Computers and Electronics in Agriculture 206 (2023) 107694

Contents lists available at ScienceDirect

Computers and Electronics in Agriculture

journal homepage: www.elsevier.com/locate/compag

Original papers

Field detection of small pests through stochastic gradient descent with genetic algorithm

Yin Ye [a,b], Qiangqiang Huang [a,b], Yi Rong [b,a,c,*], Xiaohan Yu [d], Weiji Liang [a,b], Yaxiong Chen [a,b], Shengwu Xiong [a,b,**]

[a] Sanya Science and Education Innovation Park of Wuhan University of Technology, Sanya 572025, China
[b] School of Computer Science and Artificial Intelligence, Wuhan University of Technology, Wuhan 430070, China
[c] Hainan Yazhou Bay Seed Laboratory, Sanya 572025, China
[d] School of Engineering and Built Environment, Griffith University, Brisbane, QLD 4111, Australia

ARTICLE INFO

Keywords:
Object detection
Small pests detection
Gradient descent
Genetic algorithm
Evolutionary algorithm

ABSTRACT

Pest invasion is one of the main reasons that affect crop yield and quality. Therefore, accurate detection of pests is a key technology of smart agriculture. Pests often exist as small objects with limited features in the actual field. Deep neural networks, as promising small object detectors, are adopted to fully acquire the feature information. The pest detection network has a large number of parameters to be trained, where the current stochastic gradient descent method may tend to fall into local optimum and lead to poor pest detection precision. To solve the above issue, we propose the GA-SGD algorithm to help the SGD jump out of the local optimal trap. It consists of selection operation, crossover operation and mutation operation. The selection operation selects fine solutions from the parent population, crossover operation exchanges and combines the two solutions to generate the new offspring, mutation operation replaces the original value with the random value to produce the new solutions. Experiments show the proposed GA-SGD achieves higher detection accuracy and stability than five algorithms on three object detectors. The results indicate small pests are detected with superiority. It also proves the effectiveness and value of the proposed algorithm.

1. Introduction

The development of agriculture is affected by many factors, especially pests (Li et al., 2021). The overall physiological function of crops attacked by pests will be greatly reduced and will not be able to achieve optimal production, resulting in poor yields and low economic efficiency. Due to a large number of crop plants, it is extremely difficult to identify pests by relying on human eyes. And by the time the pests are found, they have already spread to a large area. Therefore, automatic and accurate pest detection is particularly important to improve agricultural productivity. In the agricultural planting process, crop pests are analyzed in a timely and accurate manner so that a quick and accurate response can be made, and pesticides can be accurately sprayed in the affected areas to ensure efficient use of pesticides and increase crop yields, thus achieving fast and efficient precision agriculture.

With the development of artificial intelligence technology, computer vision also develops rapidly in agriculture. For example, fine-grained classification models (Yu et al., 2021b) are applied for cultivar species identification (Yu et al., 2021a), insect wing identification (Yu et al., 2020), crop disease identification (Yu et al., 2022b) and so on. computer vision opens up a new way for the identification of harmful organisms. The key feature information for object recognition in computer vision is mainly divided into manual features and deep features. In the literature Wu et al. (2019), Manual features LCH (Swain and Ballard, 1991), Gabor (Rajiv Mehrotra, 1992), GIST (Oliva and Torralba, 2001), SIFT (Lowe, 2004), SURF (Bay et al., 2006) are compared with deep features AlexNet (Krizhevsky et al., 2017), GoogleNet (Szegedy et al., 2015), VGGNet (Karen Simonyan, 2014), ResNet (He et al., 2016) on pests dataset for identification, and the results show the strong advantage of deep learning in the field of pests identification, with an accuracy exceeding manual features by about 30%. The current pests detection mainly focuses on the detection of large pests (Huang et al., 2022; Yu et al., 2022a; Wei et al., 2022). The arrangement of cameras is relatively sparse in the field, pests have relatively few pixels in the overall image. Fig. 1 shows the difference between large pests derived from literature Wu et al. (2019) and small pests derived from literature Wang et al. (2021). The resolution of Fig. 1-a is 5472*3648,

* Corresponding author.
** Corresponding author at: School of Computer Science and Artificial Intelligence, Wuhan University of Technology, Wuhan 430070, China.
E-mail addresses: yrong@whut.edu.cn (Y. Rong), xiongsw@whut.edu.cn (S. Xiong).

https://doi.org/10.1016/j.compag.2023.107694
Received 28 July 2022; Received in revised form 23 December 2022; Accepted 1 February 2023
Available online 14 February 2023
0168-1699/© 2023 Elsevier B.V. All rights reserved.

(a) big pest (b) small pest

Fig. 1. The differences of large pest and small pest.

where the pixel value of the pest is about 2861*1805. The resolution of the Fig. 1-b is 1440*1080, where the pixel value of the pest is about 30*30. Small objects (Lin et al., 2014) with an area less than or equal to 32×32 pixels.

Due to the low resolution of small pests, small pests detection is much more difficult than large pests detection and the detection accuracy is lower. At present, two-stage detectors achieved state-of-the-art performance in the small object datasets (Liu et al., 2021b). And they can also achieve better performance on the small pests dataset (Wang et al., 2021). The core steps of the two-stage detector are as follows. Stage one: The input feature map is processed by Region Proposal Network (RPN) and several Region of Interests (RoI) are output. RPN is mainly to preserve as much foreground information (pests) as possible, and filter out background information that is useless for subsequent classification. Stage two: The location and classification of each RoI is completed by several fully connected layers. The environmental backgrounds of small pests can be removed by RPN, and the accuracy of multi-classification of small pests can be improved.

The small pests detector is a complex deep neural network and the network model is trained by stochastic gradient descent. The essence of training a neural network is to optimize the parameters of the loss function. For complex models, the gradient descent method is difficult to train better network model parameters due to the tendency to fall into local optima (Cheridito et al., 2021), so the object detection accuracy is difficult to improve. To alleviate the problem of getting stuck in the local optimums, some optimization algorithms (Talpur et al., 2022) have been proposed to train deep neural network models. RMSProp (Shi and Li, 2021) uses root mean square as the denominator, which can alleviate the problem that the learning rate decreases faster during the training process. And by introducing the root mean square, the oscillation can be reduced to accelerate the convergence process. Adam (Kingma and Ba, 2015) uses first-order moment estimation and second-order moment estimation of the gradient to adjust the learning rate. Adam is a combination of Momentum and RMSprop with bias correction. Adamax (Raja and Ashok, 2021) adds a learning rate ceiling concept to Adam. AdamW (Loshchilov and Hutter, 2019) is an improved algorithm based on Adam combined with L2 regularization. Adagrad (Antonakopoulos et al., 2022) is adaptive in assigning different learning rates to each parameter. The variation of this learning rate is influenced by the size of the gradient and the number of iterations. The gradient is inversely proportional to the learning rate, and Adagrad accumulates all the previous gradients squared as the denominator, which will lead to a small learning rate problem in the late training period. In the denominator of Adadelta (Zeiler, 2012), cumulative items that are relatively close to the current time point are adopted, which can avoid too small learning rate in the later stages of training. As a biological heuristic algorithm, the evolutionary algorithm (Miikkulainen and Forrest, 2021) maintains a large number of solutions in the search process and enables extreme exploration and massive parallelization, which provides a very potential optimization method for deep learning and engineering design (Ye et al., 2022). This type of training is called neuroevolution (Stanley et al., 2019). In literature Wong et al. (2021), neuroevolution and stochastic gradient descent are used to solve several white box differential equations, and the convergence advantage of neuroevolution is proved by experimental data. An evolvable neural unit is proposed in the literature Bertens and Lee (2020) and evolutionary strategies are used to train the Spiking neural network.

The gradient descent method converges quickly, while neuroevolution converges with high accuracy, so some improvements in coevolution (Antonio and Coello, 2017) have emerged. Cui et al. (2018) combine stochastic gradient descent and evolutionary algorithms in a framework as complementary algorithms, optimization is alternated between the SGD step and the evolutionary step to improve the average fitness of the solution, it is a way to train for better neural network parameters. Cui and Picheny (2019) present an ESGD variant for acoustic model optimization in automatic speech recognition. Liu et al. (2021a) apply particle swarm optimization to improve stochastic gradient descent. Based on ensuring the classification accuracy of medical images, the optimal solution of the network can be found faster and the solving efficiency can be improved. Zhang et al. (2022) propose a framework that combines Neuroevolution with stochastic gradient descent to optimize the loss function of deep neural networks. In this study, the training effect of pest model is improved by integrating three evolutionary strategies and SGD method, so as to obtain the higher accuracy of pest detection.

The purpose of this paper is to propose an efficient training algorithm, named GA-SGD, for the pests detection models. The current training method of the pest detection model is stochastic gradient descent, which is easy to fall into local optimum, so the loss function parameters of the pest detection model are difficult to optimize, resulting in poor accuracy of pests detection. Therefore, this paper introduces three evolution strategies to improve the original SGD algorithm, which are selection operation, crossover operation and mutation operation. Firstly, the solution set of the initial population is generated according to the neural network parameters output by SGD, and then the excellent solution set is selected by the roulette wheel method. Crossover operation and mutation operation are performed on the selected solution set. The random disturbance is performed through these two operations to help SGD jump out of the local optimal trap, and better model parameters are trained by GA-SGD to obtain higher accuracy of small pests detection in the field.

Our main contributions can be summarized as follows:

- Due to the features of small pests in the field are not obvious, the small pests detection networks are usually deep models. We propose an optimization algorithm based on evolution strategies to address the problem that SGD cannot efficiently train deep models.
- For the premature convergence problem of the stochastic gradient descent method, we propose stochastic gradient descent with genetic algorithm (GA-SGD) based on selection strategy, crossover and mutation strategy to improve the performance of the basic SGD.
- Three object detectors are trained using GA-SGD for the field detection of four small pests (Sitobion avenae of wheat; Planthopper of rice; Cruciferae padi of rape; Rhopalosiphum maidis of corn). Compared with the five algorithms, the higher detection accuracy and stability are obtained by GA-SGD.

The remainder of this paper is organized as follows. Section 2 describes the pests dataset, the modified GA-SGD and the experimental details. Section 3 verifies and discusses the algorithm performance of GA-SGD on pests detection. Finally, Section 4 summarizes the findings and concludes the paper.

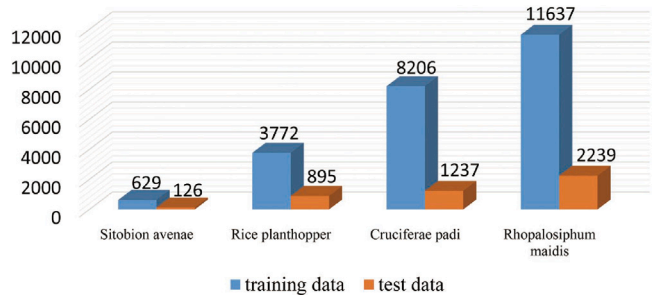

Fig. 2. The number of four pests labeled in the dataset.

2. Materials and methods

2.1. Small pests datasets

In the algorithm evaluation phase, this paper uses the small pest dataset Agripest (Wang et al., 2021) to verify the algorithm performance. The dataset consists of four tiny pests (Sitobion avenae of wheat; Planthopper of rice; Cruciferae padi of rape; Rhopalosiphum maidis of corn), and the image resolution is 1440*1080, including the training set of 772 images and the test set of 148 images. The number of four pests labeled in the dataset is shown in Fig. 2. The annotation visualization of small pests dataset is displayed in Fig. 13.

2.2. Small pests detector with GA-SGD

In this paper, three two-stage object detection networks (Ren et al., 2017) are used to detect small pests. Fig. 3 shows the architecture of the two-stage object detection network with GA-SGD. And Fig. 3 is analyzed as follows: we first input the four small pest dataset to residual neural network (resnet50, He et al., 2016) for feature extraction. Then the feature maps are obtained by resnet50, and the feature map is input into the feature pyramid network (FPN, Lin et al., 2017) for feature fusion. After that, the enhanced feature map is input into the region proposal network (RPN, Ren et al., 2017) to classify the corresponding background and foreground from the feature map. And we apply the region of interest (ROI, Ren et al., 2017) to receive the region proposals from RPN and further refine them. Finally, full Convolutional Network is employed to locate and classify the four small pests. The above description is the neural network structure for small pest detection. Next, we describe the process of GA-SGD optimizing the parameters of the detector in Fig. 3. GA-SGD is based on the original SGD. First, the output parameters W of SGD are applied to create the initial population of GA. Then GA-SGD performs three evolution strategies (selection operation, crossover operation, mutation operation) to further optimize the parameters W of the network. At last, the parameters of the neural network are optimized by GA-SGD to enhance the detection accuracy of small pests.

2.3. Optimization goal: Loss function of object detection

The object detection task is to output the location and category of the pests in the image, so the loss function of the neural network is the sum of regression loss and classification loss. The mathematical formulation of the loss function is shown as Eq. (1).

$$L\left(p_i,t_i\right)=\frac{\lambda}{N_r}\sum_i p_i^* L_r\left(t_i,t_i^*\right)+\frac{1}{N_c}\sum_i L_c\left(p_i,p_i^*\right). \tag{1}$$

Where i is the index of an anchor in a mini-batch; p_i and p_i^* are the predicted and real probability of anchor i being a pest; t_i and t_i^* are the predicted border and the real border of anchor i being a pest; N_c is the mini-batch size; N_r is the number of anchors; λ is a balancing parameter to normalize and weight N_c and N_r, usually set to 10; L_r applies smooth L1 loss as regression loss for location prediction of pests, and L_c applies cross-entropy loss as classification loss for class prediction of pests. For more detailed information, please refer to literature Girshick (2015) and Ren et al. (2017).

2.4. The proposed GA-SGD algorithm

GA-SGD performs three evolution strategies based on the original SGD. The basic information of the original SGD is as follows: a mini-batch of s examples is sampled from the training set $\{x_1,x_2,\ldots,x_s\}$ with predicted values y_j. The gradient g is computed as Eq. (2), where w is the network parameters.

$$g=\frac{1}{s}\nabla_w\sum_j^s L\left(f\left(x_j;w\right),y_j\right). \tag{2}$$

The SGD with momentum is updated as Eq. (3) and Eq. (4). Where v is the velocity, α is the momentum, and lr is the learning rate.

$$v=\alpha*v-lr*g. \tag{3}$$

$$w=w+v. \tag{4}$$

The above description is the introduction of the original SGD. However, the original SGD algorithm is prone to fall into the local optimum, and the gradient g of the local optimum is 0, which will affect the parameter optimization of the neural network and lead to the decline of the accuracy of the trained detection model. The rationale of the GA-SGD is to carry out three evolution operations on the basis of SGD optimizing the parameters of the model. These operations have disturbance effects on the search process of SGD and maybe obtain a relatively better solution. The disturbance is one of the core ideas of genetic algorithms. GA-SGD adopts three evolution strategies to help SGD jump out of the local optimum. The evolution strategies consist of selection operation, crossover operation, mutation operation.

The basic steps of GA-SGD algorithm are as follows:

- The SGD algorithm is executed by Eq. (3) and Eq. (4) to output the neural network parameter matrix W. w in Eq. (4) is a value in matrix W.
- The output value W of the SGD is used to initialize the population of GA.
- GA performs selection operation, crossover operation, and mutation operation to further optimize parameter W. Finally, W is returned to the loss function

2.4.1. SGD output network parameters

The SGD algorithm is executed by Eq. (3) and Eq. (4) to output the neural network parameter matrix W, w in Eq. (4) is a value in matrix W. The number of parameters of the neural network is at the level of millions, we adopt the mini-batch method to optimize a large number of parameters. As shown in Fig. 4, the matrix W is composed of $M*N$ w in a minibatch. In order to calculate effectively, all parameters of the neural network are divided into many minibatchs, each minibatch is $M*N$ w, this paper set M as 1, N as 1024. The original SGD also uses the mini-batch method for parallel training.

2.4.2. Population initialization

The matrix W output by SGD is input into GA-SGD to initialize the population. As shown in Fig. 5, the matrix W is multiplied by a random number uniformly distributed between 0 and 1 to generate a new matrix W. The number of random numbers is pop. The matrix

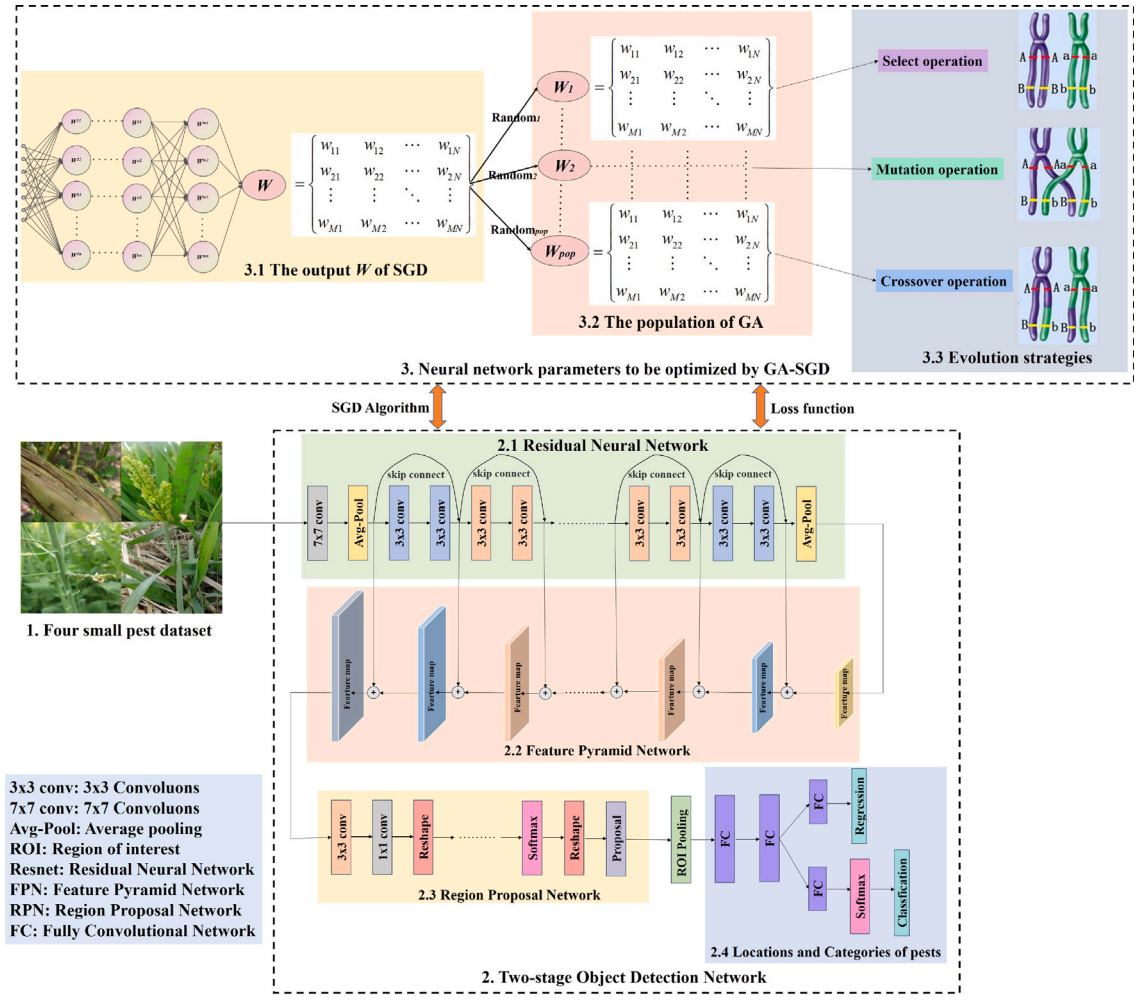

Fig. 3. The architecture of two-stage object detection network with GA-SGD.

W with pop number forms the initial population, the population is calculated by Eq. (5).

$$\begin{aligned} popolation &= W * \{random_1, random_2, \ldots, random_{pop}\} \\ &= \begin{Bmatrix} w_{11} & w_{12} & \cdots & w_{1N} \\ w_{21} & w_{22} & \cdots & w_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ w_{M1} & w_{M2} & \cdots & w_{MN} \end{Bmatrix} * \{random_1, random_2, \ldots, random_{pop}\} \\ &= W_1, W_2, \ldots, W_{pop} \end{aligned} \tag{5}$$

2.4.3. Selection operation

Selection operation selects fine individuals from the old population with a certain probability to form a new population, and reproduce the next generation of individuals. The probability of an individual being selected is related to the fitness value. The higher the fitness value of the individual, the higher the probability of being selected. The fitness value of the individual W is f_W. Since it is a minimization problem, the fitness function is the reciprocal of the objective function. The individual W is selected with the probability P_W as Eq. (6). The cumulative probability of the individual is calculated as Eq. (7). The line segments transformed by cumulative probability are shown in Fig. 6. A random number is generated between the interval [0,1], and the random number is in which probability range, which individual is selected.

$$P_{W_k} = \frac{f_{W_k}}{\sum_{k=1}^{pop} f_{W_k}}. \tag{6}$$

$$Q_{W_k} = \sum_{k=1}^{pop} P_{W_k}. \tag{7}$$

If an individual has a high probability of selection, it has a chance to be selected multiple times, then its values will be expanded in the population; if an individual has a small probability of selection, it will have a high probability of being eliminated.

2.4.4. Crossover operation

Crossover operation means that two individuals are randomly selected from the population, the superior characteristics of the parent matrix are inherited by the offspring matrix by exchanging and combining the two matrix, thus producing the new individuals. Suppose that the crossover operation is performed between W_a and W_b. P_c is the probability of crossover, when the random number d between [0, 1] is less than the crossover probability, the crossover operation is performed. c is the position of crossover and the crossover operation formula is as Eq. (8).

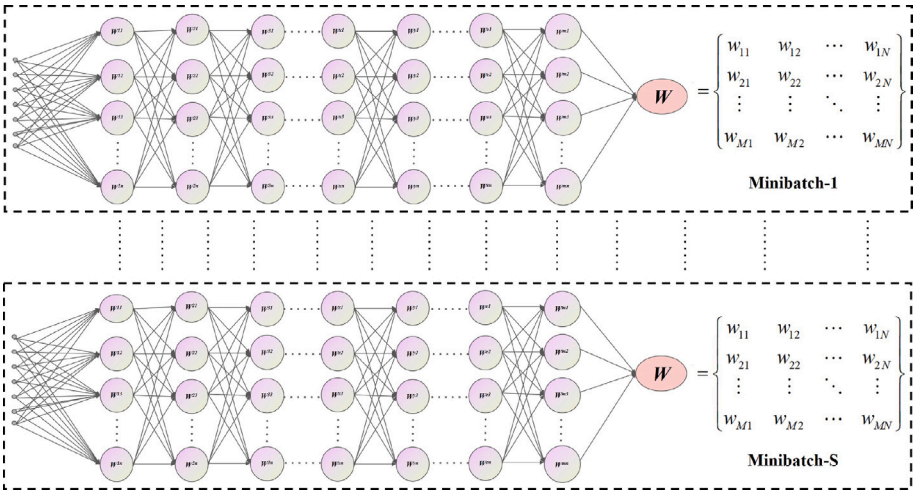

Fig. 4. The minibatch of neural network parameter.

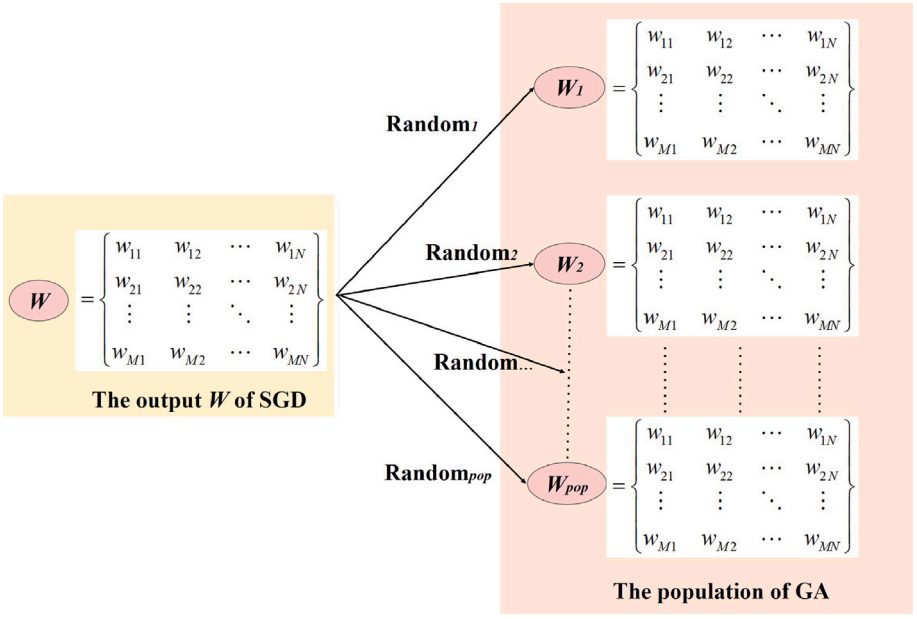

Fig. 5. The initialization of population.

$$\begin{Bmatrix} w_{11}^a & w_{12}^a & \cdots & w_{1N}^a \\ w_{21}^a & w_{22}^a & \cdots & w_{2N}^a \\ \vdots & \vdots & w_c & \vdots \\ w_{M1}^a & w_{M2}^a & \cdots & w_{MN}^a \end{Bmatrix} \begin{Bmatrix} w_{11}^b & w_{12}^b & \cdots & w_{1N}^b \\ w_{21}^b & w_{22}^b & \cdots & w_{2N}^b \\ \vdots & \vdots & w_c & \vdots \\ w_{M1}^b & w_{M2}^b & \cdots & w_{MN}^b \end{Bmatrix}$$
$$\Downarrow \text{crossover} \tag{8}$$
$$\begin{Bmatrix} w_{11}^b & w_{12}^b & \cdots & w_{1N}^a \\ w_{21}^b & w_{22}^b & \cdots & w_{2N}^a \\ \vdots & \vdots & w_c & \vdots \\ w_{M1}^b & w_{M2}^b & \cdots & w_{MN}^a \end{Bmatrix} \begin{Bmatrix} w_{11}^a & w_{12}^a & \cdots & w_{1N}^b \\ w_{21}^a & w_{22}^a & \cdots & w_{2N}^b \\ \vdots & \vdots & w_c & \vdots \\ w_{M1}^a & w_{M2}^a & \cdots & w_{MN}^b \end{Bmatrix}$$

2.4.5. Mutation operation

Mutation operation replaces the original value with a random number, when the random number d between [0, 1] is less than the mutation probability P_m, the mutation operation is performed. Suppose that the individual W, where m is the position of mutation, and the mutation value range is from $w_{\min}$ to $w_{\max}$. A new individual W' can be obtained after the mutation operation of w_m. The new value of the mutation point is calculated as Eq. (9), the crossover operation formula is as Eq. (10).

$$w'_m = w_{\min} + random * \left(w_{\max} - w_{\min}\right). \tag{9}$$

$$W = \begin{Bmatrix} w_{11} & w_{12} & \cdots & w_{1N} \\ w_{21} & w_{22} & \cdots & w_{2N} \\ \vdots & \vdots & w_m & \vdots \\ w_{M1} & w_{M2} & \cdots & w_{MN} \end{Bmatrix}$$
$$\Downarrow \text{mutation} \tag{10}$$
$$W' = \begin{Bmatrix} w_{11} & w_{12} & \cdots & w_{1N} \\ w_{21} & w_{22} & \cdots & w_{2N} \\ \vdots & \vdots & w'_m & \vdots \\ w_{M1} & w_{M2} & \cdots & w_{MN} \end{Bmatrix}$$

2.4.6. The pseudocode of GA-SGD

The pseudocode of GA-SGD is analyzed as follow: it is worth noting that the termination criterion used in the research work is the number of the current iteration, also known as the number of fitness evaluations

(NFE). In the preparation phase, the related parameters are defined and created; a minibatch of examples is sampled from the training set with predicted values and compute gradient estimate by Eq. (2); velocity is computed by Eq. (3) and network parameters are updated by Eq. (4). The neural network parameters of the previous steps are used to generate the initial population solution; the fitness value is calculated by the objective function; GA-SGD select the individual by Eq. (6) to perform crossover operation and mutation operation. The fitness value of the individual is sorted and the best fitness value is used to update the solution. Until the last iteration, the optimal solution are outputted. The structure of two-stage object detection network with GA-SGD is displayed in Fig. 3.

Algorithm 1 Pseudocode of the GA-SGD algorithm.

Initialize the parameters of GA-SGD: population size (pop), crossover probability (P_c), mutation probability (P_m), the random number d, learning rate (lr), momentum parameter (α), the maximum number of iteration (t_{max}), the number of fitness evaluations (NFE).
The examples are sampled from the training set with the predicted value.
The gradient and velocity are computed according to Eq. (2) and Eq. (3) respectively.
The network parameters are updated according to Eq. (4) and the initial solution of the population is generated by Eq. (5).
Set T as the best solution.
while $NFE < t_{max}$ **do**
  The fitness value of the individual is calculated according to the objective function.
  for $i = 0 \rightarrow pop$ **do**
    Selection operation is performed.
    if $d<P_c$, crossover operation is performed.
    if $d<P_m$, mutation operation is performed.
  end for
  Sort the fitness value of the individual.
  Update T according to the best fitness value.
  $NFE = NFE + 1$.
end while
Output the T.

2.5. Experimental environments and settings

The experiments are performed on the graphics workstation with Ubuntu20.04. The hardware environment is Intel Xeon Silver-4210R CPU, 2*RTX 5000 GPU; the programs are written in Python. We also want to use GA-SGD on MindSpore,[1] which is a new deep learning framework. These problems are left for future work. Small pests detection models use deep backbone network resnet50 to extract features, feature pyramid networks perform feature enhancement, and two-stage detection heads perform location regression and category classification of pests. The proposed algorithm GA-SGD, SGD and Adam are used to train three two-stage object detectors (Faster RCNN Ren et al., 2017, Dynamic RCNN Zhang et al., 2020, Double-Head R-CNN Wu et al., 2020) with IOU= 0.5. Each detection model runs 24 epochs, the best, worst, standard deviation, confidence interval and mean MAP of all epochs are shown in Table 3. The boxplots of MAP with three optimizers are displayed in Figs. 7–9. AP measures the performance of the model in each category. MAP measures the performance of the model across all categories. The MAP curve of six algorithms are compared in Table 3. The hyperparameters of the optimization algorithms are shown in Table 1. Where β_1 and β_2 are the exponential decay rate for the 1st moment estimates and the 2nd moment estimates. ϵ is a small value for numerical stability. $initial\_accumulator$ is a float value and the initial value of the accumulator must be positive. rho is the sliding mean hyperparameter for calculating the square of the gradient. The description of the variables and abbreviations are shown in Table 4.

2.6. Evaluation indexes

In addition, the evaluation indexes of the performance as follow:

[1] https://www.mindspore.cn/

Table 1
The hyperparameters of the optimization algorithms.

Algotithm	Parameter setting
SGD	$lr = 0.005,\ weight\_decay = 0.0001,\ \alpha = 0.9.$
Adam	$lr = 0.0001,\ weight\_decay = 0.0001,\ \alpha = 0.9,\ \beta_1 = 0.9,$ $\beta_2 = 0.999,\ \epsilon = 10^{-8}.$
AdamW	$lr = 0.0001,\ weight\_decay = 0.0001,\ \alpha = 0.9,\ \beta_1 = 0.9,$ $\beta_2 = 0.999,\ \epsilon = 10^{-8}.$
Adagrad	$lr = 0.005,\ weight\_decay = 0.0001,\ \alpha = 0.9,$ $initial\_accumulator = 0.1,\ \epsilon = 10^{-10}.$
Adadelta	$lr = 1,\ weight\_decay = 0.0001,\ rho = 0.9,\ \epsilon = 10^{-8}.$
GA-SGD	$lr = 0.005,\ weight\_decay = 0.0001,\ \alpha = 0.9,\ pop = 10,\ P_m = 0.1,$ $P_c = 0.7.$

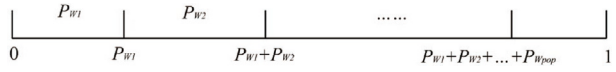

Fig. 6. The line segments of the cumulative probability.

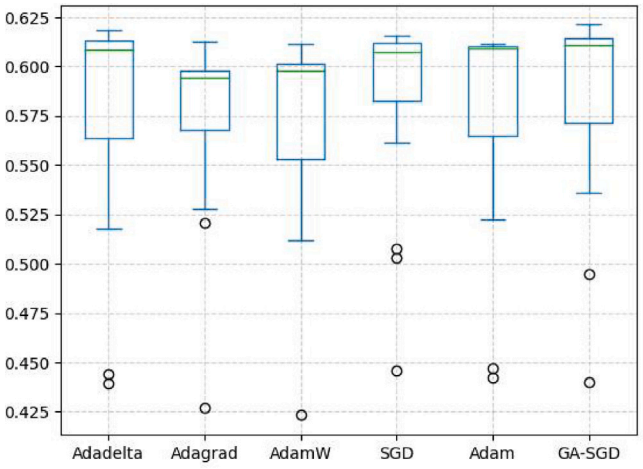

Fig. 7. The boxplot of faster RCNN for pests detection.

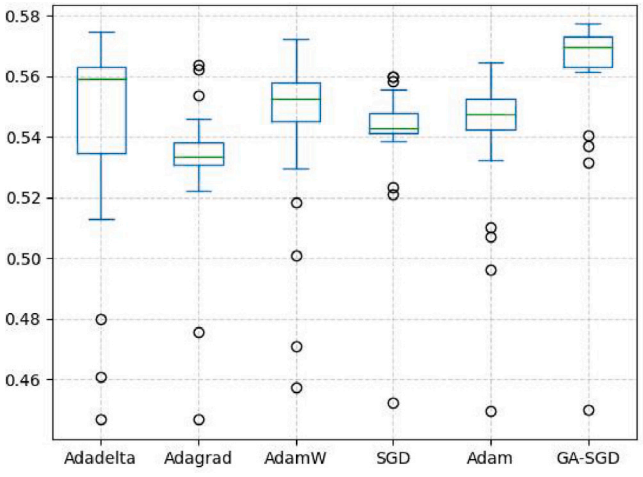

Fig. 8. The boxplot of Dynamic RCNN for pests detection.

- MAP, Mean Average Precision is the average precision of all detection categories, and MAP is the recognized index for the performance evaluation of the object detection model. MAP (Henderson and Ferrari, 2017) is calculated from Recall and Precision.

Table 2
The best, worst, mean, standard deviation and confidence interval of MAP.

Detector	Optimizer	Best	Worst	Mean	Std	CI	Δ CI
Faster RCNN	SGD	0.6155	0.4457	0.5872	0.0425	0.5872 ± 0.0179	0.0359
	Adam	0.6120	0.4422	0.5805	0.0500	0.5805 ± 0.0211	0.0422
	AdamW	0.6115	0.4232	0.5772	0.0426	0.5772 ± 0.0180	0.0360
	Adagrad	0.6127	0.4270	0.5761	0.0403	0.5761 ± 0.0170	0.0340
	Adadelta	0.6187	0.4395	0.5830	0.0497	0.5832 ± 0.0210	0.0420
	GA-SGD	**0.6212**	0.4397	**0.5875**	0.0444	0.5875 ± 0.0187	0.0375
Dynamic RCNN	SGD	0.5597	0.4522	0.5409	0.0206	0.5409 ± 0.0087	0.0174
	Adam	0.5647	0.4495	0.5396	0.0250	0.5396 ± 0.0105	0.0211
	AdamW	0.5725	0.4572	0.5426	0.0279	0.5426 ± 0.0118	0.0236
	Adagrad	0.5640	0.4467	0.5310	0.0235	0.5310 ± 0.0099	0.0199
	Adadelta	0.5745	0.4467	0.5440	0.0341	0.5440 ± 0.0144	0.0288
	GA-SGD	**0.5772**	0.4500	**0.5611**	0.0261	0.5611 ± 0.0110	0.0220
Double-Head RCNN	SGD	0.6085	0.4737	0.5881	0.0280	0.5881 ± 0.0118	0.0236
	Adam	0.6082	0.4402	0.5764	0.0321	0.5764 ± 0.0135	0.0271
	AdamW	0.6097	0.4597	0.5755	0.0275	0.5755 ± 0.0116	0.0232
	Adagrad	0.6152	0.4595	0.5795	0.0290	0.5795 ± 0.0122	0.0245
	Adadelta	0.6122	0.4410	0.5827	0.0372	0.5827 ± 0.0157	0.0314
	GA-SGD	**0.6217**	0.4535	**0.5940**	0.0423	0.5940 ± 0.0178	0.0357

Table 3
The AP of four small pests.

Detector	Optimizer	SA	RP	CP	RM	MAP
Faster RCNN	SGD	0.822	0.664	**0.458**	0.518	0.6155
	Adam	**0.844**	0.654	0.440	0.510	0.6120
	AdamW	0.829	0.678	0.427	0.512	0.6115
	Adagrad	0.831	0.678	0.426	0.516	0.6127
	Adadelta	0.843	0.672	0.444	0.516	0.6187
	GA-SGD	**0.844**	**0.682**	0.437	**0.522**	**0.6212**
Dynamic RCNN	SGD	0.803	0.628	0.398	0.410	0.5597
	Adam	0.816	0.648	0.384	0.411	0.5647
	AdamW	**0.836**	0.637	0.398	0.419	0.5725
	Adagrad	0.822	0.637	0.376	0.421	0.5640
	Adadelta	0.829	0.652	**0.402**	0.415	0.5745
	GA-SGD	**0.836**	**0.653**	0.392	**0.428**	**0.5772**
Double-Head RCNN	SGD	0.812	0.661	**0.445**	0.516	0.6085
	Adam	0.825	0.667	0.425	0.516	0.6082
	AdamW	0.825	0.675	0.425	0.514	0.6097
	Adagrad	0.838	**0.685**	0.427	0.511	0.6152
	Adadelta	0.841	0.665	0.428	0.515	0.6125
	GA-SGD	**0.861**	0.663	0.443	**0.520**	**0.6217**

SA: Sitobion avenae, RA: Rice planthopper, CP: Cruciferae padi, RM: Rhopalosiphum maidi.

The calculation formula of MAP is as Eq. (11).

$$MAP = \frac{\sum_{h=1}^{h} AP_h}{h} \tag{11}$$

Where h is the number of pest species; AP is the area under the curve formed by the horizontal coordinate of Recall and the vertical coordinate of Precision. Recall and Precision are calculated as Eqs. (12) and (13).

$$Recall = TP/(TP + FN) \tag{12}$$

$$Precision = TP/(TP + FP) \tag{13}$$

Where TP is True Positive, which refers to the number of pests that are correctly classified as Pest A, that is, the actual pest A is also classified as pest A by the model; FP is False Positive, which refers to the number of pests misclassified as Pest A, that is, the actual pest B is classified as pest A by the model; TN is True Negative, which refers to the number of pests that are correctly classified as Pest B, that is, the actual pest B is also classified as pest B by the model; FN is False Negative, which refers to the number of pests misclassified as Pest B, that is, the actual pest A is classified as pest A by the model.

- $Best$ and $Worst$ are the maximum MAP value and the minimum MAP value in all epochs and defined as Eqs. (14) and (15).

$$Best = \max_{1 \le i \le n} x_i \tag{14}$$

$$Worst = \min_{1 \le i \le n} x_i \tag{15}$$

- $Mean\ value$ is the average of the values obtained from n epochs and denoted as Eq. (16).

$$Mean = \frac{x_1 + x_2 + \cdots + x_n}{n} = \frac{\sum_{i=1}^{n} x_i}{n} \tag{16}$$

- $Standard\ deviation$ reflects the degree of dispersion between individuals within a solution set. It is calculated as Eq. (17).

$$Std = \sqrt{\frac{1}{n-1} \sum_{i=1}^{n} (x_i - Mean)^2} \tag{17}$$

- $Confidence\ interval$ refers to the estimated interval of the overall parameters constructed by the sample statistics. The formula is expressed as Eq. (18).

$$CI = mean + t * \frac{Std}{\sqrt{n}} \tag{18}$$

When the confidence level is 95%, $n = 24$, it is found that freedom is 23 and double-tail is 0.05 in the t-test table, that is, the critical value of t is 2.069 in the middle 95%.

3. Results and discussion

Table 2 shows the best, worst, mean, standard deviation and confidence interval of MAP achieved by the six optimizers. The GA-SGD obtains the relative optimal MAP than the other five algorithms in all detectors, which means that GA-SGD trains the detectors with better convergence accuracy. And the mean MAP of the GA-SGD is also higher than others, it means the MAP value of the whole epochs is relatively better. In terms of the worst value, SGD achieves relatively good values, but we can further analyze the performance from the macro perspective of the boxplots (Figs. 7–9). In these boxplots, the central mark of the box is the median; the edges of the box are the 25th and 75th percentiles; the whiskers extend to the most extreme data points that are considered outliers. According to the outlier statistics of the three detectors, the outliers of the other five algorithms are more than GA-SGD. And the worst values of all six algorithms are outliers, but the lower edge of GA-SGD is higher than other algorithms on Faster RCNN and Dynamic RCNN. The standard deviation of GA-SGD is relatively

Table 4
Description of the variables and abbreviations.

The variables and abbreviations	Description
i	The index of an anchor in a mini-batch.
p_i	The predicted probability of anchor i being a pest.
p_i^*	The real probability of anchor i being a pest.
t_i	The predicted border of anchor i being a pest.
t_i^*	The real border of anchor i being a pest.
N_c	The mini-batch size.
λ	The balancing parameter to weight N_c and N_r.
N_r	The number of anchors.
W	The network parameters to be optimized.
w	w in Eq. (4) is a value in matrix W.
v	The velocity vector.
g	The gradient vector.
α	The momentum factor.
lr	The learning rate.
$weight\_decay$	To achieve the purpose of regularization and reduce the problem of model overfitting.
pop	The size of the population.
K	The individual of the population.
K'	The mutant individual.
β_1	The exponential decay for the 1st moment.
β_2	The exponential decay for the 2nd moment.
ϵ	The small value for numerical stability.
$initial\_accumulator$	The float value and the accumulator must be positive.
rho	The sliding mean hyperparameter for calculating the square of the gradient.
f_K	The fitness value of the individual.
P_K	The probability of the selection operation.
P_c	The probability of the crossover operation.
P_m	The probability of the mutation operation.
t_{max}	The maximum number of iteration.
NFE	The number of fitness evaluation.
T	The best solution.
AP	The area under the curve formed by the Recall and the Precision.
MAP	The average precision of all detection categories.
$Best$	The maximum map value in all the epochs.
$Worst$	The minimum map value in all the epochs.
$Mean$	The average map obtained from n epochs.
Std	The standard deviation.
CI	The confidence interval.
SGD	Stochastic gradient descent algorithm.
GA	Genetic algorithm.
Adam	Adaptive momentum algorithm.
AdamW	Adam combined with L2 regularization.
Adagrad	Adaptive gradient algorithm.
Adadelta	The variant of Adagrad.
GA-SGD	The SGD with genetic algorithm.

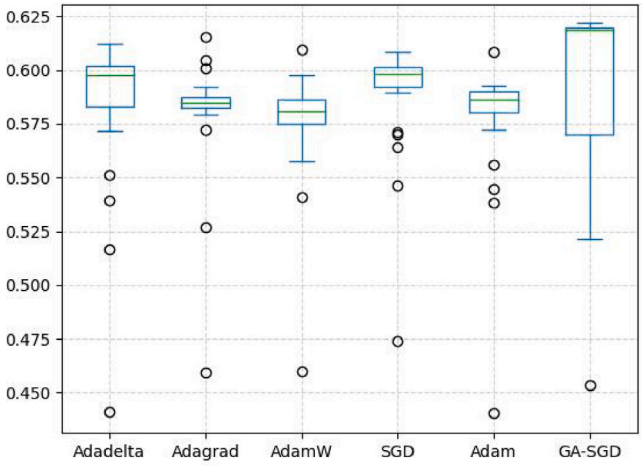

Fig. 9. The boxplot of Double-Head RCNN for pests detection.

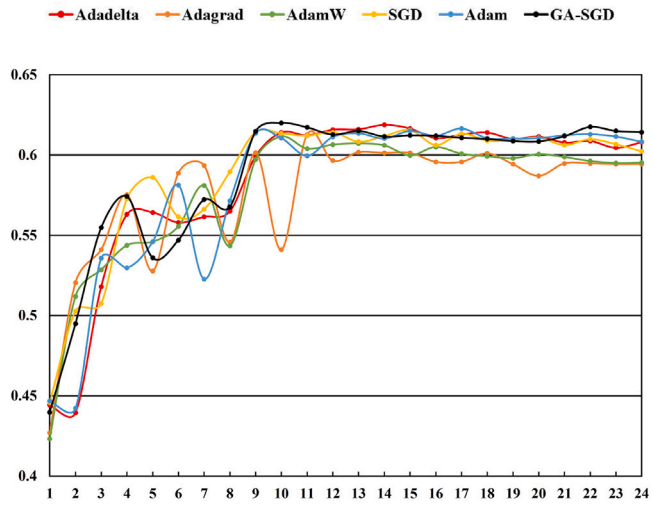

Fig. 10. The MAP curve of six algorithms in Faster RCNN.

higher, it means the MAP of GA-SGD has a high degree of dispersion; However, the high dispersion of GA-SGD has no significant impact on the performance, because the upper edge of GA-SGD is the maximum value, the lower edge is also higher than most algorithms and there are fewer outliers. Several algorithms show that the outliers are the optimal MAP in Figs. 8 and 9, rather than the upper edge of the boxplots. It means that the performance of the other algorithms is less stable than

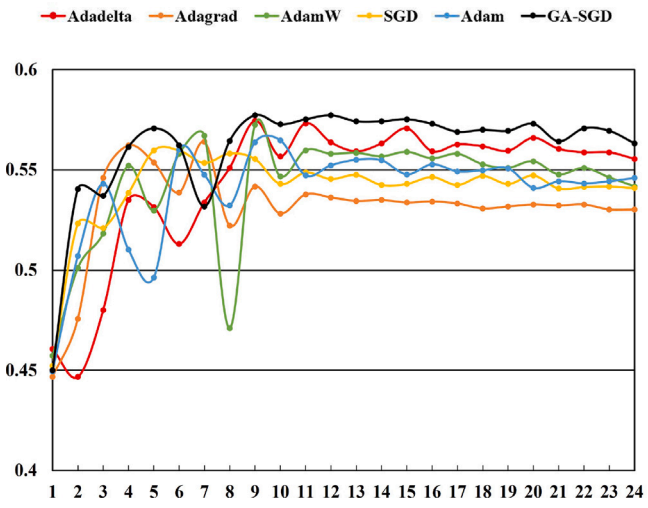

Fig. 11. The MAP curve of six algorithms in Dynamic RCNN.

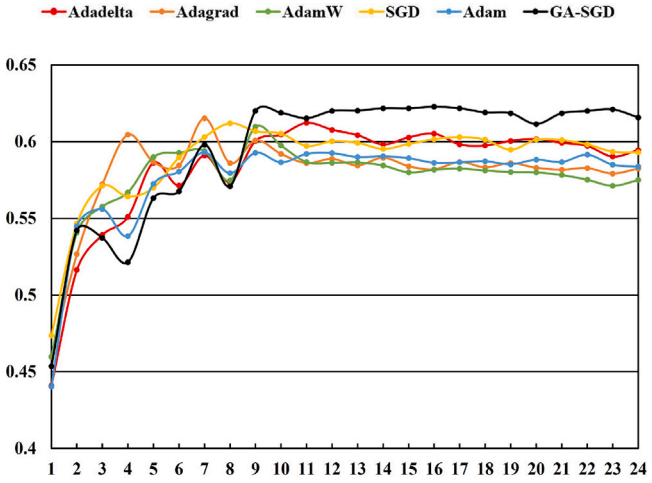

Fig. 12. The MAP curve of six algorithms in Double-Head RCNN.

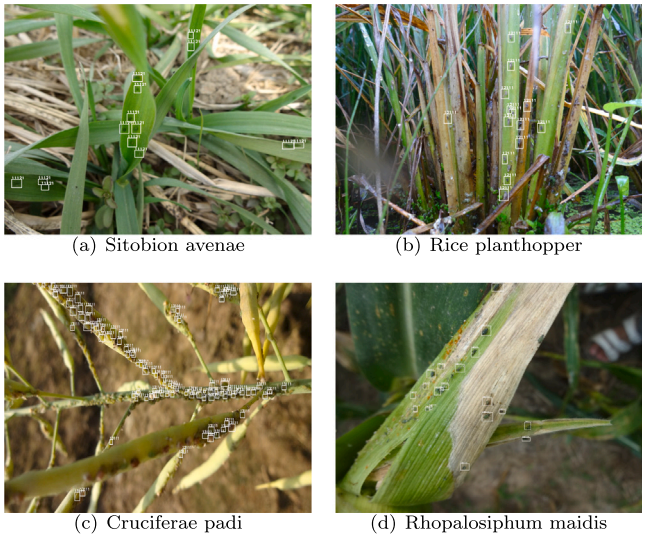

Fig. 13. The groundtruth of the four small pests.

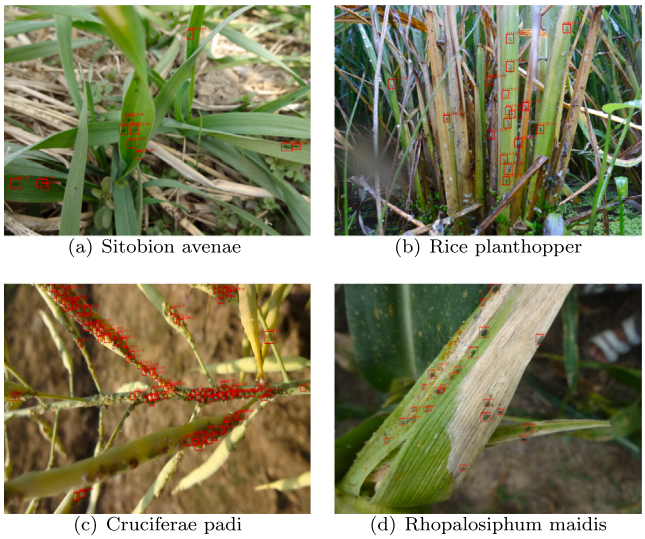

Fig. 14. The detection visualization of four small pests with GA-SGD.

GA-SGD. In contrast, the best MAP of GA-SGD in all detectors is on the upper edge of the boxplots, it can be further clearly observed from the boxplots that the obtained MAP of GA-SGD is higher than the other five algorithms. The confidence interval of GA-SGD is wider than most algorithms, it represents the strong ability to search the relatively better solutions. In summary, the overall performance of GA-SGD is superior to the other algorithms. Table 3 shows the single class AP value of four small pests under the best MAP. The GA-SGD obtains the optimal AP on the Sitobion avenae, Rice planthopper and Rhopalosiphum maidis, the AP value increased by about $1\%-5\%$. The AP of GA-SGD on Cruciferae padi is similar to the other algorithms. It reveals that GA-SGA has the fine search ability and can find more excellent solutions. The MAP of GA-SGD is better than the other five algorithms. The above shows that the GA-SGD can get effective results in four small pests detection tasks.

In Figs. 10–12, the horizontal coordinates represent the number of epochs trained by the model and the vertical coordinates represent the MAP obtained by the six algorithms. In the previous epochs, it is observed that GA-SGD has a faster convergence rate than most algorithms to find the local optimum. Then, GA-SGD continuously performs selection, crossover and mutation operations to jump out of the local optimal solutions and search for relatively better solutions. It can be found from the MAP curve of the ninth epoch to the last epoch that Dynamic RCNN and Double-Head RCNN trained by GA-SGD can obtain the sustained advantages over the other five algorithms for detecting four pests. It can be clearly seen from the figures that GA-SGD rises to the highest point and obtains the optimal MAP value on the three detectors. According to the above discussion, the proposed GA-SGD achieves better convergence rate and higher accuracy than other algorithms.

Fig. 14 displays the field detection effect of GA-SGD on four small pests. Compared with the groundtruth of the dataset (Fig. 13), most pests are located and identified, but the missing and wrong detection occurred due to the lack of feature information and the complex background environment, which led to the decline of the overall detection accuracy. In the future, we can use evolutionary algorithms to optimize the object detection network structure based on the real-time feedback of pest detection accuracy, to design a more suitable neural network structure for small pests detection. This study only optimizes the parameters of the model and does not involve the optimization of the structure, which is the main research content of our next work and can substantially improve the small pests detection accuracy.

4. Conclusion

The two-stage detectors show excellent performance on the small pests detection task, but SGD is difficult to train the deep and complex neural network effectively. For the above issue, we propose the GA-SGD combining selection operation, crossover operation, and mutation operation to enhance the convergence performance of basic SGD. The best and mean of MAP reveal that GA-SGD is superior to five algorithms in detection accuracy. The boxplots of three optimizers show that the performance of GA-SGD is relatively stable. The data reveals that four small pests are detected through GA-SGD with superiority.

The limitations of GA-SGD as follow: (1) The implementation of the three operations also has many parameters, such as crossover rate and mutation rate, and the selection of these parameters seriously affects the quality of the solution, but at present, the selection of these parameters mostly depends on experience. (2) The algorithm depends on the selection of the initial population and can be improved with some heuristic algorithms.

In future work, the evolutionary algorithm applied in small pests detection will be further extended. The evolutionary algorithm optimizes the parameters of deep neural networks in this paper. The current two-stage detection network structure is not the optimal form in the field of pests recognition. We will use evolutionary algorithms to automate the design of the neural network structure based on feedback from the accuracy of small pests detection. In the future, the evolutionary algorithm is applied to optimize both the parameters and the structure of the neural network to design the best neural network model for small pests detection.

CRediT authorship contribution statement

Yin Ye: Conceptualization, Methodology, Investigation, Formal analysis, Writing – original draft. **Qiangqiang Huang:** Data curation, Writing – original draft. **Yi Rong:** Visualization, Investigation. **Xiaohan Yu:** Resources, Supervision. **Weiji Liang:** Validation, Writing – review & editing. **Yaxiong Chen:** Visualization, Writing – review & editing. **Shengwu Xiong:** Conceptualization, Funding acquisition, Resources, Supervision, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The authors do not have permission to share data.

Acknowledgments

This work was in part supported by the Project of Sanya Yazhou Bay Science and Technology City, China (Grant No. SCKJ-JYRC-2022-76, Grant No. SCKJ-JYRC-2022-17), the Hainan Special PhD Scientific Research Foundation of Sanya Yazhou Bay Science and Technology City, China (Grant No. HSPHDSRF-2022-03-017), the National Natural Science Foundation of China (Grant No. 62101393, Grant No. 62176194), the Major project of IoV (Grant No. 2020AAA001), Sanya Science and Education Innovation Park of Wuhan University of Technology, China (Grant No. 2021KF0031, Grant No. 2022KF0020, Grant No. 2022KF0032), the Youth Fund Project of Hainan Natural Science Foundation, China (Grant No. 622QN344), CAAI-Huawei MindSpore Open Fund, China (Grant No. CAAIXSJLJJ-2022-001A), the Natural Science Foundation of Chongqing, China (Grant No. cstc2021jcyj-msxmX1148) and the Open Project of Wuhan University of Technology Chongqing Research Institute, China (ZL2021-6). We thank MindSpore for the partial support of this work, which is a new deep learning framework https://www.mindspore.cn/.

References

Antonakopoulos, K., Mertikopoulos, P., Piliouras, G., Wang, X., 2022. AdaGrad avoids saddle points. In: International Conference on Machine Learning. PMLR, pp. 731–771.

Antonio, L.M., Coello, C.A.C., 2017. Coevolutionary multiobjective evolutionary algorithms: Survey of the state-of-the-art. IEEE Trans. Evol. Comput. 22 (6), 851–865.

Bay, H., Tuytelaars, T., Gool, L.V., 2006. Surf: Speeded up robust features. In: European Conference on Computer Vision. Springer, pp. 404–417.

Bertens, P., Lee, S.-W., 2020. Network of evolvable neural units can learn synaptic learning rules and spiking dynamics. Nat. Mach. Intell. 2 (12), 791–799.

Cheridito, P., Jentzen, A., Rossmannek, F., 2021. Non-convergence of stochastic gradient descent in the training of deep neural networks. J. Complexity 64, 101540.

Cui, X., Picheny, M., 2019. Acoustic model optimization based on evolutionary stochastic gradient descent with anchors for automatic speech recognition. In: Proc. Interspeech 2019. pp. 1581–1585.

Cui, X., Zhang, W., Tüske, Z., Picheny, M., 2018. Evolutionary stochastic gradient descent for optimization of deep neural networks. Adv. Neural Inf. Process. Syst. 31.

Girshick, R.B., 2015. Fast R-CNN. In: 2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7–13, 2015. IEEE Computer Society, pp. 1440–1448.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778.

Henderson, P., Ferrari, V., 2017. End-to-end training of object class detectors for mean average precision. In: Asian Conference on Computer Vision. Springer, pp. 198–213.

Huang, M.-L., Chuang, T.-C., Liao, Y.-C., 2022. Application of transfer learning and image augmentation technology for tomato pest identification. Sustain. Comput.: Inform. Syst. 33, 100646.

Karen Simonyan, A.Z., 2014. Going deeper with convolutions. In: International Conference on Learning Representations.

Kingma, D.P., Ba, J., 2015. Adam: A method for stochastic optimization. In: International Conference on Learning Representation.

Krizhevsky, A., Sutskever, I., Hinton, G.E., 2017. ImageNet classification with deep convolutional neural networks. Commun. ACM 60 (6), 84–90.

Li, W., Zheng, T., Yang, Z., Li, M., Sun, C., Yang, X., 2021. Classification and detection of insects from field images using deep learning for smart pest management: A systematic review. Ecol. Inform. 66, 101460.

Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S., 2017. Feature pyramid networks for object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2117–2125.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L., 2014. Microsoft coco: Common objects in context. In: European Conference on Computer Vision. Springer, pp. 740–755.

Liu, X., Feng, R., Zhou, S., Yang, Y., 2021a. A novel PSO-SGD with momentum algorithm for medical image classification. In: 2021 IEEE International Conference on Bioinformatics and Biomedicine. BIBM, IEEE, pp. 3408–3413.

Liu, Y., Sun, P., Wergeles, N., Shang, Y., 2021b. A survey and performance evaluation of deep learning methods for small object detection. Expert Syst. Appl. 172, 114602.

Loshchilov, I., Hutter, F., 2019. Decoupled weight decay regularization. In: International Conference on Learning Representation.

Lowe, D.G., 2004. Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. 60 (2), 91–110.

Miikkulainen, R., Forrest, S., 2021. A biological perspective on evolutionary computation. Nat. Mach. Intell. 3 (1), 9–15.

Oliva, A., Torralba, A., 2001. Modeling the shape of the scene: A holistic representation of the spatial envelope. Int. J. Comput. Vis. 42 (3), 145–175.

Raja, R., Ashok, B., 2021. An efficient ada max based parameter tuned deep neural network for medical data classification. Ann. Roman. Soc. Cell Biol. 1946–1968.

Rajiv Mehrotra, N.R., 1992. Gabor filter-based edge detection. Pattern Recognit. 25 (12), 1479–1494.

Ren, S., He, K., Girshick, R., Sun, J., 2017. Faster R-CNN: Towards real-time object detection with region proposal networks. IEEE Trans. Pattern Anal. Mach. Intell. 39 (06), 1137–1149.

Shi, N., Li, D., 2021. Rmsprop converges with proper hyperparameter. In: International Conference on Learning Representation.

Stanley, K.O., Clune, J., Lehman, J., Miikkulainen, R., 2019. Designing neural networks through neuroevolution. Nat. Mach. Intell. 1 (1), 24–35.

Swain, M.J., Ballard, D.H., 1991. Color indexing. Int. J. Comput. Vis. 7 (1), 11–32.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A., 2015. Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1–9.

Talpur, N., Abdulkadir, S.J., Alhussian, H., Aziz, N., Bamhdi, A., et al., 2022. A comprehensive review of deep neuro-fuzzy system architectures and their optimization methods. Neural Comput. Appl. 1837–1875.

Wang, R., Liu, L., Xie, C., Yang, P., Li, R., Zhou, M., 2021. AgriPest: A large-scale domain-specific benchmark dataset for practical agricultural pest detection in the wild. Sensors 21 (5), 1601.

Wei, D., Chen, J., Luo, T., Long, T., Wang, H., 2022. Classification of crop pests based on multi-scale feature fusion. Comput. Electron. Agric. 194, 106736.
Wong, J.C., Gupta, A., Ong, Y.-S., 2021. Can transfer neuroevolution tractably solve your differential equations? IEEE Comput. Intell. Mag. 16 (2), 14–30.
Wu, Y., Chen, Y., Yuan, L., Liu, Z., Wang, L., Li, H., Fu, Y., 2020. Rethinking classification and localization for object detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10186–10195.
Wu, X., Zhan, C., Lai, Y.-K., Cheng, M.-M., Yang, J., 2019. IP102: A large-scale benchmark dataset for insect pest recognition. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition. CVPR, pp. 8779–8788.
Ye, Y., Xiong, S., Dong, C., Chen, Z., 2022. The structural weight design method based on the modified grasshopper optimization algorithm. Multimedia Tools Appl. 1–29.
Yu, H., Liu, J., Chen, C., Heidari, A.A., Zhang, Q., Chen, H., 2022a. Optimized deep residual network system for diagnosing tomato pests. Comput. Electron. Agric. 195, 106805.
Yu, X., Zhao, Y., Gao, Y., 2022b. SPARE: Self-supervised part erasing for ultra-fine-grained visual categorization. Pattern Recognit. 128, 108691.
Yu, X., Zhao, Y., Gao, Y., Xiong, S., 2021a. Maskcov: A random mask covariance network for ultra-fine-grained visual categorization. Pattern Recognit. 119, 108067.
Yu, X., Zhao, Y., Gao, Y., Xiong, S., Yuan, X., 2020. Patchy image structure classification using multi-orientation region transform. In: Proceedings of the AAAI Conference on Artificial Intelligence. pp. 12741–12748.
Yu, X., Zhao, Y., Gao, Y., Yuan, X., Xiong, S., 2021b. Benchmark platform for ultra-fine-grained visual categorization beyond human performance. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 10285–10295.
Zeiler, M.D., 2012. Adadelta: an adaptive learning rate method. ArXiv preprint arXiv: 1212.5701.
Zhang, H., Chang, H., Ma, B., Wang, N., Chen, X., 2020. Dynamic R-CNN: Towards high quality object detection via dynamic training. In: European Conference on Computer Vision. Springer, pp. 260–275.
Zhang, H., Hao, K., Gao, L., Wei, B., Tang, X., 2022. Optimizing deep neural networks through neuroevolution with stochastic gradient descent. IEEE Trans. Cogn. Dev. Syst.