = 2.0$ and then sharply to zero for $\sigma = 3.75$.

3.3 Degradation with Additive Noise

Noise is typically added when training DCNN's and thus one would expect a certain amount of tolerance to additive noise. In order to avoid saturated pixel values, the image I was scaled so that the noiseless image raw values fall in the range between 60 and 195. This scaling introduces a loss of contrast which results in a reduction of number of detected vehicles from 41 to 37. Then Gaussian noise N was added in increasing amounts with $\sigma = \{2, 4, 6, ..., 38, 40\}$. A Signal to Noise Ratio (SNR) was computed using the relationship:

$$SNR = \frac{1}{3}\left[\frac{Mean(I(red))}{STDEV(N(red))} + \frac{Mean(I(green))}{STDEV(N(green))} + \frac{Mean(I(blue))}{STDEV(N(blue))}\right]$$

.

In Figure 5 the images with detected vehicles is shown for 3 SNR values and in the right plot of Figure 6 as a function of SNR. The performance starts to degrade for an SNR less than 10 with about half of the cars being

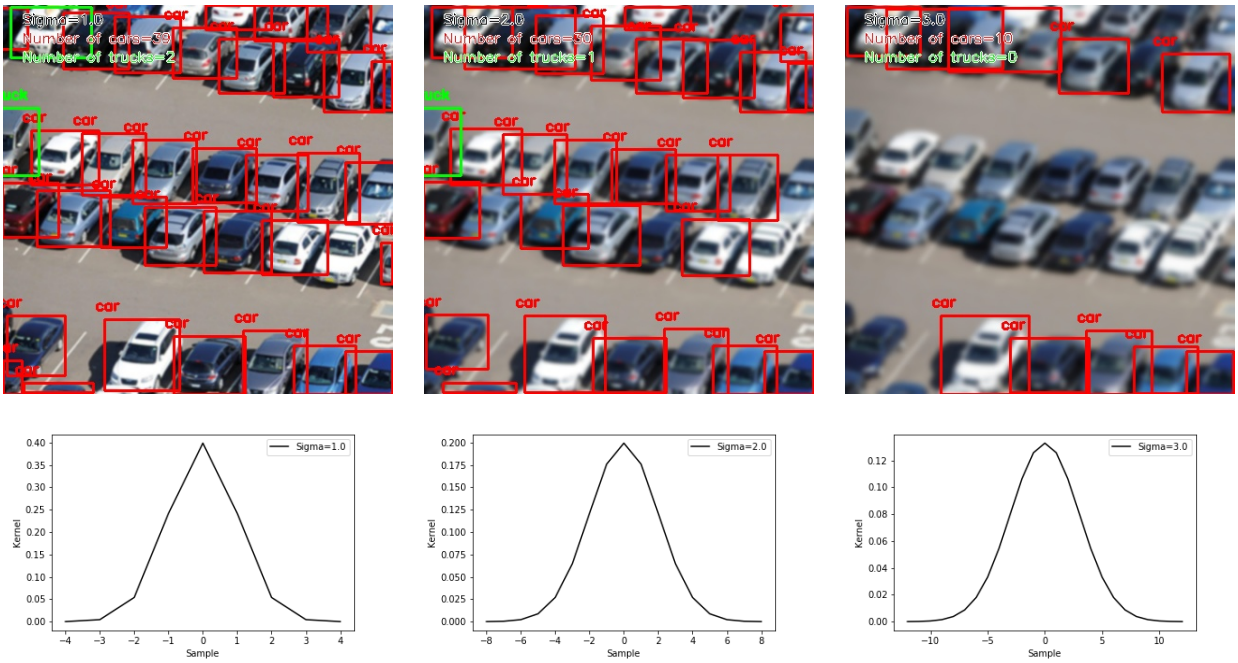

Figure 3. Degradation of the vehicle detection due to image blurring. Left column: Blurred image with kernel width $\sigma = 2.0$ detects 31 vehicles. Middle column: Blurred image with kernel width $\sigma = 1.0$ detects 41 vehicles. Right column: Blurred image with kernel width $\sigma = 3.0$ detects 10 vehicles.

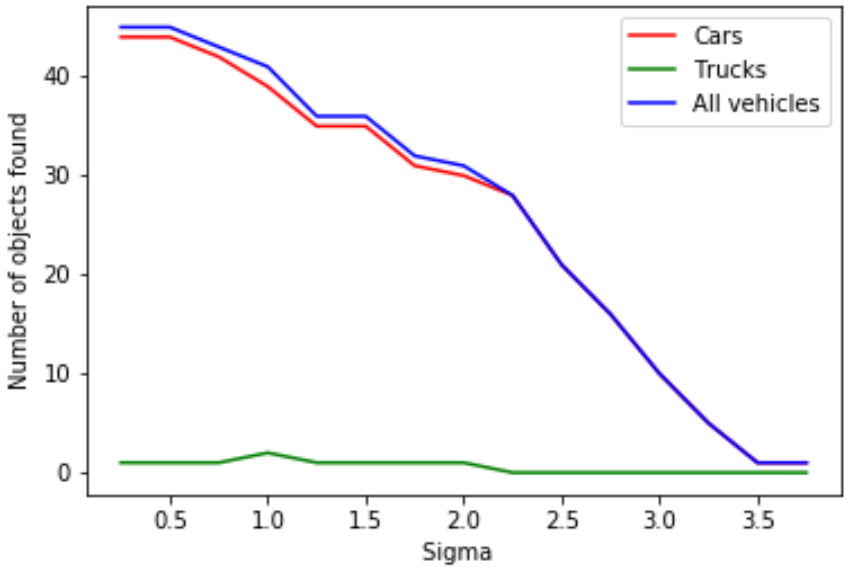

Figure 4. Number of cars detected as a function of the width σ of the Gaussian blurring kernel.

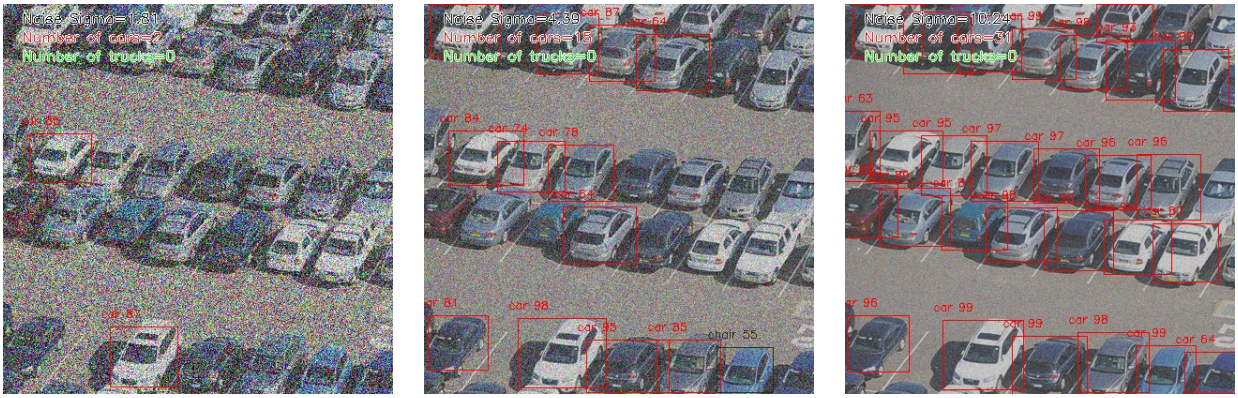

Figure 5. Vehicle detections for additive noise with $SNR = \{1.81, 4.39, 10.24\}$.

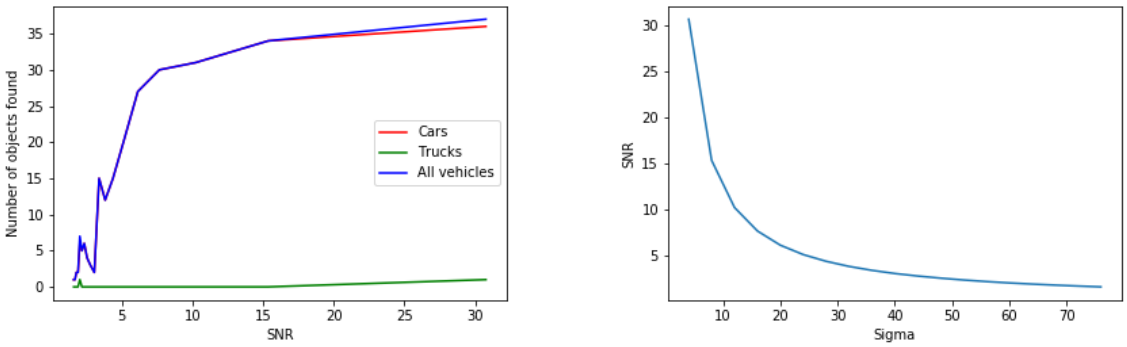

Figure 6. Left: Number of cars detected as a function of the Gaussian noise added with a signal to noise $SNR = \{1.62, ..., 30.76\}$. Right: SNR as a function of $\sigma = \{4, 8, ..., 80\}$.

detected for $SNR = 4.39$ and only 2 cars for $SNR = 1.81$. Thus it seems YOLO is quite robust with additive noise.

3.4 Degradation with Image Contrast

Images taken with digital cameras are typically scaled over a range of 256 (8 bit) levels. However depending on the visibility, exposure and contrast settings fewer levels may be available. DCNN's have convolutional layers which in effect convolve an image with a filter bank and the activation function maps the output onto a range from 0 to 1. As the image dynamic range is reduced, banding across objects with slowly varying brightness is increased which might introduce a loss of spatial information. On the other hand hard edges are preserved and many convolutional layers seem to use that information for object recognition.

In the first approach the image intensities were scaled in such a way that the image intensities varied by just 1,2,4,8,...256 levels symmetrically around 128 which approximates situations where the image contrast is low, e.g. for poor visibility or when the automatic gain control decreases the contrast because of a sun-glint or bright light. The resulting performance on object detection showed a dramatic decrease where only about half of the vehicles were detected when the contrast range of the image was 32 levels and none were found for 3 bits or 8 levels.

The second approach assumed that the images have low contrast but are contrast-stretched to cover all 256 levels. Thus an 8 bit image was divided by the number of desired levels and then scaled back to a range from 0 to 255. In this case the number of detected vehicles did not drop as dramatically and even for 4 levels most of the vehicles were found. Surprisingly 27 vehicles were found for just binary images.

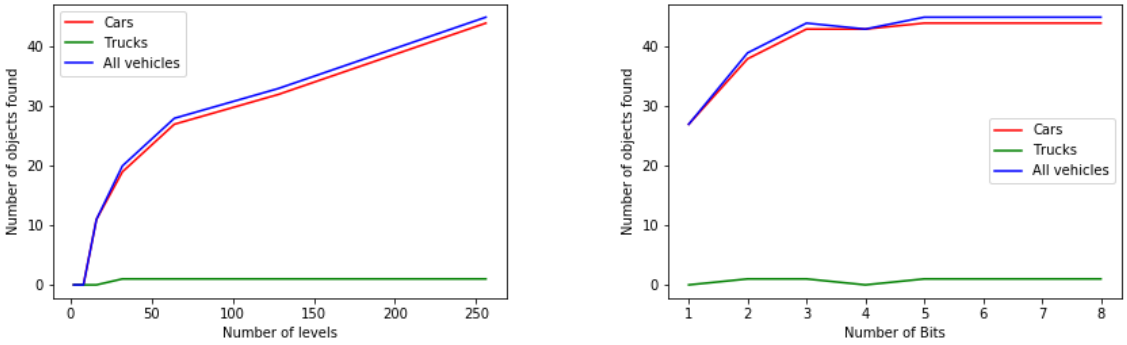

Figure 7. Left: Number of vehicles detected as a function of contrast measured in number of intensity levels. Right: Number of vehicles detected as a function of number of bits to quantize image in the range from 0 to 255.

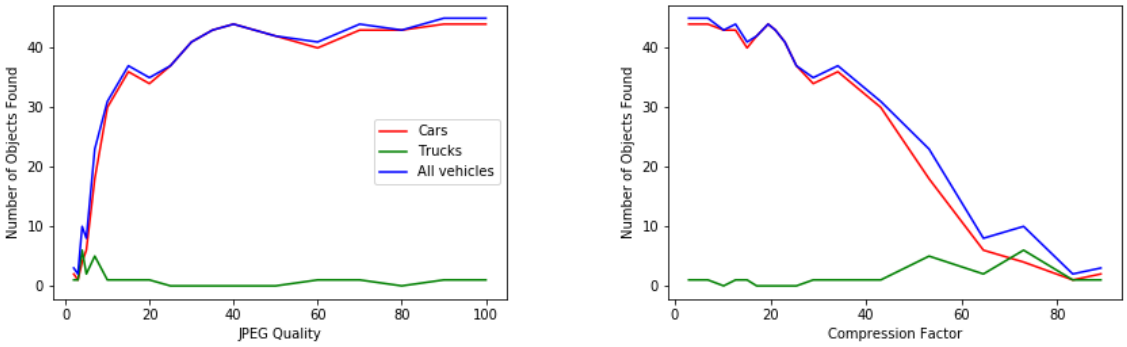

Figure 8. Left: Number of cars detected as a JPEG quality. Right: Number of vehicles detected as a function of compression ratio CR.

The results for both approaches can be found in Figure 7. Thus it seems very important to contrast-scale low-contrast images to cover the entire range of 256 levels.

3.5 Degradation with Lossy Compression

Many users of DCNN's are concerned with the degradation of performance with respect to image compression but not many results are available to quantify the effect of compression. In the previous section we found that a simple reduction in the number of quantization levels which correspond to a form of image compression hardly has any effect on the object detection performance. There are many forms of image compression available and in this study we use one method available as a function in OpenCV, namely the JPEG compression where it is possible to vary the quality parameter $q = \{2, 3, 4, 5, 7, 10, 15, 20, 25, 30, 35, 40, 50, ..., 90, 100\}$. The left plot in Figure 8 shows the number of detected cars as a function of the quality parameter q and the right side as a function of the compression ratio which is defined as:

$$CR(q) = \frac{3N_xN_y}{FileSize(q)}.$$

In Figure 8 the graph shows that the performance does not change up to a quality of 40% and a compression ratio of up to 20:1. For a quality of 7% and compression ratio of 53:1 more than 50% of the vehicles were found.

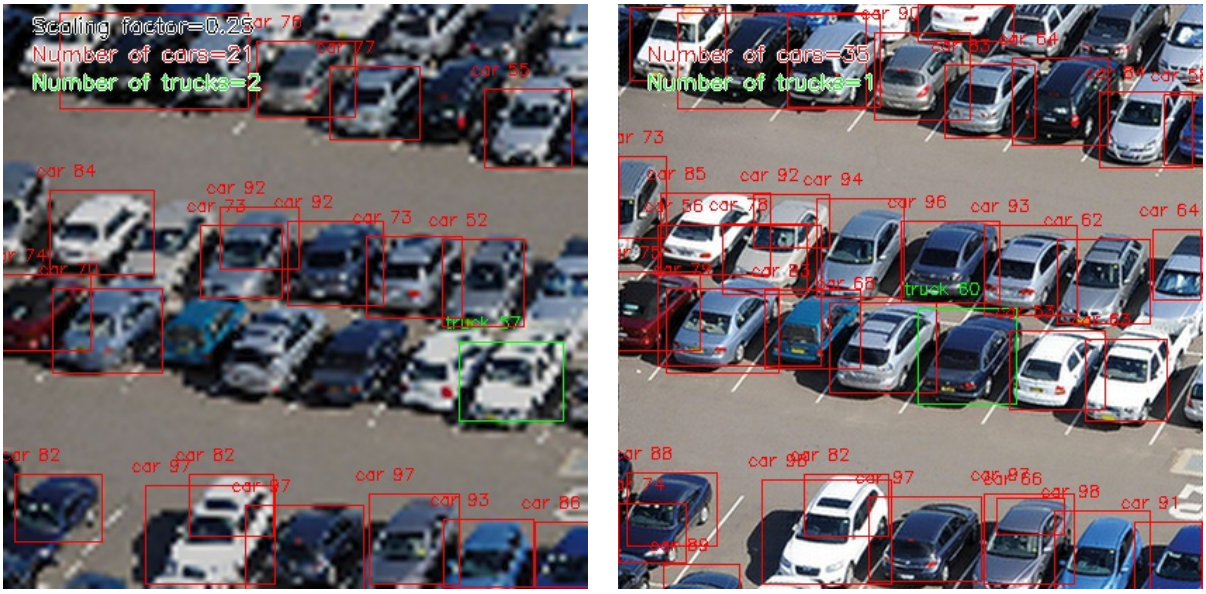

Figure 9. Left: Number of vehicles found in a single low-resolution frame of size 104 by 104 pixels. Right: Number of vehicles found in the super-resolved image of size 416 by 416 pixels.

4. IMAGE ENHANCEMENT STUDIES

4.1 Super Resolution

In subsection 3.1 it was found that at a scaling factor of $fac = 0.25$ some vehicles were no longer detected. We wondered what would happen if we applied super-resolution to a sequence of under-sampled images. Such a sequence of images were produced for scaling factors of $fac = 0.5$ (4 low-resolution frames of size 208 by 208 pixels) and $fac = 0.25$ (16 low-resolution frames of size 104 by 104 pixels). These lower resolution images were then run through a super-resolution algorithm [7] to produce a full size 416 by 416 image. To test the performance of YOLO on low-resolution images they were interpolated to a size of 416 by 416 using a cubic interpolation algorithm and then fed into the YOLO v3 network. The number of detected cars and trucks was recorded for each under-sampled low-resolution input frame. The left image in Figure 9 shows 23 vehicle detections for a single low-resolution input frame of size 104 by 104 pixels and 36 vehicles for the super-resolved image of size 416 by 416 pixels. Note that 4 of the bounding boxes overlap and thus the true count would be 32 vehicles.

The results comparing single frame versus super-resolved and fused detections are shown in Figure 10. The left bar chart shows the results for 16 frames of 104 by 104 pixels between 14 to 28 vehicles were found. A fused vehicle detection was created for all frames by aggregating all bounding boxes and their confidences and then running the NMS algorithm on the aggregate. Fusing the results for 16 frames, 36 vehicles were found. The 416 by 416 pixel super-resolved image yielded 40 detected vehicles. For 4 frames of size 208 by 208 pixels 43 vehicles for the super-resolved image versus between 37 and 40 for individual frames. Fusing the results yielded 35 vehicle detections.

4.2 Salt And Pepper Noise reduction With Median Filtering

In some sensors, in particular thermal imagers, a fraction of the pixels are dead or "hot" and will affect the object recognition performance. In visible sensors electromagnetic interference can produce impulsive noise or "white" snow. This effect is best modeled with "salt and pepper noise" where a fraction f_b of pixels are changed to a value of 0 and a fraction f_w are changed to 255. In a practical situation, however, an image correction algorithm will be used to "fix" the bad pixels in an image before the DCNN is run. In this study we varied $f = f_b = f_w = \{0, 0.01, 0.02, 0.03, ..., 0.2\}$ and computed the number of vehicles detected for the uncorrected and corrected case. The image correction used was a median filter with a kernel size of 3. The results are shown in Figure 11 and show that the performance drops with increasing f. Over 80% of the vehicles are detected for

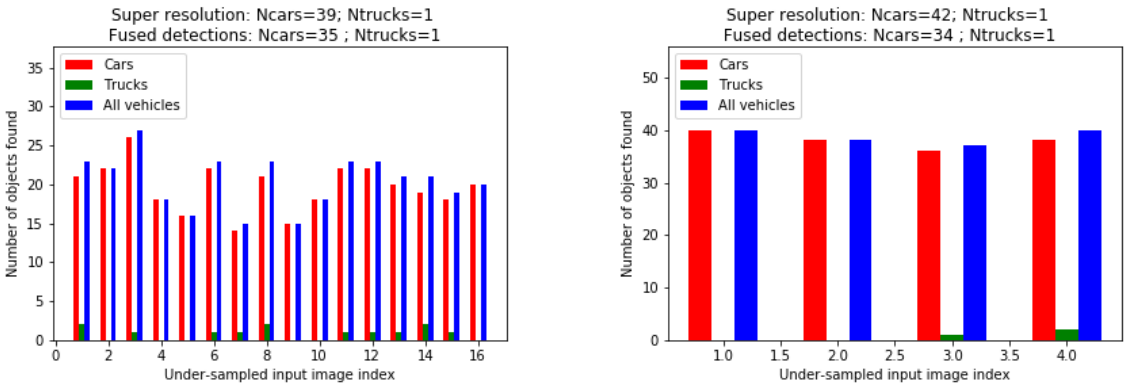

Figure 10. Number of cars found using (a) YOLO on individual frames, (b) NMS fused result and (c) super resolved image. Left: Number of vehicles for under-sampled input frame size of 104 by 104 pixels. Right: Number of vehicles for under-sampled input frame size of 208 by 208 pixels.

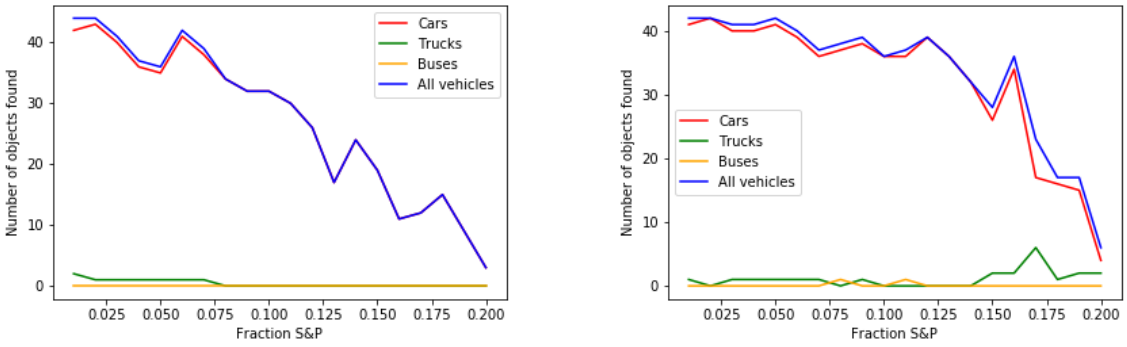

Figure 11. Left: Left: number of vehicles found as a function of salt and pepper noise fraction. Right: number of vehicles found as a function of salt and pepper noise fraction with 3 x 3 median filtering applied.

$f = 0.1$ and 50% of the vehicles are detectable with no correction and 80% for median filtering for $f = 0.15$. For higher noise fractions the median filtered images yield more truck and bus detections.

5. CONCLUSIONS

In this paper we have shown how various image degradations and enhancements affect the object recognition of vehicles. Key results are that for YOLO the number of pixels on target should be about $400 = 20x20$ pixels, the blurring filter width less than 2 pixels, and the signal to noise ratio greater than 10. We also observed that an image with low contrast has a significantly lower number of object detections than if it is contrast-stretched over the full 0 to 255 range. Image compression can be tolerated for a JPEG quality of 40% or compression ratio of $20 : 1$. If a sequence of 16 under-sampled frames are available at a quarter resolution, then super-resolution can improve the number of detected objects by almost a factor of 2, matching the detection results of a 4x larger image. Finally salt-and-pepper noise can be tolerated up to a fraction of $f = 0.1$ and median filtering can yield similar results for $f = 0.15$.

REFERENCES

[1] R. Bernardi, et al., "Automatic description generation from images: A survey of models, datasets, and evaluation measures," Journal of Artificial Intelligence Research 55, pp. 409-442, 2016.

[2] K. He et al, "Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification," The IEEE International Conference on Computer Vision (ICCV), pp. 1026-1034, 2015.

[3] A. Krizhevsky, I. Sutskever, G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks,", Advances in Neural Information Processing Systems 25 (NIPS 2012).
[4] J. Redmon and A. Farhadi, YOLO9000: Better, Faster, Stronger, arXivpreprint arXiv:1612.08242, 2016.
[5] A. Rosebrock, "YOLO object detection with OpenCV,", https://www.pyimagesearch.com/2018/11/12/yolo-object-detection-with-opencv/, 2018.
[6] B. J. Schachter, "Automatic Target Recognition,"', SPIE, Third Edition, 330 p., 2018.
[7] S. S. Young and R. G. Driggers, Super-resolution image reconstruction from a sequence of aliased imagery, Applied Optics, vol. 45, no. 21, pp. 5073 5085, July 2006.