TESTING YOLO ON RAINY CONDITION

[1] *Chia-Ching Fu* (傅家靖), [1] Chiou-Shann Fuh (傅楸善)
[1] Department of Computer Science and Information Engineering,
National Taiwan University, Taipei, Taiwan
E-mail: r07922138@csie.ntu.edu.tw, fuh@csie.ntu.edu.tw

ABSTRACT

Nowadays, image recognition had emerged and become one of the top discussed topics. Also, as the Convolutional Neural Network was introduced in [1], the new era of image recognition comes with CNN's dominating performance in terms of accuracy and feature extraction. However, in 2013, Szegezy et al. [2] found that the models trained with AlexNet and ImageNet structure are unstable. With the tiny changes in pixel value of images, the classifiers can predict the images wrong. Thus, many researchers have created ways to generate adversarial images trying to fool the deep learning models. In recent, the object detector, YOLO, had been introduced and is one of the fastest algorithms that is the most suitable to real-time usage. In this paper, we will look into the ways to alleviate the effect of the adversarial attacks on the real-time object detector YOLO trained on VOCdevkit dataset.
Keywords: *YOLO, VOCdevkit.*

1. INTRODUCTION

With the inspiration of the paper [2], which emphasizes the effect of adversarial and noise images on the recognition task, we notice the possibility that the recognition task might be affected by the real-time images taken from the environment while performing the recognition job with the well-trained model. In this paper, we test on the effect of images taken under the rainy condition on the image's classification task with mAP measurement, and we also try various kinds of noise removing masks to test the performances.

2. RELATED WORK

In this paper, we looked into the effect that the raindrops can be to the detector in terms of the accuracy, and we also look into different method that deal with noise or even adversarial noise removing from other scholars.

2-1. INTRIGUING PROPERTY OF NEURAL NETWORK

The neural network performs well in visual recognition task with its expressiveness to the input image, however, the property of the neural network would be fooled by the uninterpretable input images features caused by the learning of the interpretable images' features [2]. The paper [2] further conjectures that the input is the space rather than the individual unit to the activation of the output in the neural network. From the above notions, the stability of the Deep Convolutional network is questionable caused by the blind spots that may become the vulnerable spot to the false detection of the network. In Figure 1 from the paper [2], it explains the possibility for the trained network to falsely predict the input to the arbitrary class of objects by non- random imperceptible perturbation created by maximizing the predicting error.

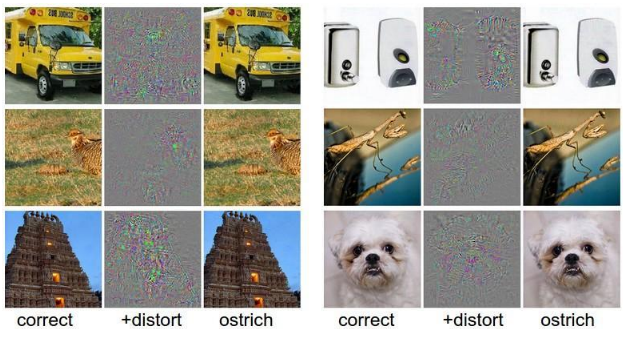


Figure 1:
The image represents the false recognition on the AlexNet after adding distortion to the original images. The left side of each block represents original images, the middle parts are distortion pixels added to the original images, and the right side are images after distortions.

From the notions above, we assume that the noise caused by the environment would affect the performance of the Convolution Neural Network such as YOLO when we are to perform image detection task outdoors.

2-2. NOISES TO DETECTION PERFORMANCE

When we try to perform the recognition tasks on the image with deep convolution network, we encounter various qualities of the real-life images taken as the network input. According to the paper [3], testing images usually have similar qualities to training images in data sets, therefore, the testing process cannot really help on verifying if the training data contains enough varieties. Sometimes the false detection can lead to severe consequences especially for the real- time detection on vehicles, so it is crucial to take the environmental interference to images into consideration while dealing with detection tasks.

In the paper [3], it experiments on the classification performance with various types of noises. After the performance testing on the network with noise images, it trains with noise images to test if the deep neural network model become more robust. Furthermore, authors of this paper perform different setups of noise restoration methods to see if it helps alleviate the down gradation2 of the network. According to the paper [3], it shows the experiment result on the accuracy after applying the Gaussian and the Salt and Pepper noises with different levels on different classifiers, and it is shown in the Figure 2.

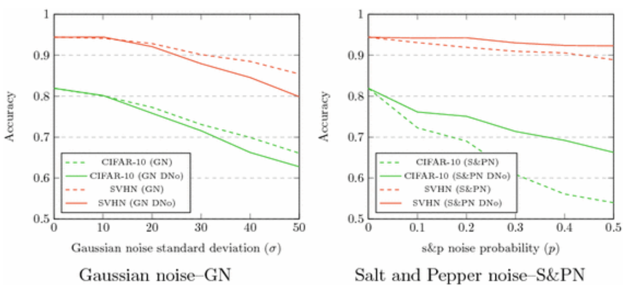


Figure 2:
The performance comparison after adding different levels of the Gaussian and the Salt-and-Pepper noises on different datasets such as CIFAR-10 and SVHN. Each dataset is specified to certain classifier.

Although paper [3] has experimented on the noise effects on detection result, our direction is different from the one done in paper [3]. We focus on the raindrops noises and further compare the detection result with various noise removing filters, while in paper [3], they tested on the Gaussian and the Salt and Pepper noises. The deep neural network model we used is also different from the one used in the paper [3]; We choose to experiment on the mainstream real- time image recognition model called YOLOV3.

2-3. ADVERSARIAL DETECTING

Some researchers try to detect the existence of the adversarial examples(noises). In the paper [4, 5], the authors train the deep neural network binary classifier to determine if there exists any adversarial images(noise).

2-4. INPUT RECONSTRUCTION

The adversarial images can be reconstructed to the original image through the reconstruction process. The deep contractive autoencoder method proposed by Gu and Rigazio [6] can make the model become more robust to the adversarial attacks by training the noise removing network to encode the adversarial images into the clean images.

3. PASCAL-VOC DATASET

The Pascal-VOC dataset, developed by Mark Everingham, aims to the supervised learning problem with labeled images. In the dataset, all the images are categorized into twenty classes including person, bird, cat cow, dog, horse, sheep, aero plane, bicycle, boat, bus, car, moto- bike, train, bottle, chairs, dining table, potted plant, sofa, and tv/monitor. In each image of all the classes, the information of the bounding boxes and the labels are provided with the annotation files.

In this paper, we decide to train on the Pascal-VOC dataset from 2007 to 2012 on the YOLOV3-TINY because the author of the YOLOV3 have trained on the Pascal-VOC dataset, thus, it provides the standard to compare the performance to verify if the trained YOLOV3-TINY has proper mAP.

4. THE MACHINE LEARNING MODEL USED

In nowadays, the object detection task often performs on the real- time situation, so it becomes the basic requirement for object detectors. Therefore, we choose the YOLOV3 object detector for our experiment since it is regarded as one of the fastest real- time object detector.

4-1. YOU ONLY LOOK ONCE

The image detection and classification software, YOLO (You Only Look Once), is the open-source software developed by Joseph Chet Redmon, which deals with the real-time object detection and classification. In YOLO, the Darknet is used as the fundamental structure for the object detection in YOLO. In traditional object detection algorithms such as DPM [7], which the classifier runs through the whole image with sliding windows to perform the object detection, and R-CNN which initially proposes potential bounding boxes and then apply the classifier on those boxes, the structural complexity and the time complexity are cumbersome to the real-time purpose. As what has been mentioned in the paper [8] proposed by YOLO inventors, the YOLO is fast and comparatively accurate because it implements only single CNN to perform the detection job and draw bounding boxes from end to end, therefore, it is a good fit for real time detection tasks. The structure of the YOLO is shown in Figure 3 [8].

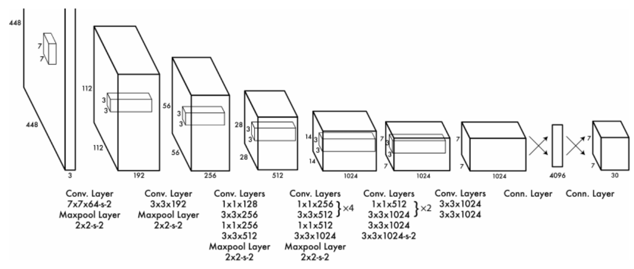


Figure 3:
The architecture of YOLO version one.

4-2. YOLO FEATURES EXTRACTION

According to [8], after the YOLO has been trained, it shows less performance downgrade on artworks testing data among other real-time object detection algorithm because it focuses on learning features of shapes and sizes of objects more than the intensity variant. Therefore, we can be sure that the effect the raindrops brings to the down gradation of YOLO is mainly due to the shape feature distortion. Therefore, we aim to noise removing methods and compare the performance of each noise removing filters.

4-3. YOLOV3

For the model to train in this paper, we decided to use the newest version of the YOLO, however, during the experiment with the YOLOV3, we found that the model structure is resource consuming to the middle level GPU. For the more flexible model on various hardware platform, we decided to train on the comparably smaller version of the YOLOV3, the YOLOV3-TINY.

According to the tech report [9], the YOLOV3 follows the method of anchor boxes that the previous YOLO version, YOLO9000 [10], uses for predicting bounding boxes. In the anchor boxes prediction, it uses predefined anchor boxes for bounding boxes prediction, which makes it easier for the network to learn better [9]. For the initialization of anchor boxes, instead of using hand chosen anchor boxes, it applies k-means algorithm to choose the good initial anchor boxes automatically.

The main change that makes the YOLOV3 more accurate is the feature extraction part of the network, and the architecture is shown in the Figure 4 from [9]. Comparing to the previous version which uses Darknet 19 with 19 convolution layers for the feature extraction, the newest version adds some shortcut connections, which makes the network larger with 53 convolution layers. The full architecture of the YOLOV3 is shown in the Figure 5 [11].

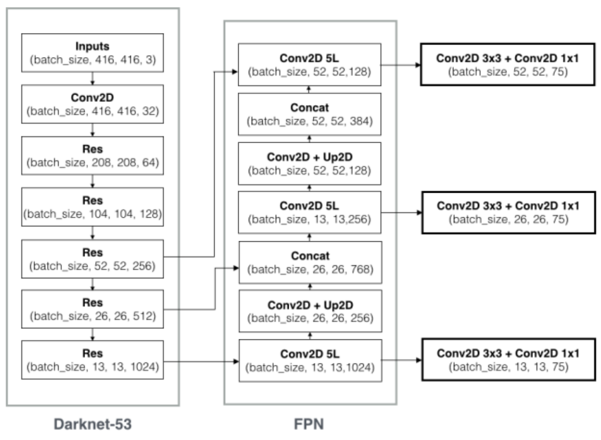
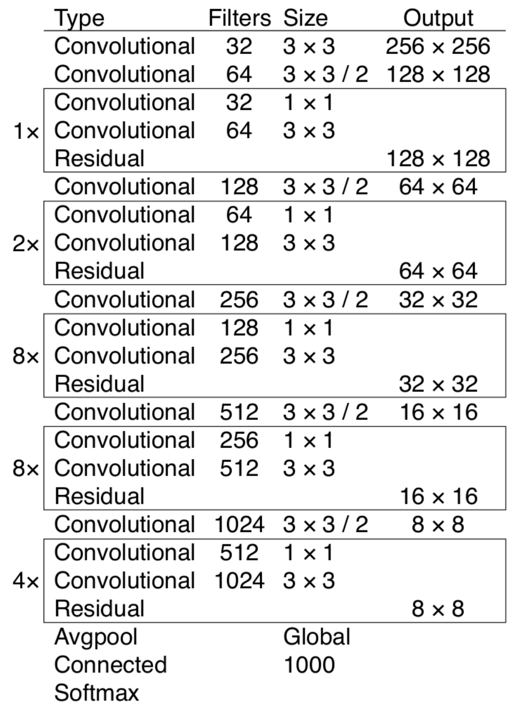


Figure 4:

The architecture of the Darknet 53.



	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Figure 5:
The architecture of the YOLOV3.

4-4. YOLOV3-TINY

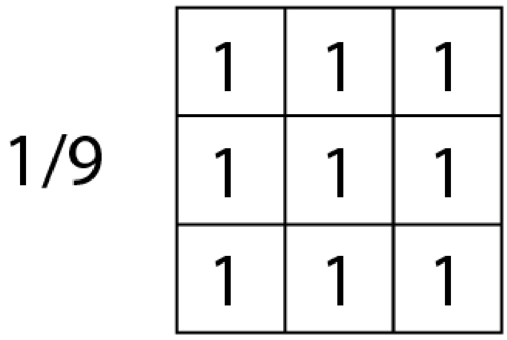
During the training of YOLOV3 on VOC 2007 to 2012 dataset, we found that the model is unable to run on NVIDIA GTX-1050 (notebook) GPU, therefore, we decide to train on the YOLOV3-TINY.

In the YOLOV3-TINY, there are only 7 convolution layers in the network for the feature extraction, and it is considerably smaller architecture comparing to the YOLOV3. From the YOLO official site [12], the author provides the source code, and we found the document about the YOLOV3-TINY configuration showing the architecture of the YOLOV3-TINY. To train the VOC dataset on the YOLOV3-TINY, we modify the original YOLOV3-TINY configuration file to fit with the dataset.

5. NOISE SMOOTHEN FILTER

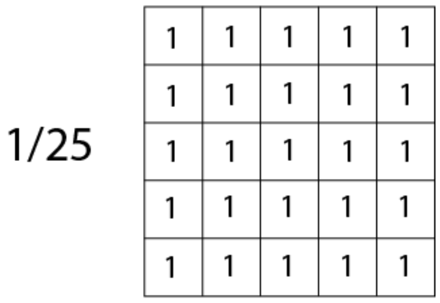
5-1. BOX FILTER

To remove the noise in the image, the most common way is to apply sliding window of matrix to perform the convolution process iteratively on the image. The idea behind it is that by averaging up the neighboring pixels, the outlier can be smoothening. Therefore, the box filter is derived from this idea, and it is shown in the Figures 6 and 7.



$$\frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$

Figure 6:
3 x 3 Box filter



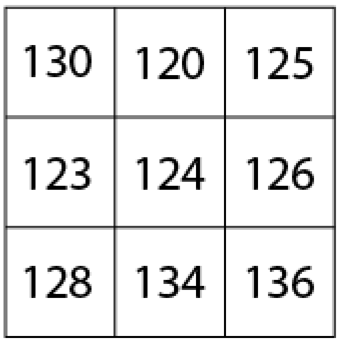
$$\frac{1}{25}\begin{bmatrix}1&1&1&1&1\\1&1&1&1&1\\1&1&1&1&1\\1&1&1&1&1\\1&1&1&1&1\end{bmatrix}$$

Figure 7:
5 x 5 box filter

However, the box filters are prone to the distortion of images, therefore, it is better to take the distance between pixels to the center of the filtering mask into the consideration. By the above notion, the Gaussian filtering mask and the Median filtering mask are better approaches to the distortion issue.

5-2. MEDIAN FILTER

The Median filter takes the neighboring pixels into consideration of sorting, and it replaces the center pixel value with the median value of the sorting result. By iteratively choosing the median value within the window, the outlier can be removed.

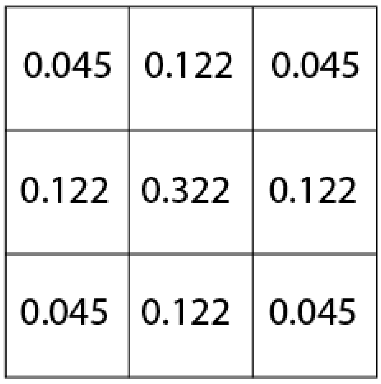


130	120	125
123	124	126
128	134	136

Figure 8:
Median filter
Neighboring values: 120, 123, 124, 125, 126, 128, 130, 134 ,136. Median value: 126

5-3. GAUSSIAN FILTER

The 3 x 3 Gaussian filter is calculated by the equation $G(x,y) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{2\sigma^2}}$ $G(x,y) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{2\sigma^2}}$, where x, y denote the relative position with respect to the center of the 3 x 3 filter (0, 0). The equation can then be simplified to $G(x,y) = \frac{1}{2\pi} e^{-(x^2+y^2)}$ $G(x,y) = \frac{1}{2\pi} e^{-(x^2+y^2)}$ if we set the sigma to 0.707 (square root of 0.5). The equation can be further simplified to $G(x,y) = e^{-(x^2+y^2)} G(x,y) = e^{-(x^2+y^2)}$ since there is a step of normalization, dividing each entry of the filter by sum of all the entries, in Gaussian filter generating algorithm. The resulting matrix is shown in Figure 9.



0.045	0.122	0.045
0.122	0.322	0.122
0.045	0.122	0.045

Figure 9:
Gaussian filter

6. EXPERIMENT

The experiment is done on Ubuntu 16.04 with NVIDIA GTX-1050 (notebook) GPU. We choose the YOLOV3-tiny for out object detector. To simulate the raindrops, we choose to run the python program and add raindrops on all the images in the testing data.

6-1. EXPERIMENT SETUP

The first step is to train the YOLOV3-TINY on the Pascal-VOC datasets from 2007 to 2012. Although the trained classifier does not show good performance with almost half of the mAP of the YOLOV3 due to its simplified architecture, failing to detects some objects such as human with sitting postures, multiple same objects in the images, false detection, etc., we still go with it for the experiment because our goal is to testify the effect of raindrop noises on the real-time detector that is applicable with average computational capability. In the Figure 11, we can see the mAP of each class and also the mAP for all objects.

The second step is to simulate the raindrop noises on testing dataset of Pascal-VOC. In the experiment, we apply raindrops with parameters of raindrops rotation degrees, raindrops densities, and raindrops lengths. The images in Figure 10 are generated by the rotation of -30 degrees, the raindrops density of 500, and the raindrops length of 50. With the above-mentioned configuration for raindrop, it simulates the heavy rain scenario that the density is high and the lengths of the raindrops are long. In Figure 10, the left part of the images is images being added the raindrops, and the right side of the images are the original images.

6-2. CALCULATING .5 MAP

After the simulation of raindrops on the testing images, we run the detector on the testing images with raindrops to calculate the mAP of each class and also the mAP of all classes combined, and the graph is shown in Figure 12. Comparing to the Figure 11, we can see that the mAPs for almost all classes drop but for buses and motor-bikes. Despite for the unexpected results of motor bikes and buses, the mAP for all classes combined drops from 0.29 to 0.25, which is not too severe but still affects the performance.

6-3. EFFECT OF THE BOX FILTER

After the calculation of the mAP on the images with raindrops, we see the performance down gradation. To mitigate this down gradation, we choose the box filter as one of the image smoothers. When we apply a 3 x 3 box filter on the images with raindrops noises, we notice that the detection performance drops a little. The result shows that the 3 x 3 box filter does not provide help on the images with raindrops, and it instead makes the detection result worse. The mAP with all classes combined drops from 0.2577 to 0.2414, and the chart is shown in the Figure 13.

For the 3 x 3 box filter, each pixel in images is averaged with its eight neighboring pixels. With this property of the algorithm, the features learned such as edges might be distorted by averaging up its neighboring pixels. However, the raindrops noises seem to have less effect on the learned feature distortion because each raindrop is thin.

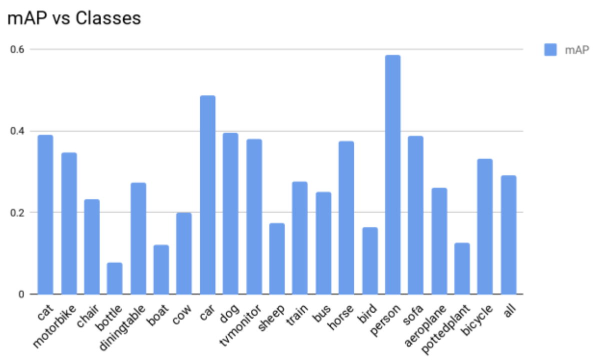


Figure 11:
mAP chart for original images

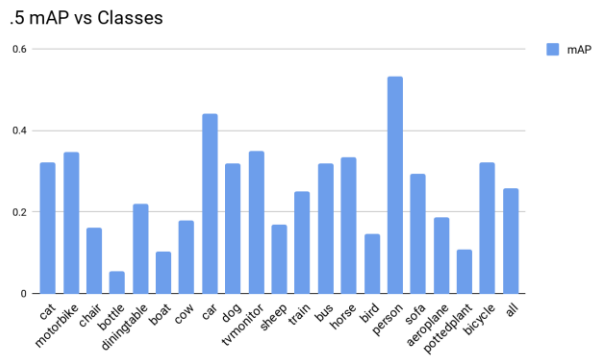


Figure 12:
mAP chart for images with raindrops.
The chart shows the 0.5 mAP for each object class and also average AP of all object classes

6-4. EFFECT OF THE MEDIAN FILTER

After we apply the 3 x 3 Median filter on the images with raindrops, we found that the mAP with all objects combined is even worse than the mAP of the raindrops images. The average AP for all object classes drops from 0.2577 to 0.2414, and the chart of the mAP after applying the Median filter on raindrops images is shown in the Figure 14.

In the Median filter algorithm, it finds the eight neighbors' RGB value and gives the median to the pixel being processed. The failure of the Median filter might be the result of the feature distortion caused by replacing the pixel value with the area median value. For the pixels within raindrops, because the color of raindrops are white (extremity of RGB value), the raindrops can be removed from images. However, in the areas that are not covered by raindrops, the edge would be distorted, and therefore, brings the detection mAP down with more wrong detection.

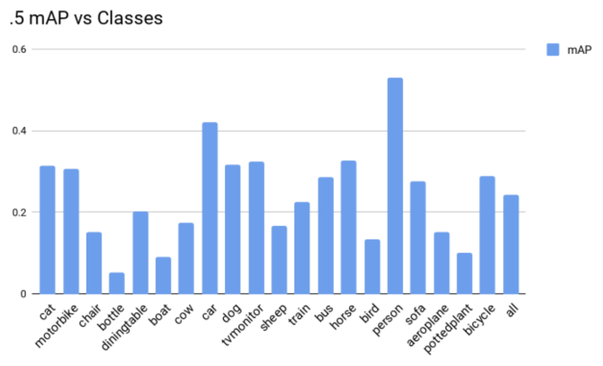


Figure 13:
mAP chart for images after the Box filter.
The chart shows the 0.5 mAP for each object class and also average AP

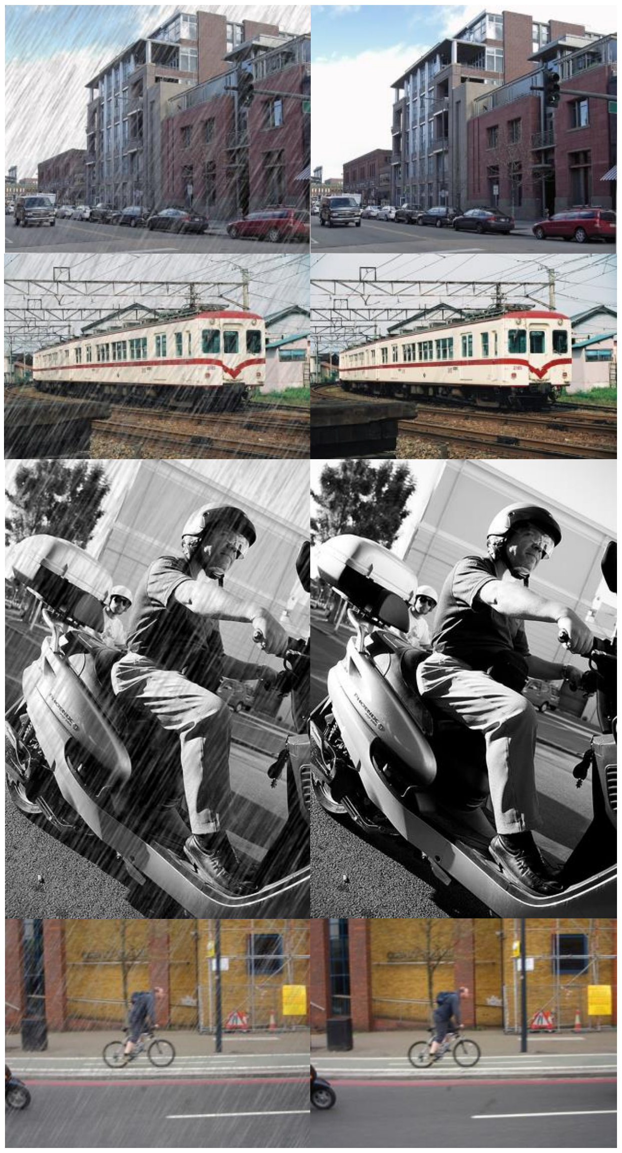


Figure 10:
Raindrop simulation
The left side of columns is images with raindrops, and the right side of columns is original images.

6-5. EFFECT OF THE GAUSSIAN FILTER

Finally, we test on the Gaussian filter with radius 1 and find that this filter does not work on increasing the mean AP of all classes. The Figure 15 shows the mAPs for each class and also the mAP for all classes (all). The mAP for all classes of the images with raindrops is 0.2577, however, after applying the Gaussian filter with radius 1, we found that the mAP for all classes drops even to 0.2401.

Different from the Box filter, the Gaussian filter takes the distance of the pixel being processed and its neighbors into the consideration. With the above definition of the Gaussian filter, it seems reasonable for it to lead the better result; however, the experiment result shows that it rather performs the worst with the lowest mAP score.

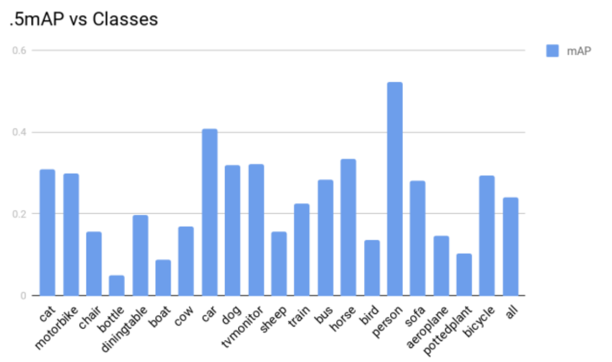


Figure 14:
mAP chart for images after the Median filter.
The chart shows the 0.5 mAP for each object class and also average AP of all object classes

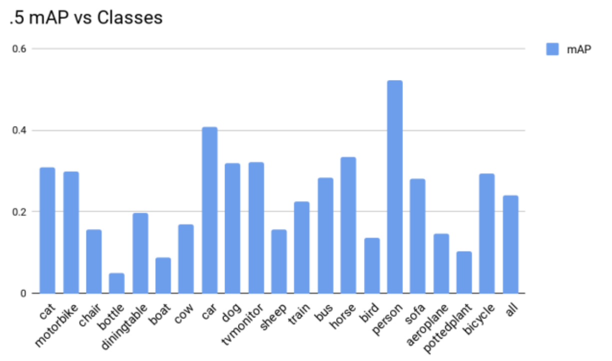


Figure 15:
mAP chart for images after the Gaussian filter, radius = 1.
The chart shows the 0.5 mAP for each object class and also average AP

7. CONCLUSION

The Image recognition is one of the main implementations of the Machine learning algorithm, and it is crucial to have a good performance in terms of the mAP and execution time. After we have trained the machine learning model, our goal is to eventually use it in the real environment. However, we always encounter uncertain condition in terms of the image noise in this paper discussion. In this paper, we discuss how raindrops influences on the detection result of YOLOv3-tiny with Pascal-VOC 2007 to 2012 datasets. The result shows that the mAP drops after adding the raindrops noise. Therefore, we conclude that the heavy rain condition affects the detection task performed on the YOLOv3-tiny. However, after we apply the Gaussian, the Box, and the Median filter, we do not see any improvement on the mAP, and it even drops a little. By the observation, we conclude that the simple sliding window filters do not help on relieving the down gradation of the model and it even make it worse.

6. FUTURE WORK

In this paper, we experiment on the effect that the rain condition can bring to the detection model; however, there are still some other uncertain conditions such as the fog and also the traffic sign physical distortion when performing traffic sign detection. We will look further on other various conditions that would happen in the real environment and look into the effect those conditions bring to the detection result to better induct the uncertain conditions in a numeric way.

7. REFERENCES

[1] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, D. Ramanan, "Object detection with discriminatively trained part-based models", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627-1645, 2010

[2] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I, Goodfellow, R. Fergus, “Intriguing Properties of Neural Networks”, arXiv preprint arXiv:1312.6199, 2014.

[3] T. Nazare, G. Paranhos da Costa, W. Contato, M. Ponti, “Deep Convolutional Neural Networks and Noisy Images”, ICRP, 2017, pp. 416-424.

[4] X. Yuan, P. He, Q. Zhu, X. Li, “Adversarial Examples: Attacks and Defenses for Deep Learning”, arXiv preprint arXiv:1712.07107, 2018.

[5] J. Lu, T. Issaranon, and D. Forsyth, “Safetynet: Detecting and rejecting adversarial examples robustly,” ICCV, 2017.

[6] S. Gu and L. Rigazio, “Towards deep neural network architectures robust to adversarial examples,” Proceedings of the International Conference on Learning Representations (ICLR), 2015.

[7] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. IEEE Transactions on Pattern Analysis and Machine Intelligence, 32(9):1627–1645, 2010.

[8] J. Redmon et al., *You Only Look Once: Unified Real-Time Object Detection*, [online] Available: https://pjreddie.com/darknet/yolov1/.

[9] J. Redmon and A. Farhadi. (2018). ‘‘Yolov3: An incremental improvement.’’ [Online]. Available: https://arxiv.org/abs/1804.02767

[10] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, 2017, pp. 6517-6525.

[11] Mr. Opengate, “Yolo：基於深度學習的物件偵測 (含 YoloV3)”, June, 2018. [Online]. Available: https://mropengate.blogspot.com/2018/06/yolo-yolov3.html. [Access May, 27, 2019].

[12] J. Redmon, “Joseph Chet Redmon”. [Online] Available: https://pjreddie.com/projects/. [Access May, 27, 2019].