Journal of Imaging Science and Technology® 69(4): 1–8, 2025.

Applying the Taguchi Method to the Optimized Design of YOLO Hyperparameters: Blood Cell Count and Detection

Fu-I Chou
Department of Electrical Engineering, National Kaohsiung University of Science and Technology, No. 415, Jiangong Road, Kaohsiung 807, Taiwan
Department of Healthcare Administration and Medical Informatics, Kaohsiung Medical University, No. 100, Shin-Chuan 1st Road, Kaohsiung 807, Taiwan

Tian-Hsiang Huang
Department of Computer Science and Information Engineering, National Penghu University of Science and Technology, No. 300, Liuhe Road, Penghu 880, Taiwan

Po-Yuan Yang
Department of Intelligent Robotics, National Pingtung University, No. 51, Min-Sheng E. Road, Pingtung 900, Taiwan

Chun-Chieh Huang
Department of Electrical Engineering, National Kaohsiung University of Science and Technology, No. 415, Jiangong Road, Kaohsiung 807, Taiwan

Meng-Hsuan Chiang
Department of Medical Imaging, Kaohsiung Medical University Hospital, No. 100, Shin-Chuan 1st Road, Kaohsiung 807, Taiwan

Wen-Hsien Ho[†]
Department of Healthcare Administration and Medical Informatics, Kaohsiung Medical University, No. 100, Shin-Chuan 1st Road, Kaohsiung 807, Taiwan
Department of Medical Research, Kaohsiung Medical University Hospital, No. 100, Shin-Chuan 1st Road, Kaohsiung 807, Taiwan
College of Professional Studies, National Pingtung University of Science and Technology, No. 1, Shuefu Road, Pingtung 912, Taiwan
E-mail: whho@kmu.edu.tw

Jyh-Horng Chou[†]
Department of Electrical Engineering, National Kaohsiung University of Science and Technology, No. 415, Jiangong Road, Kaohsiung 807, Taiwan
Department of Healthcare Administration and Medical Informatics, Kaohsiung Medical University, No. 100, Shin-Chuan 1st Road, Kaohsiung 807, Taiwan
Department of Mechanical and Computer-Aided Engineering, Feng-Chia University, No. 100, Wenhua Road, Taichung 407, Taiwan
E-mail: choujh@nkust.edu.tw

Abstract. *In recent years, deep learning has achieved excellent results in several applications across various fields. However, as the scale of deep learning models increases, the training time of the models also increases dramatically. Furthermore, hyperparameters have a significant influence on model training results and selecting the model's hyperparameters efficiently is essential. In this study, the orthogonal array of the Taguchi method is used to find the best experimental combination of hyperparameters. This research uses three hyperparameters of the you only look once-version 3 (YOLOv3) detector and five hyperparameters of data augmentation as the control factor of the Taguchi method in addition to the traditional signal-to-noise ratio (S/N ratio) analysis method with larger-the-better (LB) characteristics.*

Experimental results show that the mean average precision of the blood cell count and detection dataset is 84.67%, which is better than the available results in literature. The method proposed herein can provide a fast and effective search strategy for optimizing hyperparameters in deep learning.

[†]Corresponding authors.

Received May 16, 2024; accepted for publication Sept. 29, 2024; published online Nov. 20, 2024. Associate Editor: Jia-Shing Sheu.

1062-3701/2025/69(4)/040401/8/$25.00

Keywords: *Taguchi method, hyperparameter, robust optimization, deep learning, object detection, YOLO*

[DOI: 10.2352/J.ImagingSci.Technol.2025.69.4.040401]

1. INTRODUCTION

In recent years, deep learning (DL) has achieved excellent results in various fields, such as radiofrequency (RF) transceivers in signal processing [1]. In the medical field, stacked autoencoders are used for diagnosing cervical cancer [2] and in the arena of machine vision tasks, DL's usage extends from basic image classification [3] to object detection [4]. Moreover, significant breakthroughs have been made in applications such as generative image modeling [5] and vehicle automation technology [6].

In DL development, most convolutional neural networks are developed based on the model structure. Three types of mainstream development directions exist, i.e., width, depth, and resolution. A representative work of the width type development is GoogLeNet [7]. This model won the first place in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2014 competition. The wide structure of GoogLeNet uses different mask sizes to increase the diversity of feature maps. Conversely, 1×1 convolutional masks are used to ensure that the output dimensions of each path are the same. Moreover, GoogLeNet introduced global average pooling [8] to replace the final fully connected layer, which can significantly improve the problem of overfitting.

A representative work of the depth type development is deep residual network (ResNet) [9]. This model won the first place in the ILSVRC 2015 competition. When the residual blocks update their weights backward, they can directly pass the loss value of this "addition node" layer to the previous node. Therefore, the inherent shortcomings of gradient disappearance due to the continuous deepening of the network can be mitigated [10]. Another advantage of this model is residual coding [11], which can eliminate common parts and amplify small changes.

Resolution type development uses high-resolution images as an input in the network to allow the model to capture detailed feature maps. The input image resolution of early convolutional networks was mostly 224×224. Examples include VGG16 [12], GoogLeNet, ResNet, and DenseNet [13]. GPipe [14] is a representative work in recent years, and the resolution has been increased to 480×480 to improve the model accuracy.

Regardless of using any of the above three types in model development, the model needs to be trained. Every time the hyperparameters are adjusted and the model is retrained and evaluated, the time required increases exponentially [15]. This is due to multiple factors: the size of the model is increasing every year; the initial hyperparameter settings have an important influence on the model training results; and the typical hardware is gradually unable to handle the latest network architecture. The present and significant challenge is the selection method of hyperparameters, which must reduce the time cost of the experiment effectively and simultaneously use reduced experimental result information to infer the best hyperparameter combination.

The current hyperparameter selection methods include grid and random searches [16]. The former selects factors and levels, lists all possible permutations and combinations at the required level and trains them. The latter is an improved method based on grid search, which is less time-consuming. However, the search result is not necessarily the best solution in the whole domain because it has poor stability.

Bayesian optimization (BO) [17] is a widely used method for hyperparameter optimization, which attempts to find the best combination in the entire domain with fewest steps. However, the search results of BO are susceptible to the parameters of the surrogate model and quality of the learning model, so the use restrictions are stringent.

The hyperparameter optimization method used in this study is the Taguchi method [18], an experimental design method that reduces the time cost. It also analyzes the contribution ratio of each control factor to the target value with the tiniest practical information and can predict the target value. Chen et al. [19] used the Taguchi method for single shot detector (SSD) algorithm to count the number of blood cells, and for parameter optimization of YOLOv7 [20], though it was used for vehicle detection. This study applies the Taguchi method for hyperparameter optimization of YOLOv3 [4] and proposes a robust optimization method for object detection.

Traditionally, blood cells are counted visually by using the hemocytometer along with additional chemical compounds and other scientific equipments. Since blood cells are huge in number, this task is time-consuming and tedious but necessary for assessment of health conditions. A computer-aided system of detecting and counting can greatly facilitate the entire counting process. The method designed in this study is applied to the blood cell count and detection (BCCD) dataset and assess the performance of hyperparameter search method proposed.

2. MATERIALS AND METHODS

2.1 *Dataset*

The dataset used in this study is the BCCD [21] dataset made available by Massachusetts Institute of Technology (MIT). The objective is to automatically detect and classify three different blood cells: red blood cells (RBC), white blood cells (WBC), and platelets, which have critical medical applications.

Each image has many blood cells, and 364 images exist in the entire dataset. In subsequent experiments, the BCCD dataset is divided into 292 training (80%) and 72 test (20%) data.

2.2 *Data Augmentation*

The image of the original data is in RGB color space. This study first transforms the RGB to HSV color space, randomly perturbs the hue, saturation, and value parameters, and then converts it back to RGB format. Simultaneously, the image is

randomly cropped and zoomed through 1 ± jitter (jitter is the ratio of random cropping). Data enhancement methods such as color perturbation, random cropping, and zooming images are used to enhance the diversity of data, improve the generalization ability of the model, and solve the problem of overfitting. Ultimately, the size of augmentation dataset is extended to 32 times the original dataset due to the hardware limitation of RAM of GPU video card (6GB RAM in this study).

2.3 *Methods*

2.3.1 *Object Detection Methods*

We use an image as the input of the algorithm and go through grayscale, binarization, open operation, close operation, and feature extraction steps. The process of determining the algorithm category is called image classification. Unlike image classification, in object detection, a picture usually contains multiple categories, and the location information of the object must be determined before further classification.

We first use the region proposal network (RPN) with anchor boxes to generate the candidate boxes of interest. Subsequently, we use the center point pixel of each candidate frame as a feature representative and perform the two-class classification training to determine whether it is an object. Further, we use the bounding box generated using RPN as the input data for training the classifier network. This architecture with two neural networks is called the two-stage object detection model. The network of classifiers can use standard CNN models as the backbone, such as VGG16, Resnet, DenseNet, and NSGA-Net [22]. A very popular two-stage model is faster R-CNN [23, 24].

Moreover, as it was discovered in deep neural network [25] that neural networks can perform not only classification tasks but can also learn geometric information, single shot detector (SSD) [26] was developed. This type of model which discards RPN and only uses a single neural network to complete object positioning and classification is called one-stage object detection.

The evaluation index in the object detection field is the mean average precision (mAP), which is the average value of APs, and the area calculates AP under the curve drawn using precision and recall [27]. In the test data set of The PASCAL Visual Object Classes Challenge 2007 (VOC2007) [27], SSD achieved 74.3% mAP, surpassing faster R-CNN's 69.9%. In addition to earning a higher mAP, SSD achieved a processing speed of 59 frames per second (FPS) under the same Nvidia Titan X GPU environment, which shows an overwhelming advantage over faster R-CNN with only 7 FPS.

Since two-stage detection model is very slow [4], and the two networks are processed separately, optimizing one network connected to the second network directly is challenging. In addition to SSD, the detection technology YOLO proposed by Redmon et al. [4] treats the object detection problem as a regression task. It directly obtains the information frame through the pixel value of the entire image, including the probability of belonging to each category and positioning information [4]. This is the core spirit of YOLO: you only need to look at it once to obtain the object and its coordinate information contained in the image.

The first version of YOLOv1 [4] successfully entered the list of real-time detectors (FPS $\geq$ 30), and its accuracy significantly surpassed the deformable part models (DPM) [28]. Fast YOLO [29] has a processing speed of up to 155 FPS, five times that of 30 Hz DPM. Thus, even if a part of the accuracy is lost due to a lightweight design, mAP is still doubled.

Furthermore, YOLO has a low error rate in the foreground and background. Because fast R-CNN uses RPN to predict the information frame, the classifier network can only obtain a partial information of the entire image. YOLO uses all the pixel values of the whole image during the training process, so YOLO rarely predicts the incorrect message in the background image. Moreover, YOLO can learn the characteristics to a high degree of generalization, which can be transferred to different datasets.

In YOLO detection process, an image is first divided into an $S \times S$ grid [30]. If the object's center point falls on this grid, it must be responsible for the detection of this object. Each grid will predict B bounding boxes and the confidence score of the object. If no central point of any object exists in this grid, the confidence score is 0. Otherwise, it is the intersection over union (IOU) between the predicted and ground truth boxes.

The information frame of YOLO is encoded into five data dimensions: the center point coordinates (tx, ty) of the object, width tw of the object, height th of the object, and confidence score. Moreover, every grid containing at least one object predicts C conditional class probability belonging to a specific category, i.e., Pr(Class|Object), so YOLO can be regarded as an application of CNN in the coding system.

The output feature map of YOLOv1 [4] uses a grid of $S = 7$. Each grid predicts $B = 2$ object frames and finally uses nonmaximum suppression (NMS) to eliminate duplicate object frames. The detection model of YOLOv1 is implemented based on GoogLeNet but does not use the inception structure. Its input size is 448×448. There are 24 convolutional layers and 2 fully connected layers. Considering PASCAL VOC as an example with a total of 20 categories, $C = 20$, the final output code number is $7 \times 7 \times (2 \times 5 + 20) = 7 \times 7 \times 30$. YOLOv1 only uses one network to complete object detection, thereby significantly improving the processing speed. However, there is no RPN assistance and only the grid of $S = 7$ is used, so the positioning of the object frame may not be very accurate, which leads to the relatively weak detection accuracy of YOLOv1.

YOLOv2 [31] uses a convolutional layer to replace the fully connected layer of YOLOv1. The input size of the model is changed to 416×416, $B = 5$, $S = 13$. Furthermore, because of the introduction of anchor boxes, the number of predicted boxes improves from $7 \times 7 \times 2 = 98$ to $13 \times 13 \times 5 = 845$. The detector of YOLOv2 is based on Darknet-19. It contains 19 convolutional layers and 5 extensive pooling layers. Considering the same PASCAL

VOC 20 categories as an example, the output code number of YOLOv2 in the PASCAL VOC detection model is $13 \times 13 \times (5 \times (5 + 20)) = 13 \times 13 \times 125$. The most significant change in YOLOv2 is redefining the method of coordinate prediction and changing it to predict the offset of the anchor box. In the training process, the learning goal of the detector is to predict the four offsets of *tx*, *ty*, *tw*, *th*$\in$ [0, 1]. Among them, (*tx*, *ty*) is the center point offset of the information frame, *tw* is the width offset of the information frame, and *th* is the height offset.

The positioning method of YOLOv3 [32] is inherited from YOLOv2 but with significant changes in the feature extractor. The detector is based on Darknet-53 and is combined with the feature pyramid network structure [33] to become a multiple scale detection network. Consequently, YOLOv3 has an excellent detection speed and accuracy. For example, in the ImageNet image dataset for image classification tasks, the obtained results are as accurate as ResNet-152, but the floating point operations (FLOPS) is improved by 110.81%.

YOLOv4 [34] made improvements on the Darknet-53 of YOLOv3, drawing on the cross-stage partial network to solve other large CNN frameworks in gradient duplication problem for optimization. The gradient change integrates the feature map, reducing the amount of FLOPS values, ensuring the speed and accuracy of inference, and reducing the size of the model. YOLOv4 has greatly improved in speed and accuracy, compared with YOLOv3, in mAP and FPS performance improved by 10% and 12%.

YOLOv5 [35] came out a month later than YOLOv4, and advertised that it can reach 140 FPS, but no related official publication and is less innovative than YOLOv4. Few scholars opine that it has not evolved enough to be called YOLOv5. It is a little controversial.

The same process of optimizing hyperparameters can be applied to later versions of YOLOv4 or extended to other similar high-quality object detection algorithms. This study uses YOLOv3 to perform blood cell type identification experiments and demonstrates the use of the Taguchi method described hereinafter to optimize the hyperparameters of YOLOv3 robustly.

2.3.2 *Taguchi Method*

Dr. Genichi Taguchi first proposed the Taguchi method in 1951. Its orthogonal arrays are widely used by Japanese industries and received enthusiastic responses. By 1970, Japanese industries had popularized the Taguchi method for parameter calibration. The orthogonal nature of the orthogonal table can significantly reduce the cost of experiments and greatly simplify data analysis.

The target value of the Taguchi method is divided into three static characteristics: smaller-the-better, nominal-the-best, and larger-the-better. The larger-the-better (LB) characteristic was adopted in this study. The target value of this study is that the commonly used evaluation index in object detection is the mAP [27], so the factor effects are analyzed using the magnificent feature. Selecting control factors is a difficult task, and it usually depends on the user's rules of thumb for screening. The number of segments where the control factor changes up or down is called the levels of control factors.

The orthogonal table is expressed as $L_p(q^m)$, where p represents the number of the experiments of the orthogonal table; q represents the level of change; and m represents the number of control factors. For example, with three control factors and two variable levels, the experimental orthogonal table of $L_4(2^3)$ can be selected. Orthogonality implies that between two rows, all level combinations appear the same number of times. Moreover, under the same number of factors and levels, the number of grid search experiments is eight times. However, the Taguchi method only requires four experiments, which is half of the grid search method, significantly reducing the number of experiments.

After the orthogonal experiment table is constructed, the number of experiments, N, for each combination can be determined according to the cost of each experiment. In the Taguchi method, the target value analysis is converted into the signal-to-noise ratio (S/N ratio) form of processing. The larger the S/N ratio, the more the antinoise and the better the communication ability. The S/N ratio of LB characteristic (S/N_{LB}) used herein is shown in Eq. (1):

$$S/N_{\mathrm{LB}} = -10 \log \frac{\sum_{z=1}^{N} \frac{1}{y_z^2}}{N}, \tag{1}$$

where y_z represents the zth experiment result of each group.

Factor effects refer to changes in control factors on the S/N ratio or the target value. The response table can be further analyzed using the experimental results of the orthogonal table. We can predict the best combination of control factors from the response data and then perform confirmation experiments. If the result of the experiment is better than any combination in the orthogonal table, it implies that the predictive ability of the Taguchi method is effective.

2.3.3 *Orthogonal Table*

There is one factor of Level 2 and seven factors of Level 3, as shown in Table I. Among them, flip (A) is whether the image is flipped or not; max_box (B) is the upper limit of the sampling information frame of each category in the training process of YOLOv3; jitter (C) is the ratio of random cropping; hue (D) is the random perturbation amount of hue; saturation (E) is the random perturbation of saturation; value (F) is the random perturbation of brightness; and $\mathrm{IOU}_{\mathrm{train}}$ (G) is the lowest object threshold setting during the search and training process of YOLOv3. The last is the slope (H) of the activation function Leaky ReLU. After selecting the control factors, this study uses the $L_{18}(2^1 \times 3^7)$ orthogonal table experimental layout, as shown in Table II.

3. EXPERIMENTAL RESULTS

Each set of experiments in Table II is repeated three times; the input width and height are 320 (pixels); the

Table I. Control factor level table.

Level	Flip (A)	Max_box (B)	Jitter (C)	Hue (D)	Saturation (E)	Value (F)	IOU_{train} (G)	Leaky (H)
Level 1	True	20	0.3	0.1	0.5	0.5	0.5	0.05
Level 2	False	40	0.55	0.4	1.5	1.5	0.3	0.10
Level 3	N/A	60	0.8	0.7	2.5	2.5	0.7	0.20

Table II. $L_{18}(2^1 \times 3^7)$ orthogonal table experimental configuration.

Experiment order	A	B	C	D	E	F	G	H
1	True	20	0.3	0.1	0.5	0.5	0.3	0.05
2	True	20	0.55	0.4	1.5	1.5	0.5	0.1
3	True	20	0.8	0.7	2.5	2.5	0.7	0.2
4	True	40	0.3	0.1	1.5	1.5	0.7	0.2
5	True	40	0.55	0.4	2.5	2.5	0.3	0.05
6	True	40	0.8	0.7	0.5	0.5	0.5	0.1
7	True	60	0.3	0.4	0.5	2.5	0.5	0.2
8	True	60	0.55	0.7	1.5	0.5	0.7	0.05
9	True	60	0.8	0.1	2.5	1.5	0.3	0.1
10	False	20	0.3	0.7	2.5	1.5	0.5	0.05
11	False	20	0.55	0.1	0.5	2.5	0.7	0.1
12	False	20	0.8	0.4	1.5	0.5	0.3	0.2
13	False	40	0.3	0.4	2.5	0.5	0.7	0.1
14	False	40	0.55	0.7	0.5	1.5	0.3	0.2
15	False	40	0.8	0.1	1.5	2.5	0.5	0.05
16	False	60	0.3	0.7	1.5	2.5	0.3	0.1
17	False	60	0.55	0.1	2.5	0.5	0.5	0.2
18	False	60	0.8	0.4	0.5	1.5	0.7	0.05

optimizer uses Adam [36]; and the batch size is four. Moreover, to save computing time, the number of iterations is set to 25. First, we determine the best hyperparameter combination based on the analysis of results and adjust the settings. Subsequently, we retrain the model for conducting a confirmation experiment to verify whether the target value mAP of the best hyperparameter combination is better than any set in the orthogonal table.

Table III shows the experimental results of the BCCD dataset. We use the experimental data of the Taguchi method $L_{18}(2^1 \times 3^7)$ orthogonal table in the YOLOv3 architecture and repeat each set of experimental configurations three times. In Table III, S/N_{LB} can be obtained by Eq. (I).

Table IV uses the response table of S/N_{LB} analysis factors, and the best combination ($\mathrm{Best}_{\mathrm{LB}}$) that can be predicted is as follows: flip = true, max_box = 40, jitter = 0.55, hue = 0.1, saturation = 1.5, value = 0.5, $\mathrm{IOU}_{\mathrm{train}} = 0.3$, and leaky = 0.10.

After retraining the model according to the $\mathrm{Best}_{\mathrm{LB}}$, the mAP is 49.27%, which is higher than the maximum value of 43.51% in the orthogonal experiment listed in Table III. Therefore, the predictive ability of the Taguchi method is practical.

The confirmed experimental data of the BCCD dataset is shown in Table V. To verify the effectiveness of the proposed method, the number of iterations is set to 100. Consequently, the $\mathrm{Best}_{\mathrm{LB}}$ analysis method can obtain the highest mAP value of 84.67%. Furthermore, the results of this experiment also exceeds 74.37% of those using Resnet50 as the YOLO feature extractor [37] and 81.76% by Chou's optimization approach [38]. These results show that the rapid optimization method proposed in this study using S/N_{LB} leads to better mAP experimental results. Therefore, this method is expected to provide stable detection results in practical tasks.

Furthermore, Figures 1–3 show the AP of RBC, WBC, and platelets, respectively. Figures 4 and 5 show the loss function curves of training dataset and test dataset for the confirmation experimental, respectively. In addition, Figures 6 and 7 show that even if the WBC and platelets are blue, the model can still distinguish them. The green information box in each image represents the ground truth; the blue information box is the positioning information of

Table III. Taguchi method experimental results of the BCCD dataset.

Experiment order	mAP_1 (%)	mAP_2 (%)	mAP_3 (%)	Average mAP (%)	S/N_{LB}
1	32.62	42.38	43.30	39.43	31.70
2	24.25	28.08	25.50	25.94	28.23
3	18.86	20.48	20.53	19.96	25.98
4	33.67	32.00	35.27	33.65	30.52
5	29.07	27.45	28.13	28.22	29.00
6	27.69	35.24	32.79	31.91	29.94
7	21.69	15.92	20.95	19.52	25.55
8	35.16	41.11	41.72	39.33	31.82
9	32.03	32.88	33.97	32.96	30.35
10	27.10	24.80	28.03	26.64	28.48
11	36.65	37.78	35.85	36.76	31.30
12	24.59	25.93	22.88	24.47	27.74
13	24.19	28.04	26.63	26.29	28.35
14	37.10	39.33	33.62	36.68	31.23
15	33.81	31.53	35.53	33.62	30.50
16	25.58	29.51	23.89	26.33	28.31
17	36.34	43.51	40.85	40.23	32.02
18	14.81	17.74	15.63	16.06	24.04

Table IV. Factor response table.

Level	Flip	Max_box	Jitter	Hue	Saturation	Value	IOU_{train}	Leaky
1	29.23	28.90	28.82	31.06	28.96	30.26	29.72	29.26
2	29.11	29.92	30.60	27.15	29.52	28.81	29.12	29.41
3	N/A	28.68	28.09	29.29	29.03	28.44	28.67	28.84

Table V. Confirmation experimental results and comparisons with the related studies.

Approach	mAP_1 (%)	mAP_2 (%)	mAP_3 (%)	Average mAP (%)	Standard deviation
The proposed method ($Best_{LB}$)	84.59	84.67	81.57	83.61	1.767
Alam and Islam [37]			74.37		
Chou [38]			81.76		

the model; and the upper left of the information box is the label name.

4. CONCLUSIONS

This study used the Taguchi method to optimize the hyperparameters of the object detection deep learning algorithm. Using YOLOv3 for demonstration in the BCCD dataset, the confirmation experimental result yielded an mAP of 84.67%, which is considerably higher than that obtained by Alam and Islam (74.7%) [37] and Chou (81.76%) [38], indicating the superiority of our proposed method. It is conceivable that the performance of cell detection from image is equivalent to the performance of cell counting. Therefore, the proposed method can provide an effective strategy for blood cell counting.

Evidently, the main contributions of this study are as follows: (1) Using the predictive ability of the Taguchi method

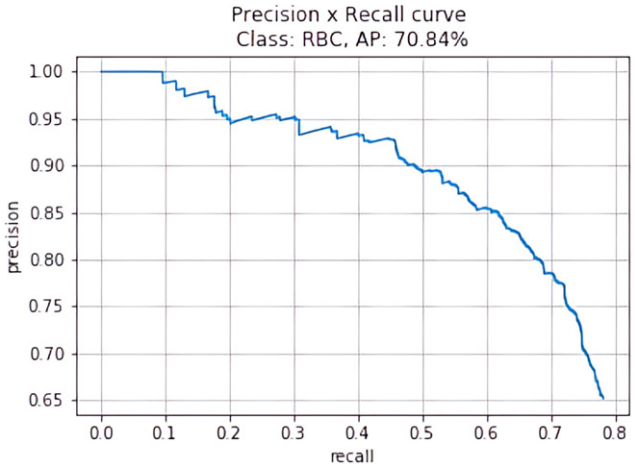

Figure 1. AP of red blood cells.

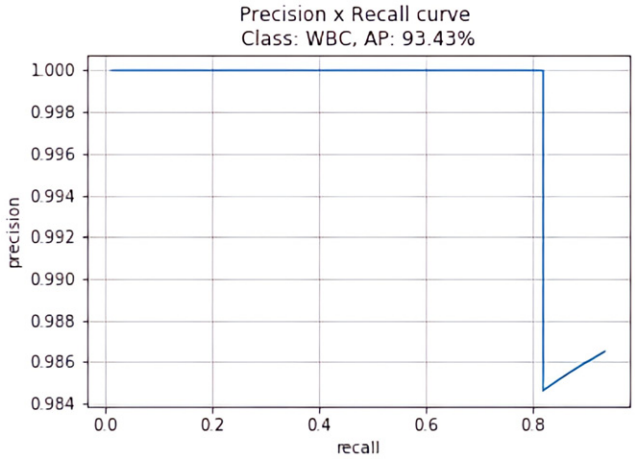

Figure 2. AP of white blood cells.

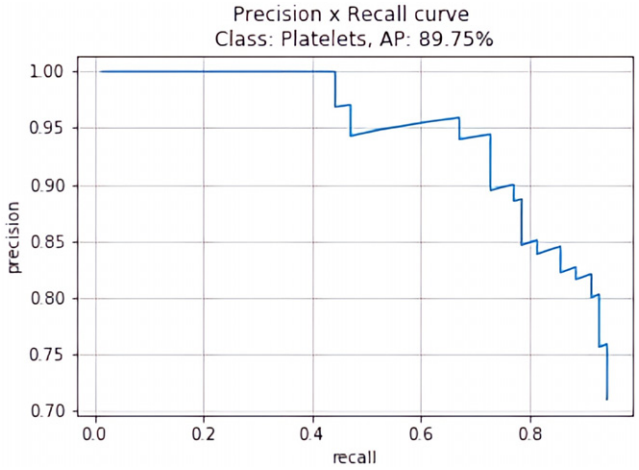

Figure 3. AP of platelets.

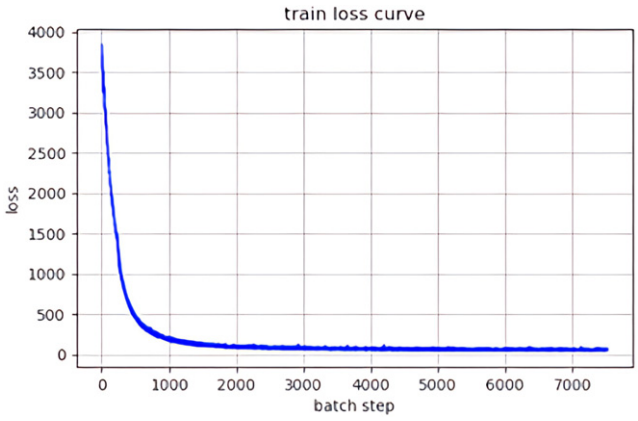

Figure 4. The loss curve of training data set.

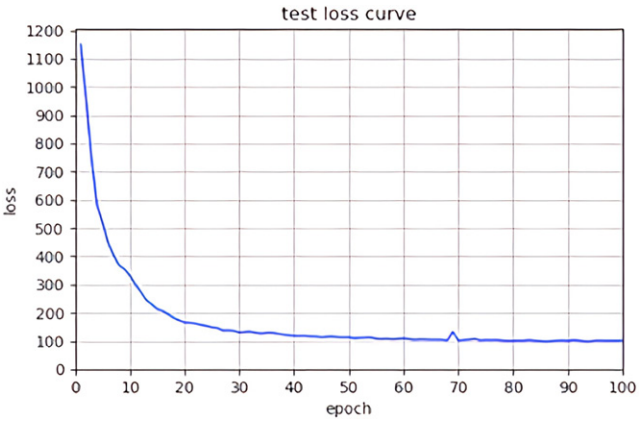

Figure 5. The loss curve of test data set.

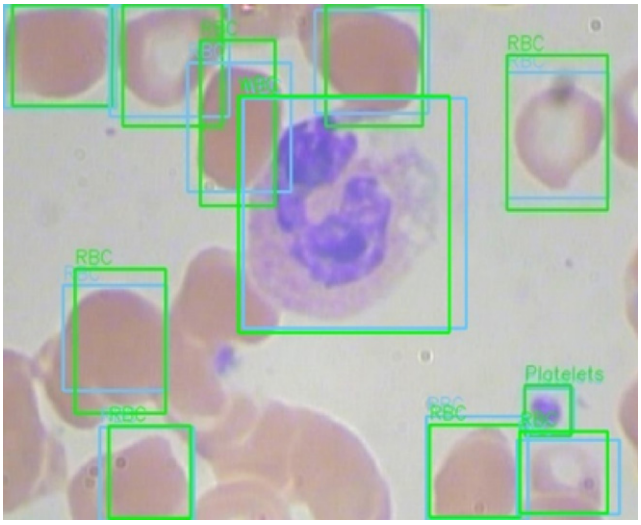

Figure 6. Test result 1.

S/N_{LB}, we determined the best mix of hyperparameter factors of the model to achieve the highest mAP. (2) The study had a total of 4374 permutations and combinations of selectable factor levels in YOLOv3. Thus, considerable time would have been required if the traditional grid search method was used; the proposed method saved significant

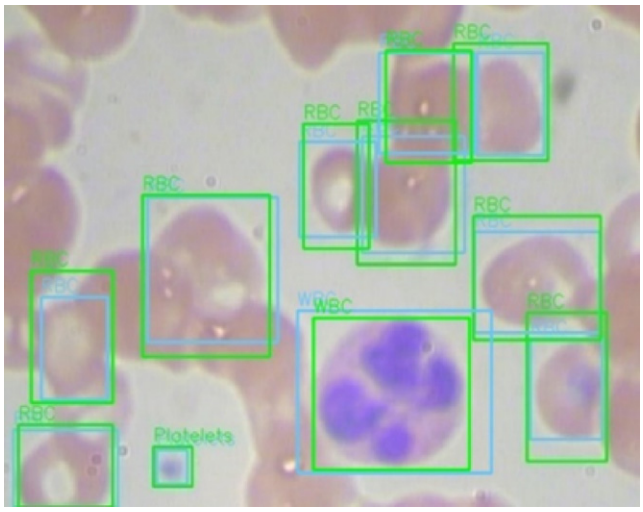

Figure 7. Test result 2.

time and cost (the number of experiments is only 18). The experimental results confirm that the proposed fast optimization method can be used as a general method for hyperparameter optimization problem of deep learning.

ABBREVIATIONS

YOLOv3: You only look once - version 3 S/N ratio: Signal-to-noise ratio LB: Larger the better RF: radiofrequency ILSVRC: ImageNet Large Scale Visual Recognition Challenge ResNet: Residual network BO: Bayesian optimization BCCD: Blood cell count and detection RPN: Region proposal network SSD: Single shot detector mAP: Mean average precision FPS: Frames per second DPM: Deformable part models IOU: Intersection over union FLOPS: Floating point operations RBC: Red blood cells WBC: White blood cells.

AVAILABILITY OF DATA AND MATERIALS

The datasets generated during and/or analysed during the current study are available in the BCCD dataset repository, https://www.kaggle.com/surajiiitm/bccd-dataset.

ACKNOWLEDGMENT

Publication costs are funded by the National Science and Technology Council, Taiwan, under grants MOST 110-2221-E-035-092-MY3, MOST 110-2221-E-153-010, MOST 111-2221-E-992-072-MY2 and NSTC 112-2221-E-037-004-MY3. The design and part of writing costs of the study are funded by NKUST-KMU JOINT RESEARCH PROJECT (#NKUSTKMU-110-KK-001).

REFERENCES

[1] P. Jueschke and G. Fischer, "Machine learning using neural networks in digital signal processing for rf transceivers," *2017 IEEE AFRICON* (IEEE, Piscataway, NJ, 2017), pp. 384–390.

[2] K. Adem, S. Kiliçarslan, and O. Cömert, "Classification and diagnosis of cervical cancer with stacked autoencoder and softmax dassification," Expert Syst. Appl. **115**, 557–564 (2019).

[3] D. Li, "The mnist database of handwritten digit images for machine learning research [best of the web]," IEEE Signal Process. Mag. **29**, 141–142 (2012).
[4] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," *2016 IEEE Conf. Computer Vision and Pattern Recognition (CVPR)* (IEEE, Piscataway, NJ, 2016), pp. 779–788.
[5] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," *2017 IEEE Int'l. Conf. Computer Vision (ICCV)* (IEEE, Piscataway, NJ, 2017), pp. 2242–2251.
[6] L. Fridman, D. E. Brown, M. Glazer, W. Angell, S. Dodd, B. Jenik, J. Terwilliger, A. Patsekin, J. Kindelsberger, L. Ding, and S. Seaman, "MIT advanced vehicle technology study: Large-scale naturalistic driving study of driver behavior and interaction with automation," IEEE Access **7**, 102021–102038 (2019).
[7] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," *2015 IEEE Conf. Computer Vision and Pattern Recognition (CVPR)* (IEEE, Piscataway, NJ, 2015), pp. 1–9.
[8] M. Lin, Q. Chen, and S. Yan, "Network in network," *2nd Int'l. Conf. Learning Representations* (ICLR, Banff, 2014), pp. 14–16.
[9] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *2016 IEEE Conf. Computer Vision and Pattern Recognition (CVPR)* (IEEE, Piscataway, NJ, 2016), pp. 770–778.
[10] J. Baxter, "Learning internal representations," *8th Annual Conf. on Computational Learning Theory* (ACM, New York, NY, 1995), pp. 311–320.
[11] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," *23rd IEEE Conf. Computer Vision & Pattern Recognition* (IEEE, Piscataway, NJ, 2010), pp. 3304–3311.
[12] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *3rd Int'l. Conf. Learning Representations* (ICLR, San Diego, CA, 2015).
[13] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)* (IEEE, Piscataway, NJ, 2017), pp. 2261–2269.
[14] Y. Huang, Y. Cheng, A. Bapna, O. Firat, D. Chen, M. Chen, H. Lee, J. Ngiam, Q. V. Le, and Y. Wu, "Gpipe: Efficient training of giant neural networks using pipeline parallelism," *33rd Int'l. Conf. Neural Information Processing Systems* (NeurIPS, San Diego, CA, 2019), pp. 103–112.
[15] M. Feurer and F. Hutter, "Hyperparameter optimization," in *Automated Machine Learning: Methods, Systems, Challenges*, edited by F. Hutter, L. Kotthoff, and J. Vanschoren (Springer, Cham, 2019), pp. 3–33.
[16] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," *Advances in Neural Information Processing Systems 24* (NeurIPS, San Diego, CA, 2011), pp. 2546–2554.
[17] R. Calandra, A. Seyfarth, J. Peters, and M. P. Deisenroth, "Bayesian optimization for learning gaits under uncertainty," Ann. Math. Artif. Intell. **76**, 5–23 (2016).
[18] R. K. Roy, *A Primer on the Taguchi Method* (Society of Manufacturing Engineers, Southfield, MI, 2010).
[19] Y. M. Chen, J. T. Tsai, and W. H. Ho, "Automatic identifying and counting blood cells in smear images by using single shot detector and taguchi method," BMC Bioinf. **22**, 635 (2021).
[20] M.-K. Chen, C.-L. Chang, C.-J. Lin, and W.-J. Chen, "Vehicle detection on express roads using yolov7 with taguchi parameter optimization method," Sens. Mater. **36**, 1605–1625 (2024).
[21] S. Mishra, BCCD dataset, https://www.kaggle.com/surajiiitm/bccd-dataset, accessed May 10 (2024).
[22] Z. Lu, I. Whalen, V. Boddeti, Y. Dhebar, K. Deb, E. Goodman, and W. Banzhaf, "NSGA-Net: Neural architecture search using multi-objective genetic algorithm," *Proc. Genetic and Evolutionary Computation Conf.* (ACM, New York, NY, 2019), pp. 419–427.
[23] R. Girshick, "Fast R-CNN," *IEEE Int'l. Conf. Computer Vision (ICCV)* (IEEE, Piscataway, NJ, 2015), pp. 1440–1448.
[24] 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. **39**, 1137–1149 (2017).
[25] C. Szegedy, A. Toshev, and D. Erhan, "Deep neural networks for object detection," *Advances in Neural Information Processing Systems 26* (NeurIPS, San Diego, CA, 2013), pp. 2553–2561.
[26] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu, and A. C. Berg, *SSD: Single shot Multibox Detector*, Lecture Notes in Computer Science (Springer, Cham, 2016), Vol. 9905, pp. 21–37.
[27] M. Everingham, L. Van_Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (VOC) challenge," Int. J. Comput. Vis. **88**, 303–338 (2010).
[28] M. A. Sadeghi and D. Forsyth, *30 Hz Object Detection with DPM v5*, Lecture Notes in Computer Science (Springer, Cham, 2014), Vol. 8689, pp. 65–79.
[29] M. J. Shafiee, B. Chywl, F. Li, and A. Wong, "Fast YOLO: A fast you only look once system for real-time embedded object detection in video," J. Comput. Vis. Imaging Syst. **3** (2017).
[30] J. Redmon and A. Angelova, "Real-time grasp detection using convolutional neural networks," *2015 IEEE Int'l. Conf. Robotics and Automation (ICRA)* (IEEE, Piscataway, NJ, 2015), pp. 1316–1322.
[31] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)* (IEEE, Piscataway, NJ, 2017), pp. 6517–6525.
[32] J. Redmon and A. Farhadi, YOLOV3: An incremental improvement, http://arxiv.org/abs/1804.02767, accessed May 11 (2024).
[33] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)* (IEEE, Piscataway, NJ, 2017), pp. 2117–2125.
[34] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, YOLOV4: Optimal speed and accuracy of object detection, https://arxiv.org/abs/2004.10934, accessed May 11 (2024).
[35] Ultralytics, YOLOV5, https://github.com/ultralytics/yolov5, accessed May 11 (2024).
[36] D. P. Kingma and L. J. Ba, "Adam: A method for stochastic optimization," *3rd Int'l. Conf. Learning Representations (ICLR)* (NeurIPS, San Diego, CA, 2015).
[37] M. M. Alam and M. T. Islam, "Machine learning approach of automatic identification and counting of blood cells," Healthc. Technol. Lett. **6**, 103–108 (2019).
[38] J. H. Chou, "Optimization approaches," *Lecture notes and handouts at National Kaohsiung University of Science and Technology* (NKUST, Kaohsiung City, 2019).