



## Article

# Robustness Enhancement of Neural Networks via Architecture Search with Multi-Objective Evolutionary Optimization

Haojie Chen <sup>1,2</sup>, Hai Huang <sup>1,\*</sup>, Xingquan Zuo <sup>1,2</sup>  and Xinchao Zhao <sup>3</sup> 

<sup>1</sup> School of Computer Science, Beijing University of Posts and Telecommunications, Beijing 100876, China; haojie\_chen@vip.henu.edu.cn (H.C.); zuoxq@bupt.edu.cn (X.Z.)

<sup>2</sup> Key Laboratory of Trustworthy Distributed Computing and Service, Ministry of Education, Beijing 100876, China

<sup>3</sup> School of Science, Beijing University of Posts and Telecommunications, Beijing 100876, China; zhaoxc@bupt.edu.cn

\* Correspondence: hh Huang@bupt.edu.cn

**Abstract:** Along with the wide use of deep learning technology, its security issues have drawn much attention over the years. Adversarial examples expose the inherent vulnerability of deep learning models and make it a challenging task to improve their robustness. Model robustness is related not only to its parameters but also to its architecture. This paper proposes a novel robustness enhanced approach for neural networks based on a neural architecture search. First, we randomly sample multiple neural networks to construct the initial population. Second, we utilize the individual networks in the population to fit and update the surrogate models. Third, the population of neural networks is evolved through a multi-objective evolutionary algorithm, where the surrogate models accelerate the performance evaluation of networks. Finally, the second and third steps are performed alternately until a network architecture with high accuracy and robustness is achieved. Experimental results show that the proposed method outperforms some classical artificially designed neural networks and other architecture search algorithms in terms of robustness.

**Keywords:** neural architecture search; surrogate model; CLEVER score; adversarial defense

**MSC:** 68T07



**Citation:** Chen, H.; Huang, H.; Zuo, X.; Zhao, X. Robustness Enhancement of Neural Networks via Architecture Search with Multi-Objective Evolutionary Optimization. *Mathematics* **2022**, *10*, 2724. <https://doi.org/10.3390/math10152724>

Academic Editor: Pedro A. Castillo Valdivieso

Received: 30 June 2022

Accepted: 28 July 2022

Published: 2 August 2022

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Copyright:** © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

The rise of deep learning has brought forth a revolution in many fields as well as some security risk because of its uncertainty. Adversarial examples expose the vulnerability of deep learning models [1]. These samples are constructed by adding noise that humans cannot perceive into the original data, resulting in misclassification of a given input for the deep learning models. This poses a big risk to the applications of neural networks. For example, by adding a small perturbation to a “stop” road sign, the models may misjudge it as a “60 km/h” sign, which will cause serious traffic accidents [2]. Therefore, determining how to improve the adversarial robustness of deep learning neural networks is worthy of further study.

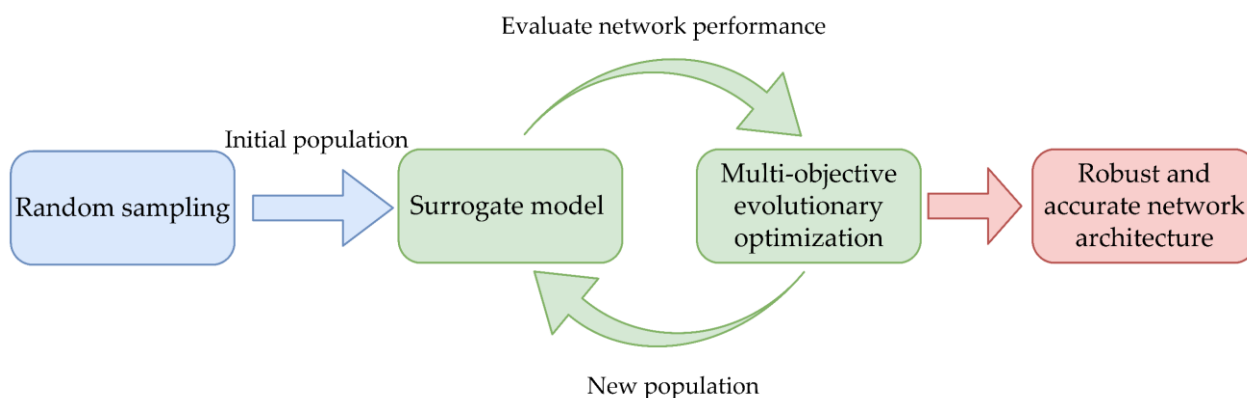
Existing adversarial defense methods mostly focus on model parameters of neural networks. For example, adversarial training methods [3–5] add adversarial samples into the training data to retrain the models to enhance their robustness. However, these methods will lead to a decline in model accuracy. Some researchers optimize, conceal, or confuse the gradient of models [6,7] to make it less easy for attackers to use gradient information to construct adversarial samples. However, this type of method can only defend against gradient-targeted attack methods.

The architecture of a neural network is a vital factor that affects its robustness [8]. Some neural architecture search approaches have been proposed to build a robust model.

In [9,10], evolutionary algorithms are used to search for a neural network with both high accuracy and high robustness. However, the robustness of a neural network is evaluated by adversarial samples generated by specific attack approaches, such that the optimized network may not resist other adversarial attacks, especially new attack approaches. Some methods [11–13], based on the Differentiable ARchiTecture Search (DARTS) [14] algorithm, assess a model's robustness using attack-independent metrics so that the optimized neural network can defend against different adversarial attack approaches. Nevertheless, in those methods, the two metrics of accuracy and robustness are aggregated into a single optimization objective by the linear-weighted method, which makes it difficult to balance the two metrics.

In this paper, we propose a novel Robustness Enhanced method of neural networks based on Architecture Search with multi-objective evOLutionary optimizationN (REASON). Figure 1 describes the overall process of REASON. First, multiple neural networks are randomly sampled to construct the initial population. Second, the surrogate models are fit or updated through the individual networks in the population. Third, a multi-objective evolutionary optimization algorithm [15–18] is implemented to evolve the population of neural networks. The surrogate models are used to accelerate the evaluation process of neural network architectures. The second and third steps are performed alternately until an optimized network architecture with high accuracy and robustness is found. The main contributions of this paper are as follows:

- (1) We propose a novel robustness-enhanced method of neural networks based on architecture search with multi-objective optimization on the robustness and accuracy.
- (2) We analyze the effectiveness of different surrogate models and select the best one to accelerate the performance evaluation of networks in the architecture search algorithm.
- (3) We utilize the CLEVER (Cross Lipschitz Extreme Value for nEtwork Robustness) [19] score, which is an attack-independent metric, to evaluate the network robustness, so that the optimized neural network can defend against various adversarial attack approaches.
- (4) We conduct extensive experiments on real-world datasets to evaluate the effectiveness of REASON.



**Figure 1.** The overall process of the proposed method.

## 2. Related Work

We review the literature on robustness-enhanced methods of neural networks from two aspects that are model-parameter related and model-architecture related.

The methods on model parameters refer to improving network robustness by optimizing model parameters. For example, Goodfellow et al. [3] proposed that the neural network is vulnerable to adversarial sample attacks due to its linear characteristics and proposed to use the adversarial samples generated by the gradient-based attack method for model training to improve the robustness of the model. Ross et al. [7] proposed a gradient regularization training method, which hides the gradient by penalizing the variation in the model output relative to the input and can make the model resistant to transfer attacks. Since

such methods need to calculate the model gradient multiple times, the computing time for model training will increase exponentially and the accuracy of models will decrease.

The methods on model architecture refer to improving the robustness in the model by designing the model architecture. Using neural architecture search technology to construct a robust neural network has become an emerging direction in recent years.

Among the existing studies, one type of approach is to use adversarial training during the search process to increase the robustness in the model. For example, some studies [8,20] adopted one-shot NAS technology. When training the supernet, adversarial training is used to enhance the robustness in the model. The study of [21] is based on the DARTS algorithm, where for lower-level optimization, adversarial training is used to enhance the robustness in the model, and for upper-level optimization, model accuracy and model computational efficiency are taken as the two goals of the multi-objective optimization algorithm. Due to the use of adversarial training, such methods result in an exponential increase in the computational complexity of the algorithm, and the searched network is only robust to some specific attack methods.

Another class of approaches focuses on the search space, which enhances the robustness in the model by enlarging the search space or introducing specific structures in the search space. For example, the authors of [9] believe that the design of existing search spaces is derived from good artificial networks, which limits the ability of search algorithms to discover architectures beyond existing knowledge. It divides the population into hierarchical populations, block-level populations, and model-level populations to ensure a larger search space. The study of [22] introduced denoising blocks, weight-free operations, and Gabor filters in the search space to improve the robustness in the model. Such methods require strong expert experience and are difficult to design.

Some scholars believe that the robustness in the model should not be implicitly improved by adversarial training during the search process but should explicitly define robustness metrics and directly optimize these metrics to obtain robust architectures. For example, the study of [11] explored the relationship among robustness, architecture parameters and Lipschitz constant, and found that an appropriate constraint on architecture parameters could reduce the Lipschitz constant to further improve the robustness. The study of [12] mapped the robustness evaluation of the model to the smoothness of its input loss landscape and used the DARTS algorithm to search. The study of [13] used two differentiable metrics, the certified lower bound and the Jacobian norm bound, to evaluate the model robustness and searched with the DARTS algorithm. Such methods combine the accuracy and robustness in a weighted manner and it is difficult to weigh the relationship between these two metrics.

In summary, the research on robust architecture search is still in its infancy. There are some disadvantages: (1) in the method of using adversarial training in the search process, huge computing power is required and the optimized model is only robust to some specific attacks and has poor scalability; (2) the method of searching by increasing the search space requires strong expert design experience; (3) the methods of optimizing the robustness metrics all adopt the method of transforming multiple metrics into a single optimization objective in a weighted manner and it is difficult to trade off the relationship of multiple metrics.

### 3. Framework of Robustness Enhancement in Neural Networks

In this section, we elaborate on the robustness enhancement of networks based on neural architecture search techniques. Generally, there are three major challenges for this issue, namely multi-objective search trade off, network evaluation cost and robustness evaluation method. First, we use the algorithm of NSGA-II (Elitist Non-Dominated Sorting Genetic Algorithm) [23] to trade off accuracy and robustness in the multi-objective search. Second, we train a surrogate model to reduce the cost of network evaluations. Third, we take the CLEVER score as the robustness evaluation method, which is an attack-independent

robustness metric. Finally, we propose a novel robust neural architecture search algorithm based on the above ideas.

### 3.1. Multi-Objective Search Trade off

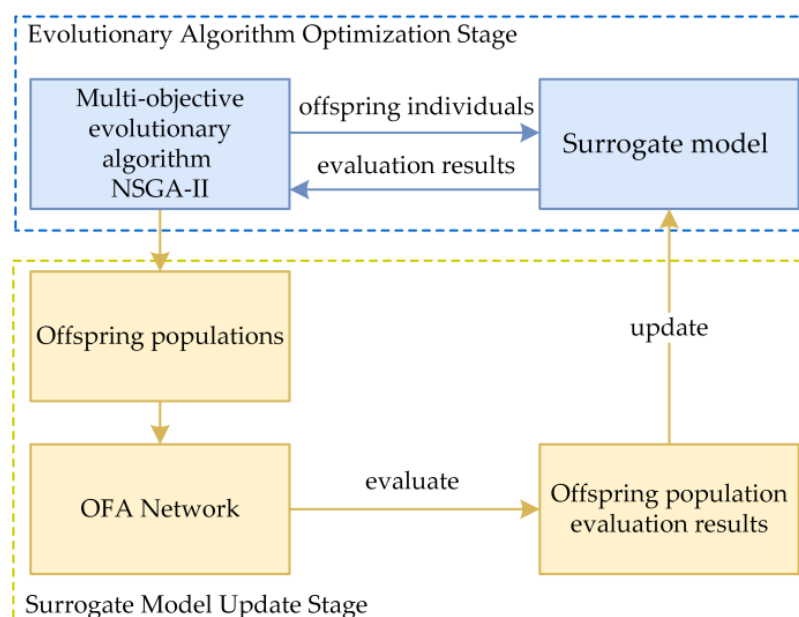
Two optimization objectives, accuracy and robustness, need to be considered in the neural architecture search algorithm. The current multi-objective optimization approaches can be categorized into two types. One is the scalar method [24–27], which transforms the multi-objective problem into a single-objective one using a linear weighted method. It requires multiple experimental attempts to predefine the weights of different objectives. The other is the population-based method [28,29], which fits the Pareto front of multiple objectives to help designers make tradeoffs within the set of solutions or works with aggregation approaches to produce single solutions.

In this paper, the two objectives of accuracy and robustness cannot be simply transformed into a single-objective one using the scalar method since they have different dimensions. We use the population-based NSGA-II algorithm, which equally considers the two optimization objectives to obtain the Pareto frontier solutions. The NSGA-II is a fast and elitist multi-objective genetic algorithm. It presents a selection operator that creates a mating pool by combining the parent and offspring populations and selecting the best solutions. Compared with other multi-objective optimization algorithms, the NSGA-II can reach a good compromise between the optimization performance and the search efficiency and can be easily implemented in our experiments.

### 3.2. Network Evaluation Cost

In the neural architecture search process, huge computing resources and time are consumed to train a new network and evaluate the performance of its architecture. In this paper, we take the advantage of surrogate models to accelerate the evaluation of neural networks without training. A surrogate model is a simple analytical method that mimics the input and output behavior of complex network evaluation systems. It consists of an offline surrogate and an online one. The offline surrogate refers to training the surrogate model before it is actually used in the search process. This method requires a large amount of training data to be prepared in advance, while it is difficult to guarantee the evaluation effect of the surrogate model. Different from the offline surrogate, the online surrogate model is directly utilized in the search algorithm to reduce the evaluation cost and it is gradually optimized by the search results at the same time. The two processes are performed alternately to achieve co-evolution. Figure 2 depicts the online surrogate process implemented in this paper. It is divided into two stages: the evolutionary algorithm optimization and the surrogate model update.

In the stage of evolutionary algorithm optimization, NSGA-II first generates a large number of offspring individuals by using the initial surrogate models to evaluate each neural network. Then, it performs operations, such as crossover, inheritance, mutation, and replication, on these individuals to construct an offspring population. In the stage of surrogate model update, the generated offspring population helps to update and fit the initial surrogate models so that they can assess the performance of networks more accurately in the first stage. Specifically, we employ the Once For All (OFA) [30] network to specialize the individual in offspring populations, which means the parameters in each descendant network architecture can be directly inherited from the OFA network without training. After that, we evaluate the accuracy and robustness in each specialized neural network and use them to update the surrogate models. Compared to offline surrogate, online surrogate requires significantly fewer training data. In this paper, we design experiments to test many surrogate models, such as multilayer perceptron (MLP), Gaussian processes (GP), classification and regression tree (CART), and analyzed their performance to select the best one in our search framework.



**Figure 2.** Online surrogate process.

### 3.3. Robustness Evaluation Method

As mentioned in the last section, when updating the online surrogate models, the accuracy and robustness in each neural network need to be calculated. For the robustness evaluation, the defense success rate of the neural network under some attacks can be viewed as the evaluation indicator. However, this method only works with some specific attacks, not for all attacks. In this paper, we adopt the CLEVER score to evaluate the robustness in neural networks, which is independent of any attack approach. The CLEVER score can estimate the lower bound of the neural network perturbation through the local Lipschitz constant to ensure that the model can defend against any attack when the perturbation size is smaller than this score. Compared with other metrics, such as CROWN [31] and Fast-Lin/Fast-Lip [32], this indicator has the advantages of fast calculation and strong generality and, therefore, is suitable for the robustness evaluation of neural networks.

### 3.4. Robust Architecture Search Algorithm

Based on the above ideas, we propose a novel robust neural architecture search algorithm. As shown in Algorithm 1, from lines 2 to 9,  $N$  neural networks are randomly selected and added into the initial population  $A$ . For each individual network architecture, the network weights are inherited from the OFA network. The network accuracy is obtained by an accuracy function that calculates the accuracy based on a test data set. The network robustness is evaluated by the CLEVER score function. Then, the  $K$ -iteration process of network architecture search starts, shown in lines 10 to 21. In each iteration, neural network individuals in population  $A$  are firstly used to fit two surrogate predictors of network accuracy and robustness, shown in lines 11 to 12. Next,  $n$  offspring are selected by implementing the NSGA-II algorithm to construct subpopulation  $\tilde{A}$ , where the two surrogate predictors of accuracy and robustness help to accelerate the evaluation of new neural networks without training, shown in line 13. Then, the network weights, accuracy and robustness in the offspring are calculated by the OFA network, accuracy function and CLEVER function, respectively, and they are added into  $A$ , shown in lines 14 to 19. In the next iteration, based on the new population  $A$ , the surrogate models are updated and a better subpopulation will be generated. Finally, when the iterative multi-objective search process is done, the non-dominated sorting result of population  $A$  is returned, shown in line 22.

**Algorithm 1.** Robust architecture search algorithm

**Given:**  $N$ : the number of random samples;  $K$ : the number of iterations of multi-objective search;  $A$ : the population set with empty initial value;  $\tilde{A}$ : the subpopulation generated in each iteration of multi-objective search, with a size of  $n$ ;  $SS$ : the search space of neural network architecture;  $\alpha$ : the individual network architecture;  $S_w$ : the OFA network;  $acc(\alpha, w)$ : the function for evaluating the accuracy of network  $\alpha$  with weights  $w$ ;  $clever(\alpha, w)$ : the function for evaluating the robustness in network  $\alpha$  with weights  $w$ ;  $S_{acc}$ : the surrogate predictor of accuracy;  $S_{rob}$ : the surrogate predictor of robustness;  $NSGA-II(A, S_{acc}, S_{rob})$ : the multi-objective search algorithm for generating offspring based on the population set  $A$  through two surrogate predictor of  $S_{acc}$  and  $S_{rob}$ .

**Output:** Pareto solutions of robust architecture search

```

1.  $i \leftarrow 0, j \leftarrow 0$ 
2. while  $i < N$  do
3.     randomly sample individual network  $\alpha_i$  from  $SS$ 
4.     get weights  $w_i$  of  $\alpha_i$  by inheriting from  $S_w$ 
5.      $acc_i = acc(\alpha_i, w_i)$ 
6.      $rob_i = clever(\alpha_i, w_i)$ 
7.      $A = A \cup (\alpha_i, acc_i, rob_i)$ 
8.  $i \leftarrow i + 1$ 
9. end while
10. while  $j < K$  do
11.     fit  $S_{acc}$  based on  $A$ 
12.     fit  $S_{rob}$  based on  $A$ 
13.      $\tilde{A} = NSGA-II(A, S_{acc}, S_{rob}), \text{ size}(\tilde{A}) = n$ 
14.     for each  $\alpha$  in  $\tilde{A}$  do
15.         get weights  $w_\alpha$  of  $\alpha$  by inheriting from  $S_w$ 
16.          $acc_\alpha = acc(\alpha, w_\alpha)$ 
17.          $rob_\alpha = clever(\alpha, w_\alpha)$ 
18.          $A = A \cup (\alpha_\alpha, acc_\alpha, rob_\alpha)$ 
19.     end for
20.  $j \leftarrow j + 1$ 
21. end while
22. return Non-Dominated-Sort( $A$ )

```

**4. Experiments**

In this section, we first introduce the experimental parameter settings and real-world datasets. Next, we compare the performance of different surrogate models to select the best one for the robust architecture search. Then, we conduct experiments to verify the effectiveness of CLEVER score in evaluating the robustness in neural networks. Finally, we analyze the results of the proposed REASON method.

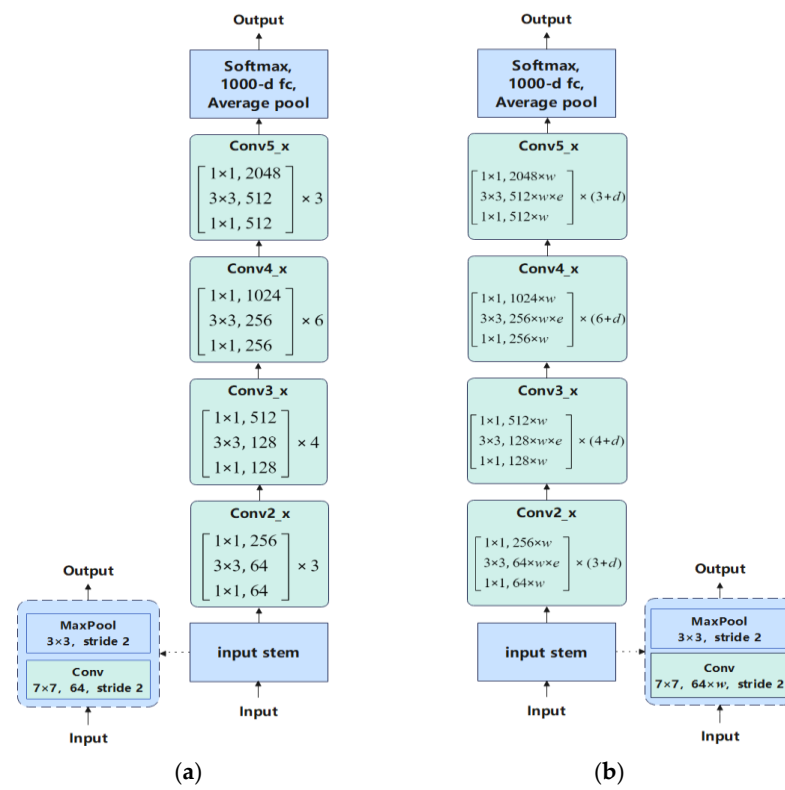
**4.1. Experimental Parameter Settings**

The search space of neural network architecture is designed based on the Resnet50 [33] structure. There are three types of variable parameters in the search space, including the input resolution, the width (the number of channels at the bottleneck) and the depth (the number of bottlenecks).

Figure 3 depicts the comparison of the Resnet50 and the search space. For the network structure of the search space, parameters in the blue part are consistent with Resnet50, while parameters in the cyan part need to be determined by the architecture search algorithm. Specifically, a network consists of multiple stacked bottlenecks. A bottleneck includes three convolution layers: a  $1 \times 1$  convolutional layer for channel reduction, a  $3 \times 3$  convolutional layer for feature extraction and a  $1 \times 1$  convolutional layer for channel expansion. The number of channels for each bottleneck is controlled by the parameters of  $w$  and  $e$ . The  $w$  represents the width multiplier, controlling the number of input and output channels of the bottleneck, with a value among  $\{0.65, 0.8, 1.0\}$ . The  $e$  represents the expand ratio, determining the number of middle channels in the bottleneck, with a value among  $\{0.2, 0.25, 0.35\}$ . The



number of bottlenecks is controlled by the parameter  $d$ , with a value range of  $\{0,1,2\}$ . Additionally, the input image size (input resolution) varies from 128 to 224.



**Figure 3.** Comparison of Resnet50 and the search space: (a) Resnet50 structure; (b) Network structure of the search space.

Except for the search space  $SS$  in Algorithm 1, other parameters in this algorithm are tuned iteratively by a grid search strategy until they have made a tradeoff between the optimization performance and the computational efficiency. Specifically, the number of random samples  $N$  is set to 150. The number of iterations of multi-objective search  $K$  is set to 60. The size of the subpopulation generated in each iteration is set to 15. The public dataset ImageNet [34] is used in all the experiments.

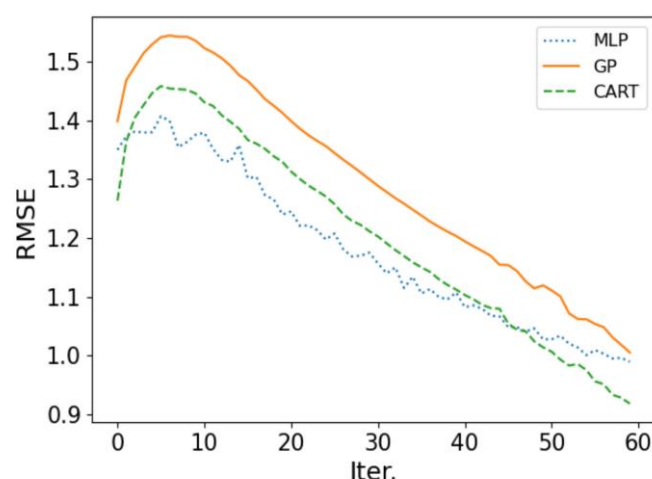
#### 4.2. Surrogate Model Performance Analysis

In this subsection, we compare the performance of three surrogate models of MLP, GP and CART so as to select the best one in the robust architecture search algorithm.

##### 4.2.1. Comparison of Root Mean Square Error

In Algorithm 1, the offspring networks are generated in each search iteration and used to fit the surrogate model. In the same iteration, we firstly adopt the updated surrogate model to predict the accuracy or robustness in the offspring network. Then, we calculate the true accuracy or robustness in them on the validation dataset. Finally, we compare the performance of three surrogate models by calculating the root mean square error between the predicted value and the true value of each model. The smaller the mean square error is, the better the surrogate model will be.

As shown in Figure 4, the RMSE of three surrogate models firstly increases and then gradually decreases. The reason is that the number of offspring networks is too small to fit the surrogate model well in the initial phase. With an increase in the number of iterations, more offspring data are generated to update the surrogate model, leading to a rapid decrease in RMSE. Although the RMSE of MLP is better than others, the predictive effect of three surrogate models is not very different.



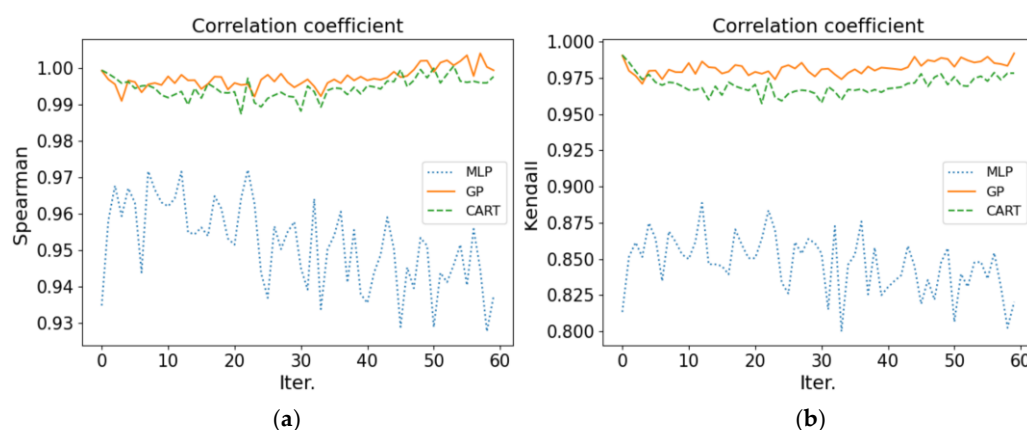
**Figure 4.** Root mean square error comparison of surrogate models.

#### 4.2.2. Comparison of Correlation Coefficient

The NSGA-II algorithm pays more attention to the “good or bad relationship” between offspring networks rather than how accurately the surrogate models predict, since it needs to find the best individual through this relationship. Therefore, we can judge whether prediction results of the surrogate model are closely related to the ranking relationship among offspring individuals by calculating the rank correlation coefficient.

Kendall’s rank correlation coefficient [35] and Spearman’s rank correlation coefficient [36] were used for analysis. Those two methods measure the degree of similarity between two rankings and can evaluate the significance of the relation between them. Specifically, for the ranking set of  $X$  and  $Y$ , if there are more pairs  $(i, j)$  that satisfy  $x_i > x_j$  and  $y_i > y_j$  or  $x_i < x_j$  and  $y_i < y_j$ , the rank correlation coefficient between  $X$  and  $Y$  will be greater, and vice versa. In our experiment,  $X$  is the prediction set of surrogate models and  $Y$  is the true value set of performance metrics for offspring networks. The coefficient is inside the interval  $[-1, 1]$ , where 1 means positive correlation,  $-1$  means negative correlation and 0 means no correlation.

As shown in Figure 5, the values of two types of rank correlation coefficients for three models are all above 0.8. It indicates that the ranking relationship between prediction values and the true results is closely related. We can utilize the estimation of surrogate models to effectively represent the ordinal association among offspring network performance. Moreover, we can see that the correlation coefficients of GP and CART keep stable with the increase in the number of iterations and is significantly better than that of MLP. The main reason is that MLP is much more complex and needs more training samples to fit it.



**Figure 5.** Correlation coefficient comparison of surrogate models: (a) Spearman’s rank correlation coefficient; (b) Kendall’s rank correlation coefficient.



In summary, the root mean square errors of the three surrogate models are not very different, while the correlation coefficients of the GP and CART outperform the MLP and the GP is the best. Therefore, we choose the GP model as the surrogate model in the subsequent experiments.

#### 4.3. Robustness Evaluation Effectiveness Analysis

In this subsection, we conduct experiments to verify whether the CLEVER score is effective in evaluating the robustness in neural networks.

##### 4.3.1. Randomness Analysis

The CLEVER score estimates the lower bound of the robustness in the model through a random sampling method, which has a certain randomness and is not a strict boundary. As mentioned above, NSGA-II focuses on the ranking relationship between the robustness in individual networks. Given the neural networks, we can calculate the CLEVER score for them multiple times. If the ranking relationship between the CLEVER scores is fixed in most test time, it indicates that the randomness of the CLEVER score has no influence on the evaluation of the robustness in neural networks.

In the experiment, three different neural networks are selected and their CLEVER scores are calculated 100 times, respectively. Each time, 100 sample points are selected and the batch size of each sample point is set to 500. Each batch of 64 samples is sampled and the perturbation radius is set to 5 with the perturbation norm of 2.

Figure 6 shows the experimental results. The CLEVER score of model 1 is the smallest and the CLEVER score of model 3 is the largest. The ranking relationship among the CLEVER scores of three models remains stable in 90 of the total 100 running times, which proves the randomness of the CLEVER score has little effect on the model robustness evaluation.

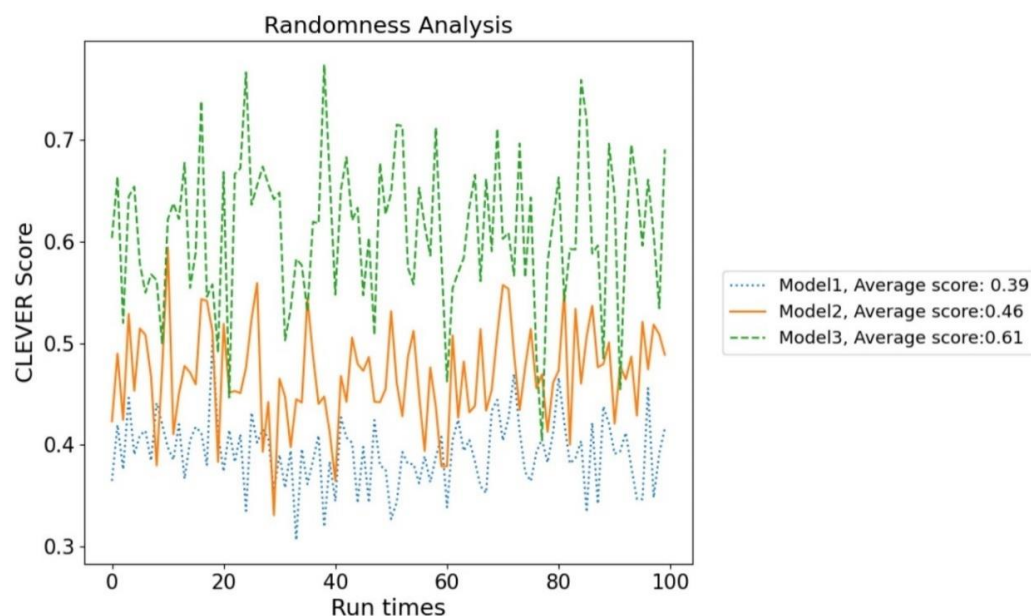


Figure 6. CLEVER score randomness analysis.

##### 4.3.2. Coefficient Analysis

To further demonstrate the effectiveness of the CLEVER method, we assess the correlation of two rankings, the CLEVER score and the defense success rate of attacked models. Specifically, highly robust neural networks should have large CLEVER scores as well as high defense success rates against various adversarial attacks. We select the FGSM [2] (Fast Gradient Sign Method) as the adversarial attack against neural network models. The defense success rate of the model is defined as  $m/n * 100\%$ , where  $n$  is the number of

adversarial samples generated by FGSM and  $m$  is the number of samples successfully defended by the model. In this experiment, we firstly sample 100 subnets from the OFA network and obtain their CLEVER scores. Then, we estimate their defense success rate under FGSM attack on the validation dataset. Finally, we assess the significance of the relation between the CLEVER score and the defense success rate rankings by calculating Spearman's and Kendall's rank correlation coefficient.

The experimental parameters are set as follows. For the CLEVER score, 100 sample points are selected and the batch size of each sample point is set to 500. Each batch of 64 samples is sampled and the perturbation radius is set to 5, with a perturbation norm of 2. The average value of five CLEVER scores for each model is taken to eliminate the influence of random sampling on the results. For the defense success rate, the perturbation size of the FGSM attack is set to 4/255 and the perturbation norm is set to 2, with the number of generated adversarial samples as 5000.

As shown in Table 1, the Spearman's rank correlation coefficient is 0.9003 and Kendall's is 0.7208, which demonstrates the ranking association between CLEVER score and the defense success rate is strongly related. The CLEVER score can effectively evaluate the robustness in neural networks.

**Table 1.** Correlation coefficient between CLEVER score and defense success rate.

| Spearman's Rank Correlation Coefficient | Kendall's Rank Correlation Coefficient |
|---|--|
| 0.9003                                  | 0.7208                                 |

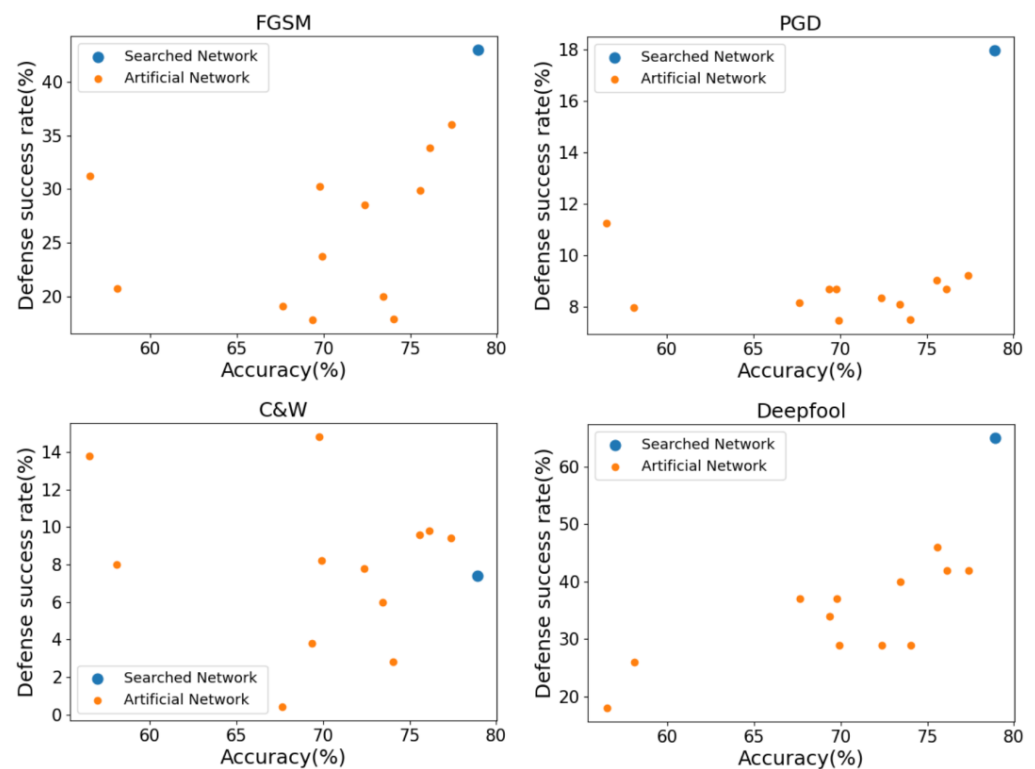
#### 4.4. Architecture Search Results Analysis

In this section, we verify the effectiveness of REASON through three comparison experiments. First, we compare the search results of REASON with some classical artificially designed neural networks. Second, we demonstrate the performance advantages of REASON compared with other robust enhance methods based on architecture search. Finally, we prove the networks searched by REASON have better robustness to various attack methods.

##### 4.4.1. Comparison with Artificially Designed Networks

We compare the robustness in networks found by REASON with that of several well-known artificially designed networks, including AlexNet [37], VGGNet [38], ResNet [33], SqueezeNet [39], DenseNet [40], GoogLeNet [41], ShuffleNetV2 [42], MobileNetV3 [27] and MnasNet [25]. The robustness evaluation metric is the defense success rates of each network under various adversarial attacks. The selected attack methods include FGSM, PGD (Projected Gradient Descent) [5], C&W (Carlini and Wagner) [43] and Deepfool [44]. For FGSM, set the perturbation size to 2/255 with the perturbation norm as infinite. For PGD, set the perturbation size to 4/255, the step size to 2/255, the number of iterations to 5 and the perturbation norm to infinite. For C&W, set the maximum number of iteration steps to 60. For Deepfool, set the overshoot parameter to 10, the class gradient to 1, and the maximum number of iteration steps to 50. FGSM and PGD select the samples from the entire validation set of the ImageNet dataset due to their fast attack speed. CW and Deepfool randomly select 500 and 100 samples in the validation set, respectively.

Figure 7 shows the experimental results. The small orange dots represent artificially designed networks, while the large blue dots depict networks searched by REASON. The horizontal axis represents the classification accuracy of neural networks under clean samples and the vertical axis describes their defense success rates under FGSM, PGD, C&W and Deepfool, respectively. It can be seen that the accuracy of the searched network is higher than that of all other artificially designed networks. For the defense success rate, the searched network also outperforms others under all attack methods, except for C&W.



**Figure 7.** Comparison with artificially designed networks.

The experimental details are shown in Table 2. The number in bold is the best value for each column. We can see that the performance of the network searched by REASON is much better than all other networks for all attack methods except for C&W. Particularly, for the accuracy under clean samples, the searched network is 22.4% higher than the AlexNet network. For the defense success rate, the searched network is 25.18% higher than the ShuffleNetV2 under FGSM attack, 10.51% higher than the VGG13 under PGD attack and 47% higher than the AlexNet under Deepfool attack, which demonstrates the superiority of REASON.

**Table 2.** Comparison of the network searched by REASON and artificially designed networks.

| Network           | Accuracy      | FGSM          | PGD           | C&W         | Deepfool    |
|-------------------|---------------|---------------|---------------|-------------|-------------|
| Searched Network  | <b>78.908</b> | <b>43.118</b> | <b>18.272</b> | 7.4         | <b>65.0</b> |
| AlexNet           | 56.522        | 31.224        | 11.254        | 13.8        | 18.0        |
| VGG13             | 69.928        | 23.746        | 7.456         | 8.2         | 29.0        |
| VGG19             | 72.376        | 28.528        | 8.33          | 7.8         | 29.0        |
| ResNet50          | 76.13         | 33.856        | 8.67          | 9.8         | 42.0        |
| ResNet101         | 77.374        | 36.016        | 9.198         | 9.4         | 42.0        |
| SqueezeNet        | 58.092        | 20.73         | 7.966         | 8.0         | 26.0        |
| DenseNet169       | 75.6          | 29.902        | 9.026         | 9.6         | 46.0        |
| GoogLeNet         | 69.778        | 30.264        | 8.68          | <b>14.8</b> | 37.0        |
| ShuffleNetV2      | 69.362        | 17.792        | 8.69          | 3.8         | 34.0        |
| MobileNetV3_small | 67.668        | 19.056        | 8.16          | 0.4         | 37.0        |
| MobileNetV3_large | 74.042        | 17.892        | 7.5           | 2.8         | 29.0        |
| MnasNet           | 73.456        | 19.99         | 8.09          | 6.0         | 40.0        |

#### 4.4.2. Comparison with Other Robust Architecture Search Algorithms

We evaluate REASON against the following same types of representative robust architecture search algorithms: (1) RobNet [8], searching targets in the supernet based on adversarial training; (2) SDARTS-ADV and PC-DARTS-ADV [45], finding networks using

DARTS algorithm; and (3) DSRNA-CB and DSRNA-Jacobian [13], performing searches based on differentiable robustness evaluation metrics.

In this experiment, two metrics are used for comparison. One is the accuracy under clean samples. The other is the defense success rate of each network under FGSM and PGD attacks. For FGSM, set the perturbation size to  $2/255$  with the perturbation norm as infinite. For PGD, set the perturbation size to  $2/255$ , the number of iteration steps to 100, the iteration step size to  $2/255$  and the perturbation norm to infinite. According to [13], the searched network is retrained with Jacobian regularization. For a fair comparison, we implement the same Jacobian regularization training on the searched network, with a regularization ratio of 0.001 and an epoch of 5.

Table 3 shows the comparison results of different search algorithms. For the accuracy under clean samples, REASON is slightly lower than other methods except for RobNet-large. DSRNA-Jacobian has the highest accuracy, only 2.8% higher than that of REASON. For the defense success rate under FGSM and PGD attacks, REASON greatly outperforms other algorithms. Specially, REASON is 22.72% higher and 22.79% higher than the RobNet-large under FGSM and PGD attacks, respectively. To sum up, REASON is comparable to other methods in accuracy but much better in robustness.

**Table 3.** Comparison of search algorithms.

| Search Algorithms | Accuracy     | FGSM          | PGD          |
|-------------------|--------------|---------------|--------------|
| REASON            | 73.066       | <b>62.458</b> | <b>59.93</b> |
| RobNet-large      | 61.26        | 39.74         | 37.14        |
| SDARTS-ADV        | 74.85        | 48.09         | 46.54        |
| PC-DARTS-ADV      | 75.73        | 48.25         | 46.59        |
| DSRNA-CB          | 75.84        | 50.89         | 45.39        |
| DSRNA-Jacobian    | <b>75.88</b> | 48.69         | 43.79        |

#### 4.4.3. Comparison with Search Algorithm Using Attack-Dependent Robustness Metric

Our search method adopts the CLEVER score, which is an attack-independent metric to evaluate the network robustness, so that the searched neural network can defend against all types of adversarial attacks. To verify it, we alter Algorithm 1 by replacing the CLEVER score with an attack-dependent robustness metric and compare it with the original REASON method. In this experiment, the attack-dependent robustness metric is the defense success rate under FGSM attack. For the parameters of FGSM, the perturbation size is  $4/255$  with the perturbation norm as infinite. The other parameters are the same as Algorithm 1.

Similarly, we use two metrics to compare the altered search algorithm with REASON, which are the accuracy under clean samples and the defense success rates under FGSM, PGD, C&W and Deepfool attacks. For FGSM, the perturbation size is  $2/255$  with the perturbation norm as infinite. For PGD, set the perturbation size to  $4/255$ , the number of iteration steps to 5, the iteration step size to  $2/255$  and the perturbation norm to infinite. For C&W, set the maximum number of iteration steps to 60, the learning rate to 0.005, the confidence of adversarial examples to 0, the relative importance constant of the perturbation size and classification confidence to 0.0001 and the number of binary search steps to 5. For Deepfool, set the overshoot parameter to 10, the number of calculated gradient categories to 1 and the maximum number of iteration steps to 50.

As shown in Table 4, the first column represents the two search algorithms with different robustness evaluation metrics. We can observe that REASON with CLEVER score is superior to the algorithm with the attack-dependent metric in almost all comparisons. For the defense success rate under FGSM attack, REASON with CLEVER score is 0.2% higher than the algorithm with the FGSM-dependent metric. The experimental results indicate that the networks searched by REASON are more robust to different adversarial attacks than networks searched by using attack-dependent metrics.

**Table 4.** Comparison of search algorithms with different robustness evaluation metrics.

| Metrics          | Accuracy      | FGSM          | PGD           | C&W | Deepfool    |
|------------------|---------------|---------------|---------------|-----|-------------|
| Attack-Dependent | 78.732        | 43.012        | 18.054        | 7.6 | 63.8        |
| CLEVER Score     | <b>78.908</b> | <b>43.118</b> | <b>18.272</b> | 7.6 | <b>65.0</b> |

## 5. Conclusions

In this paper, we proposed a novel architecture search method based on a multi-objective evolutionary algorithm for enhancing the robustness in neural networks. The method uses NSGA-II to balance the optimization objectives of network accuracy and robustness. The online surrogate models are trained to reduce the computational cost of network performance evaluation. The attack-independent metric, CLEVER score, is used to evaluate networks' robustness, which enables the optimized neural network to defend against different adversarial attacks. Experimental results on real-world datasets show that the networks searched by the proposed method have better robustness than several classical human-designed neural networks. Moreover, the proposed method outperforms other neural architecture search algorithms in the robustness in models.

The future work will focus on exploring more effective robustness evaluation metrics and search space, so as to discover a neural network architecture beyond current human knowledge.

**Author Contributions:** Conceptualization, H.C. and X.Z. (Xingquan Zuo); methodology, H.C.; software, H.C.; validation, H.C. and H.H.; formal analysis, H.H.; investigation, H.C.; resources, X.Z. (Xingquan Zuo); writing—original draft preparation, H.C.; writing—review and editing, H.H.; supervision, X.Z. (Xinchao Zhao); project administration, H.H. and X.Z. (Xingquan Zuo) All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Not applicable.

**Acknowledgments:** The authors wish to thank the anonymous reviewers and editors for their valuable comments and suggestions.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Szegedy, C.; Zaremba, W.; Sutskever, I.; Bruna, J.; Erhan, D.; Goodfellow, I.; Fergus, R. Intriguing properties of neural networks. *arXiv* **2013**, arXiv:1312.6199.
2. Eykholt, K.; Evtimov, I.; Fernandes, E.; Li, B.; Rahmati, A.; Xiao, C.; Song, D. Robust physical-world attacks on deep learning visual classification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 1625–1634.
3. Goodfellow, I.J.; Shlens, J.; Szegedy, C. Explaining and harnessing adversarial examples. *arXiv* **2014**, arXiv:1412.6572.
4. Wang, J. Adversarial Examples in Physical World. In Proceedings of the International Joint Conference on Artificial Intelligence, Virtual, 19–26 August 2021; pp. 4925–4926.
5. Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; Vladu, A. Towards deep learning models resistant to adversarial attacks. *arXiv* **2017**, arXiv:1706.06083.
6. Gu, S.; Rigazio, L. Towards deep neural network architectures robust to adversarial examples. *arXiv* **2014**, arXiv:1412.5068.
7. Ross, A.; Doshi-Velez, F. Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. In Proceedings of the AAAI Conference on Artificial Intelligence, New Orleans, LO, USA, 2–7 February 2018; p. 1.
8. Guo, M.; Yang, Y.; Xu, R.; Liu, Z.; Lin, D. When nas meets robustness: In search of robust architectures against adversarial attacks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 631–640.
9. Kotyan, S.; Vargas, D.V. Towards evolving robust neural architectures to defend from adversarial attacks. In Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion, Cancun, Mexico, 8–12 July 2020; pp. 135–136.



10. Geraeinejad, V.; Sinaei, S.; Modarressi, M.; Daneshtalab, M. RoCo-NAS: Robust and Compact Neural Architecture Search. In Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN), Shenzhen, China, 18–22 July 2021; pp. 1–8.
11. Dong, M.; Li, Y.; Wang, Y.; Xu, C. Adversarially robust neural architectures. *arXiv* **2020**, arXiv:2009.00902.
12. Mok, J.; Na, B.; Choe, H.; Yoon, S. AdvRush: Searching for Adversarially Robust Neural Architectures. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, QC, Canada, 11–17 October 2021; pp. 12322–12332.
13. Hosseini, R.; Yang, X.; Xie, P. Dsrna: Differentiable search of robust neural architectures. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 6196–6205.
14. Liu, H.; Simonyan, K.; Yang, Y. DARTS: Differentiable architecture search. *arXiv* **2018**, arXiv:1806.09055.
15. Wang, Z.; Zhang, Q.; Ong, Y.S.; Yao, S.; Liu, H.; Luo, J. Choose appropriate subproblems for collaborative modeling in expensive multiobjective optimization. *IEEE Trans. Cybern.* **2021**. [\[CrossRef\]](#)
16. Leung, M.F.; Wang, J. A collaborative neurodynamic approach to multiobjective optimization. *IEEE Trans. Neural Netw. Learn. Syst.* **2018**, *29*, 5738–5748. [\[CrossRef\]](#) [\[PubMed\]](#)
17. Gunantara, N. A review of multi-objective optimization: Methods and its applications. *Cogent Eng.* **2018**, *5*, 1502242. [\[CrossRef\]](#)
18. Marler, R.T.; Arora, J.S. Survey of multi-objective optimization methods for engineering. *Struct. Multidiscip. Optim.* **2004**, *26*, 369–395. [\[CrossRef\]](#)
19. Weng, T.W.; Zhang, H.; Chen, P.Y.; Yi, J.; Su, D.; Gao, Y.; Daniel, L. Evaluating the robustness of neural networks: An extreme value theory approach. *arXiv* **2018**, arXiv:1801.10578.
20. Xie, G.; Wang, J.; Yu, G.; Zheng, F.; Jin, Y. Tiny adversarial multi-objective oneshot neural architecture search. *arXiv* **2021**, arXiv:2103.00363.
21. Yue, Z.; Lin, B.; Huang, X.; Zhang, Y. Effective, efficient and robust neural architecture search. *arXiv* **2020**, arXiv:2011.09820.
22. Chen, H.; Zhang, B.; Xue, S.; Gong, X.; Liu, H.; Ji, R.; Doermann, D. Anti-bandit neural architecture search for model defense. In *European Conference on Computer Vision*; Springer: Cham, Switzerland, 2020; pp. 70–85.
23. Deb, K.; Pratap, A.; Agarwal, S.; Meyarivan, T.A.M.T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* **2002**, *6*, 182–197. [\[CrossRef\]](#)
24. Cai, H.; Zhu, L.; Han, S. Proxylessnas: Direct neural architecture search on target task and hardware. *arXiv* **2018**, arXiv:1812.00332.
25. Tan, M.; Chen, B.; Pang, R.; Vasudevan, V.; Sandler, M.; Howard, A.; Le, Q.V. Mnasnet: Platform-aware neural architecture search for mobile. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 2820–2828.
26. Wu, B.; Dai, X.; Zhang, P.; Wang, Y.; Sun, F.; Wu, Y.; Keutzer, K. Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 10734–10742.
27. Howard, A.; Sandler, M.; Chu, G.; Chen, L.C.; Chen, B.; Tan, M.; Adam, H. Searching for mobilenetv3. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Long Beach, CA, USA, 15–20 June 2019; pp. 1314–1324.
28. Chu, X.; Zhang, B.; Xu, R. Fairnas: Rethinking evaluation fairness of weight sharing neural architecture search. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Virtual, 11–17 October 2021; pp. 12239–12248.
29. Dong, J.D.; Cheng, A.C.; Juan, D.C.; Wei, W.; Sun, M. Dpp-net: Device-aware progressive search for pareto-optimal neural architectures. In Proceedings of the European Conference on Computer Vision (ECCV), Munchen, Germany, 8–14 September 2018; pp. 517–531.
30. Cai, H.; Gan, C.; Wang, T.; Zhang, Z.; Han, S. Once-for-all: Train one network and specialize it for efficient deployment. *arXiv* **2019**, arXiv:1908.09791.
31. Zhang, H.; Weng, T.W.; Chen, P.Y.; Hsieh, C.J.; Daniel, L. Efficient neural network robustness certification with general activation functions. *Adv. Neural Inf. Process. Syst.* **2018**, *31*, 4939–4948.
32. Weng, L.; Zhang, H.; Chen, H.; Song, Z.; Hsieh, C.J.; Daniel, L.; Dhillon, I. Towards fast computation of certified robustness for relu networks. In Proceedings of the International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; pp. 5276–5285.
33. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26–30 June 2016; pp. 770–778.
34. Available online: <https://image-net.org/> (accessed on 20 May 2021).
35. Kendall, M.G. A new measure of rank correlation. *Biometrika* **1938**, *30*, 81–93. [\[CrossRef\]](#)
36. Sedgwick, P. Spearman's rank correlation coefficient. *BMJ* **2014**, *349*, g7327. [\[CrossRef\]](#)
37. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 1097–1105. [\[CrossRef\]](#)
38. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
39. Iandola, F.N.; Han, S.; Moskewicz, M.W.; Ashraf, K.; Dally, W.J.; Keutzer, K. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5 MB model size. *arXiv* **2016**, arXiv:1602.07360.
40. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4700–4708.



41. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
42. Ma, N.; Zhang, X.; Zheng, H.T.; Sun, J. Shufflenet v2: Practical guidelines for efficient cnn architecture design. In Proceedings of the European Conference on Computer Vision (ECCV), Munchen, Germany, 8–14 September 2018; pp. 116–131.
43. Carlini, N.; Wagner, D. Towards evaluating the robustness of neural networks. In Proceedings of the 2017 IEEE Symposium on Security and Privacy, San Jose, CA, USA, 22–24 May 2017; pp. 39–57.
44. Moosavi-Dezfooli, S.M.; Fawzi, A.; Frossard, P. Deepfool: A simple and accurate method to fool deep neural networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 2574–2582.
45. Chen, X.; Hsieh, C.J. Stabilizing differentiable architecture search via perturbation-based regularization. In Proceedings of the International Conference on Machine Learning, Virtual, 13–18 July 2020; pp. 1554–1565.