Explore Object Detection Performances In Relation To Input Image Deterioration

Zizhun Guo
Department of Computer Science
Golisano College of Computing and Information Sciences
Rochester Institute of Technology
Rochester, NY 14586
zg2808@cs.rit.edu

***Abstract*—Object Detection nowadays has been applied in many scenarios such as self-driving vehicles, traffic safety surveillance and so on. The visual system is typically trained and tested on input images in relatively high quality. This may trigger the visual machines malfunction if they receive the adversarial behaviors by manipulated input images in lower quality when operates in public. Therefore, this works evaluates the performance affections of object detectors by fitting the input images in variant distortion types. In our experiments, we adopt distortion images in noise, grayscale, contrast, packet loss to simulates the images of low quality, and employ YOLO v4 object detector to evaluates the corresponding performances in metrics of mean average precision to compare the results.**

Keywords— object detection; YOLO; distortion; Google Open Image

I. INTRODUCTION

A. Background

Object detection is the main direction of computer vision and digital image processing. It has important practical significance to reduce the consumption of human capital through computer vision in many fields such as robot navigation, intelligent video surveillance, industrial detection, and aerospace. Therefore, object detection has become a research hotspot in theory and application. It is an important branch of image processing and computer vision and is also a core part of intelligent monitoring systems. At the same time, object detection is also a basic algorithm in the field of generic recognition. It plays a vital role in subsequent tasks such as face recognition, gait recognition, crowd counting, and instance segmentation. The detection method mainly introduces two target detection algorithm ideas based on deep learning, a one-stage object detection algorithm, and a two-stage target detection algorithm.

The task of object detection can be divided into two key subtasks: object classification and object localization. The object classification task is responsible for judging whether there are objects of the interest category in the input image or the selected image area (Proposals) and output the possibility of a series of scored labels indicating that the objects of the interest category appear in the input image or the selected image area (Proposals). The object localization task is responsible for determining the position and range of the object of interest in the input image or the selected image area (Proposals), the bounding box of the output object, or the center of the object, or the closed boundary of the object, etc., usually a square bounding box, namely Bounding Box is used to represent the location information of an object.

Fig. 1. Example of Object Detection Output

B. Existing Approaches

The current mainstream object detection algorithms are mainly based on deep learning models, which can be roughly divided into two categories: (1) One-Stage target detection algorithms: this type of detection algorithm does not require the Region Proposal stage, and can directly generate the category probability of the object and the position coordinate value, the more typical algorithms are YOLO, SSD and CornerNet; (2) Two-Stage target detection algorithm: this kind of detection algorithm divides the detection problem into two stages, the first stage generates the candidate region (Region Proposals), which contains the approximate location information of the object, and then classifies and refines the location of the candidate area in the second stage. Typical representatives of this type of algorithm are R-CNN, Fast R-CNN, Faster R-CNN, etc.

The One-Stage object detection algorithm can directly generate the category probability and position coordinate value of the object in one stage. Compared with the two-stage object detection algorithm, the Region Proposal stage is not required,

and the overall process is simpler. As shown in the figure above, the input image is output through the backbone and neck of the networks, and the corresponding detection frame can be generated by decoding (post-processing) in dense prediction, whereas the two-staged object detection has one more step sparse prediction to process.

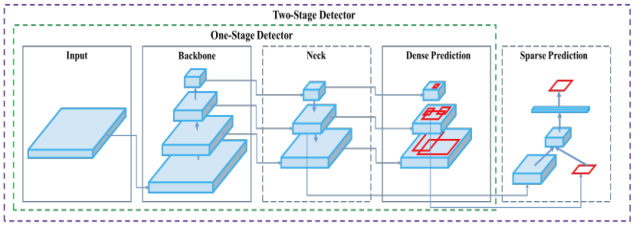

Fig. 2. One-Stage Detector vs Two-Stage Detector

C. *Related Works*

Some previous works have proved that the neural networks work successfully applied on object detection tasks. These include, but are not limited to, YOLO v2 [3], YOLO v3 [4], YOLO v4 [5] as a one-staged object detector, Feature Pyramid Network structure [7], and improved Weighted Bidirectional Feature Pyramid Network [6].

[3] takes Darknet-19 as the backbone of the feature extractor. It puts input image size into (416, 416), employs Batch Normalization layer instead of dropout layer for regularization, uses Region Proposal Network applying k-means to fetch anchor boxes to predict offsets of the bounding boxes.

[4] adopts Darknet-53 as the new backbone of the feature extractor which takes the advantages of the residual network by adding shortcut connections between layers. It also uses 9 types of anchor boxes for better recognizing the objects in different types and distances.

[5] uses CSPDarknet-53 as the improved backbone for the feature extractor. It also has employed the Mosaic data augmentation method.

[8] discovers the relationships between input images in distortion with neural networks designed for classifications. The research paper takes similar distortion methods as blur, noise, contrast, JPEG, and JPEG2000. By comparing the performances in Accuracy for each distorted image set, it shows the deep neural network is susceptible to distortions.

[9] evaluates the effect for the classifiers fitted with artificially low-resolution images. It finds out the indeed the low quality would negatively affect the performances of the classifiers. The trends are similar across the different architectures of the neural networks. It also finds that employing the state-of-art neural networks, mitigates the affection but with the images' qualities continue to get lower, the networks are still susceptible.

II. DATASET

The Google Open Images Dataset contains 9 million images, all of which have been annotated and are suitable for use in image classification, object detection and segmentation, and other purposes. In terms of object detection, 1.9 million images with borders have been annotated, covering more than 600 object categories. In our experiment, we use subsets of the Open Image V6, use the original images to train neural networks and use a separate set of testing images in different distortion qualities to fit the model.

A. *Data Format*

The descriptive table stores the information about each image in CSV file. For our task, we only select the features below:

ImageID: the image this box lives in.

LabelName: the MID of the object class this box belongs to. Stops signs is /m/02pv19 and Traffic signs is /m/015qff.

XMin, **XMax**, **YMin**, **YMax**: coordinates of the box, in normalized image coordinates. XMin is in [0,1], where 0 is the leftmost pixel, and 1 is the rightmost pixel in the image. Y coordinates go from the top pixel (0) to the bottom pixel (1).

TABLE I. SOURCE DATA EXAMPLE

ImageID	*LabelName*	*XMin*	*XMax*	*YMin*	*YMax*
001fba4728544c2e	**/m/02pv19**	**0.615**	**0.635**	**0.525**	**0.573**
0026b117bb17880a	**/m/02pv19**	**0.039**	**0.108**	**0.442**	**0.554**
0062d9a5fb4913ce	**/m/02pv19**	**0**	**0.167**	**0.291**	**0.881**
0121f9c615042091	**/m/02pv19**	**0.685**	**0.781**	**0.526**	**0.669**
01550f1b9680c4ce	**/m/02pv19**	**0**	**0.573**	**0.279**	**0.794**
017f45b1410d3ddb	**/m/02pv19**	**0.446**	**0.608**	**0.294**	**0.492**
01a021697ff040bd	**/m/02pv19**	**0.390**	**0.735**	**0.382**	**0.913**

B. *Distorted Images*

We apply both classical distortion methods and the network packet loss method to process our original images. The classical distortion methods include random noise adding, gray scaling, and contrast increases. The packet loss method used a simulating program transferring images through the network with given various amount of packet loss probabilities.

Original Stop Sign Image

Fig. 3. Original Input Image vs Distortion Images

III. Model Architecture

Fou our study, we choose to use a One-Stage object detection model to conduct the experiements.

A. *CSPDarknet53-tiny*

Darknet53 was first proposed by YOLOv3. Its main goal is to use as a Feature Extractor to embed images into vectors to perform post-prediction regression calculations after decoding. Therefore, as the backbone feature extractor, its role is to encode the image into smaller sized feature maps which contain the unique information about the input image.

B. *CSP and FPN*

The CSP in CSPdarknet represents CSPNet architecture, which stands for Cross Stage Partial Network [1], its main purpose is to enable the architecture to achieve richer gradient combination information while reducing the amount of calculation. This goal can be achieved by dividing the feature map of the base layer into two parts and then merging them through the proposed cross-stage hierarchical structure. See CSPBlock structure in Figure 4, as the module adopting CSP architecture as part of the CSPDarknet53-tiny, it is composed of partial dense layer and partial convolutional layer. It is similar to the proposed CSPDenseNet which keeps the benefits of DenseNet's feature for reusing characteristics, but meanwhile prevents an additional amount of duplicate gradient information by truncating the gradient flow [1]. In CSPDarknet53-tiny, the CSPBlock module divides the feature map into two parts, and combines the two parts by cross stage residual edge. This makes the gradient flow can propagate in two different network paths to increase the correlation difference of gradient information [2].

Through the FPN algorithm, the high-layer features of low-resolution and high-semantic information and the low-layer features of high-resolution and low-semantic information are connected from the top to the bottom side, so that to make the semantic information from features at all scales rich. Prediction on feature layers of different scales makes the effect of generating proposals better than the YOLOv2 Darknet19 backbone network and other traditional feature extraction algorithms such as VGG16 that only predict at the top level. As Figure 4 shows, using FPN can effectively reduce detection time [2], it employs two-scale predictions (shape size in 26x26 and 13x13), conducting convolution layer and up-sampling layer at last feature layer in shapes of 26x26 and 13x13 so that to perform feature fusion.

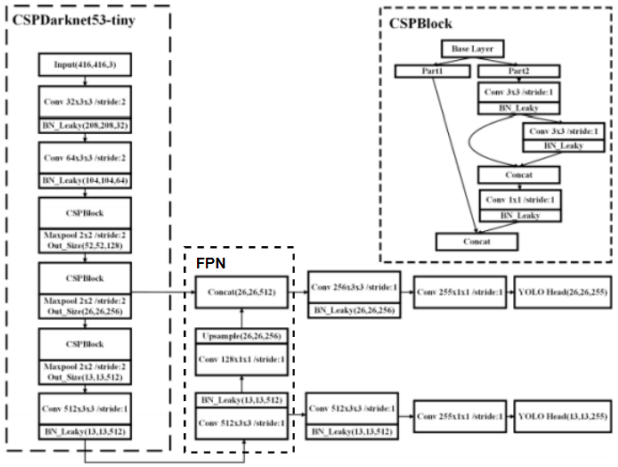

Fig. 4. Example: Architecture of CSPDarknet53-tiny for COCO Dataset

Through the above modules, the input image is extracted from the backbone feature network through the backbone feature network, and the two branch layer features are extracted through the CSP structure. After feature fusion is performed through the FPN algorithm, the two encoded image features have been obtained. The next part is to perform decoding process on these feature maps and run YOLO algorithm to make predictions. These processes are implemented by the YOLO Head in the last part of the result.

C. *YOLO Head*

The shape of the feature maps output from the above feature detectors is either (26, 26, 255) or (13, 13, 255). Since YOLO, it has a grid system that divides the image into smaller cells, therefore the first two coordinates represent the number of cells in total. The last coordinate represents the encoded information

about the images. The details of how encoding works would be illustrated in the next part.

IV. Model Mechanism

Object Detection is a regression task that neural networks are required to output not only the predicting score and class for an object but also the bounding boxes. Figure 5 shows the basic mechanism of YOLO networks as the one-staged object detector.

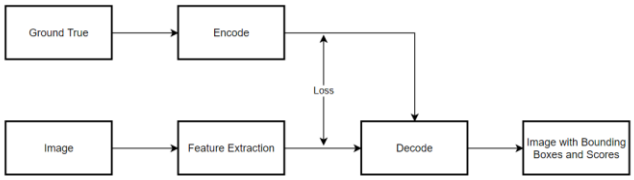

Fig. 5. YOLO Mechanism

A. Label y

As the Figure 5 shows, the output label is not the results output from the YOLO head layer, since the decode component would convert the bounding boxes' encoding offsets into the real coordinates of the input image, YOLO Head works for filtering the real bound boxes and select ones that are expected to remain, therefore using the intermediate result – encodings as the labels ignores the additional operation and keeps the offsets as the normalized target values, which improved the efficiency to calculate the loss for the training process.

B. The shape of Input and Output

The input image is sized in (m, 416, 416, 3), whereas m represents the number of batches set by the practitioners, (416, 416) represents the pixel numbers for the weight and height of the input images, and the 3 represents the channels numbers as RGB channels.

Fig. 6. Original Input Image vs Images in cells

The output encoding is sized in (m, 26, 26, 21) or (m, 13, 13, 21), whereas m is the same as the input image, due to the object detector is FPN structured, therefore two scaled feature maps are extracted but the network, however, both (26, 26) and (13, 13) represents the same meaning, which is the cells numbers of the original images. YOLO algorithm requires the image to be divided into cells into variable scales so that to enable extracting objects in different distances since the objects at a far distance would be small-sized in the image which takes fewer numbers of pixels than the objects in closer distance.

C. Encodings

The encodings are fetched by the encoding process (see Encode in figure 5) while assigning the ground true samples during the training session or they are the intermediate results representing the extracted features process (see Feature Extractor in figure 5) which are output by the network backbone. The tensor size for the embeddings contains the same number of ranks of the input image but the actual features are thereby translated and embedded into the special object detection format.

The last dimension number of the output encoding represents the list of bounding boxes along with the recognized classes. In our case, we have two classes: Traffic Signs and Stop Signs. As the specific flatten format of the encodings can be reorganized in a more meaningful way, from figure 7, the last element of the output encoding is 21, which also can be represented in (3, 7). The first element of 3 indicates that there are three anchor boxes, which are set to vary the scales of detecting results, **whereas the number of 7** represents $\boldsymbol{p_c}$, $\boldsymbol{t_x}$, $\boldsymbol{t_y}$, $\boldsymbol{t_w}$, $\boldsymbol{t_h}$, $\boldsymbol{c_1}$, $\boldsymbol{c_2}$.

$\boldsymbol{p_c}$: the confidence if it exists an object in current cell.

$\boldsymbol{t_x}$: the x coordinate of the offset to anchor box grid cell.

$\boldsymbol{t_y}$: the y coordinate of the offset to anchor box grid cell..

$\boldsymbol{t_w}$: the relative weight ratio for the anchor box.

$\boldsymbol{t_h}$: the relative height ratio for the anchor box.

$\boldsymbol{c_1}$: the probability for the object being as a Traffic Sign.

$\boldsymbol{c_2}$: the probability for the object being as a Stop Sign.

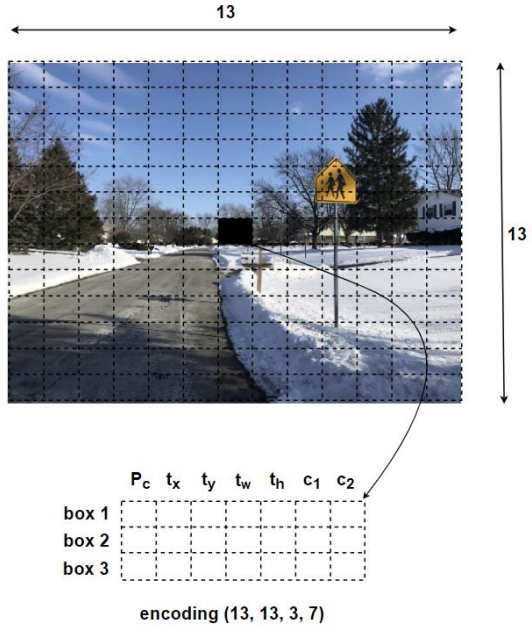

Fig. 7. Encoding in (13, 13, 3, 7)

D. Anchor Box

The concept of anchor box was originally introduced by Faster RCNN. The anchor box (also known as the bounding box prior in the paper, and the anchor box is used later) is the statistics (using k-means) from all the ground truth boxes in the

training set and the most frequently appearing box shapes and sizes in the training set. These statistical prior (or human) experiences can be added to the model in advance so that when the model is learning, the model converges quickly.

Another understanding about using anchor boxes is that the traditional object detection head used as the regressor can only detect one object. The performance for multiple object detection would be affected and interference due to the variant shape of the bounding boxes. To solve this issue, multiple regressors can be used which are all limited to specific detecting regions. To achieve such a mechanism, for each grid cell, there could set multiple anchor boxes in different shapes specifically in charge for detecting objects around the positions.

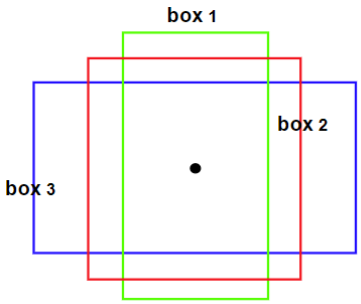

Fig. 8. 3 Anchor Boxes for single route of feature maps

From 4.2 FPN and 5.3 Encodings parts, we have two routes of feature map output for each input image sample. Therefore, we have two features that are in tensors' shape in (13, 13, 3, 7) and (26, 26, 3, 7). The third number 3 implies there are three anchor boxes that are used for specifying the regressors.

For the route feature maps in the shape of (26, 26, 3, 7), we select (23,27), (37,58), (81,82) sized bounding boxes as the anchor boxes. For the route feature maps in the shape of (13, 13, 3, 7), we select (81,82), (135,169), (344,319) as the anchor boxes.

As can be seen by the size values, for images divided into (26, 26) cells, each cell **is** smaller than the ones processed into (13, 13), hence the anchor boxes for (26, 26) cells are smaller in the number of pixels, whereas the (13, 13) cells have greater sized anchor boxes.

E. Intersection over Union for bounding boxes

The bounding box is represented in a format of 4 numbers. In the different processes of the YOLO neural networks, the tensors of the bounding boxes are variant processed. For example, the bounding box values as the intermediate results that are produced after the encoding process are represented in the format of $(\boldsymbol{t_x}, \boldsymbol{t_y}, \boldsymbol{t_w}, \boldsymbol{t_h})$**.** This tensor cannot be directly used to reference the exact bounding box's coordinates so that to use it as the prediction results. Therefore, such intermediate should be decoded into the real coordinates that have the midpoints, weights, and heights in size of the same scales of the real image. The actual bounding box is set to be in the format of $(\boldsymbol{b_x}, \boldsymbol{b_y}, \boldsymbol{b_w}, \boldsymbol{b_h})$ whereas b_x represents the bounding box mid point x-coordinate, b_y represents the bounding box mid point y-coordinate, b_w represents the weight of the bounding box and b_h represents the height of the bounding box.

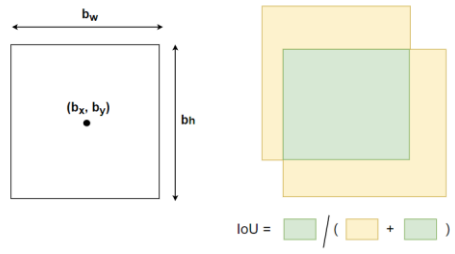

Fig. 9. Bounding Box example

Intersection over Union (IoU) is an evaluation metric that can be used in the prediction task of bounding boxes in values of ranges. The metric applies to all shapes of objects. In YOLO neural networks, it serves the same purpose by calculating the ratio between the intersection area over the union area to measure the accuracy of bounding boxes regressor.

F. Decode

As the anchor boxes have different sizes of the boxes, YOLO only predicts the bounding boxes in the format of $(\boldsymbol{t_x}, \boldsymbol{t_y}, \boldsymbol{t_w}, \boldsymbol{t_h})$ that bonds to different anchor boxes where each value represent the offsets and relative values, due to the model instability when in early training iteration. However, this form of coordinates does not affect the final location prediction. The decoding process would translate the temporary coordinates into the bounding boxes coordinates relevant to the real image size.

$$b_x = \sigma(t_x) + c_x$$

$$b_y = \sigma(t_y) + c_y$$

$$b_w = p_w\, e^{t_w}$$

$$b_h = p_h\, e^{t_h}$$

$\boldsymbol{b_x}$ and $\boldsymbol{b_y}$ represent the bounding box center coordinates, $\boldsymbol{b_w}$ and $\boldsymbol{b_h}$ represent the bounding box weight and height. $\boldsymbol{t_x}$ and $\boldsymbol{t_y}$ represent the level of the center point shifts reletively to the $\boldsymbol{c_x}$ and $\boldsymbol{c_y}$ which represent that the cell is offset from the top left corner of the image. The $\boldsymbol{b_w}$ and $\boldsymbol{b_h}$ represent the weight and height of the bounding box prior (also known as the anchor box).

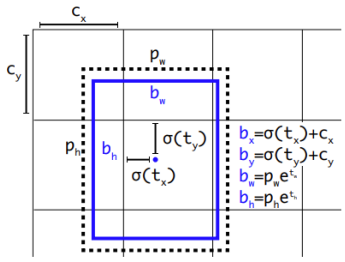

Fig. 10. Bounding Box Decode

The Decode process transforms the feature maps in the shape of $(\boldsymbol{t_x}, \boldsymbol{t_y}, \boldsymbol{t_w}, \boldsymbol{t_h})$ into the shape of $(\boldsymbol{b_x}, \boldsymbol{b_y}, \boldsymbol{b_w}, \boldsymbol{b_h})$, which makes it possible to not only take the advantage of anchor box mechanism but also by constraining the values of bounding boxes so that to make the network more stable.

G. Confidence Scores

YOLO defines the Confidence Scores as the product of the confidence and the Intersection of Union (IOU) of prediction and ground truth bounding boxes' pairs.

$$\Pr(\text{Object}) * IOU_{Pre}^{True}$$

$$\Pr(Class_i|Object) * \Pr(\text{Object}) * IOU_{Pre}^{True}$$

$$= \Pr(Class_i) * IOU_{Pre}^{True}$$

The $\mathbf{Pr}(\boldsymbol{Class_i}|\boldsymbol{Object})$ from the equation in our case is $\boldsymbol{c_1}$ and $\boldsymbol{c_2}$. Therefore, the final confidence score is determined how much the overlapping ratio between the prediction and ground true bounding boxes and the object-specific probability.

H. Non-Max Suppression

Non-maximum suppression (NMS) is an algorithm for removing non-maximum values. The simple understanding is to select all the partial substitutions by defining the part. In the process of object detection, a large number of candidate bounding boxes will be generated at the same object position. These candidate boxes may overlap with each other. In this situation, the NMS would first generates a detection box with the object detection confidence score, the detection boxes with the highest scores are replaced, and other detection boxes that have obvious overlap with the replaced detection boxes are suppressed.

The progress of the Non-maximum suppression algorithm : (1) Sort bounding box list by confidence scores; (2) Select the bounding box with the highest confidence score and add it to the final output list, and delete it from the bounding box list; (3) Calculate the area of all bounding boxes; (4) Calculate the IoU of the bounding box with the highest confidence score and other additional boxes; (5) Delete bounding boxes with IoU greater than the threshold; (6) Repeat the above process until the bounding box list is empty.

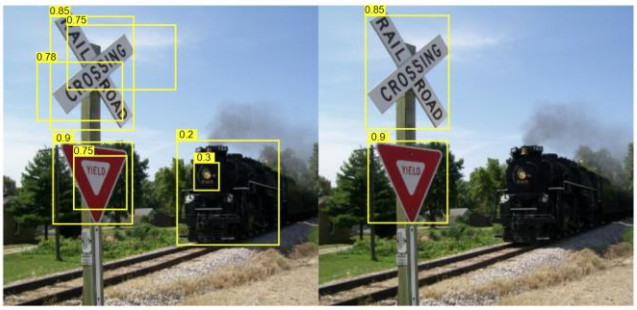

Fig. 11. NMS

The image on the left is the result of the candidate boxes of YOLO detection. Each bounding box has a confidence score. If NMS is not used, multiple candidate boxes will appear such as duplicate bounding boxes on the same object and bounding boxes on other objects which should not be seen as the target objects. The image on the right-hand side is the result of using non-maximum suppression, which fits the expectation to have two bounding boxes attached on two traffic signs.

I. Ground Truth Boxes Assignment

Before the training session, one necessary step is to define the positive and negative cases for target variables for each image sample. It is essential to assign the ground true bounding box in the appropriate anchor box. Since each object, has only one corresponding bounding box for labeling but exists multiple anchor box regressors. The strategy is to calculate the IoU between ground true bounding box with each anchor box and select the anchor box with the greatest IoU as the target.

J. Loss

The loss function of YOLO v4 is mainly divided into three parts: bounding box regression loss, confidence loss and classification loss.

$$\begin{aligned} Loss = &\sum_{i=0}^{S^2}\sum_{j=0}^{B}\mathbb{I}_{ij}^{obj}[(\sigma(t_x)_i^j - \sigma(\hat{t}_x)_i^j)^2 + (\sigma(t_y)_i^j - \sigma(\hat{t}_y)_i^j)^2]+ \\ &\sum_{i=0}^{S^2}\sum_{j=0}^{B}\mathbb{I}_{ij}^{obj}[(t_{w_i}^{\ j} - \hat{t}_{w_i}^{\ j})^2 + (t_{h_i}^{\ j} - \hat{t}_{h_i}^{\ j})^2]+ \\ &\sum_{i=0}^{S^2}\sum_{j=0}^{B}\mathbb{G}_{ij}(C_i^j - \hat{C}_i^j)^2+ \\ &\sum_{i=0}^{S^2}\sum_{j=0}^{B}\sum_{c\in classes}\mathbb{I}_{ij}^{obj}(p_i^j(c) - \hat{p}_i^j(c))^2 \end{aligned}$$

The $\boldsymbol{S}$ and $\boldsymbol{B}$ represent the grid cells number and the bounding box priors (anchor boxes) number. The value of $\mathbb{I}_{ij}^{obj}$ parameter decides whether count the loss of the bounding boxes. If $\mathbb{I}_{ij}^{obj}$ equals 1, it means that the predicting bounding box matches the current anchor box, whereas if it is 0, the loss does not take into account.

The $\mathbb{G}_{ij}$ plays a similar role as the $\mathbb{I}_{ij}^{obj}$, it defines whether the confidence scores of the prediction result should be considered. For $\mathbb{C}_i^j$ =1, the predicted bounding box has the greatest IoU with ground truth box, hence $\mathbb{C}_i^j$ =0 for the cells of other anchor box types. One particular case is, the current anchor box which does not assign to detect a specific object class produces the IoU that is greater than the IoU threshold (YOLO paper is 0.5), $\mathbb{G}_{ij}$ would be 0 to neglect the loss calculation for it.

In the original YOLO paper [3], the loss function employs sum-squared error for it is easy to calculate. In our project, we use the binary cross-entropy as the cost for each grid cell and bounding box prior.

V. TRAINING

A. Use Pretrained Model

The pre-trained model is a deep learning architecture that has been trained to perform specific tasks on large amounts of data (for example, identifying classification problems in pictures). This kind of training is not easy to perform, and usually requires a lot of resources, beyond the resources available to many people who can use deep learning models, such as not having a large number of such GPUs.

It should be noted that the weight of the pre-training model needs to conform to the model structure used in the current task. In other words, the pre-training model must also be trained with the model structure of the current architecture. Therefore, the only difference is whether the machine equipment can support large-scale dataset training.

The purpose of this is to increase training efficiency. In fact, many pre-training models already have relatively good prediction efficiency, but it should be noted that the data set during training of the pre-training model should have a target value similar to the current prediction task. In our case, there are two target categories in the Coco dataset that are similar to the two categories of the currently used Open Image dataset, namely Street Sign (COCO) corresponding to Traffic sign (Open Image) and Stop sign (COCO) and Stop sign (Open Image).

B. Training Work Flow

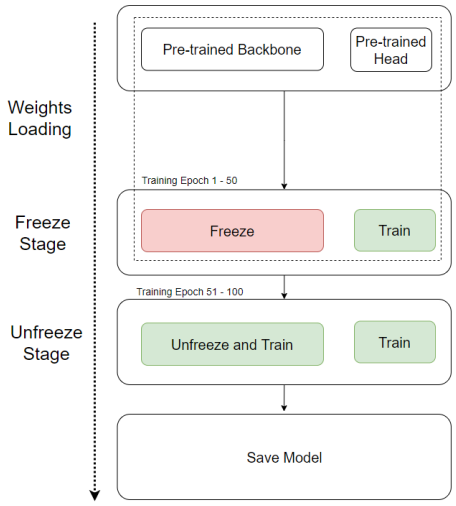

Fig. 12. Training workflow

1. Warm-up stage: Load the weights of the pre-trained model. For the first 50 epochs, the first 60 layers of pre-trained model hidden layers are frozen. These 60 layers of hidden layers are the backbone feature extraction network. The purpose of this is to speed up the training speed, because only the weights of the YOLO head part are updated, which protects the network weights from being influenced at the initial stage of training.

2. Unfreeze stage: Unfreeze the front 60 layers of backbone feature extraction network starting from the 51st epoch, and let the entire neural network participate in training. The training time will be greatly increased, but all weights will be updated in which the purpose is to make the backbone network fully adapt to the dataset of the current task.

VI. Evaluation Metrics

Generally speaking, for neural network models, it is hoped that the network model has a fast recognition speed, a small memory usage, and high detection accuracy. In terms of the current target detection problem, the general commonly used evaluation indicators are mean average precision (map) and floating point operations per second (FLOPS). For the reason that our task only detects objects in the format of image files instead of video streaming, it is no longer needed to evaluate the detecting speed of the YOLO detector in our case, therefore mAP is the only metric we use to evaluate the performance.

A. Precision and Recall

Below follows the definition of precision and recall:

$$\text{Presion} = \frac{tp}{tp + fp}$$

$$\text{Recall} = \frac{tp}{tp + fn}$$

$\boldsymbol{tp}$ represents true positive, $\boldsymbol{fp}$ represents false positive, $\boldsymbol{fn}$ represents false negative. The precision measures of all predictions made the percentage of correct positive predictions. The recall measures of all positive cases in reality the percentage of the correct positives cases are predicted. The trade-off between the two metrics depends on the task goal. In our case, the YOLO object detector is designed for traffic signs and stop signs detection which is related to traffic scenarios that are safety-sensitive, therefore to detects as many positive cases as possible from all positive cases is more important to increase the detection percentage among its own prediction, hence recall should be more primary to consider than precision. However, we only use precision and recall to calculate AP by drawing the PR-Curve.

B. Average Precision (AP) and PR-Curve

The definition of **AP** is defined as the area under the precision-recall curve (PR-curve). For multiple pairs of precision and recalls values, we get them through changing the confidence **score** threshold. The object that has a lower score than the threshold would be detected as a false positive.

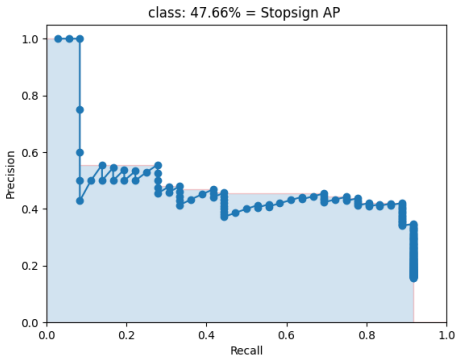

Fig. 13. PR-Curve Example

Given a PR-curve example shows above, the precision goes down once the recall goes up and vice versa. As a simple explanation to this pattern is, as more cases being detected positive cases, the precision increases due to at the same time, the overall detection cases increase as well but in smaller percentages. This would cause the recall to become smaller. Hence, the curve looks monotonically decreasing.

Noted that the later number simply represents the IoU threshold values. For example, AP50 represents the IoU threshold that equals 50%.

C. Mean Average Precision (mAP)

The metric mAP the averaged AP across all categories. In our cases, we have two categories Traffic sign and Stop signs, therefore the mAP in our experiment it simply means the summation of Traffic sign mAP and Stop sign mAP that divided by 2. The mAP is calculated after running the NMS algorithm. The mAP is the final evaluation metric of the detection model. After operations are completed, the final detection result is used to calculate the mAP value.

VII. EXPERIMENT

A. Baseline Model and Tuning

We consider having the model trained based on the original dataset without processing it with distorted quality and treat it as the baseline model. A fine-tune process is also necessary to conduct in order to get the optimal performance for object detection. This allows us to fit the same dataset processed in distorted methods in further step and make it possible for us to compare how the dataset in the variable of distortion types would impact the performance of the baseline model.

B. Learning Rate Annealing

During the training process, the adjustment of the learning rate is best to have. The learning rate represents the speed at which information accumulates in the neural network over time. In an ideal situation, we would start with a large learning rate and gradually reduce the speed until the loss value no longer diverges.

The technique we use is learning rate annealing, which starts with a relatively high learning rate and then slowly reduces the learning rate during training. The idea behind this method is to move from the initial parameter to a "good" range of parameter values. We use the ReduceLROnPlateau callback method of the TensorFlow API to implement this step. This callback monitors a quantity and if no improvement is seen for a 'patience' number of epochs, the learning rate is reduced.

C. Early Stopping

In order to get better generalization performance to prevent the model from overfitting, we use the early stopping method. It is implemented by the TensorFlow **EarlyStopping** callback function. It stops training when a monitored metric has stopped improving based on the 'patience' - number of epochs with no improvement after which training will be stopped. We set it 3 for giving the 3 epochs training threshold times.

D. NMS Parameters Tuning: Score and IoU

Once the YOLO model has finished the training process, it is necessary to fine-tune the IoU threshold and Score threshold to improve the overall performance in mAP metric. We choose the strategy by individually incrementing the values of each parameter in the stride of 0.1 and evaluate its model performance. During the individual tuning of each parameter, we select the current parameter value that can make mAP reach the highest value as the baseline setting. Therefore, for the final tuning result, we expect to get IoU and Score setting as in the format of values of pairs.

TABLE II. SCORE THRESHOLD TUNING RESULTS

Score	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
AP(SS)	84.5	84.5	84.5	84.5	84.5	84.5	84.5	82.0	79.3
AP(TS)	70.4	69.0	66.7	64.5	61.8	58.3	54.1	48.0	36.6
mAP	77.5	76.7	75.6	74.5	73.1	71.4	69.3	65.0	57.9

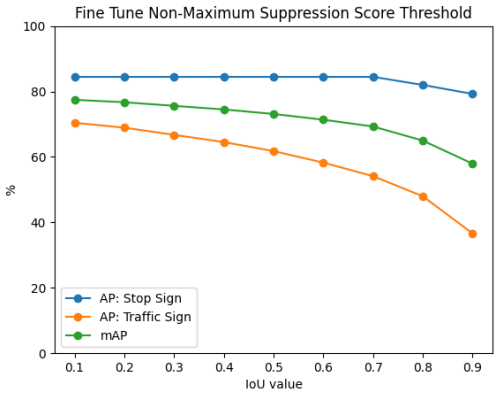

Fig. 14. Score Threshold Tuning Line Graph

It can be seen from Figure 14 that the overall line graph follows a monotonous downward trend which also applies to both categories of the Traffic sign and Stop sign.

The Step sign category has the lowest degree of decline in AP value, and the decline rate is slower, so the curve looks smoother. From table 1, the results show Stop sign performs consistently steady until the score threshold reaches the values that close to the end.

Traffic sign behaves in an opposite way, the AP value is decreasing from the beginning, as the score threshold increases linearly, the AP value decreases more rapidly.

When the score value is 0.1, the mAP value of the YOLO model is the highest and performs best. Therefore 0.1 is selected for the score threshold value for the baseline YOLO model.

TABLE III. IOU THRESHOLD TUNING RESULTS

IoU	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
AP(SS)	84.5	84.5	84.5	84.5	84.5	84.4	78.7	65.1	47.7
AP(TS)	69.4	70.4	70.4	70.7	70.5	70.0	65.7	55.3	40.2
mAP	76.9	77.4	77.5	77.6	77.5	77.2	72.2	60.2	44.0

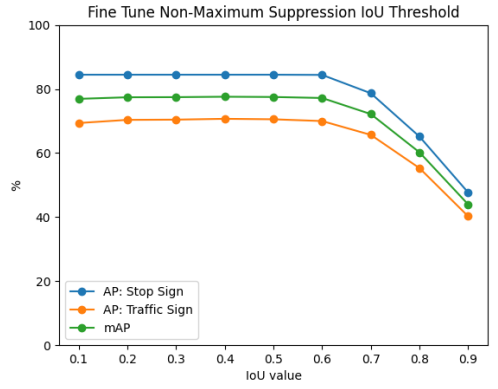

Fig. 15. IoU Threshold Tuning

It can be seen from Figure 15 all three lines display a two-phased behavior. In the first phase, they both demonstrate a smooth upwards trend in values. The increasing rate is too small to be noticed in the figure, but the performance improves from the results table.

In the second phase, the overall mAP and AP of all categories drop rapidly. The starting point for having such a situation is when IoU threshold equals around 0.6.

When the IoU value is 0.3, the mAP value of the YOLO model is the highest and performs best. Therefore, the IoU threshold setting for the baseline YOLO model is 0.3.

E. Score and IoU Threshold Selection Strategy

Both score threshold and IoU threshold are set manually as the parameters for YOLO model, they both have an impact on the final performance of the YOLO model. The specific values to choose in our case are defined to be determined based on the greatest mAP value, in other words, we only care about the absolute values of the mAP but avoid discussing the relation between neighboring values for two thresholds.

One phenomenon observed from the results from Figures 14 and 15 is, the difference among neighbor values of the highest score value and IoU values, are not significant. This implies once the values of score and IoU threshold gets smaller, the performances of the model tend to gets more stable. For the baseline model to be used for the comparing type of experiment, the stability is an essential standard to be primarily considered.

However, it is not strictly necessary to select the set of score and IoU threshold values that produce the highest mAP for the YOLO baseline model, since the exploration is focused on exploration for the possible relationship between distortion images which are examined based on their mAP difference rather than the absolute values. Therefore, the goal for fine-tuning the model is to fetch a trained model that is reasonably stable. This requirement matches what we previous found from the performance trend. Hence, the pair of two thresholds are selected as:

TABLE IV. SCORE AND IOU THRESHOLD SETTING

Model Parameters	*Value*
NMS score	**0.1**
NMS IoU	**0.4**

F. Fitting Dataset in Distortion

As it is mentioned in part 2, the dataset that employed for the experiment contains 3200 images, in which 10 percent of them are 320 images that are selected as the test set. The images in the test set will first be processed by different distortion methods, and then the test images of different types and levels of distortion will be input into the neural network for prediction, and the results will be used to measure the performance of the object detector using the final statistics of mAP.

TABLE V. DISTORTION IMAGE COMPARISON RESULTS

Distortion Type	*Dataset Size*
Original	**320**
Noise100	**320**
Noise200	**320**
Grayscale	**320**
Contrast	**320**
2% Packet Lost	**310(10 broken)**
5% Packet Lost	**307(13 broken)**

Table 3 : Distortion Image Comparison Results

Noted that all 320 test images serve only for testing the performance of the YOLO model. The model is trained and validated by the other 90 percent of the dataset.

G. Results

TABLE VI. DISTORTION IMAGE COMPARISON RESULTS

Category	*Distortion Type*	*AP/mAP*
Stop Sign	**Original**	**84.46**
	Noise100	**99.32**
	Noise200	**92.57**
	Grayscale	**99.03**
	Contrast	**99.26**
	2% Packet Lost	**41.38**
	5% Packet Lost	**30.77**
Traffic Sign	**Original**	**70.69**
	Noise100	**61.29**
	Noise200	**43.55**
	Grayscale	**63.25**
	Contrast	**68.04**
	2% Packet Lost	**21.78**
	5% Packet Lost	**10.14**
Overall	**Original**	**77.58**
	Noise100	**80.31**
	Noise200	**68.06**
	Grayscale	**81.14**
	Contrast	**83.65**
	2% Packet Lost	**31.58**
	5% Packet Lost	**20.46**

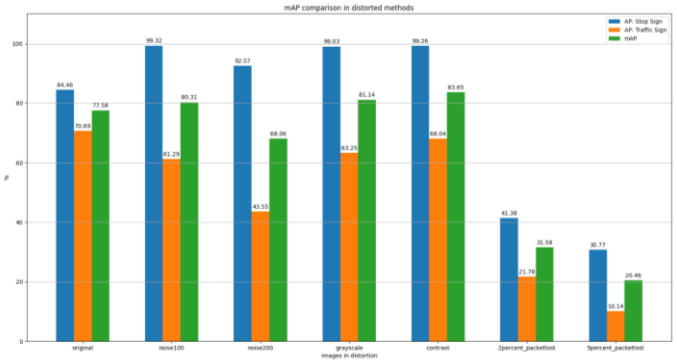

Fig. 16. Distortion Image Comparison Bar Graph

From the experiment results, it is interesting to observe that the object detector is remarkably affected by the images of different distorted types in various behaviors.

There are 3 distortion types that affect the detector in a positive way by increasing the mAP values. These three distortion types are Noise100, Grayscale, and Contrast. However, the increase of values is not significant therefore the impact levels are considered to be limited and weak.

The other three distortion types of images – Noise200, 2percentPackLost, 5percentPackLost – suffer extreme decrease. The 2percentPackLost and 5percentPackLost are even close to half and one-third of the performance with original images.

Horizontally comparing detector's performances with Stop signs (SS) and Traffic signs (TS), one using SS for all distortion types and levels beats what use TS. Surprisingly to observe that the 4 distortion types of images in Noise100, Noise200, Grayscale, and Contrast boosts the performances using SS. In other words, these 4 distortion types improved the detector's ability to predict. However, it works in the opposite way when employs TS and exhibits a similar decrease in performance of overall mAP.

VIII. Discussion

It is not surprising to see that for the images in the last two distortion types, which are 2 percent and 5 percent probability packet loss, all model's performances suffer huge significant damage. The reason can be explained in an intuitive way, the distorted images are too corrupted that it sometimes gets tough for human beings to recognize. The distortion may lead to situations as parts of images would be shifted to the wrong location, which would divide a complete object into several pieces so that a single object could be recognized as multiple objects or the background. Another distortion effect from packet loss is that the damaged areas would lose the RGB information that all pixel values are left set the same number hence lose the feature information. This is similar to fit an input image that is in pure colors which also triggers the malfunction of the object detector.

Another remarkable discovery is about 4 manipulated distortion types. There are disparate outcomes from two categories for fitting these images into the detector. For Stop Signs, the output assessment shows it gets even better performative than fitting the original images. This means a deliberately distorted image in types of noise100, noise200, grayscale, and contrast would be more capable recognized by the machine. Since we only changed the image itself hence the reason can only come from the image. Therefore, we believe one possible explanation for this phenomenon is that the distortion algorithms filtered some less essential features and kept (or even created) more unique features for the Stop Sign.

To help examine our assumptions, we need to compare the results with Traffic signs to ones with Stop signs. However, the mAP values from Traffic Sign images demonstrated an opposite pattern – all types of distortion images harm the performances. The decrease does not supports the previous assumptions that the distortion would help filter the features of the images, otherwise, the performances should be increased.

Based on the observation from the comparison, we believe the nature of the image, which is the category the sign belongs to, is the parameter that decides how the performance trend to become. For images in Stop Sign, due to their shapes, colors and descriptions are all similar across all countries, the first four distortions would not affect too much and benefit the feature extraction based on the experiment. When it comes to Traffic Sign, since they are all largely distinctive in shapes, colors, and descriptions, hence the distortions would enhance the chao of the image nature and increase the diversity of the test images. Therefore, in some way, the distortion images are no longer similar to the original version and hence the distortions would be detrimental.

IX. Conclusion and Future works

The YOLO v4 object detector in CSPDarknet53-tiny is susceptible to distortion images. The scale of such affection is determined by the size of training image sets and the invariance of the nature of the images from their category. A small training set may trigger an overfitting issue for the model but less susceptible to the manipulated distortions. A small training set with a category of objects being nature diverse may suffer a strong negative affection. The packet loss distortion would be detrimental to the model performances of which the harm would be overwhelming significant.

For future works, one possible move is to conduct the same experiment with the replaced dataset in two new categories whereas the size of the source images is much larger than Stop Sign or Traffic Sign. This move is designed to eliminate the possible overfitting issue. Another possible move is to try out more options of distortion types. Not only the noises in different scales can be considered, but also other adversarial image processing algorithms can be used to compare the results.

References

[1] Wang, Chien-Yao, et al. "CSPNet: A new backbone that can enhance learning capability of CNN." Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops. 2020.

[2] Jiang, Zicong, et al. "Real-time object detection method based on improved YOLOv4-tiny." arXiv preprint arXiv:2011.04244 (2020).

[3] Redmon, Joseph, and Ali Farhadi. "YOLO9000: better, faster, stronger." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.

[4] Redmon, Joseph, and Ali Farhadi. "YOLOv3: An incremental improvement." arXiv preprint arXiv:1804.02767 (2018).

[5] Bochkovskiy, Alexey, Chien-Yao Wang, and Hong-Yuan Mark Liao. "YOLOv4: Optimal speed and accuracy of object detection." arXiv preprint arXiv:2004.10934 (2020).

[6] Tan, Mingxing, Ruoming Pang, and Quoc V. Le. "Efficientdet: Scalable and efficient object detection." Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2020.

[7] Lin, Tsung-Yi, et al. "Feature pyramid networks for object detection." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.

[8] Dodge, Samuel, and Lina Karam. "Understanding how image quality affects deep neural networks." 2016 eighth international conference on quality of multimedia experience (QoMEX). IEEE, 2016.

[9] Koziarski, Michał, and Bogusław Cyganek. "Impact of low resolution on image recognition with deep neural networks: An experimental study." International Journal of Applied Mathematics and Computer Science 28.4 (2018): 735-744.