IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received June 7, 2021, accepted June 26, 2021, date of publication July 2, 2021, date of current version July 14, 2021.

*Digital Object Identifier 10.1109/ACCESS.2021.3094201*

# Yolo V4 for Advanced Traffic Sign Recognition With Synthetic Training Data Generated by Various GAN

**CHRISTINE DEWI[1,2], RUNG-CHING CHEN[1], (Member, IEEE), YAN-TING LIU[1], XIAOYI JIANG[3], (Senior Member, IEEE), AND KRISTOKO DWI HARTOMO[2]**

[1]Department of Information Management, Chaoyang University of Technology, Taichung 41349, Taiwan
[2]Faculty of Information Technology, Satya Wacana Christian University, Central Java, Salatiga 50711, Indonesia
[3]Department of Mathematics and Computer Science, University of Münster, 48149 Münster, Germany

Corresponding authors: Rung-Ching Chen (crching@cyut.edu.tw) and Kristoko Dwi Hartomo (kristoko@uksw.edu)

This work was supported in part by the Ministry of Science and Technology, Taiwan, under Grant MOST-110-2927-I-324-501 and Grant MOST-109-2622-E-324-004, in part by the European Union (EU) Horizon 2020 Program RISE Project ULTRACEPT under Grant 778062, and in part by the Education and Culture Ministry Republic Indonesia for Grant Research Penelitian Terapan Unggulan Perguruan Tinggi (PTUPT) at 2019–2021.

**ABSTRACT** Convolutional Neural Networks (CNN) achieves perfection in traffic sign identification with enough annotated training data. The dataset determines the quality of the complete visual system based on CNN. Unfortunately, databases for traffic signs from the majority of the world's nations are few. In this scenario, Generative Adversarial Networks (GAN) may be employed to produce more realistic and varied training pictures to supplement the actual arrangement of images. The purpose of this research is to describe how the quality of synthetic pictures created by DCGAN, LSGAN, and WGAN is determined. Our work combines synthetic images with original images to enhance datasets and verify the effectiveness of synthetic datasets. We use different numbers and sizes of images for training. Likewise, the Structural Similarity Index (SSIM) and Mean Square Error (MSE) were employed to assess picture quality. Our study quantifies the SSIM difference between the synthetic and actual images. When additional images are used for training, the synthetic image exhibits a high degree of resemblance to the genuine image. The highest SSIM value was achieved when using 200 total images as input and 32 × 32 image size. Further, we augment the original picture dataset with synthetic pictures and compare the original image model to the synthesis image model. For this experiment, we are using the latest iterations of Yolo, Yolo V3, and Yolo V4. After mixing the real image with the synthesized image produced by LSGAN, the recognition performance has been improved, achieving an accuracy of 84.9% on Yolo V3 and an accuracy of 89.33% on Yolo V4.

**INDEX TERMS** DCGAN, LSGAN, synthetic images, traffic sign, WGAN, Yolo V3, Yolo V4.

## I. INTRODUCTION

Traffic sign identification has emerged as a critical study area in the science of computer vision in recent years. Moreover, it plays a critical part in advanced driver assistance systems, self-driving vehicles, and traffic safety [1]–[3].

In the previous research, Convolutional Neural Networks (CNN) [4] has achieved very good research results in traffic sign detection and recognition. A considerable quantity of data must be utilized in the neural network to ensure that the model is adequately trained and capable of recognizing traffic signs. As a result, if a significant quantity of labeled data is available, developing a CNN-based traffic detection model should not be difficult. Many researchers have done extensive research and discussion on the identification of traffic signs. They also provide many data sets for public use, such as the German Traffic Signals Dataset (GTSRB) [5], [6], the Chinese Traffic Sign Database (TSRD) and Tsinghua Tencent 100K (TT100K) [7]. Researchers usually use open datasets or collect traffic signs through roads to do their experiments. Nevertheless, obtaining a huge quantity of high-quality images of traffic signs is not straightforward [8], [9]. It takes a considerable amount of time, whether it is using a dashcam or on-site filming. In addition, the design and color

The associate editor coordinating the review of this manuscript and approving it for publication was Krishna Kant Singh.

of traffic signs are different for each country, meaning that it is necessary to collect and mark traffic signs from different countries.

Synthesizing images is a prominent problem in computer vision [10], [11]. To acquire more varied and inexpensive training data, traffic sign pictures generated from standard templates have been routinely employed to train machine learning classification algorithms [12], [13].

Deep Convolutional Generative Adversarial Network (DCGAN) was proposed by Radford *et al.* [14] and Karras *et al.* [15] in 2016. DCGAN combines the Generative Adversarial Network (GAN) with CNN so that all GANs can get better and more stable training results. Other versions of GAN are Least Squares Generative Adversarial Networks (LSGAN) and Wasserstein Generative Adversarial Networks (WGAN) [16], [17]. Both of them can better solve the problem of instability training due to GAN. Each GAN has achieved excellent results in producing synthetic image. Because of the absence of a training dataset, our studies use DCGAN, LSGAN, and WGAN to generate synthetic pictures.

For Traffic Sign Detection (TSD), it is very important to detect small objects at high speeds accurately and quickly. Moreover, CNN can effectively detect and classify object such as Faster R-CNN [18], Single Shot Multibox Detector (SSD) [19], and You Only Look Once (Yolo) [20]. Yolo has the most significant influence under conditions that require faster time detection. It has a high-efficiency detection speed and high accuracy. The newest version of Yolo, Yolo V4 was proposed in 2020 [21]. The majority of modern scientific models need several GPUs for training with large mini-batch size. Usually, when training with one GPU makes the training process slow, heavy, and ineffective. Yolo V4 [21] approaches this problem by constructing an object detector trained on a single GPU with a smaller mini-batch size. This method makes it potential to train a super quick and precise object detector with a single 1080 Ti or 2080 Ti GPU.

This paper analyzes in detailed CNN models and feature extractors, specifically, Yolo V3 and Yolo V4 for object identification. Our study refined them using Taiwan prohibitory sign datasets that we created to detect traffic signs. Our dataset consists of no entry (Class P1), no stopping (Class P2), no parking (Class P3), and speed limit (Class P4). We have been unable to locate a research article that assesses a large number of object detectors based on deep learning that is expressly tuned to the traffic sign recognition problem domain while taking into account numerous crucial aspects such as *mAP, IoU*, and detection time.

The primary contributions of this research are as follows: (1) High-quality prohibitory sign pictures are synthesized using DCGAN, LSGAN, and WGAN. (2) The development of a CNN-based solution for traffic sign classification tasks, as well as the augmentation of the CNN training set using created synthetic data, in order to enhance classification and recognition performance. (3) We proposed an experimental setting with various GAN style to generate synthetic image, after that we evaluate the synthetic image using SSIM and MSE. (4) The Yolo V3 and Yolo V4 model evaluation includes the *mAP*, detection time, *IoU*, and floating-point operations (FLOPS). (5) Experiments show that using synthetic image data generation using various GAN can improve all models' *IoU* and performance.

This research work is structured as follows. Section II discusses contemporarily published works. Section III details our recommended technique. The experiment and its findings are described in Section IV. Detailed discussion about our research describes in Section V. In Section VI, conclusions are stated and recommendations for further study are made.

## II. RELATED WORKS

### A. IMAGE RECOGNITION

Image recognition is an important role in the field of computer vision. For humans, it is easy to recognize objects, but it is very difficult for machines. The machine needs to learn the meaning of each image, observe slowly, and test. According to the learning results, the machine can learn how to recognize the object.

Shijin Song *et al.* [22] proposed a better CNN network architecture that allows small objects to be better detected, with less computation, and easier deployment. They eliminated the CNN network, significantly lowering the model's size and operation time while preserving accuracy. At the same time, the fully connected layer is replaced with a fully convolutional layer, which improves the efficiency of calculation. Tao *et al.* [23] were inspired by the Fully Convolution Network (FCN) and used the combination of Yolo to produce a new optimized Yolo. This combination models allows object detection to provide higher accuracy in traffic scenes. An average accuracy of 69.3% was obtained on the VOC07 and VOC12 datasets, while the traditional Yolo was only 64%.

In traffic scenes, traffic signs are relatively small and make it impossible to detect the sign precisely. Hasegawa *et al.* [24] proposed a better recognition method in complex traffic scenes in Japan. The experiment use Yolo V2 and 5 different image sizes as input. The multiscale images also implemented to make the model steady and train better. Chih-Chung Hsu *et al.* proposed a new architecture based on Densenet [25], called Common Fake Feature Network (CFFN). The architecture uses pairwise learning to optimize the network, and matching real images with fake images. The neural network captures the features and recognize them as fake images and real images well. The cross-entropy loss function used to optimize the classifiers method. Finally, the experimental results are significantly higher than other traditional methods.

### B. VARIOUS GAN

Alec Radford *et al.* assessed the convolutional GAN's architectural and topological restrictions in 2016. The technology, called Deep Convolutional GAN (DCGAN), is more stable in most situations [14], [26].

GAN [27], [28] has two parts that are simultaneously trained, namely generative ($G$) and discriminative *(D).* The discriminatory model is used to detect if a sample contains valid or invalid data. The generative model captures certain target information distribution to puzzle the discriminative model [29], [30]. The D model is a binary classifier that classifies the G model's data in the training system as either realistic or unrealistic. G minimizes its loss function by supplying data that D classifies as real, as modeled by Equation (1).

$$minmax\,(D,G) = E_{x-pdata(x)}\,[logD\,(x)] + E_{x-p(x)}\left[\log(1-D\,(G\,(z)))\right] \quad (1)$$

We employ DCGAN to build a synthetic traffic sign picture in this work [31]. Following that, we will integrate the synthetic image with the real image in order to expand our dataset and enhance traffic sign recognition algorithms. DCGAN is a baseline model, other models build on it by adding additional restrictions or making enhancements.

Numerous research efforts have been directed toward different GAN variations to improve the overall performance of GANs. Brock *et al.* [32] introduced models named BigGANs, which realized the work of generating high-resolution and different images from heterogeneous datasets. The image recognition algorithm can process high-resolution images and a wide range of samples from the difficult dataset, ImageNet. Karras *et al.* [15] suggested an alternative generator design called StyleGAN. The authors designed a new generator architecture that can dynamically vary the style of the generated picture depending on the latest information in all convolutional layers. By starting with low resolution, it helps to guide the full picture synthesis process which begins with low resolution and works its way up to high resolution.

Li and Wand [33] proposed an efficient texture synthesis method named Markovian Generative Adversarial Networks (MGANs). It can decode brown noise straight into a realistic texture, but it can also decode pictures into the painting, which increases the texture synthesis quality. Bergmann *et al.* [34] introduced an architecture named spatial GAN (SGAN) which is very good for texture synthesis. This approach is capable of creating high-quality texture pictures and fusing numerous diverse source photos to create complex textures.

There are two benefits of Least Squares Generative Adversarial Networks (LSGAN) over regular GANs. The first benefit of LSGANs is that they can create higher-quality pictures than normal GANs. Second, LSGANs are more stable in their performance throughout the learning process [35], [36]. In reality, training GANs is a difficult challenge due to the instability of GAN learning. Recently, many articles have shown that the goal function affects the uncertainty of GANs learning [37]. In particular, reducing the usual GAN objective functions can cause gradient loss problems, which makes it difficult to update the generator. LSGANs overcome this barrier by penalizing samples depending on their distances to the decision boundary, which results in more gradients being generated when the generator is updated. Furthermore, demonstrate theoretically that the training instability of standard GANs is attributable to the objective function's mode-seeking tendency, while LSGANs display less mode-seeking activity.

The algorithm of Wasserstein Generative Adversarial Networks (WGAN) [17] has been created to confront the instability in training networks [38], which is believed to be associated with the existence of undesirable sharp gradients of the GAN discriminator function. Yang *et al.* [39] adopted Wasserstein GAN for denoising low-dose CT images and attained a successful application in medical imaging reconstruction. In the synthesis data production module, WGAN is used to produce simulated fault signals in order to incrementally supplement minority fault classes, and the synthetic signals are used to balance the training dataset.

### C. PERFORMANCE EVALUATION OF SYNTHETIC IMAGES

Structural Similarity Index (SSIM) [40], [41] is employed to determine the structural similarity between two photographs. The traditional picture's SSIM assessment method calculates the SSIM index of the local block via the sliding window in the deformed picture. After obtaining the SSIM evaluation value of the whole picture, it normalizes all the local block evaluation indications to produce the overall SSIM evaluation value [41], [42]. The SSIM metrics for brightness, contrast, and structural comparison are presented in Equation (2) and are computed as follows:

$$SSIM\,(x,y) = \frac{\left(2\mu_x\mu_y + C_1\right)\left(2\sigma_{xy} + C_2\right)}{\left(\mu_x^2 + \mu_y^2 + C_1\right)\left(\sigma_x^2 + \sigma_y^2 + C_2\right)} \quad (2)$$

where $\mu_x$ is the average of x, $\mu_y$ is the average of y, $\sigma_x^2$ is the variance of x, $\sigma_y^2$ is the variance of y, and $\sigma_{xy}$ is the covariance of x and y. $C_1$, $C_2$, and $C_2$ are small constants.

The Mean Square Error (MSE) is obtained to compute the difference between estimated values and the real values of the quantity being estimated, which is squared as the square of the difference of pixels. The error is the difference between the estimator's inferred value and the quantity to be estimated as indicated in Equation (3) [11].

$$MSE = \frac{\sum_{i=1}^{n}(P_i - Q_i)^2}{n} \quad (3)$$

where $P_i$ describes the observed value, $Q_i$ represents predicted value, and $n$ is the number of data points.

In this research, DCGAN, LSGAN, and WGAN-generated synthetic images will be assessed using SSIM and MSE. SSIM has a value between $-1$ and 1, which is meaning the higher is better. Smaller MSE values, on the other hand, imply a more positive outcome.

### D. YOLO V3 AND YOLO V4

Yolo V3 was introduced by Redmon and Farhadi [43] in 2016. It splits the input image into ($N \times N$) grids cells [44] with the similar size. Yolo V3 forecast bounding boxes and probabilities for each grid cell. Also, Yolo V3 utilizes multi-scale fusion to provide predictions, and a single neural network is used to gather and preprocess the holistic picture. In the earlier box forecasting, the dimension clusters are used as boxes to which the border boxes are assigned. Furthermore, the K-means algorithm is used to perform dimensional clustering on the objective boxes in the dataset, yielding nine prior boxes of varying sizes that are distributed uniformly among feature graphs of various scales. Additionally, Yolo V3 allows for the establishment of a customized bounding box anchor for each ground truth item [45].

The latest version of Yolo is Yolo V4, released by [21] in 2020. Further, the Yolo V4 structure as follows: (1) Backbone: CSPDarknet53 [46]. (2) Neck: SPP [47], PAN [48]. (3) Head: Yolo V3 [43]. In the backbone, Yolo V4 utilizes a Mish [49] activation function. Mish is a contemporary, soft, and non-monotonic property of the neural activation function, as shown in Equation (4).

$$f(x) = x\tanh(\ln(1 + e^x)) \tag{4}$$

Hence, $\ln(1 + e^x)$ is the soft plus activation function [50].

Complete Yolo V4 specifications are explains as follows: (1) Bag of Freebies (BoF) [51] Backbone: CutMix [52] and Mosaic data augmentation, DropBlock [53] regularization, and Class label smoothing. Detector: CIoU-loss, CmBN, DropBlock regularization, Mosaic data augmentation, Self-Adversarial training, Eliminate grid, Sensitivity, Using multiple anchors for a single ground/truth, Cosine annealing scheduler [54], Optimal hyperparameters, and Random training shapes. (2) Bag of Specials (BoS) Backbone: Mish activation, Cross-stage partial connections (CSP), and Multi input weighted residual connections (MiWRC). Detector: Mish activation, SPP-block, SAM-block, PAN path-aggregation block, and DIoU-NMS.

Yolo algorithm [55] is a standard network design throughout the whole process. This algorithm is shorter than the R-CNN algorithm [56], [57]. Yolo V4 adopts Yolo V3 as a one-stage dense prediction in the head. Further, Yolo V3 divides the input image into $S \times S$ grids cells of the same size [44], predicts bounding boxes and possibilities to every grid cell. Besides, Yolo V3 utilizes multiscale fusion to predict the whole image. This algorithm adopts a single CNN to prepare the entire image. The clusters are used to evaluate boundary lines.

## III. METHODOLOGY

The next sections discuss our system's synthetic data generation methodologies for enhanced traffic sign identification utilizing DCGAN, LSGAN, and WGAN. Figure 1 depicts a high-level overview of system techniques.

The BBox mark tool [58] was adopted to create a bounding box for all signs. All classes are labeled, and a single picture may have many marks. In the detection phase, a single class detector model was used, and each class label corresponds to a single training model. The bounding box labeling tool's return values are object coordinates $(\boldsymbol{x_1},\boldsymbol{y_1},\boldsymbol{x_2},\boldsymbol{y_2})$. These item coordinates are separate from Yolo's input value. The Yolo input value is the center point, width, and height (x, y, w, h). As a consequence, the system's bounding box coordinates in the Yolo input format must be adjusted. The adjustment process is based on Equations (5) – (10).

**TABLE 1.** Yolo V4 Dataset combination.

| Group | Original Image | Synthetic Image | | |
|---|---|---|---|---|
| | | DCGAN | LSGAN | WGAN |
| 1 | ✓ | | | |
| 2 | ✓ | ✓ | | |
| 3 | ✓ | | ✓ | |
| 4 | ✓ | | | ✓ |

$$dw = 1/W \tag{5}$$

$$x = \frac{(x_1 + x_2)}{2} \times dw \tag{6}$$

$$dh = 1/H \tag{7}$$

$$y = \frac{(y_1 + y_2)}{2} \times dh \tag{8}$$

$$w = (x_2 - x_1) \times dw \tag{9}$$

$$h = (y_2 - y_1) \times dh \tag{10}$$

*H* stands for the image's height, dh for the image's absolute height, *W* for the image's width, and *dw* for the image's absolute width. As a result, float values relative to the width and height of the picture (*dw, dh*) might range from 0.0 to 1.0.

In the data preparation step, we employ several GAN to produce synthetic prohibitory sign pictures. Additionally, the dataset was divided into four groups. The first group dataset contains solely original photos. The second group dataset contains both the original picture and a DCGAN-generated synthetic picture. Next, the third groups combined the original image with synthetic image produced by LSGAN. The fourth group employ the dataset that mix real image with synthetic image generation by WGAN. The first group dataset comprises just original images, and the others consists of merging the original image with a synthetic image generated by various GAN. The complete dataset combination shown in Table 1.

Object detection using Yolo V3 and Yolo V4 proceeds as follows.

*Step 1:* Splits the input image into $S \times S$ grids. By calculating anchor boxes, each grid generates K bounding boxes. It predicts B boundary boxes for each grid cell, with one box confidence score for each box.

*Step 2:* Regardless of the number of boxes B, detects just one item. Additionally, forecasts C conditional class probabilities (one per class for the similarity of the object class).

*Step 3:* The system executes the CNN layers to obtain all features from the image and predicts the $b = [b_x, b_y, b_w, b_h, b_c]^T$ and the $class = [class_1, class_2, \ldots, class_c]^T$.

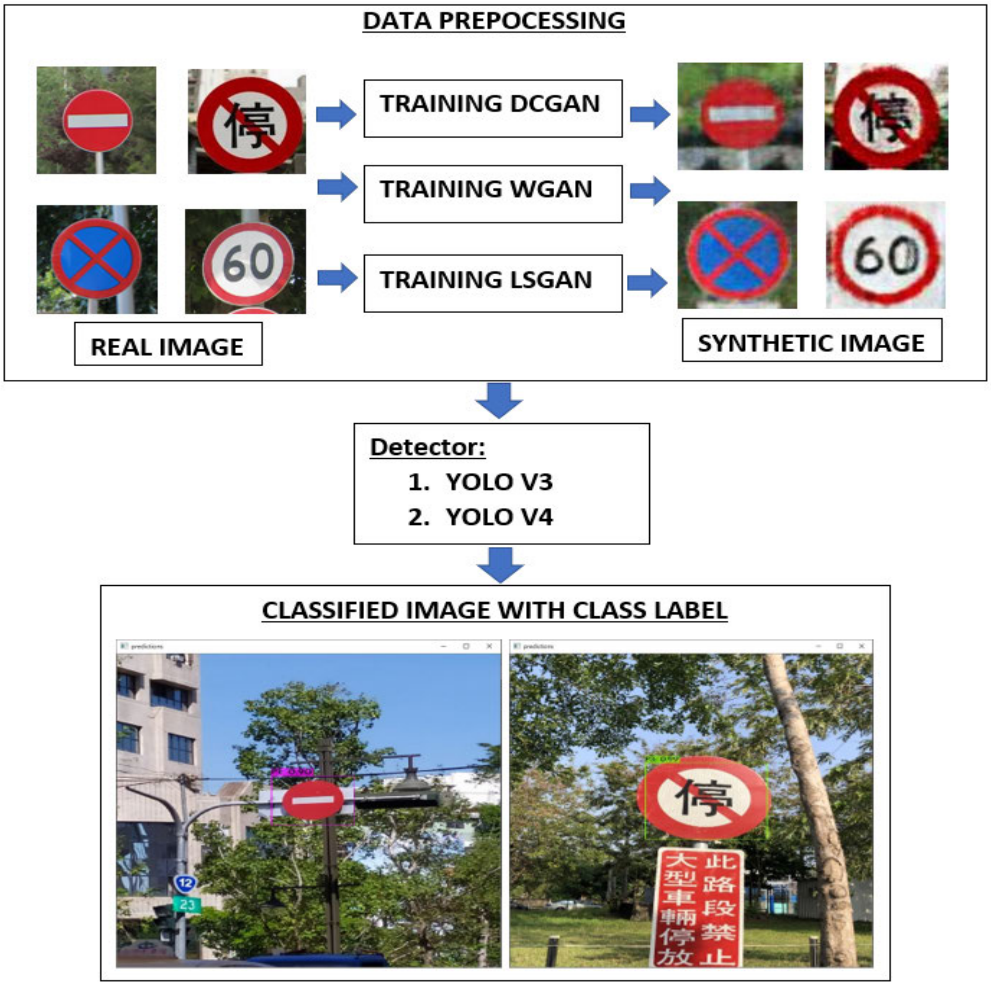


**FIGURE 1.** An overview of the system.

*Step 4:* Compares the optimum confidence $IoU^{truth}_{pred}$ of the K bounding boxes with the threshold $IoU_{thres}$. If $IoU^{truth}_{pred} > IoU_{thres}$, meaning that the bounding box contains the object. The bounding box would not contain the object otherwise.

*Step 5:* After finding out the potential category for the object's category, the computer will identify the specific item. Non-Maximum Suppression (NMS) is employed in our study to search for and uncover potential problem drop boxes, redundant output, and discoveries of object detection.

Additionally, the NMS is implemented as follows: (1) Sort predictions according to their confidence ratings. (2) If we evaluate the same class predictions and IoU $>$ 0.5 with the current prediction, we should start with the top scores and dismiss any current forecast. (3) Further, repeat step 2 to see whether the predictions hold. Before training was completed, our designs were fine-tuned using pre-trained Taiwan prohibitory sign weights, which sped up training by a significant amount.

*Step 6:* The last step results in a categorized image labeled with the class.

### A. DATA GENERATION BY DCGAN, LSGAN, AND WGAN

The DCGAN gives a list of conditions to the CNN backbone network, making sure that the network can be trained, and learning features from previously illustrated pictures can categorize pictures. FDCGAN enhances the picture quality by adopting the following improvements. To begin, DCGAN replaces pooling layers with strangled convolutions on the discriminator and fractional stride convolutions on the generator. CNN is often used to extract characteristics. Second, DCGAN employs the Batch Normalization technique to address the gradient disappearance issue. BN incorporates a gradient propagator in each layer, ensuring that the gradient makes it to each layer, and prohibiting the generator from gathering all samples to the corresponding point. The third issue is that DCGAN uses distinct activation functions, such as Adam optimization, ReLU activation function, and leakyReLU, for distinct neural networks. The findings reveal that DCGAN offers improved performance. DCGAN is widely regarded to be the standard when used in conjunction with other GAN models.

**TABLE 2.** GANs Experiment setting.

| No | Total Image | Image size (px)/ Generate Image (px) | Total Generate Image |
|---|---|---|---|
| 1 | 200 | 64x64 | 1000 |
| 2 | 200 | 32x32 | 1000 |
| 3 | 100 | 64x64 | 1000 |
| 4 | 100 | 32x32 | 1000 |
| 5 | 50 | 64x64 | 1000 |
| 6 | 50 | 32x32 | 1000 |

The advantages of using LSGAN are as follows: (1) LSGAN increases the original GAN loss function by replacing the original cross-entropy loss function with the least-squares loss function. This way corrects two major traditional GAN problems. (3) LSGAN makes the result image quality better, the training process stable, and the convergence speed is faster. ReLu and Leaky ReLu parameters are used in generators and discriminators the same as traditional GAN. On the other hand, the disadvantage of LSGAN is that excessive penalties for outliers lead to reduced sample diversity.

WGAN [59] solves the problem of training instability due to its efficient network architecture. In this model, the sigmoid function eliminates the discriminator's last layer. The Wasserstein distance formula can effectively narrow the generation, distribution and ensure the diversity of the resulting image. The disadvantage of WGAN is that the training time is longer. Due to inappropriate pruning of weight gradient may disappears or exploded [60].

### B. EXPERIMENT SETTING

This study classifies images according to the overall image used to train. The first set consists of 200 photos with 64 × 64 and 32 × 32-pixel dimensions. Then, for each combination of the same size, it will generate 1000 images. The second category uses 100 images of 64 × 64 and 32 × 32 dimensions.

Following that, it will generate 1000 identical photos for each combination. The last group uses 50 images of 64 × 64 and 32 × 32 dimensions. Further, 1000 prints of a similar combination size will be produced. The image size was chosen to reflect the reality that traffic signs often constitute a minor fraction of the picture. Table 2 describes various GANs experiment setting in our work.

## IV. EXPERIMENT RESULTS

### A. DATA GENERATION RESULTS

The training model environment for data generation by various GAN was Nvidia GTX2070 Super GPU accelerator, an AMD Ryzen 7 3700X Central Processing Unit (CPU) with 8 Core Processor, and the RAM is using 32GB DDR4-3200 memory. Further, our method is implemented in Torch and TensorFlow [61]. The training setting is the same for DCGAN, LSGAN and WGAN. The generative network and discriminative network are trained with Adam [20] optimizer with $\beta 1 = 0.5$, $\beta 2 = 0.999$, and learning rate of 0.0002. The batch size is 32, hyperparameter $\lambda$ is set to 0.5 and the normalization method using layer norm. The iterations for training are set as 2000. Then, the total images are 200, 100, and 50. Further, the images sizes are 64 × 64 and 32 × 32, respectively, for input and output. Figure 2, Figure 3, and Figure 4 displays synthetic traffic sign images generated by DCGAN, LSGAN, and WGAN with size (a) 32 × 32, and (b) 64 × 64. Additionally, the picture is pretty authentic, since we cannot determine which picture is fake and which is genuine. The photos seem to be very crisp, natural, and realistic. The synthetic picture created using different GAN techniques will be utilized for training and combined with the actual picture to improve the performance of the traffic sign recognition system.

Our research evaluated the data-generating capabilities of several GANs by comparing the synthesized pictures to their matching actual photos. In order to determine the SSIM value, we produced a picture of a similar type to the original picture and compared it to the original picture. SSIM contains luminosity and contrast masking. However, the error computation requires strong connections between nearby pixels, and the metric is based on short picture windows.

Table 3 represents the complete SSIM and MSE calculation for various GAN. Based on Table 3, the best performance of synthetic image creation is achieved by LSGAN with 200 total images as input, dimensions 32 × 32, and 2000 epoch. These groups obtain the maximum SSIM values at 0.498 and minimum MSE values at 4.453.

Figure 5 describes the SSIM and MSE calculation for original image and synthetic image by DCGAN. All original image in Figure 5a, indicate the same MSE = 0 and SSI = 1. Hence, Figure 5b reveal MSE = 2.16 and SSI = 0.76 for class P1.

### B. DATASET

The dataset is made up of 70% for training and 30% for testing, and the experiment with the Taiwan prohibitory sign was performed using photos that were created, pictures that were synthesized, and a mixture of the two.

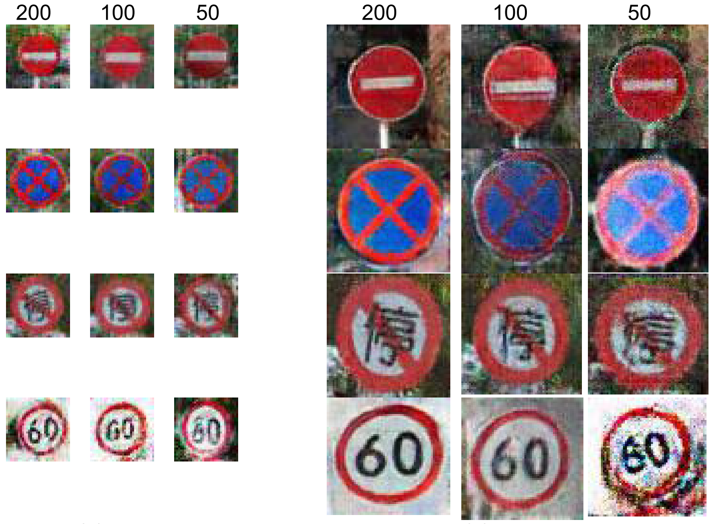
We use 200 synthetic images from various GAN and mix it with real image. The original images consist of 235 no entry images, 250 no stopping images, 185-speed limit images, and 230 no parking images. Next, for the synthetic image, we use 100 images at 64 × 64 sizes and 100 images with 32 × 32 sizes. Moreover, Table 4 explains the Taiwan prohibitory signs in detail. In our experiment Yolo V3 use width = 416, height = 416, and Yolo V4 employ width = 512, height = 512. So, the image is converted to 416 × 416 for Yolo V3 and 512 × 512 for Yolo V4.

### C. YOLO V3 AND YOLO V4 TRAINING RESULTS

During the training stage, our work enhances the Yolo V3 and Yolo V4 model using a learning rate of 0.001 for analysis, learning rate decay of 0.1 at each epoch, and 0.9 for momentum. To solve the over-fitting problem, we implement the cross validation and early stopping in our experiment. 5-fold

**TABLE 3. Various GAN performance evaluations with 2000 epoch.**

| No | Total Image | Image size (px) | DCGAN |  | LSGAN |  | WGAN |  |
|---|---|---|---|---|---|---|---|---|
|  |  |  | MSE | SSIM | MSE | SSIM | MSE | SSIM |
| P1 |  |  |  |  |  |  |  |  |
| 1 | 200 | 64x64 | 9.342 | 0.483 | 8.156 | 0.497 | 7.485 | 0.509 |
| 2 | 200 | 32x32 | 3.502 | 0.558 | 4.019 | 0.529 | 4.009 | 0.533 |
| 3 | 100 | 64x64 | 8.285 | 0.502 | 8.569 | 0.475 | 7.653 | 0.504 |
| 4 | 100 | 32x32 | 5.126 | 0.449 | 4.102 | 0.557 | 4.081 | 0.531 |
| 5 | 50 | 64x64 | 9.776 | 0.366 | 9.93 | 0.336 | 9.39 | 0.41 |
| 6 | 50 | 32x32 | 3.924 | 0.562 | 8.619 | 0.26 | 4.877 | 0.453 |
| P2 |  |  |  |  |  |  |  |  |
| 1 | 200 | 64x64 | 8.969 | 0.385 | 8.96 | 0.436 | 8.385 | 0.475 |
| 2 | 200 | 32x32 | 4.724 | 0.383 | 4.213 | 0.423 | 4.639 | 0.432 |
| 3 | 100 | 64x64 | 8.907 | 0.391 | 8.943 | 0.362 | 8.094 | 0.449 |
| 4 | 100 | 32x32 | 7.999 | 0.123 | 4.408 | 0.402 | 4.425 | 0.393 |
| 5 | 50 | 64x64 | 9.549 | 0.394 | 9.313 | 0.272 | 9.013 | 0.377 |
| 6 | 50 | 32x32 | 4.29 | 0.375 | 4.999 | 0.243 | 4.601 | 0.336 |
| P3 |  |  |  |  |  |  |  |  |
| 1 | 200 | 64x64 | 9.966 | 0.452 | 9.222 | 0.461 | 8.977 | 0.469 |
| 2 | 200 | 32x32 | 5.239 | 0.478 | 4.644 | 0.504 | 4.865 | 0.459 |
| 3 | 100 | 64x64 | 9.895 | 0.392 | 9.941 | 0.38 | 8.321 | 0.478 |
| 4 | 100 | 32x32 | 4.651 | 0.494 | 4.571 | 0.469 | 4.579 | 0.47 |
| 5 | 50 | 64x64 | 10.503 | 0.339 | 12.537 | 0.233 | 9.936 | 0.377 |
| 6 | 50 | 32x32 | 5.484 | 0.434 | 5.874 | 0.363 | 5.503 | 0.391 |
| P4 |  |  |  |  |  |  |  |  |
| 1 | 200 | 64x64 | 10.055 | 0.463 | 11.888 | 0.362 | 9.649 | 0.48 |
| 2 | 200 | 32x32 | 5.698 | 0.453 | 4.934 | 0.535 | 4.89 | 0.504 |
| 3 | 100 | 64x64 | 11.181 | 0.431 | 11.255 | 0.39 | 10.834 | 0.459 |
| 4 | 100 | 32x32 | 7.358 | 0.459 | 5.762 | 0.469 | 5.527 | 0.47 |
| 5 | 50 | 64x64 | 16.311 | 0.326 | 13.637 | 0.313 | 13.035 | 0.405 |
| 6 | 50 | 32x32 | 6.428 | 0.456 | 6.399 | 0.362 | 6.102 | 0.445 |
| **Average** |  |  |  |  |  |  |  |  |
| 1 | 200 | 64x64 | 9.583 | 0.446 | 9.557 | 0.439 | 8.624 | 0.483 |
| **2** | **200** | **32x32** | 4.791 | 0.468 | **4.453** | **0.498** | 4.601 | 0.482 |
| 3 | 100 | 64x64 | 9.567 | 0.429 | 9.677 | 0.402 | 8.726 | 0.473 |
| 4 | 100 | 32x32 | 6.284 | 0.381 | 4.711 | 0.474 | 4.653 | 0.466 |
| 5 | 50 | 64x64 | 11.535 | 0.356 | 11.354 | 0.289 | 10.344 | 0.392 |
| 6 | 50 | 32x32 | 5.032 | 0.457 | 6.473 | 0.307 | 5.271 | 0.406 |



(a) 32×32 (b) 64×64

**FIGURE 2. Synthetic images generated by DCGAN with size (a) 32 × 32, and (b) 64 × 64.**

cross-validation is a conventional procedure to obtain out-of-sample prediction error. Early stopping rules indicate the maximum number of iterations that may be performed before the learner becomes to overfit.

This experiment applies *max_batches* = 8000 iterations, policy = steps, steps = 6400, 7200, scales = 0.1,0.1, momentum = 0.949, decay = 0.0005, and mosaic = 1. Typically, m-class object detectors need 2000 × m as the maximum

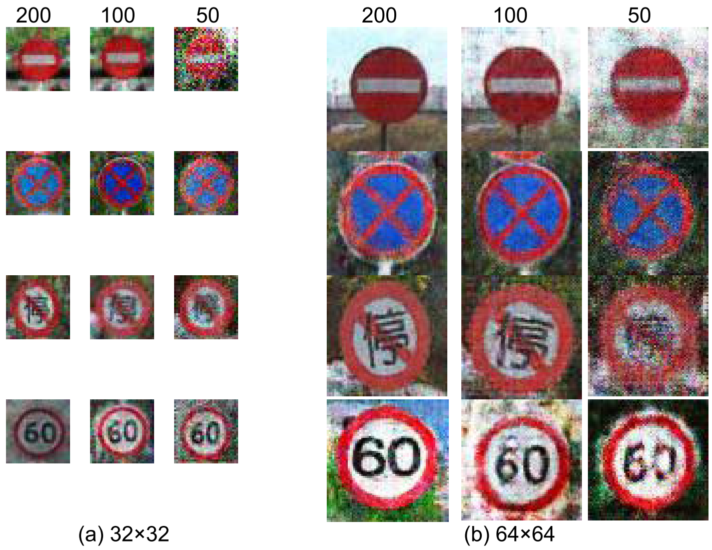


**FIGURE 3.** Synthetic images generated by LSGAN with size (a) 32 × 32, and (b) 64 × 64.

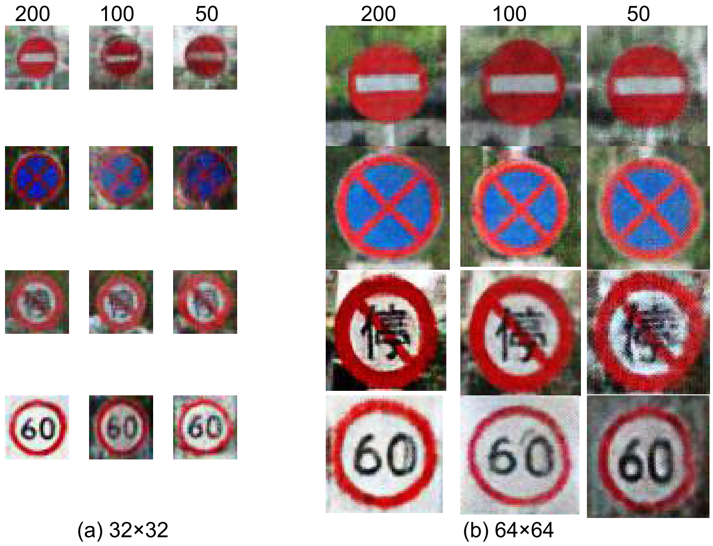


**FIGURE 4.** Synthetic images generated by WGAN with size (a) 32 × 32, and (b) 64 × 64.

batches. In the experiment, the training process stops at 8000 iterations (2000 × 4 classes). Further, the scale (0.1, 0.1) and the current iteration number 0.001 batches are used in the training process. The calculation of the current learning rate becomes learning rate × scales [0] × scales [1] = 0.00001, and the learning rate value will be updated regularly.

*IoU* calculates the overlap ratio between the boundary box of the prediction (*pred*), ground-truth (*gt*), and shown in Equation (11) [62], [63].

$$IoU = \frac{Area_{pred} \cap Area_{gt}}{Area_{pred} \cup Area_{gt}} \tag{11}$$

Nevertheless, the output examples can be classified into three classes. True positive (TP) is the number of correctly recognized samples; false positive (FP), which assigns to the number of samples with incorrect identification; true negative (TN) is the number of unrecognized samples. Precision and recall are represented by [64], [65] in Equation (12)-(13).

$$Precision(P) = \frac{TP}{TP + FP} \tag{12}$$

$$Recall(R) = \frac{TP}{TP + FN} \tag{13}$$

Another evaluation index, F1 [66]–[68] is shown in Equation (14).

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{14}$$

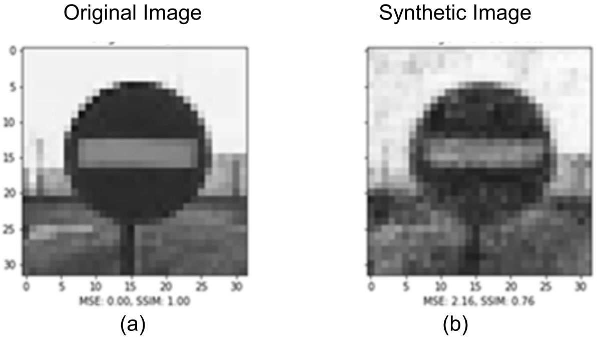


**FIGURE 5.** SSIM and MSE calculation.

**TABLE 4.** Taiwan prohibitory sign.

| Class ID | Class Name | Sign | Original Image | Synthetic Image DCGAN, LSGAN, WGAN |
|---|---|---|---|---|
| P1 | No entry | | 235 | 200 |
| P2 | No stopping | | 250 | 200 |
| P3 | No parking | | 230 | 200 |
| P4 | Speed Limit | | 185 | 200 |

Yolo loss function based on Equations (15) [69].

$$\lambda_{coord}\sum_{i=0}^{s^2}\sum_{j=0}^{B}\mathbb{I}_{ij}^{obj}\left[\left(x_i-\hat{x}_i\right)^2+\left(\mathrm{y}-\hat{y}_i\right)^2\right]$$
$$+\boldsymbol{\lambda_{coord}}\sum_{\boldsymbol{i=0}}^{\boldsymbol{s^2}}\sum_{\boldsymbol{j=0}}^{\boldsymbol{B}}\mathbb{I}_{\boldsymbol{ij}}^{\boldsymbol{obj}}\left[\left(\sqrt{\boldsymbol{w_i}}-\sqrt{\boldsymbol{\hat{w}_i}}\right)^2+\left(\sqrt{\boldsymbol{h_i}}-\sqrt{\boldsymbol{\hat{h}_i}}\right)^2\right]$$
$$+\sum_{i=0}^{s^2}\sum_{j=0}^{B}\mathbb{I}_{ij}^{obj}\left(C_i-\hat{C}_i\right)^2$$
$$+\lambda_{\text{noobj}}\sum_{i=0}^{s^2}\sum_{j=0}^{B}\mathbb{I}_{ij}^{\text{noobj}}\left(C_i-\hat{C}_i\right)^2$$
$$+\sum\nolimits_{i=0}^{s^2}\mathbb{I}_i^{obj}\sum\nolimits_{c\epsilon classes}\left(p_i(\mathrm{c})-\hat{p}_i(c)\right)^2 \qquad (15)$$

where $\mathbb{I}_{\boldsymbol{ij}}^{\boldsymbol{obj}}$ denotes if the object appears in cell $i$, and $\mathbb{I}_{\boldsymbol{ij}}^{\boldsymbol{obj}}$ denotes that the $\boldsymbol{j^{th}}$ bounding box predictor in cell $i$ is responsible for the prediction. Next, $\left(\boldsymbol{\hat{x}},\boldsymbol{\hat{y}},\boldsymbol{\hat{w}},\boldsymbol{\hat{h}},\boldsymbol{\hat{c}},\boldsymbol{\hat{p}}\right)$ are used to express the anticipated bounding box's center coordinates, width, height, confidence, and category probability. True labels are those without the cusp. Furthermore, our works set the $\boldsymbol{\lambda coord}$ to 0.5, indicating that the width and height errors are less effective in the calculation. Then, $\boldsymbol{\lambda noobj} = 0.5$ is used to mitigate the effect of several grids devoid of objects on the loss value.

Additionally, this study separated the model into four groups, each with its dataset. The first group uses just synthetic pictures, whereas the second group uses both the actual picture and the DCGAN synthetic picture. Next, the third groups utilize the original image mix with LSGAN synthetic image and the last groups combined with synthetic image generated by WGAN. Training performance result shown in Table 5. Moreover, Group 3 (original images, LSGAN) dataset obtains the maximum *mAP*, around 99.98% with *IoU* 90.35% for Yolo V4, followed by Yolo V3 at 99.83% with *IoU* 73.11%. As a result, Table 5 demonstrates that combining authentic and synthetic pictures strengthens all models and increases the IoU and mAP percentages. This study used IoU to determine the extent to which our anticipated border overlaps with the ground truth, which is the boundary of the actual object. Yolo V4 demonstrated superior mAP over Yolo V3 in almost all testing groups.

## V. DISCUSSIONS

We use sixty prohibitory sign pictures to test Yolo V3 and Yolo V4 in various sizes and environments. Table 6 presents the testing accuracy results performance of the experiments using the images beyond our datasets. Yolo V4 is generally more accurate than other versions. Yolo V4 increases the accuracy of Yolo V3 in all groups except group 1. The highest average accuracy is Group 3 (Original Image, LSGAN) with the accuracy of Yolo V4 model at 89.33%, followed by Yolo V3 at 84.9%. Previously there are seven errors detection on Yolo V3, but using Yolo V4 there are only two detection error in Group 3. The second highest model is achieved by Group 4 (Original Image, WGAN) with 88.2% accuracy for Yolo V4 and 77.41% accuracy for Yolo V3. In Group 4, there were seven detection errors on Yolo V3 and two detection errors on Yolo V4. The worst group in our experiment was Group 1 which only used the original image. Group 3 shows the optimal accuracy because it uses the LSGAN synthetic image combined with original images. As shown in Table 3, the synthetic image performance evaluation by LSGAN has a maximum SSIM value and a minimum MSE. Hence, this affects the recognition performance result.

The big dataset, which contains both the original picture and the synthetic picture created by several GANs, will improve the detection and recognition performance of both versions. Overfitting occurs when a neural network is trained with an inadequate dataset. Small datasets may also act as a mapping impediment for neural networks when they attempt to find the object. One technique that makes the images simpler to learn the input picture is to apply noise or create a synthetic image during preparation. The inclusion of noise during training will improve the training phase and minimize general errors. Therefore, combining original images and synthetic images in the dataset improves object recognition performance.

Figure 6 shows the result of Group 3 (Original Image, LSGAN). Furthermore, Figure 6(a), Figure 6(c), and Figure 6(e) displays the testing result using Yolo V3. Next, Figure 6(b), Figure 6(d), and Figure 6(f) explains the experiment result employing Yolo V4. For Yolo V3 Figure 6(a) obtains

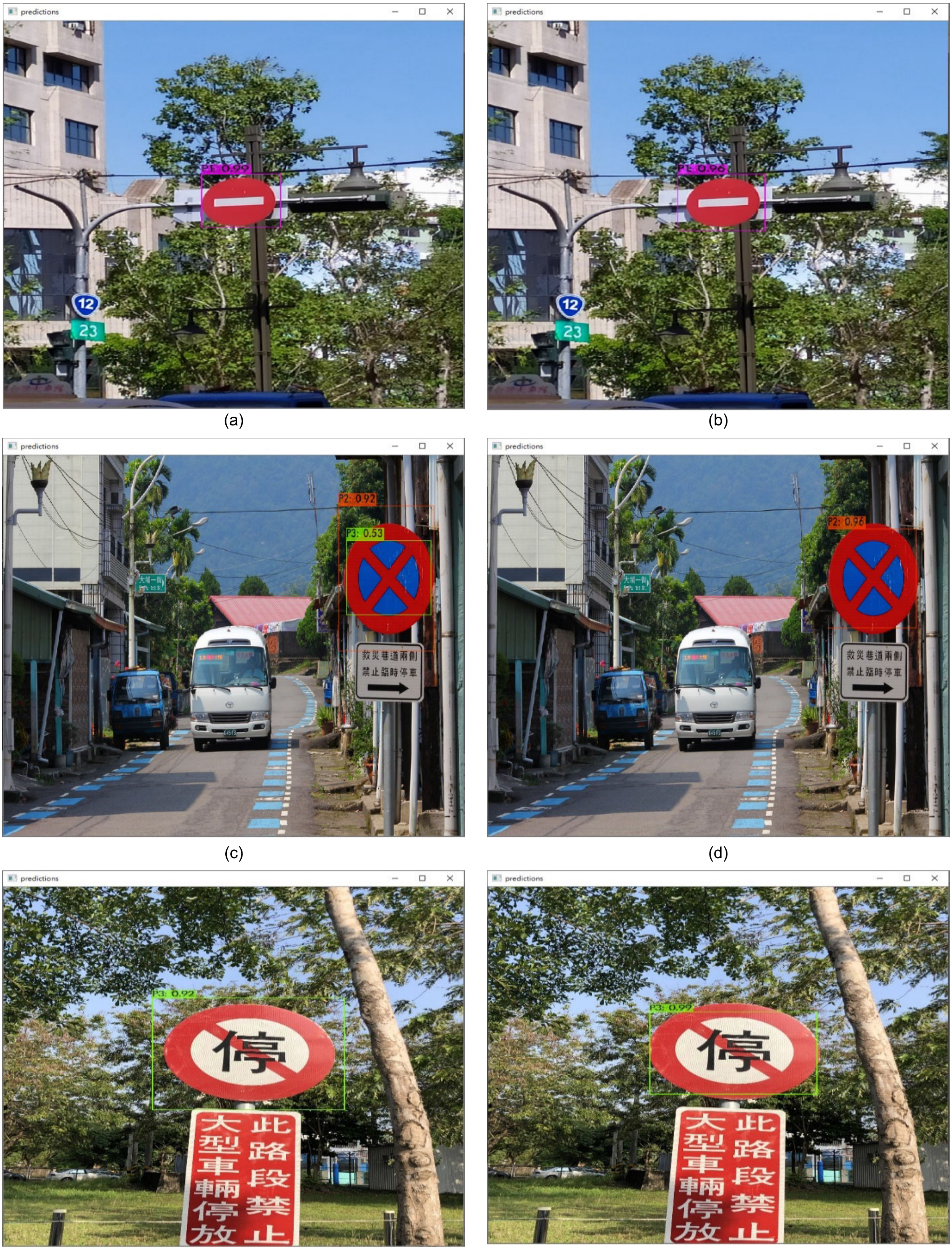
**TABLE 5. Training performance results.**

| Model | Dataset | Loss Value | Name | AP (%) | TP | FN | FP | Recall | Precision | IoU (%) | F1-score | mAP (%) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Yolo V3 | Group 1 (Original Image) | 0.0162 | P1 | 97.5 | 299 | 2 | 3 | 0.99 | 0.99 | 90.93 | 0.99 | 99.08 |
| | | | P2 | 100 | | | | | | | | |
| | | | P3 | 99.8 | | | | | | | | |
| | | | P4 | 99.04 | | | | | | | | |
| Yolo V4 | Group 1 (Original Image) | 0.1441 | P1 | 98.75 | 299 | 2 | 3 | 0.99 | 0.99 | 92.4 | 0.99 | 99.55 |
| | | | P2 | 100 | | | | | | | | |
| | | | P3 | 100 | | | | | | | | |
| | | | P4 | 99.46 | | | | | | | | |
| Yolo V3 | Group 2 (Original Image, DCGAN) | 0.0269 | P1 | 94.9 | 558 | 10 | 12 | 0.98 | 0.98 | 86.45 | 0.98 | 98.44 |
| | | | P2 | 100 | | | | | | | | |
| | | | P3 | 99.97 | | | | | | | | |
| | | | P4 | 98.88 | | | | | | | | |
| Yolo V4 | Group 2 (Original Image, DCGAN) | 0.3601 | P1 | 97.25 | 563 | 5 | 6 | 0.99 | 0.99 | 88.33 | 0.99 | 99.07 |
| | | | P2 | 100 | | | | | | | | |
| | | | P3 | 100 | | | | | | | | |
| | | | P4 | 99.05 | | | | | | | | |
| Yolo V3 | Group 3 (Original Image, LSGAN) | 0.0217 | P1 | 99.99 | 564 | 4 | 7 | 1 | 0.99 | 89.48 | 1 | 99.83 |
| | | | P2 | 100 | | | | | | | | |
| | | | P3 | 98.23 | | | | | | | | |
| | | | P4 | 99.99 | | | | | | | | |
| **Yolo V4** | **Group 3 (Original Image, LSGAN)** | 0.2173 | P1 | 100 | 566 | 2 | 5 | 1 | 0.99 | 90.35 | 0.99 | **99.98** |
| | | | P2 | 99.33 | | | | | | | | |
| | | | P3 | 100 | | | | | | | | |
| | | | P4 | 100 | | | | | | | | |
| Yolo V3 | Group 4 (Original Image, WGAN) | 0.025 | P1 | 98.63 | 565 | 2 | 5 | 1 | 0.99 | 89.51 | 0.99 | 99.51 |
| | | | P2 | 100 | | | | | | | | |
| | | | P3 | 99.98 | | | | | | | | |
| | | | P4 | 99.43 | | | | | | | | |
| Yolo V4 | Group 4 (Original Image, WGAN) | 0.2239 | P1 | 98.63 | 565 | 2 | 4 | 1 | 0.99 | 90.4 | 0.99 | 99.45 |
| | | | P2 | 100 | | | | | | | | |
| | | | P3 | 99.99 | | | | | | | | |
| | | | P4 | 99.2 | | | | | | | | |

**TABLE 6. Testing accuracy results performance.**

| Dataset | Accuracy (%) | | Not detect | |
|---|---|---|---|---|
| | Yolo V3 | Yolo V4 | Yolo V3 | Yolo V4 |
| Group 1 (Original Image) | 84.31 | 69.7 | 7 | 8 |
| Group 2 (Original Image, DCGAN) | 74.34 | 82.5 | 7 | 7 |
| **Group 3 (Original Image, LSGAN)** | **84.9** | **89.33** | **7** | **2** |
| Group 4 (Original Image, WGAN) | 77.41 | 88.2 | 7 | 4 |

the highest accuracy 99%, followed by Figure 6(b) 92%, and Fig. 6(c) 92%. Next, the optimum accuracy of Yolo V4 gained by Figure 6(f) 99%, followed by Figure 6(b) 96%, and Figure 5(d) 96%. Therefore, from the test result in Figure 6, we can summarize that every model can identify all class properly with various bounding box coordinate and accuracy. Figure 7 represents the result of Group 4 (Original Image, WSGAN). Hence, Figure 7(a), Figure 7(c), and Figure 7(e) displays the testing result using Yolo V3. Then, Figure 7(b), Figure 7(d), and Figure 7(f) explains the experiment result employing Yolo V4. Figure 7 represents the recognition result employing Group 4 (Original Image, WGAN) dataset. The Yolo V3 underperforms on Figure 7 (c) with an accuracy rate of 42%. Using the same image Yolo V4 exhibits the



(a) (b) (c) (d) (e) (f)

**FIGURE 6.** The recognition results of Group 3 (Original Image, LSGAN) using Yolo V3 and Yolo V4.

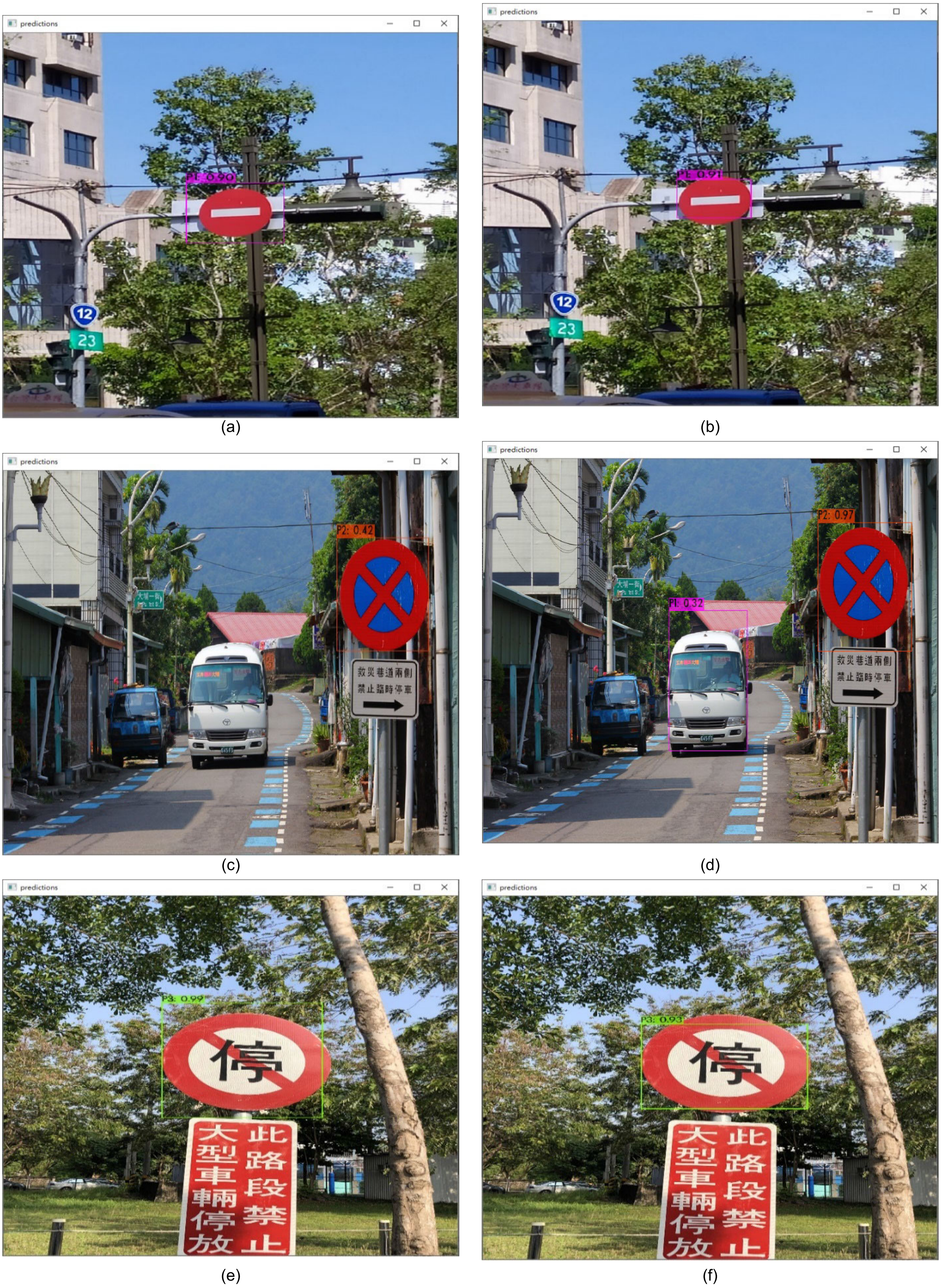


**FIGURE 7.** The recognition results of Group 4 (Original Image, WGAN) using Yolo V3 and Yolo V4.

maximum accuracy rate of 97% in Figure 7 (d). However, in Figure 7(d), Yolo V4 exhibited false detection.

## VI. CONCLUSION

The major goal of this study is to explore how the quality of synthetic pictures generated by DCGAN, LSGAN, and WGAN. Our work combines synthetic images with original images to enhance datasets and verify the effectiveness of synthetic datasets. We use different numbers and sizes of images for training. Our research investigates and analyses CNN models for object identification when paired with different backbone architectures and extractor features, notably Yolo V3 and Yolo V4. This experiment investigates the detector's primary characteristics, such as precision accuracy, detection time, workspace size, and BFLOP numbers. Meanwhile, we are developing a CNN-based solution for traffic sign classification and expanding the CNN training set using synthetic data collected to improve classification and identification outcomes. Yolo V4 is generally more accurate than other models using original image and synthetic image produced by LSGAN. Our study demonstrates that training with a mixture of original and synthetic pictures improves traffic sign identification ability.

Based on our experiment result we summarized as follows: (1) The best dataset in the experiment is Group 3, combined the original image with synthetic image produced by LSGAN. (2) The highest SSIM value was achieved when using 200 total images as input and 32 × 32 image size. (3) After combining the original image with the synthesized image produced by LSGAN, the recognition performance has developed, obtaining an accuracy of 84.9% on Yolo V3 and an accuracy of 89.33% on Yolo V4. (4) The addition of noise during training will gain the training phase and minimize general errors. Hence, integrating the various original images and synthesized images in the dataset improves object identification performance.

In the future, we want to combine synthetic images of different sizes for training. Currently, only images with a total input of 200 and 2000 epochs were used. Through a model trained on synthetic images with different sizes, we will understand about the size or quality of synthetic images affects the model. We will compare with other traffic sign benchmarks to reflect the advantages of synthetic imagery. Future research also tries other detection methods combined with Explainable AI (XAI) and the other GAN method (BigGAN, styleGAN, MGAN).

## REFERENCES

[1] H. Luo, Q. Kong, and F. Wu, "Traffic sign image synthesis with generative adversarial networks," in *Proc. Int. Conf. Pattern Recognit. (ICPR)*, Aug. 2018, pp. 2540–2545.

[2] C. Liu, S. Li, F. Chang, and Y. Wang, "Machine vision based traffic sign detection methods: Review, analyses and perspectives," *IEEE Access*, vol. 7, pp. 86578–86596, 2019.

[3] C. Dewi, R.-C. Chen, and S.-K. Tai, "Evaluation of robust spatial pyramid pooling based on convolutional neural network for traffic sign recognition system," *Electronics*, vol. 9, no. 6, p. 889, May 2020.

[4] M. Haloi, "Traffic sign classification using deep inception based convolutional networks," *CoRR*, vol. abs/1511.0, pp. 1–5, 2015.

[5] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "The German traffic sign recognition benchmark: A multi-class classification competition," in *Proc. Int. Joint Conf. Neural Netw.*, Jul. 2011, pp. 1453–1460.

[6] C. Dewi, R. C. Chen, and Y.-T. Liu, "Taiwan stop sign recognition with customize anchor," in *Proc. ICCMS*, Brisbane, QLD, Australia, Feb. 2020, pp. 51–55.

[7] Z. Zhu, D. Liang, S. Zhang, X. Huang, B. Li, and S. Hu, "Traffic-sign detection and classification in the wild," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2110–2118.

[8] A. Mukherjee, A. Joshi, S. Sarkar, and C. Hegde, "Attribute-controlled traffic data augmentation using conditional generative models," in *Proc. CVPR Workshop Vis. All Seasons*, 2019, pp. 83–87.

[9] L. Abdi and A. Meddeb, "Deep learning traffic sign detection, recognition and augmentation," in *Proc. Symp. Appl. Comput.*, Apr. 2017, pp. 131–136.

[10] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-attention generative adversarial networks," in *Proc. 36th Int. Conf. Mach. Learn. (ICML)*, 2019, pp. 12744–12753.

[11] C. Dewi, R.-C. Chen, and H. Yu, "Weight analysis for various prohibitory sign detection and recognition using deep learning," *Multimedia Tools Appl.*, vol. 79, nos. 43–44, pp. 32897–32915, Nov. 2020.

[12] H. Luo, Y. Yang, B. Tong, F. Wu, and B. Fan, "Traffic sign recognition using a multi-task convolutional neural network," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 4, pp. 1100–1111, Apr. 2018.

[13] C. Dewi, R.-C. Chen, Hendry, and Y.-T. Liu, "Similar music instrument detection via deep convolution YOLO-generative adversarial network," in *Proc. IEEE 10th Int. Conf. Awareness Sci. Technol. (iCAST)*, Oct. 2019, pp. 1–6.

[14] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional GANs," in *Proc. Int. Conf. Learn. Represent.*, 2016, pp. 1–16.

[15] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4401–4410.

[16] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley, "Least squares generative adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 2813–2821.

[17] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proc. 34th Int. Conf. Mach. Learn. (ICML)*, 2017, pp. 298–321.

[18] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[19] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in *Proc. Eur. Conf. Comput. Vis.*, in Lecture Notes in Computer Science: Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics, 2016, pp. 21–37.

[20] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 1–10.

[21] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal speed and accuracy of object detection," 2020, *arXiv:2004.10934*. [Online]. Available: http://arxiv.org/abs/2004.10934

[22] S. Song, Z. Que, J. Hou, S. Du, and Y. Song, "An efficient convolutional neural network for small traffic sign detection," *J. Syst. Archit.*, vol. 97, pp. 269–277, Aug. 2019.

[23] J. Tao, H. Wang, X. Zhang, X. Li, and H. Yang, "An object detection system based on YOLO in traffic scene," in *Proc. 6th Int. Conf. Comput. Sci. Netw. Technol. (ICCSNT)*, Oct. 2017, pp. 315–319.

[24] R. Hasegawa, Y. Iwamoto, and Y.-W. Chen, "Robust detection and recognition of japanese traffic sign in the complex scenes based on deep learning," in *Proc. IEEE 8th Global Conf. Consum. Electron. (GCCE)*, Oct. 2019, pp. 575–578.

[25] C.-C. Hsu, Y.-X. Zhuang, and C.-Y. Lee, "Deep fake image detection based on pairwise learning," *Appl. Sci.*, vol. 10, no. 1, p. 370, Jan. 2020.

[26] Q. Mao, H.-Y. Lee, H.-Y. Tseng, S. Ma, and M.-H. Yang, "Mode seeking generative adversarial networks for diverse image synthesis," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 1429–1437.

[27] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets (NIPS version)," in *Proc. Adv. Neural Inf. Process. Syst. 27*, vol. 27, 2014, pp. 2672–2680.

[28] D. Bau, J.-Y. Zhu, H. Strobelt, B. Zhou, J. B. Tenenbaum, W. T. Freeman, and A. Torralba, "GaN dissection: Visualizing and understanding generative adversarial networks," in *Proc. 7th Int. Conf. Learn. Represent. (ICLR)*, 2019, pp. 1–18.

[29] G. Wei, M. Luo, H. Liu, D. Zhang, and Q. Zheng, "Progressive generative adversarial networks with reliable sample identification," *Pattern Recognit. Lett.*, vol. 130, pp. 91–98, Feb. 2020.

[30] D. Volkhonskiy, I. Nazarov, and E. Burnaev, "Steganographic generative adversarial networks," in *Proc. 12th Int. Conf. Mach. Vis. (ICMV)*, vol. 11433, Jan. 2020, pp. 1–12.

[31] C. Dewi, R.-C. Chen, Y.-T. Liu, and S.-K. Tai, "Synthetic data generation using DCGAN for improved traffic sign recognition," *Neural Comput. Appl.*, vol. 33, no. 8, pp. 1–15, 2021.

[32] A. Brock, J. Donahue, and K. Simonyan, "Large scale GaN training for high fidelity natural image synthesis," in *Proc. 7th Int. Conf. Learn. Represent. (ICLR)*, 2019, pp. 1–35.

[33] C. Li and M. Wand, "Precomputed real-time texture synthesis with Markovian generative adversarial networks," in *Proc. Eur. Conf. Comput. Vis.*, in Lecture Notes in Computer Science: Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics, vol. 9907, 2016, pp. 702–716.

[34] U. Bergmann, N. Jetchev, and R. Vollgraf, "Learning texture manifolds with the periodic spatial GAN," vol. 1, 2017, *arXiv:1705.06566*. [Online]. Available: http://arxiv.org/abs/1705.06566

[35] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley, "On the effectiveness of least squares generative adversarial networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 12, pp. 2947–2960, Dec. 2019.

[36] R.-C. Chen, C. Dewi, W.-W. Zhang, and J.-M. Liu, "Integrating gesture control board and image recognition for gesture recognition based on deep learning," *Int. J. Appl. Sci. Eng.*, vol. 17, no. 3, pp. 237–248, 2020.

[37] G.-J. Qi, "Loss-sensitive generative adversarial networks on Lipschitz densities," *Int. J. Comput. Vis.*, vol. 128, no. 5, pp. 1118–1140, May 2020.

[38] W. Wang, C. Wang, T. Cui, and Y. Li, "Study of restrained network structures for wasserstein generative adversarial networks (WGANs) on numeric data augmentation," *IEEE Access*, vol. 8, pp. 89812–89821, 2020.

[39] Q. Yang, P. Yan, Y. Zhang, H. Yu, Y. Shi, X. Mou, M. K. Kalra, Y. Zhang, L. Sun, and G. Wang, "Low-dose CT image denoising using a generative adversarial network with wasserstein distance and perceptual loss," *IEEE Trans. Med. Imag.*, vol. 37, no. 6, pp. 1348–1357, Jun. 2018.

[40] H. Zhao, O. Gallo, I. Frosio, and J. Kautz, "Loss functions for image restoration with neural networks," *IEEE Trans. Comput. Imag.*, vol. 3, no. 1, pp. 47–57, Mar. 2017.

[41] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.

[42] A. Deshmukh and J. Sivaswamy, "Synthesis of optical nerve head region of fundus image," in *Proc. IEEE 16th Int. Symp. Biomed. Imag. (ISBI)*, Apr. 2019, pp. 583–586.

[43] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," *CoRR*, vol. abs/1804.02767, pp. 1–6, Apr. 2018.

[44] H. Chen, Z. He, B. Shi, and T. Zhong, "Research on recognition method of electrical components based on YOLOv3," *IEEE Access*, vol. 7, pp. 157818–157829, 2019.

[45] L. Zhao and S. Li, "Object detection algorithm based on improved YOLOv3," *Electronics*, vol. 9, no. 3, p. 537, Mar. 2020.

[46] C.-Y. Wang, H.-Y. Mark Liao, Y.-H. Wu, P.-Y. Chen, J.-W. Hsieh, and I.-H. Yeh, "CSPNet: A new backbone that can enhance learning capability of CNN," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2020, p. 2.

[47] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1904–1916, Sep. 2015.

[48] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8759–8768.

[49] D. Misra, "Mish: A self regularized non-monotonic neural activation function," 2019, *arXiv:1908.08681*. [Online]. Available: https://arxiv.org/abs/1908.08681

[50] Q. Liu and S. Furber, "Noisy softplus: A biology inspired activation function," in *Proc. Int. Conf. Neural Inf. Process.*, in Lecture Notes in Computer Science, 2016, pp. 405–412.

[51] Z. Zhang, T. He, H. Zhang, Z. Zhang, J. Xie, and M. Li, "Bag of freebies for training object detection neural networks," 2019, pp. 1–9, *arXiv:1902.04103*. [Online]. Available: http://arxiv.org/abs/1902.04103

[52] S. Yun, D. Han, S. Chun, S. J. Oh, Y. Yoo, and J. Choe, "CutMix: Regularization strategy to train strong classifiers with localizable features," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 6022–6031.

[53] G. Ghiasi, T. Y. Lin, and Q. V. Le, "DropBlock: A regularization method for convolutional networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 10727–10737.

[54] I. Loshchilov and F. Hutter, "SGDR: Stochastic gradient descent with warm restarts," in *Proc. 5th Int. Conf. Learn. Represent. (ICLR)*, 2017, pp. 1–16.

[55] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. 30th IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6517–6525.

[56] S. Ren, K. He, and R. Girshick, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 91–99.

[57] Q. Xu, R. Lin, H. Yue, H. Huang, Y. Yang, and Z. Yao, "Research on small target detection in driving scenarios based on improved YOLO network," *IEEE Access*, vol. 8, pp. 27574–27583, 2020.

[58] (2019). *BBox Label Tool*. [Online]. Available: https://github.com/puzzledqs/BBox-Label-Tool

[59] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GaN," 2017, *arXiv:1701.07875*. [Online]. Available: https://arxiv.org/abs/1701.07875

[60] Y.-J. Cao, L.-L. Jia, Y.-X. Chen, N. Lin, C. Yang, B. Zhang, Z. Liu, X.-X. Li, and H.-H. Dai, "Recent advances of generative adversarial networks in computer vision," *IEEE Access*, vol. 7, pp. 14985–15006, 2019.

[61] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: A system for large-scale machine learning," in *Proc. 12th USENIX Symp. Operating Syst. Design Implement. (OSDI)*, 2016, pp. 1–21.

[62] Á. Arcos-García, J. A. Álvarez-García, and L. M. Soria-Morillo, "Evaluation of deep neural networks for traffic sign detection systems," *Neurocomputing*, vol. 316, pp. 332–344, Nov. 2018.

[63] C. Dewi, R. C. Chen, Hendry, and H. T. Hung, "Comparative analysis of restricted Boltzmann machine models for image classification," in *Proc. Asian Conf. Intell. Inf. Database Syst. (ACIIDS)*, in Lecture Notes in Computer Science, vol. 12034, 2020, pp. 285–296.

[64] H. Yang, L. Chen, M. Chen, Z. Ma, F. Deng, M. Li, and X. Li, "Tender tea shoots recognition and positioning for picking robot using improved YOLO-V3 model," *IEEE Access*, vol. 7, pp. 180998–181011, 2019.

[65] Y. Yuan, Z. Xiong, and Q. Wang, "An incremental framework for video-based traffic sign detection, tracking, and recognition," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 7, pp. 1918–1929, Jul. 2017.

[66] Y. Tian, G. Yang, Z. Wang, H. Wang, E. Li, and Z. Liang, "Apple detection during different growth stages in orchards using the improved YOLO-V3 model," *Comput. Electron. Agricult.*, vol. 157, pp. 417–426, Feb. 2019.

[67] R. Shi, T. Li, and Y. Yamaguchi, "An attribution-based pruning method for real-time mango detection with YOLO network," *Comput. Electron. Agricult.*, vol. 169, pp. 1–11, Feb. 2020.

[68] H. Kang and C. Chen, "Fast implementation of real-time fruit detection in apple orchards using deep learning," *Comput. Electron. Agricult.*, vol. 168, pp. 1–10, Jan. 2020.

[69] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

**CHRISTINE DEWI** received the bachelor's degree (S.Kom.) from the Informatics Engineering Study Program and the master's degree (M.Cs.) degree from the Master of Information Systems Study Program, Faculty of Information Technology, Satya Wacana Christian University, Salatiga, Indonesia, in 2010 and 2012, respectively. She is currently pursuing the Ph.D. degree with the Department of Information Management, Chaoyang University of Technology, Taiwan. Her current research interests include image processing, computer vision, object detection and recognition, artificial intelligence, and machine learning.

**RUNG-CHING CHEN** (Member, IEEE) received the B.S. degree from the Department of Electrical Engineering, and the M.S. degree from the Institute of Computer Engineering, National Taiwan University of Science and Technology, Taipei, Taiwan, in 1987 and 1990, respectively, and the Ph.D. degree in computer science from the Department of Applied Mathematics, National Chung Hsing University, in 1998. He is currently a Distinguished Professor with the Department of Information Management, Chaoyang University of Technology, Taichung, Taiwan. His research interests include network technology, pattern recognition, knowledge engineering, the Internet of Things, data analysis, and artificial intelligence.

**YAN-TING LIU** received the B.I.M degree from the Department of Information Management, Chaoyang University of Technology, in 2019, where he is currently pursuing the master's degree. His research interests include object detection and recognition, artificial intelligence, and machine learning.

**XIAOYI JIANG** (Senior Member, IEEE) received the bachelor's degree in computer science from Peking University, Beijing, China, and the Ph.D. and Venia Docendi (Habilitation) degrees in computer science from the University of Bern, Bern, Switzerland, in 1989 and 1997, respectively. He was an Associate Professor with the Technical University of Berlin, Berlin, Germany. Since 2002, he has been a Full Professor of computer science with the University of Münster, Münster, Germany, where he is currently the Dean of the Faculty of Mathematics and Computer Science. He is a fellow of the International Association for Pattern Recognition. He is currently the Editor-in-Chief for the *International Journal of Pattern Recognition and Artificial Intelligence*. Besides, he also serves on the Advisory Board and the Editorial Board of several journals, including IEEE TRANSACTIONS ON MEDICAL IMAGING and the *International Journal of Neural Systems*.

**KRISTOKO DWI HARTOMO** received the Ph.D. degree in computer science from the Faculty of Science, Gadjah Mada University, Yogyakarta, in 2019. He has been active on research, since 2008, until now on geography information system. He is an Associate Professor with the Faculty of Information Technology, Satya Wacana Christian University, Central Java, Indonesia. He has published his articles on international journals. Moreover, he had five copyrights and written some reference books on computer science.