Contents lists available at ScienceDirect

Neurocomputing

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

# Adversarial defense via self-orthogonal randomization super-network

Huanyu Bian [a], Dongdong Chen [b,⁎], Kui Zhang [a], Hang Zhou [a], Xiaoyi Dong [a], Wenbo Zhou [a], Weiming Zhang [a,⁎], Nenghai Yu [a]

[a] University of Science and Technology of China, China
[b] Microsoft Cloud AI, United States of America

ARTICLE INFO

*Article history:*
Received 26 November 2020
Revised 7 February 2021
Accepted 17 April 2021
Available online 22 April 2021
Communicated by Zidong Wang

*Keywords:*
Adversarial Defense
Self-Orthogonal Randomization Super-network

ABSTRACT

Deep neural networks are demonstrated to be vulnerable to adversarial examples. In this paper, starting from the robustness analysis about the model ensemble, we propose a novel type of defense method named "Self-Orthogonal Randomization Super-network" (**SORS**). More specifically, we think the main robustness benefit from the model ensemble comes from two aspects: smaller adversarial subspace and gradient orthogonality. However, the naive model ensemble has two fundamental limitations: 1) Though ensembling more models will introduce more robustness, training too many models is infeasible and resource-consuming. 2) Since these models are usually trained independently, the gradient orthogonality among them is often partial and weak. Motivated by this, we propose to train one single super-network that consists of the exponential number of sub-networks, and explicitly constrain the gradient of different sub-networks with respect to the same input to be orthogonal. In the inference stage, at each forward pass, one sub-network will be randomly sampled. Through extensive experiments, we demonstrate that the proposed method can achieve significantly better robustness than the vanilla single model baseline and the naive model ensemble baseline. Moreover, this new type of defense strategy is also complementary to other types of defense methods and achieves state-of-the-art performance.

## 1. Introduction

Deep neural networks have revolutionized many sub-fields of artificial intelligence and achieved significant success, such as image recognition [21,16], neural machine translation [47], and autonomous driving [19]. However, recent research [38,11] has shown that deep neural networks can be easily fooled by adversarial examples. By adding small perturbations to the original images, they may be visually indistinguishable but are able to make the target neural network get totally incorrect recognition results. The existence of adversarial examples can pose severe security threats for applications based on visual understanding, such as autonomous driving [10,3] and face verification [37,12].

Recently, much attention has been drawn to study how to defend adversarial examples. Generally, these defense methods can be roughly categorized into four types: input transform-based methods [7,29,24,20], adversarial training-based methods [11,22,44], gradient mask-based methods [50,35] and model ensemble [4,39]. The motivation of this paper is developed from the naive model ensemble. Below let us first analyze "Why model ensemble can be more robust?" and "What are its limitations?".

Given an input image, model ensemble means getting the final recognition results by considering multiple different models with some naive or advanced voting strategies. In the analysis below, we think the main robustness benefit from the model ensemble comes from two aspects: *smaller adversarial subspace* and *gradient orthogonality*. More specifically, as indicated in precedent works [31,40,49], adversarial examples may exist in some continuous subspace and adversarial attack is essentially finding this adversarial subspace. For the model ensemble case, adversarial attack has to find the overlapped adversarial subspace of all the models involved. Therefore, as shown in Fig. 1, the shared adversarial subspace becomes much smaller if more models are ensembled. On the other hand, we find the gradients from different models are often partially orthogonal [28]. As shown in Fig. 2, when using the gradient-based methods like FGSM or I-FGSM, they have to iterate more steps to find a suitable adversarial example with such partial orthogonal gradients.

To further verify the orthogonality observation, we conduct a simple analysis experiment to reveal the relation between the attack transferability and gradient orthogonality in Table 1. The

⁎ Corresponding author.
*E-mail addresses:* hybian@mail.ustc.edu.cn (H. Bian), cddlyf@gmail.com (D. Chen), zk19@mail.ustc.edu.cn (K. Zhang), zh2991@mail.ustc.edu.cn (H. Zhou), light@mail.ustc.edu.cn (X. Dong), welbeckz@ustc.edu.cn (W. Zhou), zhangwm@ustc.edu.cn (W. Zhang), ynh@ustc.edu.cn (N. Yu).

https://doi.org/10.1016/j.neucom.2021.04.062
0925-2312/© 2021 Elsevier B.V. All rights reserved.

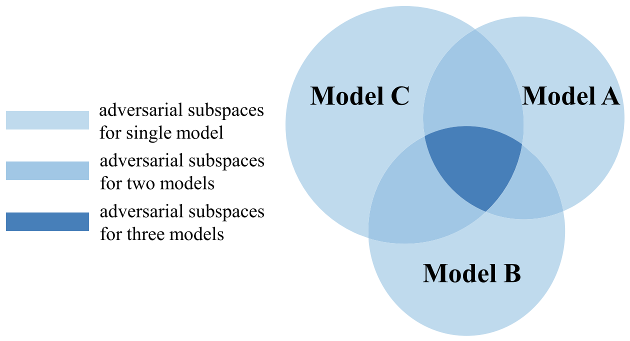

adversarial subspaces for single model
adversarial subspaces for two models
adversarial subspaces for three models
Model C
Model A
Model B

**Fig. 1.** A schematic diagram of the adversarial subspace of ensemble-based model. The shared adversarial subspace is smaller when multiple models are ensembled.

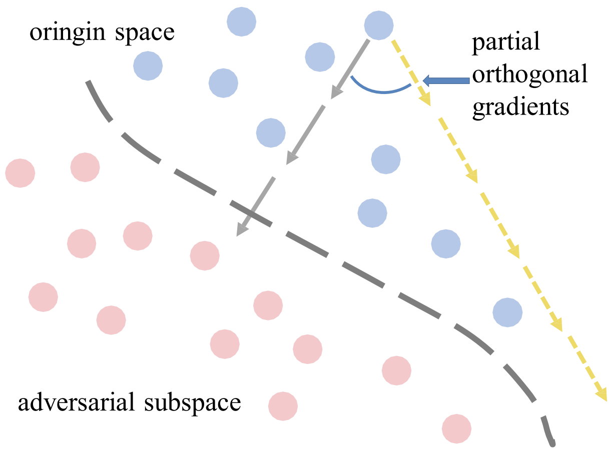


**Fig. 2.** Illustration of adversarial attacks that using partial orthogonal gradients are harder than the original gradients.

gradient orthogonality refers to that given an input image, the gradients of different models with respect to it are partial orthogonal. Specifically, we calculate the gradient orthogonality between different models and ResNet-152, and utilize the adversarial examples generated by ResNet-152 to attack these models. The results confirm that a higher gradient orthogonality often leads to a lower attack success rate.

Despite its robustness, naive model ensemble still has two limitations: 1) Training too many models is resource-consuming, so the number of models that can be ensembled is often limited. 2) Since these models are usually trained independently, it is hard to explicitly control the gradient orthogonality among them, thus making the final gradient orthogonality relatively partial and weak.

To inherit the profitable properties of model ensemble while circumventing the above limitations, this paper proposes a novel scheme called “Self-Orthogonal Randomization Super-network” (**SORS**). It aims to train one big super-network that consists of an exponential number of sub-networks and explicitly restrict them to being gradient-orthogonal. Specifically, we equip each layer of **SORS** with $m$ parallel operation blocks. At each forward pass in the training and testing stages, we randomly sample one operation block for each layer and connect them to form a specific forward path, which can be regarded as a sub-network. Assuming the super-network has a total of $n$ layers, it can theoretically form $m^n$ possible sub-networks. During the training stage, these sub-networks will be sampled uniformly, and an explicit loss term is used to constrain their gradient orthogonality.

To demonstrate the effectiveness, extensive experiments are conducted with respect to different datasets and attack methods. It shows that the proposed method is capable of obtaining better robustness than both the vanilla single model baseline and model ensemble baseline. Besides, we show this self-robust strategy is also complementary to other types of defense methods and achieves state-of-the-art performance.

To sum up, our main contributions are threefold as below:

- We have provided detailed analysis about the robustness of ensemble based models and their limitations, which may inspire researchers in this field.
- Motivated by the strengths and weaknesses of vanilla model ensemble, we propose a novel type of defense method “Self-Orthogonal Randomization Super-networks”. It theoretically consists of exponential number of sub-networks whose gradients are explicitly guaranteed to be orthogonal during training.
- Experiments demonstrate that the proposed **SORS** can achieve significantly better robustness than the vanilla single model and model ensemble baseline. It can also be combined with other types of defense methods and obtain state-the-the-art defense performance.

The rest of the paper is organized as follows. In Section 2, we describe the most related works to our proposed method. Then in Section 3, we elaborate our **SORS** which aims to train one big super-network and explicitly constrain each sub-network to be gradient-orthogonal to get better robustness. To demonstrate the effectiveness, Section 4 shows the experimental details and results of our SORS compared with other baseline methods on MNIST, CIFAR-10 and CIFAR-100 datasets. Next in Section 5, we conduct extensive ablation studies to analyze our SORS and its variants for better understanding. Finally, we draw the conclusion in Section 6.

## 2. Related work

### 2.1. Adversarial attack

Despite the great success of deep neural networks, they can be easily attacked by adversarial examples. For better efficiency and effectiveness, various types of adversarial attack methods have been proposed, which can be roughly divided into three categories: optimization-based methods, gradient-based methods, and generation-based methods. Optimization-based methods regard the generation of adversarial examples as an optimization problem and solve it by minimizing or maximizing one specific objective or performance criteria. Classical methods include L-BFGS [38] and C&W [6]. Compared to optimization-based methods, gradient-based methods add perturbations iteratively by directly using the gradients with respect to the input as guidance. Therefore they are much faster, representative works include FGSM [11], BIM [22], PGD [32] and MIM [9]. To further speed up the adversarial

**Table 1**
The relationship between the attack success rate and gradient orthogonality. It shows that a higher gradient orthogonality often leads to a lower attack success rate.

| | GoogLeNet | VGG-16 | ResNet-50 | ResNet-101 | ResNet-152 |
|---|---|---|---|---|---|
| Gradient orthogonality | 0.99 | 0.98 | 0.97 | 0.96 | 0 |
| Attack success rate | 47% | 60% | 60% | 65% | 77% |

example generation process, generation-based methods [2,36,43,15] directly train a generative model to learn how to transform input images to adversarial examples.

### 2.2. Adversarial defense

Along with the rapid development of adversarial attack methods, many different types of adversarial defense methods [7,45,29,11,22,44,50,35,33] have been proposed. For input transform-based methods, they try to remove the adversarial disturbance of an adversarial example by using pre-processing techniques before feeding it into the target network. Popular techniques include JPEG compression [7], BitSqueezing [45] and DNN-Oriented JPEG Compression [29]. Adversarial training-based methods [11,22,44] are more direct and improve the robustness of the network by directly adding the adversarial examples into the training set. Regarding the fact that gradient is the most important clue in gradient-based attack methods, gradient mask-based methods [50,35] construct models that have less available gradients for adversarial defense.

Model ensemble [4,39] is another widely used technique to boost the robustness. As analyzed before, the key reason why the model ensemble is more robust than the single model is the smaller adversarial subspace and partial gradient orthogonality among the models involved. However, the drawbacks of the ensemble-based model exist: linearly increased resource consumption making it harder to use too many models, and uncontrollable gradient orthogonality among these models. Recently, RSE [27] proposed one method that does not need to store extra models by adding random noise layers, but it still requires explicit ensemble during the runtime. Pang et al. [34] increases the ensemble diversity by encouraging non-maximal predictions of each sub-network to be mutually orthogonal in the adversarial setting. But the gradient orthogonality we proposed is totally different with the orthogonality defined in [34]. [34] use prediction orthogonality but we use gradient orthogonality, which is more natural and effective to resist gradient based adversarial attack. From another perspective, our gradient orthogonality is potentially complementary to prediction orthogonality.

Recently, there have been works [13,8] that use neural network search (NAS) to improve network robustness. [13] attempts to systematically analyze and understand the adversarial robustness of neural networks from the perspective of neural network structure. They searched and designed a series of robust network structures called RobNets. Although we both have used NAS, there are several differences between the SORS we proposed and RobNets: 1) Different motivation. RobNets study attack-resistant network architecture from an architectural perspective. The SORS we proposed is based on the advantages of the model ensemble, taking advantage of neural network search to ensemble these models into one model and trained together. 2) Different network structure. RobNets' search space is different from our SORS, and it is more difficult to train and requires more computing resources. 3) Different calculation cost. RobNets need to generate adversarial examples during training, which greatly increases the computational cost, while we use gradient orthogonality to enhance robustness without using specific types of adversarial examples. 4) Different types of adversarial attack. RobNets focus on $L_\infty$ attacks, while SORS is effective against all common adversarial example attacks. [8] aims to use the NAS framework to improve the network's adversarial robustness from an architectural perspective, and explores the relationship among adversarial robustness, Lipschitz constant, and architecture parameters. Although we have used neural network search, the SORS we proposed and the RACL they proposed have the following differences: 1) Different motivation. RACL initializes the network with robust architecture to further obtain adversarial robustness, while our proposed SORS starts from analyzing the advantages and disadvantages of the model ensemble. 2) Different constraints. RACL constructs new constraints to reduce the Lipschitz constant to further improve robustness. SORS improves robustness by constraining gradient orthogonality. 3) Different calculation cost. RACL, like RobNets, needs to use adversarial training, which greatly increases the computational cost. And we use gradient orthogonality to enhance robustness, without using specific types of adversarial examples.

In contrast, though our method utilizes one single large super-network to cover an exponential number of sub-networks, it only samples one specific sub-network randomly at each forward pass. More importantly, our method is complementary to most aforementioned defense methods.

### 2.3. Dynamic neural networks

Our super-network is partially inspired by the dynamic neural network. In [26], Liu et al. propose to learn a gating module to drop some network blocks adaptively. Lin et al. [25] propose a runtime neural pruning (RNP) framework which can be dynamically pruned at runtime. Yu et al. [46] propose a Slimmable-Net which can dynamically adjust the network width at runtime based on accuracy and speed to achieve a balance between performance and efficiency across different devices. Super-network has also been widely used in neural architecture search [41,14,5].

Different from the above, we studies the super-network from the adversarial robustness perspective and explicitly incorporates a gradient orthogonality constraint among different sub-networks.

## 3. Self-orthogonal randomization super-network

### 3.1. Motivation

As analyzed before, ensemble-based models can significantly boost the robustness against adversarial attacks based on the following two observations: 1) The shared adversarial subspace of multiple models is much smaller than that of one single model, hence it is more difficult for existing attack methods to find a suitable adversarial example that can fool all the models involved. 2) Given an input image, the gradients of different models with respect to it are partial orthogonal. Therefore, when using gradient-based adversarial attack methods to attack it, more iteration steps are needed.

Intuitively, if we can involve more models and guarantee that the gradients of these models are orthogonal as much as possible, the defense robustness will be significantly boosted. However, the resource (storage and computation) consumption will be linearly increased if more models are used. Moreover, in modern machine learning systems, it is still difficult to explicitly guarantee the gradients of many independent models to be orthogonal during the training time. This is because, due to the memory and GPU number limit, these models are often trained across different machines and synchronizing across different machines often needs very complex system optimization and infrastructure.

In this paper, we take a step further and think about the question "Whether these models can be ensembled into one model and trained together?" In fact, to accelerate the searching process of neural architecture search (NAS), there is one single-path based NAS method [14]. In this method, given a macro network structure, in order to search for an optimal operator for each layer, they add all the possible operators into each layer in parallel, and train the big super-network as a whole by uniform sampling.

Inspired by [14], we propose a novel framework called "Self-Orthogonal Randomization Super-network" (**SORS**). Basically, it follows the idea in [14] and build a special type of super-network that has multiple operators for each layer. During the training and inference stage, at each forward pass, one operator will be randomly selected for each layer, and all the selected operators will form one sub-network. Assuming that the total layer number is $n$ and $m$ different operators exist for each layer, **SORS** can be decomposed into $m^n$ models considering all possibility. But actually, the total model size is only $m$ times bigger. Moreover, since only one sub-network will be sampled and used at the forward pass, the memory consumption and computation cost are both identical to the single sub-network during runtime. Another advantage is that all these sub-networks can be jointly trained and constrained with the gradient orthogonality because they share a common weight space.

It's worth noting that although we are inspired by [14], the goal of [14] is to search a good-performance network among different candidate operators. It has no gradient orthogonality constraint among sub-networks during training and no randomness during inference, therefore it cannot defend against adversarial attack at all.

### 3.2. Self-orthogonal randomization super-network

The overall framework of **SORS** is depicted in Fig. 3. Generally, **SORS** is often built based on one specific macro network structure, such as ResNet-18 and ResNet-34. This macro network structure defines the overall network information, including the total layer number $n$ and some operator-specific parameters like channel number, kernel size, and stride. Then for each layer $i$, we will choose multiple suitable operator blocks to form an operator set $OP^i$:

$$OP^i = \{op_1^i, op_2^i, \ldots, op_m^i\}. \tag{1}$$

Here $m$ is the total number of operator blocks involved and can be different for different layers. For each operator $op_k^i$, it may be identical or different from each other. Theoretically, $op_k^i$ can be any type of operator once it can guarantee its output has the correct shape. But to achieve better recognition results, we often use some popular operators like standard convolutional layer and advanced blocks such as residual block [16] and shuffle block [48,30].

Unlike vanilla deep neural networks, each forward pass will not activate all the operators contained in **SORS**. Instead, only one operator will be activated for each layer. Therefore, each forward pass can be actually defined as a path $\mathcal{P}$:

$$\mathcal{P} = \{id_1, id_2, \ldots, id_n\}, \tag{2}$$

where $id_i$ indicates the selected operator index. By sequentially connecting the operators along this path, a sub-network $\mathcal{N}_\mathcal{P}$ can be defined:

$$\mathcal{N}_\mathcal{P} = (op_{id_1}^1, op_{id_2}^2, \ldots, op_{id_n}^L). \tag{3}$$

As mentioned before, if all the layers have $m$ operators, there are $m^n$ possible sub-networks in total. And to make sure all these sub-networks can learn good enough recognition capacity, each operator will be uniformly sampled during the training stage.

*Normalization Layer.* In modern deep neural networks, the batch normalization (BN) layer plays a key role in stabilizing the training for better performance. However, using BN is not that easy in super-network, because it uses different mean and variance statistics for feature normalization in the training and testing stage. In detail, during training, it uses the mean and variance calculated by the batch statistics of that training iteration. But for testing, it instead uses the moving average of mean and variance statistics along the whole training process. Theoretically, each operator $op_k^i$ in **SORS** is included in $m^{n-1}$ different sub-networks and the feature statistics of different sub-network may be significantly different, so the moving average of BN statistics within each operator will be meaningless.

To address this issue, existing NAS methods re-calculate the BN statistics for each sub-network on a subset of training images before testing or use a private BN for each operator in each sub-network. However, the former method is very time-consuming and the latter method is infeasible because of the exponential sub-network number. In this paper, we propose to replace all the BN layer with group normalization (GN) layer [42]. Compared to BN, GN can achieve comparable recognition performance but normalizes the features based on the mean and variance statistics of the test image itself. Therefore, its behavior is consistent during training and testing.

### 3.3. Objective loss function

To train the **SORS**, our objective loss function $\mathcal{L}$ consists of two parts: the normal recognition loss $\mathcal{L}_{rec}$ that ensures each sub-network can achieve great recognition performance by itself and self-orthogonal loss $\mathcal{L}_{ort}$ that constrains the gradients of different sub-networks to be orthogonal as much as possible:

$$\mathcal{L} = \sum_i \mathcal{L}_{rec}(\mathcal{N}_p^i) + \lambda * \sum_{i,j} \mathcal{L}_{ort}(\mathcal{N}_p^i, \mathcal{N}_p^j). \tag{4}$$

Here $\lambda$ is the weight to balance the importance of these two terms, and $\mathcal{N}_p^i, \mathcal{N}_p^j$ denote any possible sub-network. By default, we directly adopt cross entropy loss for different sub-network:

$$\mathcal{L}_{rec}(\mathcal{N}_p^i) = \sum_k^C - y_k * \log(\mathcal{N}_p^i(x)), \tag{5}$$

where $x$ is the input image and $y$ is its ground truth label. And $y_k$ is 1 if $k = y$ else 0. To calculate $\mathcal{L}_{ort}$, we will calculate the gradient of $\mathcal{L}_{rec}$

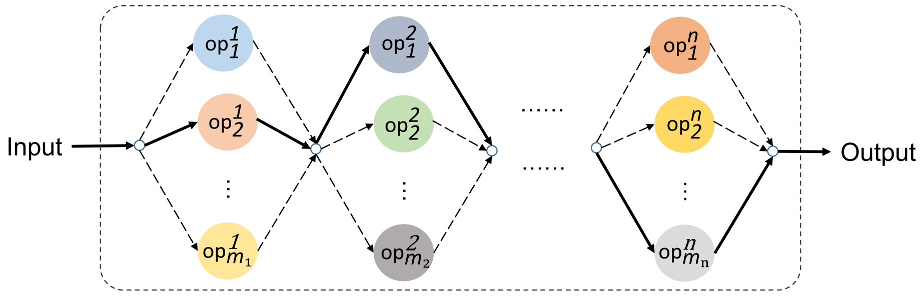


**Fig. 3.** The overall framework of **SORS**. For each layer of **SORS**, it consists of multiple possible parallel operators. By randomly sampling one operator for each layer, a sub-network can be built. Considering all the possibilities, an exponential number of sub-networks can be obtained within such a single super-network.

with respect to the input $x$ for each sub-network then constrain the gradients of any two networks are orthogonal for the same $x$.

$$\begin{aligned} &\mathcal{G}_x(\mathcal{N}_p^i) = \frac{\partial \mathcal{L}_{rec}(\mathcal{N}_p^i)}{\partial x}, \\ &\mathcal{L}_{ort}(\mathcal{N}_p^i, \mathcal{N}_p^j) = \left( \frac{\mathcal{G}_x(\mathcal{N}_p^i)}{\|\mathcal{G}_x(\mathcal{N}_p^i)\|^2} \cdot \frac{\mathcal{G}_x(\mathcal{N}_p^j)}{\|\mathcal{G}_x(\mathcal{N}_p^j)\|^2} \right)^2. \end{aligned} \tag{6}$$

Here we normalize the gradient $\mathcal{G}_x(\mathcal{N}_p^i), \mathcal{G}_x(\mathcal{N}_p^j)$ before calculating their dot product. In fact, the dot product of two normalized gradients is just the cosine similarity between them. And the smaller cosine similarity also represents stronger orthogonality. As mentioned in the Introduction part, we want the gradient of two different sub-networks to be orthogonal, i.e, the cosine angle is 90 degree or the cosine similarity is 0. Using square of cosine similarity, [0,1], makes sure the minimal loss value will be achieved when orthogonal.

### 3.4. Training strategy

The training strategy of SORS is similar to training regular networks. But like other super-networks [14,5], SORS is generally harder to train because all the sub-networks need to have good performance. Therefore we also adopt some advanced training strategies in [5]. In detail, we use three times epoch number than the regular sub-network training. Moreover, we leverage the common teacher-student training strategy [17] to improve the recognition performance, i.e., add an extra loss term by using the soft label predicted by the teacher model as the ground truth of the student model. Therefore, the final loss function is the combination of $\mathcal{L}$ in Eq. (4) and this extra loss term $\mathcal{L}_{soft}$:

$$\mathcal{L}_{total} = \mathcal{L} + \mathcal{L}_{soft}, \tag{7}$$

where $\mathcal{L}_{soft}$ adopts the same formulation as $\mathcal{L}_{rec}$, but $y_k$ becomes the soft logistic value of $k$-th class predicted by the teacher model. In all the following experiments, the default teacher model is a pretrained ResNet-18 for each dataset and the student model is just our target model SORS.

## 4. Experiment

**Datasets.** Due to limited computation resource, we use the popular MNIST, CIFAR-10 and CIFAR-100 datasets to conduct comparison experiments with other defense methods. The training set of MNIST contains 60,000 images, while CIFAR-10 and CIFAR-100 contain 50,000 images. All the datasets have 10,000 images for testing.

**Details of network architectures.** For MNIST, we use LeNet-5 as the baseline model. As depicted in Fig. 4, the depth of SORS is set exactly the same as LeNet-5 and four parallel operators are added in each layer, whose kernel sizes remain unchanged as that in LeNet-5.

For CIFAR-10 and CIFAR-100, ResNet-18 is used as the baseline model. As shown in Fig. 5, similar to the setting on MNIST, the single convolutional layer in the residual block of ResNet-18 is replaced with several paralleled convolutional operators. For the first convolutional layer of the residual block, four convolutional operators with unchanged kernel size are provided for random selection. For the second convolutional layer, the number of alternative convolutional operators reduces to three, and the kernel sizes are 3, 5, 7 respectively. Moreover, in our SORS residual block, we replace the batch normalization layer after each convolutional layer in the original residual block with group normalization layer.

**Training Details.** In our experiments, for MNIST, the models were trained using SGD optimizer with default settings in Pytorch. The learning rate is set to 0.01. For CIFAR-10 and CIFAR-100, in details, the models are trained using SGD optimizer with momentum of 0.9, and weight decay is set to be 0.0001. The learning rate is set to start at 0.1, then is divided by 10 at the 150th and 200th epochs.

**Evaluation metric.** We adopt a series of adversarial attack methods for evaluating the adversarial defense performance of our method, including gradient-based methods and optimization-based methods. For gradient-based methods, we adopt FGSM, PGD and MIM under the $L_\infty$ norm constraint. For MIM, we follow the setting in [9], the attack iteration number $T$ is 10 and the step size is $\epsilon/T$, where $\epsilon$ is the perturbation threshold. For PGD, we follow the setting in [32], the attack iteration number $T$ is 7 and the

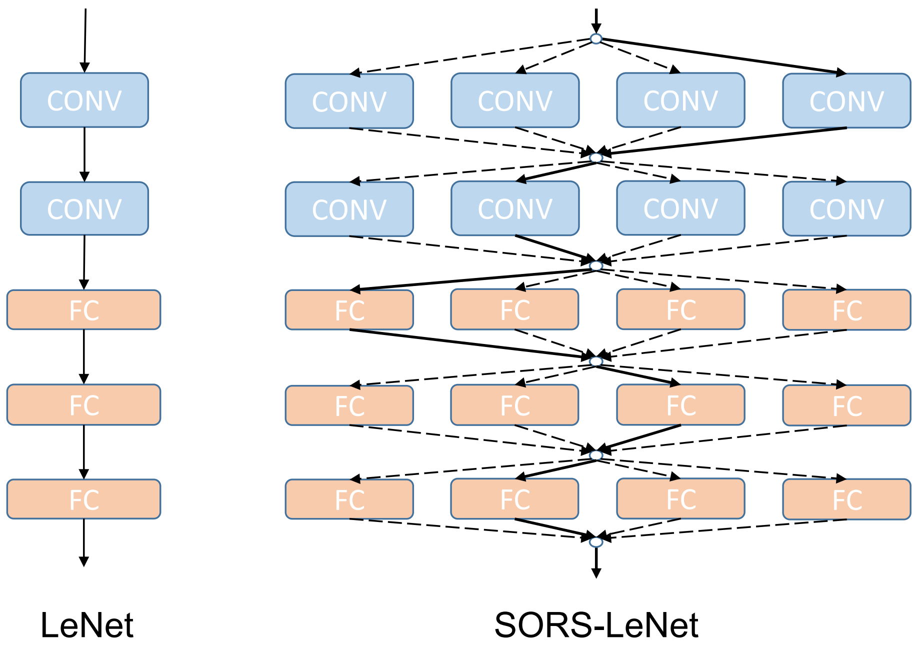


**Fig. 4.** *LeNet vs SORS-LeNet.* Left: LeNet-5. Right: SORS-LeNet.

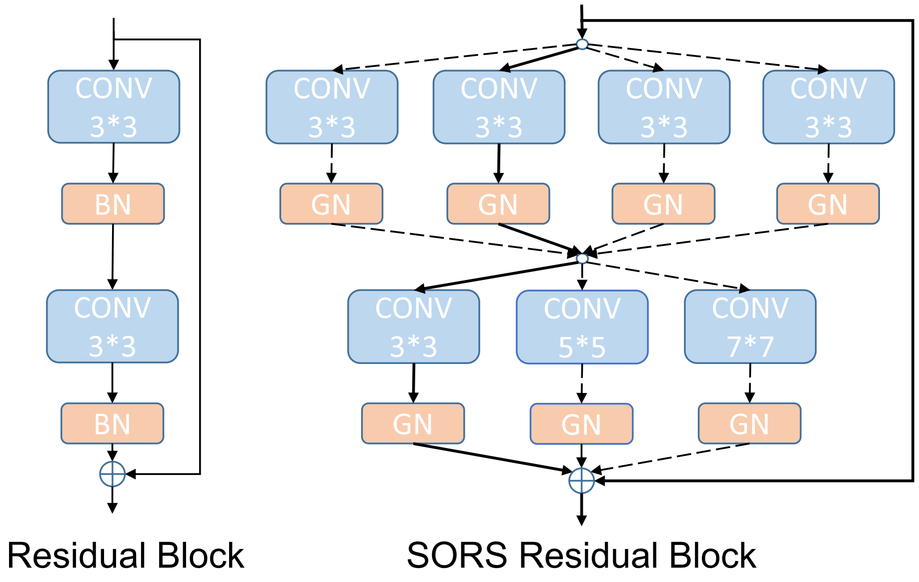


**Fig. 5.** *Residual Block vs SORS Residual Block.* Left: Residual Block. Right: SORS Residual Block.

step size is $\epsilon/4$. For optimization-based methods, we use C&W under the $L_2$ norm constraint. The confidence $c$ of C&W is 0 and the iteration number is 10000 for the MNIST dataset, the confidence $c$ is 0 and the iteration number is 100 for the CIFAR-10 dataset, and the confidence $c$ is 1 and the iteration number is 100 for CIFAR-100 dataset. By default, we adopt the classification accuracy as the evaluation metric in all following experiments.

### 4.1. Robustness evaluation on white-box attacks

We first evaluate the robustness of our SORS under the white-box attack setting, where the network structure and weights are fully exposed to the attackers. But under the common definition mentioned in [1], for white-box attack, attackers cannot change the underlying working principle. In other words, we will still keep the randomness during the sub-network sampling process.

**Defence on MNIST.** As described above, we use the macro network structure of LeNet-5 [23] to build our target SORS model for MNIST. And the operator set is just built by repeating the operator in each layer four times. Here we compare our method with three simple baselines, i.e., the vanilla single model baseline (LeNet-5) and two naive model ensemble baselines (LeNet-5×2, LeNet-5×4). For each independent LeNet-5 involved for ensemble, different initialization is used for training. As shown in Tables 2–4, RS represents one variant of SORS without the gradient orthogonality constraint during the training stage. RS* represents ensembling two randomly selected sub-networks from RS at every turn. RS and RS* are the randomization super-network baseline. SORS* represents ensembling two randomly selected sub-networks from SORS.

As shown in Table 2, our SORS outperforms all the baselines by a large margin for different adversarial attack methods. For example, when threshold $\epsilon = 75$, the accuracy of LeNet-5 is only 1.0% under PGD attack. Even when four LeNet-5 models are ensembled, the accuracy is still only 1.4%. Yet our SORS is 38.2%, which is nearly thirty times higher. For the classification accuracy on the clean data, though the accuracy of our SORS slightly degrades to 96.7%, it is still acceptable and on par with the baseline LeNet-5. In our understanding, the slight performance decrease mainly comes from the aforementioned training difficulty and the extra gradient orthogonality constraint.

**Defence on CIFAR-10 and CIFAR-100.** Similar to the above MNIST setting, three different baselines are considered: single ResNet-18, model ensemble with ResNet-18 and ResNet-34 and model ensemble with four ResNet-18 models. We also conduct comparative experiments with RSE [27] with the default settings.

In Tables 3 and 4, we show the comparison results on the CIFAR-10 and CIFAR-100 datasets respectively. Similar to the results on MNIST, the performance of our SORS will be slightly worse than the baseline methods on the clean data. But for adversarial examples, the classification accuracy of our SORS surpasses the baseline methods by about 30% $\sim$ 50%. Compared to RSE, our SORS is much more robust on both the CIFAR-10 and CIFAR-100 datasets. In addition, RSE needs explicit model ensemble as naive ensemble method, their computation cost is linearly proportional to the number of model ensembled. In contrast, our SORS only needs one single sub-network during inference and is much more

**Table 2**
Classification accuracy (%) of both clean images and adversarial examples generated by different attack methods on the MNIST dataset under white-box setting. For FGSM, PGD and MIM, we show the results with four different threshold $\epsilon$ = 25, 50, 75 and 100.

| | Clean | FGSM | PGD | MIM | C&W |
|---|---|---|---|---|---|
| LeNet-5 | 99.0 | 85.2/53.0/27.4/14.7 | 69.9/10.1/1.0/0.4 | 78.2/27.1/5.3/1.4 | 0 |
| LeNet-5×2 | 99.1 | 88.2/62.0/36.8/20.5 | 81.7/23.9/2.5/0.5 | 84.8/38.8/9.7/2.2 | 35.0 |
| LeNet-5×4 | **99.2** | 92.7/61.5/22.8/8.9 | 84.8/26.7/1.4/0.0 | 90.0/45.0/8.9/1.1 | 49.6 |
| RS | 98.9 | 88.1/63.3/41.0/26.0 | 75.6/18.2/2.3/0.3 | 80.4/37.3/13.0/4.0 | 87.1 |
| RS* | 99.0 | 84.3/56.6/34.8/22.5 | 89.5/46.6/13.7/3.1 | 77.3/27.8/8.0/2.3 | 90.9 |
| SORS | 96.7 | 94.4/86.3/70.4/**48.6** | 92.4/**73.8**/**38.2**/**13.2** | 90.1/**69.8**/**36.4**/**13.5** | **96.1** |
| SORS* | 97.4 | **95.3**/**87.2**/**70.5**/46.2 | **93.1**/70.8/28.2/7.3 | **91.5**/68.1/30.6/8.4 | 93.5 |

**Table 3**
Classification accuracy (%) of both clean images and adversarial examples generated by different attack methods on the CIFAR-10 dataset under white-box setting. For FGSM, PGD and MIM, we show the results with four different threshold $\epsilon$ = 1, 2, 4 and 8.

| | Clean | FGSM | PGD | MIM | C&W |
|---|---|---|---|---|---|
| ResNet-18 | 93.8 | 66.3/48.7/34.5/25.2 | 49.9/17.5/1.5/0.0 | 55.8/26.6/6.3/0.4 | 0 |
| ResNet-18 + ResNet-34 | 94.8 | 76.7/60.8/45.4/32.8 | 60.6/25.6/4.2/0.1 | 67.3/35.4/9.3/0.7 | 33.0 |
| ResNet-18$\times$4 | **95.1** | 84.1/66.4/45.1/29.6 | 64.5/32.8/5.9/0.1 | 77.4/48.5/15.7/1.4 | 45.0 |
| RSE | 89.5 | 67.7/40.1/19.1/10.0 | 60.8/22.3/2.4/0.1 | 63.7/29.0/5.6/0.5 | 75.0 |
| RS | 92.6 | 77.1/60.2/44.4/33.0 | 59.1/23.4/2.9/0.0 | 63.3/33.4/9.9/1.3 | 77.9 |
| RS* | 93.1 | 74.2/55.9/38.8/27.2 | 62.4/26.7/4.1/0 | 61.6/29.8/7.2/5.4 | 76.6 |
| SORS | 91.3 | 90.1/**83.8**/**72.5**/**55.8** | **86.9**/**71.7**/**41.5**/**9.4** | **83.1**/**67.4**/**40.2**/**12.6** | **79.9** |
| SORS* | 92.5 | **91.0**/83.7/70.9/53.7 | 86.4/68.6/22.6/5.4 | 82.3/64.4/33.5/8.3 | 75.8 |

**Table 4**
Classification accuracy (%) of both clean images and adversarial examples generated by different attack methods on the CIFAR-100 dataset under white-box setting. For FGSM, PGD and MIM, we show the results with four different threshold $\epsilon$ = 1, 2, 4 and 8.

| | Clean | FGSM | PGD | MIM | C&W |
|---|---|---|---|---|---|
| ResNet-18 | 71.6 | 47.4/29.7/18.2/12.1 | 31.4/8.4/1.1/0.0 | 36.2/13.1/2.9/0.3 | 0 |
| ResNet-18 + ResNet-34 | **75.21** | 62.7/46.1/30.6/18.3 | 44.9/48.7/3.6/0.3 | 52.4/26.7/7.7/1.1 | 3.0 |
| ResNet-18$\times$4 | 72.34 | 64.2/49.9/33.1/18.3 | 49.1/22.2/5.7/0.6 | 56.0/32.5/11.8/2.1 | 0.1 |
| RSE | 65.6 | 40.4/20.8/11.42/7.9 | 30.2/6.2/0.7/0.1 | 33.9/10.1/1.7/0.29 | 0.0 |
| RS | 69.17 | 58.7/42.2/27.5/17.4 | 46.1/17.4/2.8/0.2 | 42.8/17.8/4.0/0.3 | 67.0 |
| RS* | 70.68 | 56.1/37.7/23.7/15.4 | 41.7/13.5/1.8/0.0 | 40.1/15.5/3.0/0.2 | 29.1 |
| SORS | 69.0 | 79.0/**69.0**/**55.3**/**40.7** | **71.6**/**50.2**/**23.5**/**5.3** | **65.2**/**44.3**/**22.6**/**6.8** | **71.4** |
| SORS* | 70.3 | **79.6**/68.1/53.0/36.0 | 67.6/44.6/17.4/2.9 | 62.1/39.8/16.9/4.5 | 46.2 |

efficient. Last but not least, our method is theoretically complementary to RSE, i.e., we can train multiple super-networks and use super-network level ensemble to get better results.

Taking all the above results into consideration, we can draw the conclusion that, though the naive model ensemble can boost the robustness of the single model baseline, the performance gain from our SORS is far more significant with the same amount of parameters. We need to emphasize that, though our target SORS model is built with existing networks by simply repeating their operators in the above experiments, it is a general idea and supports combination of not only standard convolutional layer, but also different types of blocks, such as shuffle block [30] and mobilenet block [18].

### *4.2. Robustness evaluation on black-box attacks*

For black-box setting, we generate adversarial examples with the ResNet-101 model then attack methods with other network structures listed in Table 5. Similar to the results on white-box attack, our SORS outperforms almost all the baselines by a large margin for different adversarial attack methods when resisting the black-box adversarial attack especially for large $\epsilon$.

### *4.3. Resources consumption analysis*

For fair resource consumption analysis, we assume that our SORS model has $m$ parallel operators at each layer and the model ensemble baseline also contains $m$ independent models. Under this setting, our SORS has the same model size as the ensemble baseline. Regarding the training time, since each model is independently trained in the ensemble baseline, the total training time is $m$ times longer than the single model. But empirically, we find SORS needs less than $m\times$epochs to converge. Therefore, the training computation cost of SORS is smaller or at least comparable to the ensemble baseline. However, at each forward pass in the inference stage, our SORS will only activate one sub-network, so its memory consumption and computation cost are identical to the single model. In contrast, the naive model ensemble baseline is more expensive because the same input needs to be fed into $m$ different models.

## 5. Ablation study

In this part, we will first conduct extensive ablation studies to analyze the underlying working principles of SORS and its variants. Then we demonstrate it is complementary to other types of adversarial defense methods.

**Importance of randomization and gradient orthogonality.** As analyzed in the introduction and motivation part, the robustness of our method benefits from two aspects: the randomization super-network consisting of an exponential number of sub-networks and the gradient orthogonality constraint among these sub-networks. To study these two components independently, we further train another variant (RS) without the gradient orthogonality constraint during the training stage. Its classification results are also given in Tables 2–4. It can be seen that, compared to the single model baseline, only using the randomization super-network can already significantly boost the robustness against adversarial

**Table 5**
Classification accuracy (%) of adversarial examples generated by different attack methods on the CIFAR-10 dataset under black-box setting. For FGSM, PGD and MIM, we show the results with four different threshold $\epsilon$ = 1, 2, 4 and 8.

| | FGSM | PGD | MIM |
|---|---|---|---|
| ResNet-18 | 91.4/83.7/72.7/57.2 | 90.2/80.9/64.7/41.9 | 89.8/78.6/57.3/31.3 |
| ResNet-18 + ResNet-34 | 91.6/82.7/68.5/52.6 | 90.2/75.5/50.5/23.7 | 89.7/72.6/42.4/16.0 |
| ResNet-18$\times$4 | **92.9**/**86.8**/75.2/59.5 | 91.6/82.7/65.0/41.9 | 91.3/80.4/57.4/30.4 |
| RS | 91.4/84.1/71.3/55.1 | 90.1/79.6/63.3/41.2 | 90.1/77.2/56.2/32.0 |
| SORS | 92.6/86.7/**77.1**/**62.3** | **91.7**/**86.4**/**77.1**/**62.9** | **91.4**/**84.2**/**70.3**/**48.9** |

attacks, and adding the gradient orthogonality constraint can promote robustness even more.

Furthermore, in order to verify whether the gradient orthogonality among different sub-networks is explicitly boosted by adding the self-orthogonal loss, we randomly sample some sub-networks from RS and SORS and plot the gradient angle between any two of them in Fig. 6 by using a small adversarial example set. It can be easily observed that SORS obtains better gradient orthogonality than RS. For the naive model ensemble baseline, we find that their gradient orthogonality is pretty low, which means all the involved models have very similar behavior even though different initialization is used. By taking all the points into consideration, the average gradient orthogonality of RS and SORS is 0.771 and 0.985 respectively.

**Ensemble sub-networks of SORS.** As demonstrated in Tables 2–4, the robustness of model ensemble is stronger than that of one single model. Since our SORS can be decomposed into an exponential number of sub-networks, it can be regarded as a special type of model ensemble. But during runtime, only one sub-network will be sampled and tested. So we are curious about whether ensembling the classification results of multiple sub-networks can bring extra robustness. To answer this question, we conduct one experiment by ensembling two randomly selected sub-networks at every turn, and also put its results in Tables 2–4 (marked as "SORS*"). Interestingly, we find SORS is even slightly better than SORS* in most cases. In our understanding, it is because using more sub-networks at each forward pass will leak more model information and gradient direction to the attackers. A similar conclusion can also be drawn in the randomization super-network baseline (RS and RS*).

**Different number of operators.** By default, in this paper, we use four different operators for each layer. In the extreme case, when the operator number decreases to one, SORS will degrade into the single model baseline. In this experiment, we will study the relationship between the defense robustness and the number of parallel operators used. In details, we try the operator number to be $2, 3, 4, 5$ respectively and compare their robustness, and the results are shown in Table 6. It can be seen that when the parallel operator number increases from 2 to 4, the classification accuracy also increases. However, when the operator number becomes 5, the classification accuracy degrades a lot. It is because the training difficulty will significantly increase if too many operators are used in the super-network.

**Table 6**
Relation between operator number and model classification accuracy on the MNIST dataset. We show the results with four different threshold $\epsilon$ = 25, 50, 75 and 100. Higher accuracy indicates better robustness. In most cases, when the operator number is 4, the best results can be achieved.

| Operator number | FGSM | PGD | MIM |
|---|---|---|---|
| 2 | 88.4/78.1/60.7/41.3 | 85.3/57.1/21.9/6.5 | 82.9/53.5/21.2/7.1 |
| 3 | **94.9**/86.1/70.3/**53.1** | 84.6/62.5/29.5/9.4 | 90.0/64.1/32.8/**16.2** |
| 4 | 94.4/**86.3**/**70.4**/48.6 | **92.4**/**73.8**/**38.2**/**13.2** | **90.1**/**69.8**/**36.4**/13.5 |
| 5 | 89.4/68.6/39.3/20.6 | 86.0/44.7/8.8/1.0 | 82.2/35.4/6.9/1.4 |

**Classification accuracy vs weight of self-orthogonal loss $\lambda$.** In Eq. (4), $\lambda$ is used to control the strength of the gradient orthogonality among different sub-networks. By default, we set $\lambda$ as 10 in our implementation. In this experiment, we want to study the influence of $\lambda$ on the final classification accuracy for both clean images and adversarial examples. Intuitively, if $\lambda$ is too small, SORS will degrade to RS and get worse gradient orthogonality and robustness. However, if $\lambda$ is too large, the training difficulty will be much higher and incur worse recognition results on clean images because the constraint among different sub-networks is too strict. In Fig. 7, we have shown the classification accuracy of different $\lambda$ on the clean images and adversarial examples. It can be seen that, as $\lambda$ increases, the classification accuracy on the clean images decreases. But for the adversarial examples, it first increases then decreases, which indicates the robustness gain is smaller than the classification ability loss if $\lambda$ is too large. By contrast, when setting $\lambda$ as 10, it can achieve good classification results on both clean images and adversarial examples.

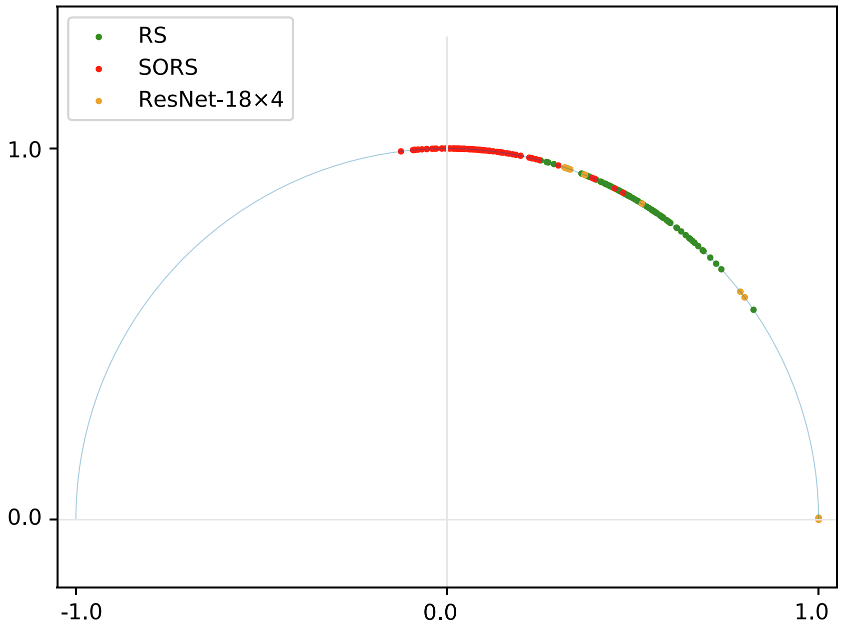


**Fig. 6.** The scatter plot of the gradient angle on the unit circle. The $x$ axis represents the gradient angle of two random sub-networks with range $[-1, 1]$ and the point indicates the angle of the gradient between any two randomly sampled sub-networks. For better visualization, we plot each data point on the unit circle and different positions on the circle (different values) indeed represent different gradient angles. Note that [0, 1] of y axis represents [0, 90] degrees. Obviously, our SORS has the best gradient orthogonality.

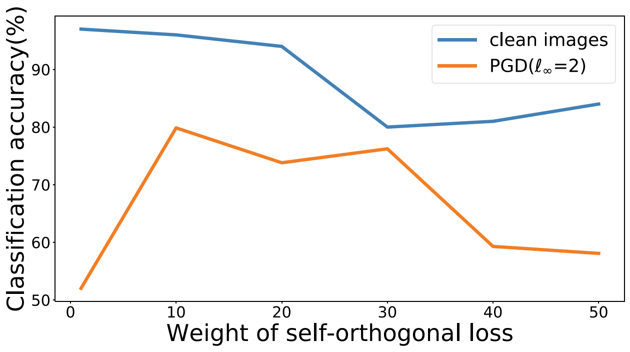


**Fig. 7.** The relationship between classification accuracy and the self-orthogonal loss weight $\lambda$. For clean images, larger self-orthogonal loss weight will incur worse classification accuracy. But for adversarial examples, the classification accuracy first increases then decreases.

**Complementary to other adversarial defense methods.** As introduced before, many different types of adversarial defense methods have been proposed in recent years, such as input transform-based and adversarial training-based. Since the proposed SORS is designed from the network structure design perspective, it can be regarded as one special type of self-robust network and complementary to other types of defense methods. To demonstrate it, we combine the proposed SORS in conjunction with four input transform-based adversarial defense methods, including MedianSmoothing [45], BitSqueezing [45], JPEG compression [7], and DNN-oriented JPEG [29]. Before feeding the adversarial examples into the final image recognition network, they will be first processed by these transform methods. It can be seen from Table 7 that the classification accuracy can still be significantly boosted upon these transform-based defense methods.

**Robustness vs model parameters.** In Table 8, we compare the robustness of our SORS on MNIST dataset with the model ensemble baseline LeNet-5×4, which has the same amount of parameters. But obviously, our SORS has much better robustness. Moreover, we compare our method to one bigger single model (VGG). Though VGG has more parameters than our LeNet-5 based SORS, its robustness is far behind us. Therefore, the robustness improvement of our SORS does not attribute to the introduction of more model parameters.

**Robustness evaluation on adaptive attack.** Adaptive attack assumes that the attacker knows the working mechanism of SORS, and then proposes adaptive attack methods to attack it. In this experiment, we design a simple gradient ensemble adaptive attack for SORS, which uses the ensemble gradient of multiple subnetworks instead of a single sub-network for attacking. In details, we consider the number of ensembled sub-networks as 2,3,4,5 respectively and compare the robustness of SORS against such attacks. It can be seen from Table 9 that, when the ensemble sub-network number increases from 2 to 5, the classification accuracy decreases, which shows more gradient leak will hurt the robustness. However, considering the accuracy decline is relatively small and computational complexity increases much in such adaptive attacks, the robustness of our SORS is still pretty satisfactory.

**Influence of teacher-student training strategy.** As mentioned before, in order to improve the recognition performance of subnetworks on clean images, we adopt the common teacher-student training strategy. To verify how much robustness is from teacher-student learning, we also use teacher-student learning on baseline methods "ResNet-18" and "ResNet-18×4" (marked with "*"). As shown in Table 10, the teacher-student learning itself does not improve the baseline models' robustness. In other words, the robustness of SORS is from the underlying working principle but not training strategy.

**Convergence analysis.** Though our SORS is more difficult to train than the single model baseline, we find it can converge very well in all the cases we tried. Based on some other super-

**Table 9**
Classification accuracy (%) of adversarial examples on the MNIST dataset under adaptive attack. We show the results with four different threshold $\epsilon$ = 25, 50, 75 and 100. Higher accuracy indicates better robustness. It's easy to find out that gradient ensemble attack cannot destroy the robustness of SORS.

| Ensemble | PGD |
|---|---|
| 2 | 86.35/68.61/22.60/5.44 |
| 3 | 84.52/64.90/30.0/3.81 |
| 4 | 83.59/62.83/27.18/2.93 |
| 5 | 83.10/61.17/25.31/2.11 |

**Table 7**
Gray-box classification accuracy (%) of adversarial examples generated by different attack methods on CIFAR-10 dataset by combining SORS with four existing input transform-based defense methods. Here we show the results with four different thresholds $\epsilon = 1, 2, 4, 8$. Obviously, our SORS is complementary to these defense methods and can bring significant extra robustness.

| | FGSM | PGD | MIM |
|---|---|---|---|
| MS | 85.3/74.5/57.9/40.0 | 84.5/72.5/53.1/28.9 | 84.0/68.4/39.9/11.3 |
| MS + SORS | **88.6**/**85.5**/**77.7**/**62.9** | **88.4**/**82.3**/**67.1**/**33.9** | **87.3**/**79.9**/**61.4**/**30.5** |
| BitS | 70.7/51.1/35.2/25.6 | 60.9/23.7/2.0/0.0 | 64.0/31.9/7.3/0.5 |
| BitS + SORS | **89.9**/**84.0**/**72.8**/**55.7** | **87.8**/**75.6**/**45.2**/**10.6** | **85.8**/**70.3**/**42.6**/**13.7** |
| JPEG | 80.9/77.5/70.0/54.8 | 82.2/78.4/72.4/**61.5** | 81.4/78.1/69.6/47.8 |
| JPEG + SORS | **82.1**/**80.2**/**76.9**/**69.5** | **83.1**/**80.5**/**74.5**/60.4 | **81.6**/**78.8**/**72.7**/**55.5** |
| DO-JPEG | 82.7/77.7/67.1/43.6 | 80.8/77.9/72.6/**63.3** | **83.1**/78.4/66.5/27.8 |
| DO-JPEG + SORS | **83.3**/**81.7**/**77.0**/**66.5** | **81.7**/**79.7**/**74.1**/61.7 | 82.7/**80.1**/**71.5**/**49.7** |

**Table 8**
Classification accuracy (%) of adversarial examples generated by different attack methods on the MNIST dataset and model parameter size. For FGSM, PGD and MIM, we show the results with four different threshold $\epsilon$ = 25, 50, 75 and 100. Obviously, the robustness improvement of our SORS does not attribute to the introducing of more model parameters.

| | Parameters | FGSM | PGD | MIM |
|---|---|---|---|---|
| VGG | 165.76M | 77.7/39.6/19.1/11.2 | 62.2/48.5/0.4/0 | 70.0/17.8/6.9/4.7 |
| LeNet-5×4 | 0.177 M | 92.7/61.5/22.8/8.9 | 84.8/26.7/1.4/0.0 | 90.0/45.0/8.9/1.1 |
| SORS | 0.177M | **94.4**/**86.3**/**70.4**/**48.6** | **92.4**/**73.8**/**38.2**/**13.2** | **90.1**/**69.8**/**36.4**/**13.5** |

**Table 10**
Classification accuracy (%) of adversarial examples on the CIFAR-10 dataset. We show the results with four different threshold $\epsilon$ = 1, 2, 4 and 8. "*" denotes the model trained with teacher-student learning.

| | PGD |
|---|---|
| ResNet-18* | 46.3/14.3/0.0/0 |
| ResNet-18×4* | 64.6/32.6/5.8/0.1 |
| SORS | **86.9**/**71.7**/**41.5**/**9.4** |

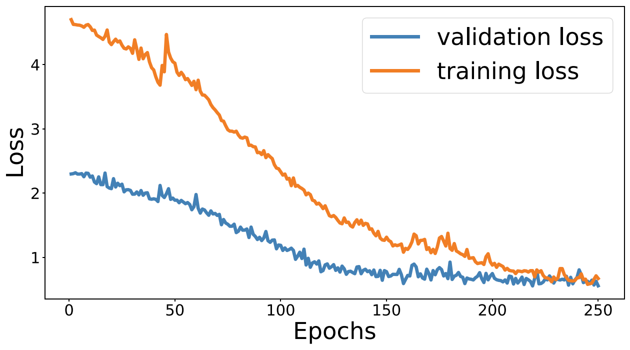


**Fig. 8.** The convergence curve of the training and validation loss. It shows that with four operators at each layer, our SORS can converge at about 250 epochs.

network results reported in [5,14], we think satisfactory results are always achievable by following proper training strategy as [5,14]. In Fig. 8, we show the convergence curve of the SORS built upon ResNet-18 for the CIFAR-10 dataset. It can be observed that our SORS converges at about 250 epochs.

## 6. Conclusion and discussion

In this paper, we first analyze the superior robustness of the model ensemble compared to the single model baseline and observe two crucial points: 1) The shared adversarial subspace of multiple models is much smaller than that of one single model. 2) The gradient orthogonality among different models will make the adversarial attack more difficult. Based on these two observations, we then propose a new type of defense method named **SORS**, which can be seen as a special type of self-robust recognition network design scheme. It trains one single super-network that can be decomposed into an exponential number of sub-networks. And because these sub-networks share the same weight space, it is easy to explicitly constrain the gradient orthogonality among them. In the inference stage, one sub-network will be randomly sampled at each forward pass. Experiments demonstrate that the proposed method can achieve better robustness than the single model and naive model ensemble baseline. Moreover, it is also complementary to other types of defense methods and further boosts their robustness. However, some limitations need to be addressed in the future. For example, we need to find more advanced training strategies to achieve the comparable recognition performance as the single model baseline on the clean data.

## CRediT authorship contribution statement

**Huanyu Bian:** Conceptualization, Methodology, Writing - original draft. **Dongdong Chen:** Writing - review & editing. **Kui Zhang:** Data curation. **Hang Zhou:** Software. **Xiaoyi Dong:** Formal analysis. **Wenbo Zhou:** Visualization. **Weiming Zhang:** Supervision, Project administration. **Nenghai Yu:** Funding acquisition.

## 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.

## Acknowledgments

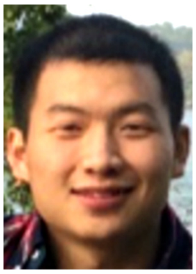
This work was supported in part by the Natural Science Foundation of China under Grant U20B2047, 62072421 and 62002334, Exploration Fund Project of University of Science and Technology of China under Grant YD3480002001, and by Fundamental Research Funds for the Central Universities under Grant WK2100000011.

## References

[1] Anish Athalye, Nicholas Carlini, David Wagner, Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples, arXiv preprint arXiv:1802.00420, 2018. .
[2] Shumeet Baluja, Ian Fischer, Adversarial transformation networks: Learning to generate adversarial examples, arXiv preprint arXiv:1703.09387, 2017. .
[3] Nicola Bezzo, James Weimer, Miroslav Pajic, Oleg Sokolsky, George J. Pappas, Insup Lee, Attack resilient state estimation for autonomous robotic systems, in: 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2014, pp. 3692–3698..
[4] Gavin Brown, Jeremy L Wyatt, Peter Tiňo, Managing diversity in regression ensembles, Journal of Machine Learning Research 6 (2005) 1621–1650.
[5] Han Cai, Chuang Gan, Song Han, Once for all: Train one network and specialize it for efficient deployment, 2019, arXiv preprint arXiv:1908.09791.
[6] Nicholas Carlini, David Wagner, Towards evaluating the robustness of neural networks, arXiv preprint arXiv:1608.04644, 2016. .
[7] Nilaksh Das, Madhuri Shanbhogue, Shang-Tse Chen, Fred Hohman, Li Chen, Michael E. Kounavis, Duen Horng Chau, Keeping the bad guys out: Protecting and vaccinating deep learning with jpeg compression, arXiv preprint arXiv:1705.02900, 2017. .
[8] Minjing Dong, Yanxi Li, Yunhe Wang, Chang Xu, Adversarially robust neural architectures, arXiv preprint arXiv:2009.00902, 2020. .
[9] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, Jianguo Li, Boosting adversarial attacks with momentum, 2018, pp. 9185–9193. .
[10] Ivan Evtimov, Kevin Eykholt, Earlence Fernandes, Tadayoshi Kohno, Bo Li, Atul Prakash, Amir Rahmati, Dawn Song, Robust physical-world attacks on deep learning models, arXiv preprint arXiv:1707.08945, 1, 2017. .
[11] Ian J Goodfellow, Jonathon Shlens, Christian Szegedy, Explaining and harnessing adversarial examples (2014), arXiv preprint arXiv:1412.6572, 2014. .
[12] Gaurav Goswami, Akshay Agarwal, Nalini Ratha, Richa Singh, Mayank Vatsa, Detecting and mitigating adversarial perturbations for robust face recognition, International Journal of Computer Vision 127 (6–7) (2019) 719–742.
[13] Minghao Guo, Yuzhe Yang, Xu. Rui, Ziwei Liu, Da.hua. Lin, 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, 2020, pp. 631–640.
[14] Zichao Guo, Xiangyu Zhang, Haoyuan Mu, Wen Heng, Zechun Liu, Yichen Wei, Jian Sun, Single path one-shot neural architecture search with uniform sampling, arXiv preprint arXiv:1904.00420, 2019. .
[15] Jiangfan Han, Xiaoyi Dong, Ruimao Zhang, Dongdong Chen, Weiming Zhang, Nenghai Yu, Ping Luo, Xiaogang Wang, Once a man: Towards multi-target attack via learning multi-target adversarial network once, in: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2019..
[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.
[17] Geoffrey E Hinton, Oriol Vinyals, Jeffrey Dean, Distilling the knowledge in a neural network. arXiv, Machine Learning (2015).
[18] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications, arXiv preprint arXiv:1704.04861, 2017..
[19] Brody Huval, Tao Wang, Sameep Tandon, Jeff Kiske, Will Song, Joel Pazhayampallil, Mykhaylo Andriluka, Pranav Rajpurkar, Toki Migimatsu, Royce Cheng-Yue, et al., An empirical evaluation of deep learning on highway driving, arXiv preprint arXiv:1504.01716, 2015. .
[20] Xiaojun Jia, Xingxing Wei, Xiaochun Cao, Hassan Foroosh, Comdefend: An efficient image compression model to defend adversarial examples, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 6084–6092.
[21] Alex Krizhevsky, Ilya Sutskever, Geoffrey E Hinton, Imagenet classification with deep convolutional neural networks, Neural Information Processing Systems 141 (5) (2012) 1097–1105.

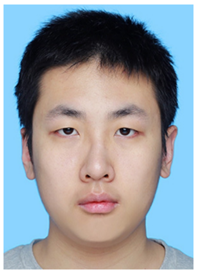
[22] Alexey Kurakin, Ian Goodfellow, Samy Bengio, Adversarial machine learning at scale, arXiv preprint arXiv:1611.01236, 2016. .
[23] Yann Lecun, Leon Bottou, Yoshua Bengio, Patrick Haffner, Gradient-based learning applied to document recognition, Proceedings of the IEEE 86 (11) (1998) 2278–2324.
[24] Fangzhou Liao, Ming Liang, Yinpeng Dong, Tianyu Pang, Hu. Xiaolin, Jun Zhu, Defense against adversarial attacks using high-level representation guided denoiser, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 1778–1787.
[25] Ji Lin, Yongming Rao, Jiwen Lu, Jie Zhou, Runtime neural pruning, 2017, pp. 2181–2191. .
[26] Lanlan Liu, Jia Deng, Dynamic deep neural networks: Optimizing accuracy-efficiency trade-offs by selective execution, arXiv: Learning, 2017..
[27] Xuanqing Liu, Minhao Cheng, Huan Zhang, Cho-Jui Hsieh, Towards robust neural networks via random self-ensemble, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 369–385.
[28] Yanpei Liu, Xinyun Chen, Chang Liu, Dawn Song, Delving into transferable adversarial examples and black-box attacks, arXiv preprint arXiv:1611.02770, 2016. .
[29] Zihao Liu, Qi Liu, Tao Liu, Yanzhi Wang, Wujie Wen, Feature distillation: Dnn-oriented jpeg compression against adversarial examples, arXiv: Computer Vision and Pattern Recognition, 2018..
[30] Ningning Ma, Xiangyu Zhang, Hai-Tao Zheng, Jian Sun, Shufflenet v2: Practical guidelines for efficient cnn architecture design, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 116–131.
[31] Xingjun Ma, Bo Li, Yisen Wang, Sarah M Erfani, Sudanthi Wijewickrema, Grant Schoenebeck, Dawn Song, Michael E. Houle, James Bailey, Characterizing adversarial subspaces using local intrinsic dimensionality, arXiv preprint arXiv:1801.02613, 2018. .
[32] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, Adrian Vladu (Eds.), Towards deep learning models resistant to adversarial attacks, 2017, arXiv preprint arXiv:1706.06083.
[33] Jörg Martin, Clemens Elster, Inspecting adversarial examples using the fisher information, Neurocomputing 382 (2020) 80–86.
[34] Tianyu Pang, Kun Xu, Chao Du, Ning Chen, Jun Zhu, Improving adversarial robustness via promoting ensemble diversity, arXiv preprint arXiv:1901.08846, 2019. .
[35] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Z. Somesh Jha, Berkay Celik, Ananthram Swami, Practical black-box attacks against machine learning, in: Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, ACM, 2017, pp. 506–519.
[36] Omid Poursaeed, Isay Katsman, Bicheng Gao, Serge Belongie, Generative adversarial perturbations, arXiv preprint arXiv:1712.02328, 2017. .
[37] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, Michael K Reiter, Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition, in: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, ACM, 2016, pp. 1528–1540.
[38] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, Rob Fergus, Intriguing properties of neural networks, arXiv preprint arXiv:1312.6199, 2013. .
[39] Florian Tramer, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, Patrick Mcdaniel, Ensemble adversarial training: Attacks and defenses, arXiv: Machine Learning, 2017..
[40] Florian Tramèr, Nicolas Papernot, Ian Goodfellow, Dan Boneh, Patrick McDaniel, The space of transferable adversarial examples, arXiv preprint arXiv:1704.03453, 2017. .
[41] Bichen Wu, Xiaoliang Dai, Peizhao Zhang, Yanghan Wang, Fei Sun, Yiming Wu, Yuandong Tian, Peter Vajda, Yangqing Jia, Kurt Keutzer, Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 10734–10742..
[42] Yuxin Wu, Kaiming He, Group normalization. arXiv: Computer Vision and Pattern Recognition, 2018..
[43] Chaowei Xiao, Bo Li, Jun-Yan Zhu, Warren He, Mingyan Liu, Dawn Song, Generating adversarial examples with adversarial networks, arXiv preprint arXiv:1801.02610, 2018. .
[44] Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan L. Yuille, Kaiming He, Feature denoising for improving adversarial robustness, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp. 501–509. .
[45] Xu. Weilin, David Evans, Yanjun Qi, Feature squeezing: Detecting adversarial examples in deep neural networks, in: Network and Distributed System Security Symposium, 2018.
[46] Jiahui Yu, Linjie Yang, Ning Xu, Jianchao Yang, Thomas Huang, Slimmable neural networks, arXiv preprint arXiv:1812.08928, 2018. .
[47] Jiajun Zhang, Chengqing Zong, et al., Deep neural networks in machine translation: An overview, 2015.
[48] Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, Jian Sun, Shufflenet: An extremely efficient convolutional neural network for mobile devices, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 6848–6856.
[49] Chenxiao Zhao, P. Thomas Fletcher, Mixue Yu, Yaxin Peng, Guixu Zhang, Chaomin Shen, The adversarial attack and detection under the fisher information metric, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, 2019, pp. 5869–5876. .
[50] Stephan Zheng, Yang Song, Thomas Leung, Ian Goodfellow, Improving the robustness of deep neural networks via stability training, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 4480–4488.

**Huanyu Bian** is currently a Ph.D. student with the University of Science and Technology of China (USTC). He received the B.S. degree in 2015 from Anhui University. His primary research interests include AI security and multimedia security.


**Dongdong Chen** is currently a senior researcher at Microsoft Research. Before that, he received a Ph.D. degree from the University of Science and Technology of China (USTC) under the joint PhD program between MSRA and USTC in 2019. His research interests include image generation, style transfer, AI security and general representation learning.

**Kui Zhang** is currently a Graduate student with the University of Science and Technology of China (USTC). He received the B.S degree in 2019 from JiLin University, Jilin. His primary research interests include AI security and robustness of deep learning.

**Hang Zhou** received his B.S. degree in 2015 from Shanghai University (SHU) and a Ph.D. degree in 2020 from the University of Science and Technology of China (USTC). Currently, he is a postdoctoral researcher at Simon Fraser University (SFU). His research interests include computer graphics, multimedia security and deep learning.



**Xiaoyi Dong** received his B.S. degrees in information security from the University of Science and Technology of China (USTC) in 2018. He is currently pursuing the Ph. D. degree in information security in USTC. His research interests include adversarial sample, 3D point cloud recognition, and DeepFake detection.

**Wenbo Zhou** received his B.S. degree in 2014 from Nanjing University of Aeronautics and Astronautics, China,and Ph. D degree in 2019 from University of Science and Technology of China, where he is currently postdoctoral researcher. His research interests include information hiding and AI security.

**Weiming Zhang** received his M.S. degree and Ph.D. degree in 2002 and 2005, respectively, from the Zhengzhou Information Science and Technology Institute, P.R. China. Currently, he is a professor with the School of Information Science and Technology, University of Science and Technology of China. His research interests include information hiding and multimedia security.

**Nenghai Yu** received his B.S. degree in 1987 from Nanjing University of Posts and Telecommunications, an M.E. degree in 1992 from Tsinghua University and a Ph. D. degree in 2004 from the University of Science and Technology of China, where he is currently a professor. His research interests include multimedia security, multimedia information retrieval, video processing and information hiding.