Machine Learning (2022) 111:685–711
https://doi.org/10.1007/s10994-021-06049-9

Achieving adversarial robustness via sparsity

Ningyi Liao[1] · Shufan Wang[1] · Liyao Xiang[1] · Nanyang Ye[1] · Shuo Shao[1] · Pengzhi Chu[1]

Received: 31 January 2021 / Revised: 26 July 2021 / Accepted: 27 August 2021 /
Published online: 12 October 2021

Abstract
Network pruning has been known to produce compact models without much accuracy degradation. However, how the pruning process affects a network's robustness and the working mechanism behind remain unresolved. In this work, we theoretically prove that the sparsity of network weights is closely associated with model robustness. Through experiments on a variety of adversarial pruning methods, image-classification models and datasets, we find that weights sparsity will not hurt but improve robustness, where both weights inheritance from the lottery ticket and adversarial training improve model robustness in network pruning. Based on these findings, we propose a novel adversarial training method called inverse weights inheritance, which imposes sparse weights distribution on a large network by inheriting weights from a small network, thereby improving the robustness of the large network.

Keywords Adversarial learning · Neural network pruning · Robustness · Sparsity

1 Introduction

It is widely recognized that deep neural networks (DNNs) are usually over-parameterized, and network pruning has been adopted to remove insignificant weights from a large neural network without hurting the accuracy. Despite its success, pruning strategies have been rarely discussed in the adversarial learning setting where the network is trained against adversarial examples, and the robustness of the network is as important as accuracy.

It is unclear what pruning methods are effective and which factors are critical for retaining model robustness. Believing that the inherited model weights may not be effective in preserving network accuracy (Ye et al. 2019; Liu et al. 2019), Ye et al. (2019) propose a concurrent adversarial training and weight pruning framework to seek a compressed robust

Editor: Sergio Escalera.

Ningyi Liao, Shufan Wang contribute equally to this work.

✉ Liyao Xiang
xiangliyao08@sjtu.edu.cn

[1] Shanghai Jiao Tong University, Shanghai, China

model. Gui et al. (2019) further incorporates pruning and several other techniques into a unified optimization framework to preserve high robustness while achieving a high compression ratio. However, the conventional three-stage 'training—pruning—fine-tuning' pipeline has not been closely examined in the adversarial context. More crucially, it is unclear which components in the network pruning methods are critical to preserving model performance. To this end, we design a comprehensive set of experiments to answer these questions.

Despite some adversarial pruning methods that have been proposed, there is still a lack of theoretical foundation to explain the working mechanism behind those methods. In fact, there are seemingly contradictory opinions on the robustness of pruned networks: Madry et al. (2018) suggest network capacity is crucial to robustness, and a wider network is more likely to obtain higher accuracy and robustness than a simple network. In contrast, Guo et al. (2018) theoretically prove that an appropriately higher weight sparsity implies stronger robustness on naturally trained models. Theories on clean training models such as the 'Lottery Ticket Hypothesis' (Frankle and Carbin 2019) point out that, a subnetwork extracted from a large network can always achieve comparable performance with the original one in the natural setting. However, it remains unknown if the hypothesis holds true for adversarially robust networks. We are motivated to explore how adversarial pruning affects the intrinsic characteristics of the network and its impact on model robustness.

In this study, we find that the robustness of the model improves as its weights become sparser. We show that weights sparsity not only includes the traditional L_0-sparsity, *i.e.*, the number of parameters retained, but also a weight distribution closer to zero, represented generally by the L_p norm of weights. These forms of sparsity can lead to robustness improvement, which is verified theoretically and experimentally. By extensive experiments on a variety of state-of-the-art pruning methods, models, and datasets, we also demonstrate that a pruned network inheriting weights from a large robust network has improved robustness than a network with the same structure but randomly initialized weights. Moreover, weight inheritance implicitly produces sparser weights distributions on adversarially pruned models.

Inspired by the connection between model sparsity and robustness, we propose a new adversarial training strategy called *inverse weights inheritance* (**IWI**): by inheriting weights from a pruned model, a large network can achieve higher robustness than being adversarially trained from scratch. The pruned model can be the 'winning ticket' of the large network, as we verify that 'Lottery Ticket Hypothesis' (Frankle and Carbin 2019) holds true in the adversarial learning context. The performance results of our proposed training strategy corroborate that sparse weights and high capacity are not contradictory, but contribute joint efforts to model robustness.

Overall, we made the following contributions. *First,* we analyze the theoretical connection between network robustness and sparsity and generalize it to L_p norm. *Second,* through comprehensive experiments, we find that weights inheritance and adversarial training are important in adversarial pruning, which implicitly provide weights sparsity. *Finally,* based on our findings, we propose a new adversarial training strategy that achieves improved robustness efficiently in large networks.

This paper is organized as follows. Section 2 introduces the related works. Preliminaries are given in Sect. 3 for a better understanding of our work. In Sect. 4, we raise the theoretical relation between sparsity and robustness. Then we report the experimental results on the one-shot pruning in Sect. 5. The following Sect. 6 explores improved means to achieve sparsity and robustness, including adversarial lottery tickets on small models, and IWI on large ones. Finally, in Sect. 7 we summarize our conclusions, and discuss possible

improvements and future work. Detailed proofs and additional discussion on experimental settings and results are presented in the appendix.

2 Related work

2.1 Network robustness and generalization

Adversarial examples (Goodfellow et al. 2015) are generated by adding imperceptible perturbations to the input which leads to performance reduction. Madry et al. (2018) motivate projected gradient descent (PGD) as a universal 'first-order adversary,' and propose the saddle point formulation to train a robust network. Adversarial training and its variants are proposed to improve network robustness against adversarial examples (Kurakin et al. 2017; Fawzi et al. 2018). Goldblum et al. (2020) observe robustness can transfer and even improve between networks by knowledge distillation.

Network robustness against adversaries also has a strong relation with generalization as revealed in Xu and Mannor (2012). Various approaches have been proposed to improve generalization capability, including optimizing loss surface (Keskar et al. 2017; Petzka et al. 2020), and imposing weights sparsity (Arora et al. 2018; Morcos et al. 2019; Bartoldson et al. 2020). There are also works on deriving the generalization bound in adversarial settings, *i.e.*, the adversarial risk bound (Fawzi et al. 2016; Balda et al. 2019), which is also evaluated empirically in varied experimental settings (Bastani et al. 2016; Moosavi-Dezfooli et al. 2016; Salman et al. 2019). In this paper, we apply distortion bound as an empirical metric to evaluate the model robustness against adversarial perturbation.

2.2 Adversarial network compression

Network compression in adversarial context has been recently discussed in search of small and robust models (Wang et al. 2018; Zhao et al. 2018; Ye et al. 2019; Sehwag et al. 2019). Several frameworks (Rakin et al. 2019; Madaan et al. 2019; Gui et al. 2019) have been proposed to adversarially train a neural network while constraining its size by pruning and/or quantization. However, these works do not answer which pruning factors are important for robust networks, nor which pruning methods are effective.

From the theoretical aspect, Hein and Andriushchenko (2017) propose a formal guarantee of adversarial robustness in terms of the local Lipschitz constant. By building a bridge between the local Lipschitz constant and weight sparsity, Guo et al. (2018) consider that an appropriately higher weight sparsity on naturally trained networks implies higher robustness. Dinh et al. (2020) also find an adversarially trained network with sparser weights distribution tends to be more robust, such as EnResNet20 (Wang et al. 2019). Different from compression, Dhillon et al. (2018) propose dynamic sparsity as an approach to improve robustness. By supplementing the concept of 'sparsity,' we found empirical evidence of the link between robustness and sparsity, as well as training strategies to boost robustness.

While Liu et al. (2019) claim that for network pruning in the natural setting, weights inherited by unstructured and predefined structured pruning may not be useful, as it may trap the pruned network to bad local minima, we show with experiments that weight inheritance improves the robustness in the adversarial setting, which we conjecture that it is because the inverse weights inheritance embraces larger networks during training, which can help the model jump out of local minima and achieve better generalization performance.

3 Preliminaries

3.1 Network robustness evaluation

Following the convention of previous literature, we adopt L_∞-PGD attack (Madry et al. 2018), *i.e.,* the strongest attack utilizing the local first-order information of the network, both in adversarial training and robustness evaluation. The PGD attack constructs adversarial examples in t iterations, with a step size s and total perturbation strength ϵ. We utilize the accuracy on the PGD perturbed testing dataset to evaluate model robustness against adversarial attacks.

The distortion bound of adversarial examples (Bastani et al. 2016; Salman et al. 2019) also serves as a robustness metric which represents the model's ability to correctly classify perturbed inputs. We estimate the distortion bound by searching the minimum ϵ of PGD attack that crafts a valid adversarial example on a given batch.

3.2 Network pruning methods

Network pruning is a set of widely implemented compression methods for its excellent performance and plasticity in practice. Pruning methods related to this paper can be divided into two categories: structured pruning and unstructured pruning. Structured pruning prunes a network at the level of filters (Lang 2018; Li et al. 2017; Luo et al. 2017), channels (Liu et al. 2017) or columns (Wen et al. 2016), depending on their respective importance. The importance of a filter or a channel can be determined by the norm of the weights (Li et al. 2017) or the channel scaling factor (Ye et al. 2018; Liu et al. 2017) (sometimes the scaling factor in batch normalization layers). The unstructured pruning (LeCun et al. 1990; Hassibi and Stork 1993) prunes at the level of individual weight according to the Hessian matrix of the loss function. Han et al. (2015) propose to prune weights with small magnitude, and the compression ratio is further enhanced in Han et al. (2016) by quantization and Huffman coding.

We pick four representative and intrinsically different pruning methods: Local Unstructured Pruning (**LUP**) (Han et al. 2016), Global Unstructured Pruning (**GUP**) (Frankle and Carbin 2019), Filter Pruning (**FP**) (Li et al. 2017) and Network Slimming (**NS**) (Liu et al. 2017). LUP and GUP are unstructured pruning, whereas FP and NS are structured pruning. Both GUP and NS prune globally according to the importance of weights or channels across all convolutional layers, while LUP and FP prune an identical percentage of weights or filters per layer locally. FP is a predefined pruning method while GUP, LUP and NS are automatic pruning methods where the structure is determined by the pruning algorithm at runtime.

3.3 Lottery ticket hypothesis

The 'Lottery Ticket Hypothesis' (Frankle and Carbin 2019) shows the existence of a sparse subnetwork (or 'winning ticket') in a randomly initialized network that can reach comparable performance with the large network. Nevertheless, Ye et al. (2019) argue against the existence of 'winning ticket' in adversarial settings. On the other hand, Cosentino et al. (2019) manage to acquire adversarial winning tickets of simple models without harming model robustness. Li et al. (2020) further propose an optimized learning rate schedule to

boost the searching performance of lottery tickets, while demonstrating why the method in Ye et al. (2019) fails to find them.

4 Study of robustness and sparsity

In this section, we theoretically prove that sparser weights distribution indicates an improved level of robustness. In the theoretical deduction, we assume DNNs with ReLU activation functions, but the conclusion can be generalized to a variety of models, as we verify by experiments.

We focus on nonlinear DNNs with ReLU activation functions for classification tasks as an example to study the connection between sparsity and robustness. Let us consider a multi-layer perceptron $g(\cdot)$ trained with labeled training datasets $\{(x_i, y_i)\}$. Each layer of the network is parameterized by a weight matrix $W_d \in \mathbb{R}^{n_{d-1} \times n_d}$ and $w_k = W_d[:, k]$ represents the weights associated with the k-th class in the final layer. σ denotes the ReLU function. Then the prediction scores of x_i for class k can be denoted as

$$g_k(x_i) = w_k^T \sigma\left(W_{d-1}^T \sigma\left(...\sigma\left(W_1^T x_i\right)\right)\right). \tag{1}$$

Let $\hat{y} = \arg\max_{k \in \{1,...,c\}} g_k(x)$ denote the class with the highest prediction score. Assuming the classifier is Lipschitz continuous, the local Lipschitz constant of function $g_{\hat{y}}(x) - g_k(x)$ over the neighborhood of x is defined as $L_{q,x}^k = \max_{x \in B_p(0,R)} \|\nabla g_{\hat{y}}(x) - \nabla g_k(x)\|$, where $B_p(x, R)$ denotes a ball centered at x with radius R under L_p norm. Previous works (Hein and Andriushchenko 2017; Guo et al. 2018) have associated robustness with the local Lipschitz constant by the following proposition:

Proposition 1 (Hein and Andriushchenko 2017; Guo et al. 2018) *Let* $\hat{y} = \arg\max_{k \in \{1,...,c\}} g_k(x)$ *and* $\frac{1}{p} + \frac{1}{q} = 1$. *For any perturbation* $\delta_x \in B_p(0, R)$, $p \in \mathbb{R}^+$ *and a set of Lipschitz continuous functions* $\{g_k : \mathbb{R}^n \mapsto \mathbb{R}\}$, *the classification decision on* $x + \delta_x$ *will not change from* $\hat{y}$ *with*

$$\|\delta_x\|_p \leq \min\left\{\min_{k \neq \hat{y}} \frac{g_{\hat{y}}(x) - g_k(x)}{L_{q,x}^k}, R\right\}, \tag{2}$$

where $L_{q,x}^k$ *is the local Lipschitz constant of function* $g_{\hat{y}}(x) - g_k(x)$.

Eqn. (2) has clearly depicted the relation between robustness and the local Lipschitz constant — a smaller $L_{q,x}^k$ represents a higher level of robustness as a larger distortion can be tolerated without changing the prediction. Guo et al. (2018) further give the relation between the local Lipschitz constant and the weights. We further deduct that the relation satisfies the following theorem:

Theorem 1 (The robustness and weights distribution of ReLU networks.) *Letting* $\frac{1}{p} + \frac{1}{q} = 1$, *for any* $x \in \mathbb{R}^n$, $k \in \{1, ..., c\}$ *and* $q \in \{1, 2\}$, *the local Lipschitz constant of function* $g_{\hat{y}}(x) - g_k(x)$ *satisfies*

$$L_{q,x}^{k} \leq \left\| w_{\hat{y}} - w_k \right\|_q \prod_{j=1}^{d-1} \left(\left\| W_j \right\|_p \right). \tag{3}$$

Note that the local Lipschitz constant is upper bounded by the product of the L_p-norm of the weights matrices. That is to say, if $\left\| W_j \right\|_p$ is small, $L_{q,x}^{k}$ is constrained to be small, leading to a higher level of robustness. The proof of Thm. 1 is omitted here due to space constraint and we refer readers to the supplementary document for the detailed proof.

We have at least two interpretations of Thm. 1: if we let $p = 0$, Eqn. (3) is bounded by the number of non-zero weights of the model, and hence the higher the proportion of non-zero weights, the more robust the model is. On the other hand, a smaller value of $\left\| W_j \right\|_p$ suggests the distribution of weights is closer to zero. This indicates that if a model has a weights distribution closer to zero, it may be more robust than other models with the same structure. We will respectively show how the two points are supported by the experimental results.

5 Adversarial pruning improves robustness by imposing sparsity

Although Thm. 1 establishes a preliminary link between sparsity and robustness, it does not tell us how to achieve sparsity and robustness by the equation. An intuitive way is to prune a network to reduce the number of non-zero weights of the model, which is also done in (Guo et al. 2018) but only in the natural setting. In this section we apply the method in adversarial contexts to show that adversarial pruning and retraining are able to improve model sparsity, hence providing robustness as Thm. 1 suggests.

5.1 Implementation details

In this part, we describe the implementation details in examining adversarially robust network pruning. To obtain objective results, we mostly follow the experimental settings in previous works (Liu et al. 2019; Yang et al. 2019; Ye et al. 2019; Zhang and Zhu 2019). Our experiments are carried out with PyTorch 1.0 on NVIDIA GeForce 2080 Ti GPUs.

5.1.1 Datasets and networks

We conduct experiments on the image classification task with datasets CIFAR-10, Tiny-ImageNet, and CIFAR-100, which are representatives for small-scale datasets, large-scale datasets, and datasets somewhere in between. Three state-of-the-art network architectures are chosen: VGG (Simonyan and Zisserman 2015), ResNet (He et al. 2016), and DenseNet (Huang et al. 2017) as the base large networks. A DenseNet-BC with depth 40 and growth rate $k = 12$ is also used on CIFAR-10.

5.1.2 One-shot pruning methods

We conduct the pruning methods defined by Sect. 3.2 in an one-shot manner that removes the parameters at one step, followed by post retraining to convergence. We re-implement each pruning methods to achieve comparable performance with that reported in the current

literature. For FP, we conduct it on every two consecutive convolutional layers and skip the shortcuts in ResNet according to (Luo et al. 2017), also it is not available on DenseNet as pruning one filter would lead to input channel changes in all subsequent layers (Li et al. 2017; Liu et al. 2019). For NS, the highest pruning ratio is selected according to the maximum channel pruning ratio to avoid the removal of layers (Liu et al. 2017).

5.1.3 Adversarial training and evaluation

We employ the widely used L_∞-PGD adversary with perturbation $\epsilon = 8/255$ and step size $s = 2/255$ in our experiments. Following recent works (Guo et al. 2020), we utilize iteration $t = 10$ for training, and evaluate robustness on $t = 100$, while other testing iterations are also included in the appendix. The distortion bound is also computed under same adversarial setting. We report the average of distortion bounds across all samples.

For all trainings, we adopt an SGD optimizer with momentum of 0.9 and weight decay of 5×10^{-4}. The batch sizes for CIFAR-10 and CIFAR-100 are both 128. On Tiny-ImageNet, the batch size is 128 for ResNet18 and 32 for DenseNet121 following Zhang and Zhu (2019) and Yang et al. (2019).

5.1.4 Stopping criteria

Typically, it is not well-defined how to train models to 'full convergence' when stepwise decaying learning rate schedule is applied. Hence we adopt two stopping criteria indicating models have been sufficiently trained for ease of comparison. **Stop-E** denotes the network is trained for a fixed number of epochs. For CIFAR-10, CIFAR-100, and Tiny-ImageNet, we set the start learning rate to be 0.1, 0.1, and 0.01, respectively. The learning rate is divided by 10 for every 1/3 of the total epochs. **Stop-C** monitors the validation loss changes to automatically adjust the learning rate. For example, when defining patience to be 10 epochs and relative threshold to be 10^{-5}, the learning rate decays only when the average validation loss does not decrease by more than 0.001% for consecutive 10 epochs. Models stop training after 2 learning rate decays.

5.2 Adversarial one-shot pruning

To obtain small and robust models, we first adversarially train each base network until reaching the state-of-the-art clean and adversarial accuracy, and then prune each network by different means. We show that pruning is able to maintain model robustness in some adversarial settings. Beyond that, adversarial retraining after pruning mostly improves robustness, at a sparser weights distribution than models with the same structure.

When networks are pruned, we immediately test their accuracies on clean and adversarial samples. The results on CIFAR-10 and Tiny-ImageNet are shown in Table 5 in the supplementary material. Compared to the performance on the base network (marked under the model name), the pruned network only moderately suffers from accuracy loss when the pruning ratio is not very high, and the pruning methods are automatic, *i.e.,* LUP, GUP, and NS. Note that those methods automatically extract a subnetwork without altering the weights, resulting in a higher L_0 sparsity compared to base models. Considering pruning reduces model capacity, the mild performance loss is reasonable.

Although pruning shows a promising method to introduce sparsity, it does not end up in robust models each time. We hence impose adversarial retraining on pruned networks to

enhance robustness. The results are provided in Table 1. Since there is a tradeoff between accuracy and robustness (Zhang et al. 2019), and some models tend to sacrifice one for the other, we choose to report the performance where the sum of adversarial accuracy and clean accuracy is the highest. Distortion bound is also reported for a complete view. We refer readers to the supplementary material for further discussions on the results.

Most networks in Table 1 obtain higher accuracy and robustness than pruning without retraining, and a large proportion of them can achieve better performance than the base networks. Specifically, LUP and NS only suffer notable performance degradation at high pruning ratios, whereas GUP remains a remarkable high performance across all pruning ratios. FP cannot preserve network performance well.

To see whether the weights inherited from a large network truly help the pruned network converge, we conduct a series of comparison experiments, as shown in Table 2. Compared to FP, FP-rand initializes a small network with the same structure as the corresponding pruned network. For automatic pruning methods including LUP, GUP, and NS, we re-use the pruned network structure with re-initialized weights. As we found, compared with FP-rand, FP provides little or no improvement with the inherited weights. On the contrary, automatic pruning with inherited weights almost always performs better than that with randomly initialized weights.

We evaluate the relation between model robustness and model norm according to Eqn. (2) and (3). In Fig. 3, we demonstrate the changes of model L_2 norm brought by adversarial retraining after pruning at different pruning ratios for GUP. We also plot the empirical model norms and their adversarial distortion bounds. The latter serves as an indicator for the robustness. We found the results are in accord with Eqn. (2) and (3) that a smaller model norm usually indicates a higher upper bound of robustness.

Although Table 1 and Table 2 experimentally found effective methods or factors to gain robustness for pruned networks, it still remains unclear how it relates to sparsity. Interestingly, by examining the weights distribution after adversarial retraining, we found that most automatically pruned networks with inherited weights have similar or higher sparsity than those with randomly initialized weights, as some examples shown in Fig. 2, while the networks pruned by predefined pruning (FP) show the opposite trend. This could be explained by Thm. 1, since a weight distribution closer to zero implies higher robustness. Therefore, weight inheritance and adversarial retraining implicitly provide a way to obtain sparse networks.

5.2.1 Comparison with previous results

We also compare our conclusion with previous works and summarize the difference as follows. We find inherited weights by automatic pruning (LUP, GUP, NS) provide better initialization for small networks, while predefined pruning does not. Liu et al. (2019) argue that weights inherited from structured pruning have little impact on the performance of the pruned network. While the experiments on FP agree with the conclusion, that on NS does not. Wang et al. (2018) also suggests inherited weights are important to preserving network accuracy and robustness in adversarial settings, but they do not discuss the working mechanism behind.

6 Lottery tickets and inverse weights inheritance

Although adversarial one-shot pruning produce convincing results, it lacks stability in scenarios such as where the pruning ratio is high. In this section, we aim to explore improved methods to obtain sparse and robust neural networks beyond adversarial pruning and retraining, on both small and large network architectures.

Algorithm 1 Lottery Ticket in Adversarial Settings

Input: A large network $f(x; \theta_0 \odot M_0)$ where x is the input, θ_0 is the randomly initialized weights, and $M_0 = 1^{|\theta_0|}$ denoting weight masks.
Iterative pruning ratio $p\%$.
Pruning iteration K.
Training epochs N per pruning iteration.
Output: A winning ticket $f(x; \theta_0 \odot M_K)$.
1: **for** k in $\{1, \ldots, K\}$ **do**
2: Conduct adversarial training on $f(x; \theta_0 \odot M_{k-1})$ for N epochs and obtain the network $f(x; \theta_k \odot M_{k-1})$.
3: Prune $p\%$ weights from the current network and obtain a new weights mask M_k.
4: Re-initialize weights of f as $f(x; \theta_0 \odot M_k)$.
5: **end for**
6: **return** $f(x; \theta_0 \odot M_K)$.

6.1 Lottery tickets in adversarial settings

We seek that in a randomly-initialized large network, if a subnetwork exists achieving comparable robustness as the large one, which is also known as the 'winning ticket' in Frankle and Carbin (2019) in a natural setting. More specifically, we perform Alg. 1 to find out the 'winning ticket' in the adversarial setting. A discussion of hyperparameters can be found in the supplementary material.

The results on CIFAR-10 and CIFAR-100 are displayed in Table 3. On ResNet18 and VGG16 trained on CIFAR-10, no noticeable performance degradation occurs when the pruning ratio is as high as 80%. This is slightly different from pruning on natural models (Frankle and Carbin 2019), where accuracies do not drop until pruning ratio reaches around 88.2% and 92% respectively on Resnet18 and VGG16. We think the difference may be explained by the more complicated decision boundary of a robust model (Fawzi et al. 2016), and hence its 'winning ticket' requires a higher capacity.

To ensure our conclusion holds against randomness, we perform repeated experiments in representative settings where the average and standard deviation of 5 trials are shown in Fig. 4. The former conclusion remains valid that both clean and adversarial accuracy almost do not decrease from baseline when $p \leq 80\%$.

To better understand lottery tickets in adversarial settings, we compare the weights distribution between one-shot pruned model and the winning ticket at the same pruning ratio. Fig. 5 illustrates the example of two models pruned at the same pruning ratio by GUP and Alg. 1 respectively on CIFAR-10, with adversarial accuracy 47.09% versus 47.36% on ResNet18, and 44.36% versus 45.15% on VGG16, correspondingly. As we observe,

whereas GUP models tend to have a flatter distribution which is consistent with Ye et al. (2019), the winning tickets have more near-zero valued weights, indicating a higher level of sparsity. To further demonstrate the relation between the L_2 norm sparsity and adversarial robustness, we also display the results for lottery tickets in Fig. 3 to compare with GUP. It can be observed that the adversarial lottery tickets produces slightly lower weight norms than one-shot GUP, while enjoying a higher distortion bound, which agrees with the theoretical resutls in Eqn. (2) and (3). Thus we conclude that it is able to achieve preferable adversarial robustness through the lottery tickets settings.

6.1.1 Comparison with previous results

Ye et al. (2019) argue against the existence of 'winning ticket' in adversarial settings. Nevertheless, through experiments we show that 'winning ticket' exists in adversarial settings and can be obtained efficiently with a few rounds of pruning and less retraining. Our conclusion is different mostly because we search 'winning ticket' by iterative global unstructured pruning as in Frankle and Carbin (2019), while Ye et al. (2019) use a layer-wise pruning method. As indicated in Frankle and Carbin (2019), layers with fewer parameters may become bottlenecks under a layer-wise pruning method, and thus winning tickets fail to emerge. We also compare our work with Li et al. (2020), and find the few-shot pruning in Li et al. (2020) does not outperform iterative pruning results in our setting.

We also plot the results in Table 1 and Table 3 by showing the relation between the number of remaining parameters of the pruned models against the adversarial testing accuracy in Fig. 1. By comparing with recent works including RobNet (Guo et al. 2020) and ATMC (Gui et al. 2019) utilizing the same training and testing metrics, which is PGD10 and PGD100, respectively, we demonstrate that our approach is able to acquire smaller networks with robustness comparable to the original dense models through adversarial network pruning, extensively effective under different current model structures among ResNet, VGG, and DenseNet.

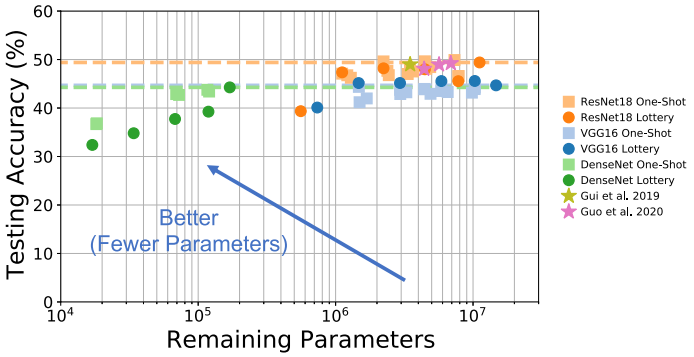

Fig. 1 The relation between parameter numbers and adversarial robustness in our approach and the state-of-the-art methods on different architectures. The dotted lines represent the baselines of three base (large) models. Models residing at the upper left corner have higher adversarial accuracies and smaller sizes. All models on CIFAR-10 are adversarially trained by PGD with $\epsilon = 8/255$ and $t = 10$ steps, and evaluated by PGD attack of $\epsilon = 8/255$ and $t = 100$ steps. We also mark the results Guo et al. (2020) and Gui et al. (2019) by stars in the same settings. Our experiments show that adversarial pruning methods are effective in obtaining networks of smaller parameters with comparable or even better robustness than the baselines

Table 1 Clean testing accuracy/adversarial testing accuracy (in %)/distortion lower bound of pruned networks with adversarial retraining

Network	p%	LUP	GUP	FP	NS
(a) One-Shot Pruning on CIFAR-10 w/ Stop-E					
ResNet18 (82.84/49.40/2.519)	30	**82.13/49.90/3.221**	**81.92**/46.56/2.402	**83.62**/46.61/2.505	**84.18/49.92**/2.023
	60	82.21/48.44/**2.777**	**84.73/49.64/2.612**	82.61/48.08/2.501	**83.57/49.46/2.666**
	90	80.09/46.76/1.533	**83.89**/47.09/**2.940**	78.87/46.24/1.764	-
VGG1 (78.57/44.68/3.471)	30	**79.81**/43.17/1.982	**80.43**/44.24/1.630	77.05/43.91/3.002	**80.10**/43.81/2.991
	60	**78.78**/43.30/2.136	**80.26**/43.51/2.275	77.13/44.21/2.032	**79.56**/44.29/2.607
	90	72.1/41.98/2.510	**79.83**/44.36/2.501	69.38/41.20/2.270	**79.54**/43.76/2.443
DenseNet-BC (76.01/44.26/1.109)	30	74.42/43.76/**1.525**	74.68/43.40/**2.928**	-	73.86/43.08/**2.572**
	60	73.16/42.70/**1.734**	73.24/42.88/**1.781**	-	66.33/37.54/1.059
	90	63.15/36.68/**2.000**	65.19/36.85/**1.784**	-	-
(b) One-Shot Pruning on Tiny-ImageNet w/ Stop-C					
ResNet18 (41.94/14.43/2.594)	30	**42.72/14.87**/2.356	**43.18/15.82/2.713**	**42.68/14.91/3.300**	**41.89/15.23**/2.397
	60	**42.28/15.51/3.022**	**42.80/16.12/3.272**	40.88/**15.87**/1.172	37.92/14.11/**3.250**
	90	40.32/**16.11**/2.581	**42.21/17.26/2.797**	36.79/14.43/1.819	-
DenseNet121 (49.48/19.65/1.922)	30	**48.86/20.03/3.616**	**48.19/20.52/2.519**	-	46.43/**19.71**/1.597
	60	**48.63/19.98/2.300**	**48.96/19.92/1.984**	-	40.82/16.51/**3.334**
	90	45.72/18.75/1.722	46.99/18.65/1.478	-	-

Accuracy and distortion bounds higher than the base model are in bold

Table 2 Clean testing accuracy/adversarial testing accuracy (in %) of scratch networks that shares the same structure with the corresponding pruned network, with the weights randomly initialized.

One-Shot Pruning w/ Randomly Initialized Weights on CIFAR-10 w/ Stop-E					
Network	p%	LUP-rand	GUP-rand	FP-rand	NS-rand
ResNet18	30	81.91/46.54	81.41/48.78	82.24/46.58	**83.21**/46.28
	60	82.58/46.40	81.91/47.46	80.64/46.22	82.61/46.04
	90	78.36/44.97	81.99/46.90	80.34/45.50	–
VGG16	30	**79.67**/42.37	**79.56/45.26**	**80.27**/44.38	**78.68**/43.52
	60	78.26/43.79	**80.23/45.22**	78.52/44.21	**78.77**/43.74
	90	72.32/41.38	77.78/43.97	74.24/42.39	77.37/43.52

Accuracy higher than the base model is in bold

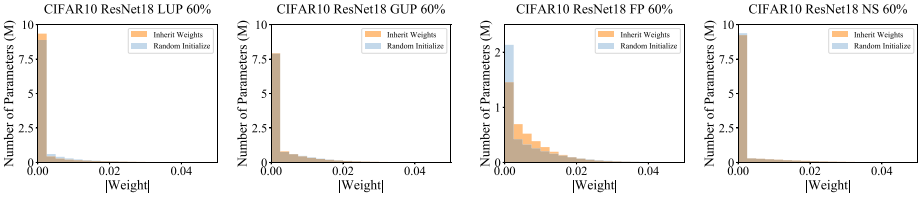

Fig. 2 Weights distribution of the pruned network adversarially trained with inherited weights or randomly initialized weights. In general, networks with inherited weights from automatic pruning methods including LUP, GUP, NS have an equivalent or higher sparsity than their counterparts with randomly initialized weights. FP has lower sparsity than FP-rand

Fig. 3 Effect on model norms by retraining at different pruning ratios, and the practical relation between model norms and adversarial distortion bound

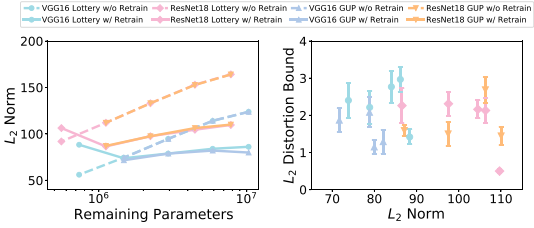

Table 3 Clean testing accuracy/adversarial testing accuracy (in %) of adversarially trained 'winning ticket.' $p\%$ is the pruning ratio

Winning Tickets on CIFAR-10 and CIFAR-100 w/ Stop-E			
Network	$p\%$	CIFAR-10	CIFAR-100
ResNet18	0 (baseline)	82.84/49.40	50.50/21.13
	30 (30×1 iter)	**84.29**/45.54	**50.62/21.72**
	60 (20×3 iter)	**84.03**/47.99	**52.68/21.54**
	80 (20×4 iter)	**82.83**/48.19	**52.23**/20.80
	90 (30×3 iter)	81.41/47.36	49.43/21.27
	95 (31.7×3 iter)	78.43/46.49	-
VGG16	0 (baseline)	78.57/44.68	44.44/18.86
	30 (30×1 iter)	**80.90/45.58**	42.21/**19.16**
	60 (20×3 iter)	**80.05/45.56**	42.65/**19.12**
	80 (20×4 iter)	**79.30/45.16**	45.90/**18.93**
	90 (30×3 iter)	**78.87/45.15**	45.76/**18.89**
	95 (31.7×3 iter)	77.23/45.02	-

'60 (20×3 iter)' means iteratively remove 20% of the weights in each for 3 iterations to achieve a final pruning ratio of 60%. Each iteration of pruning is preceded by 1 epoch of training, and the total training epoch is 240. Accuracy higher than the base model is in bold

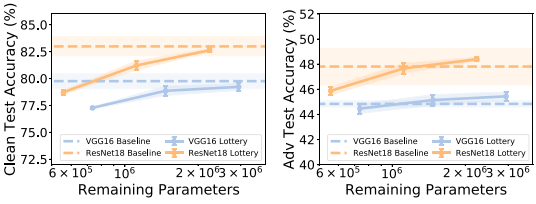

Fig. 4 Repeated results of clean testing accuracy/adversarial testing accuracy and deviation (in %) of adversarially trained 'winning ticket' on CIFAR-10 with Stop-E corresponding to $p = 80, 90, 95\%$. The dotted lines represent baselines

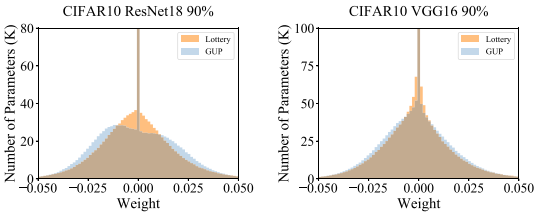

Fig. 5 Weights distribution example of the pruned network obtained by one-shot GUP and adversarial lottery at the same pruning ratio. The distribution indicates that the adversarial winning tickets have higher sparsity than corresponding GUP pruned models

Algorithm 2 Inverse Weights Inheritance (w/ Lottery Ticket)

Input: $\{f(x;\theta_0 \odot M_0)$, $p\%$, K, $N\}$ same as in Alg. 1.
Adversarial fine-tuning epochs N_f.
Output: A robust network $f\left(x;\theta''\right)$.
1: Find the winning ticket $f(x;\theta_0 \odot M_K)$ by Alg. 1.
2: Fine-tune the 'winning ticket' adversarially for N_f epochs, obtain a robust small network $g\left(x;\theta' \odot M_K\right)$.
3: Load the weights of the pruned network g to the corresponding place in the large network f and obtain $f\left(x;\theta' \odot M_K\right)$.
4: Re-initialize remaining weights of f as $f\left(x;\theta' \odot M_K + \theta_0 \odot (M_0 - M_K)\right)$.
5: Train f until stopping criterion meets, obtain $f\left(x,\theta''\right)$.
6: **return** $f\left(x;\theta''\right)$.

6.2 Inverse weights inheritance

According to our experimental results in one-shot adversarial pruning, it seems that networks with smaller capacities (higher L_0-sparsity) can also have an equivalent or even higher accuracy and robustness than large networks. This appears to be contradictory to the conclusion in Madry et al. (2018) that classifying examples in a robust way requires the model to have a larger capacity, as the decision boundary is more complicated. We ask the question that, *can a network be sparse and have larger capacity at the same time?* As we analyze, it is indeed possible to have such networks with superior performance.

We introduce a new training strategy called *inverse weights inheritance* (**IWI**), which is inspired by Thm. 1 and adversarial network pruning results. By the strategy, a large network acquires sparse weights distribution by inheriting weights from a small robust network, which is pruned from the same large network in the first place and is adversarially trained. Alg. 2 gives an example of using the lottery ticket to obtain such a small network. For a fair comparison, we train the base networks with Stop-C and Stop-E (240 epochs) and report the one with higher performance. To train the large network with inherited weights, we first run Alg. 1 to obtain the 'winning ticket' and then train the 'winning ticket' (a small network) for 120 epochs. Then the weights of the trained 'winning ticket' are loaded back to the large network to train for another 45 epochs (Stop-E) or until convergence (Stop-C). In Table 4, the large network with inherited weights not only outperforms the 'winning ticket' but also exceeds the base network.

We perform repeated experiments and display the average results on CIFAR-10 with Stop-E as shown in Fig. 6. We also provide the baselines and winning tickets results for clearer comparison. Significance test shows that the accuracy increase brought by IWI has p-value $p < 0.01$ in T-test compared with winning tickets for both architectures, and $p < 0.01$ over VGG16 baselines. Thus we conclude our improvements are valid against randomness.

To find out the reason, we measure the weight distributions of each network and partial results are given in Fig. 7. We conclude from the figure that, with inherited weights as initialization, the distribution of the final weights for the large networks is sparser (closer to zero) than those with random initialization, which is in accord with Thm. 1. The results suggest that for networks with the same structure, IWI implicitly finds sparse weights distribution for the large networks, and the network can achieve an improved level of clean and adversarial accuracies. Moreover, it is evident that those networks are sparse and have large capacities at the same time.

Table 4 Clean testing accuracy/adversarial testing accuracy (in %)/distortion lower bound of inverse weights inherited networks

Network	$p\%$	Stop-C	Stop-E
(a) Inverse Weights Inheritance on CIFAR-10			
ResNet18 82.84/49.40/2.519	80	**84.05/50.30/2.728**	**83.14/49.59**/2.303
	90	**83.56/49.89/2.819**	81.68/49.03/**2.662**
	95	**84.60**/49.34/**2.659**	**83.19**/48.93/1.734
VGG16 78.57/44.68/3.471	80	**81.21/47.38**/2.338	**81.15/47.46**/1.600
	90	**81.36/47.54**/2.597	**80.74/47.53**/2.622
	95	**81.29/46.98**/3.341	**80.68/47.59**/3.219
(b) Inverse Weights Inheritance on CIFAR-100			
ResNet18 50.50/21.13/3.047	30	**53.49/22.07/3.872**	**51.57**/20.68/1.191
	60	**52.98/21.65/3.197**	**50.74/21.28/3.472**
	80	50.06/21.03/1.900	**50.78/21.15**/1.881
	90	**52.91/21.53**/2.812	50.16/**21.39**/2.056
VGG16 44.44/18.86/2.338	30	**47.18/18.91**/1.491	**44.79/19.10**/1.534
	60	**45.97/19.38**/1.147	**45.16**/18.68/1.191
	80	**46.51/19.22/2.631**	43.79/**18.90/2.759**
	90	**47.64/19.26/3.169**	43.30/**18.94**/2.000

Performance of base networks are marked under the model name. The performance of the inherited Winning Tickets is shown in Table 3. Accuracy and distortion bounds higher than the base model are in bold

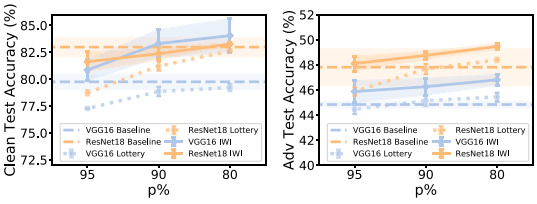

Fig. 6 Multiple-run results of clean testing accuracy/adversarial testing accuracy and deviation (in %) of IWI on CIFAR-10 with Stop-E corresponding to $p = 80, 90, 95\%$. The dash lines represent baselines and the dotted lines are lottery ticket results

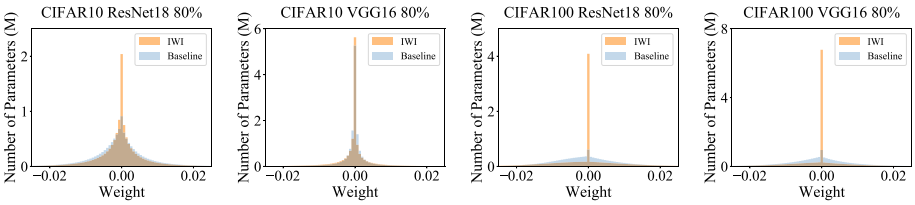

Fig. 7 Weights distribution of large networks trained by inverse weights inheritance (IWI) and adversarially trained with random initialization (Baseline). Both are trained for 240 epochs (Stop-E). Net XX% denotes a large network trained by inheriting the weights of a 'winning ticket' with XX% weights pruned

Beyond performance boost, IWI also accelerates the adversarial training process, mainly due to the lower expense of adversarially training a small network, and less training epochs required after the large network inheriting weights. The detail of training effort saved can be found in the supplementary material. We have also tried other methods, such as using an

additional regularization term to impose sparsity in large networks, but it failed. Interested readers may refer to the supplementary material for more details.

7 Conclusions, limitations and future work

In this work, we conducted comprehensive studies on adversarial network pruning. The contributions are three-fold: First, we give a novel theoretical explanation on the connection between adversarial robustness and network sparsity, which is supported by substantial empirical evidence. Second, we demonstrate the efficacy of training network with robustness via our proposed algorithm including one-shot pruning and searching the 'winning ticket.' Third, we design a new adversarial training strategy to achieve sparsity and large capacity at the same time for robustness.

The limitations of the work lie in the following aspects. We specifically look into the DNN models with ReLU activation functions when developing the theory of sparsity and robustness. The theory may be applied on general DNN models as experimental evidence supports it in Sect. 5. Meanwhile, due to a lack of specific sparsity methods directly optimizing model norm by Thm. 1, we only apply model pruning and fine-tuning as such approaches achieve sparsity intuitively.

Our results may trigger some interesting future studies. First, experimental observation shows that applying different sparsity methods on the same network structure leads to different weights sparsity, consequently affecting model robustness. Our findings and explanations may support research on how to design approaches to better manipulate weights distribution, and therefore form principled adversarial defences. Also, the relation of robustness, generalization, and pruning is a popular topic in many studies (Xu and Mannor 2012; Morcos et al. 2019; Arora et al. 2018). The discussion on whether and how our work on sparsity and robustness will improve model generalization capability would be meaningful to that line of research. In addition, as adversarial attacks emerge on deep learning tasks such as NLP (Miyato et al. 2017; Alzantot et al. 2018), it offers the possibility to extend our theory and experiments to other types of data and models for further exploring the relation between sparsity and robustness.

Appendix

Proofs

Proof of Thm. 1

Let us denote layer-wise activation output as $a_0 = x$ and $a_j = \sigma\left(W_j^T a_{j-1}\right)$ for $0 \leq j \leq d-1$. We also define D_j as

$$D_j(x) := diag\left(1_{W_j[:,1]^T a_{j-1}>0}, ..., 1_{W_j[:,n_j]^T a_{j-1}>0}\right) \tag{4}$$

which is a diagonal matrix whose entries will take value one when the activation is nonzero within j-th layer. Note that if for any x and any $j \in \{1, ..., d-2\}$, $D_j(x) = 0^{n_j \times n_j}$,

$D_{d-1}(x) = 0^{n_{d-1} \times n_{d-1}}$ must hold. In this setting, we have Lemma 1 proved by (Guo et al. 2018).

Lemma 1 (A local Lipschitz constant for ReLU networks (Guo et al. 2018)) *Letting* $\frac{1}{p} + \frac{1}{q} = 1$, *for any* $x \in \mathbb{R}^n$, $k \in \{1, ..., c\}$ *and* $q \in \{1, 2\}$, *the local Lipschitz constant of functiong*$_{\hat{y}}(x) - g_k(x)$ *satisfies*

$$L_{q,x}^k \leq \left\| w_{\hat{y}} - w_k \right\|_q \sup_{x' \in B_p(x,R)} \prod_{j=1}^{d-1} \left(\left\| D_j(x') \right\|_p \left\| W_j \right\|_p \right) \tag{5}$$

where all the matrix norms are induced norms.

In our settings, the network is pruned to avoid layer removal and thereby for any $j \in [0, d-1]$, W_j cannot be all-zero matrices. Then we assert that, for any $j \in [0, d-1]$, there must be at least one a_j which is not an all-zero matrix. We prove by contradiction. Assume for some layer j, a_j is an all-zero matrix. Then $D_j(x) = 0^{n_{d-1} \times n_{d-1}}$ which yields $D_{d-1}(x) = 0^{n_{d-1} \times n_{d-1}}$ and the prediction score $g_k(x)$ will be all-zero for any k since

$$g_k(x_i) = w_k^T \sigma\left(W_{d-1}^T \sigma\left(...\sigma\left(W_1^T x_i\right)\right)\right). \tag{6}$$

For a well-trained network that make valid predictions, it is impossible to have all-zero prediction scores. Hence the assumption is not true for the network in our settings and $a_j, \forall j \in [0, d-1]$ cannot be an all-zero matrix. The conclusion implies that, for any $j \in [0, d-1]$, there must be at least one entry in $D_j(x)$ taking the value 1. By the definition of matrix induced norms, we have that for any x and particular values of p,

$$\left\| D_j(x) \right\|_2 := \sigma_{\max}(D_j) = 1 \tag{7}$$

$$\left\| D_j(x) \right\|_\infty := \max_{1 \leq i \leq n_j} \sum_{m=1}^{n_j} |D_j[i, m]| = 1 \tag{8}$$

where $\left\| D_j(x) \right\|_2$ is the spectral norm, the greatest singular value of $D_j(x)$. Then for $p = 2$ and ∞, the right side of Eqn. (5) is independent of input x and we have the relation between the Lipschitz constant and weight sparsity described by Thm. 1.

Experiments of One-Shot Pruning

Pruned Networks Without Retraining

See Table 5

In this section, we report the testing accuracy on clean and adversarial samples of networks after they are pruned and without retraining. The corresponding results to Table 1 on CIFAR-10 and Tiny-ImageNet of PGD-10 attack are shown in Table 5. Compared to performance on the base network (marked under the model name), the pruned network only moderately suffers from accuracy loss when the pruning ratio is not very high, and the pruning methods are automatic, *i.e.,* LUP, GUP, and NS. Note that those methods

Table 5 Clean testing accuracy/adversarial testing accuracy (in %) of pruned networks without retraining

Network	$p\%$	LUP	GUP	FP	NS
(a) One-Shot Pruning on CIFAR-10 w/o Retraining					
ResNet18 (82.84/50.87)	30	79.58/50.21	82.83/50.83	21.05/15.30	80.39/39.38
	60	59.10/33.17	**82.89**/50.63	12.28/8.47	10.00/10.00
	90	10.00/10.00	54.18/28.62	10.00/10.00	–
VGG16 (78.57/48.04)	30	77.59/47.75	**78.58**/48.03	10.00/10.00	**78.57/48.33**
	60	66.59/38.63	**78.69/48.50**	10.00/10.00	76.95/47.92
	90	10.00/10.00	58.41/36.26	10.00/10.00	10.00/10.00
(b) One-Shot Pruning on Tiny-ImageNet w/o Retraining					
ResNet18 (41.94/16.04)	30	33.44/15.36	41.32/**17.57**	0.73/0.46	0.61/0.31
	60	6.59/2.56	38.67/**16.13**	0.61/0.60	0.50/0.50
	90	0.50/0.50	0.97/0.56	0.50/0.50	–
VGG16 (49.48/19.90)	30	43.71/18.51	47.95/**20.12**	–	46.43/**20.57**
	60	6.35/2.21	42.39/16.78	–	40.82/17.43
	90	0.58/0.53	1.11/0.51	–	–

$p\%$ denotes the pruning ratio. The accuracy of base networks by adversarial training are besides the model name. Accuracy higher than the base model is in bold

automatically extract a subnetwork without altering the weights, resulting in a higher L_0 sparsity compared to base models. Considering pruning reduces model capacity, the mild performance loss is reasonable.

Complete Results of One-Shot Pruning

See Table 6

We extend Table 1 to additionally include results on three different datasets, *i.e.*, CIFAR-10, CIFAR-100, and Tiny-ImageNet, and their corresponding models. We evaluate the adversarial robustness by PGD of 10, 50, and 100 iteration steps. The complete testing accuracies are shown in Table 6. We conclude from the results that most of our observations in the paper hold true for different evaluation metrics, in particular under the strong attack of PGD-100.

Trade-off between Accuracy and Robustness

As it has been pointed out that a trade-off exists between accuracy and robustness (Zhang et al. 2019), we report results when the sum of the clean testing accuracy plus the adversarial testing accuracy is the highest. This is because by the conclusion of Zhang et al. (2019), the sum of risks on both natural examples and adversarial examples is bounded. Throughout our experiments, we have observed this trade-off. For a better view, we depict all the adversarial accuracy-clean accuracy result pairs near the end of training where the models nearly or already converge. We observe three typical accuracy distributions shown in Fig. 8. Note that all results in the figure are consistent with that of Table 2 in the paper.

Fig. 8a displays the cases where the pruning method fails to retain the performance of the base network, where we observe both clean and adversarial accuracy decline as the

Table 6 Clean/PGD-10/PGD-50/PGD-100 testing accuracy (in %) of pruned networks with adversarial retraining

Network	p%	LUP	GUP	FP	NS
(a) One-Shot Pruning on CIFAR-10 w/ Stop-E					
ResNet18 82.84/50.87/49.50/49.40	30	82.13/**51.09**/49.16/**49.90**	**84.20**/49.64/46.77/46.56	**83.62**/50.00/46.65/46.61	**84.18**/49.44/**50.08/49.92**
	60	82.21/50.64/48.47/48.44	**84.73**/49.28/**49.77/49.64**	82.61/49.37/48.19/48.08	**83.57**/49.32/**49.51/49.46**
	70	**83.41**/49.52/47.68/47.54	**84.16**/49.94/47.08/47.02	82.12/49.11/47.92/47.82	82.31/47.73/45.53/45.61
	80	82.48/48.76/46.91/46.78	**83.91**/49.87/**49.80/49.62**	81.22/48.10/47.71/47.67	73.29/43.27/42.74/42.64
	90	80.09/47.03/46.78/46.76	**83.89**/49.74/47.22/47.09	78.87/46.65/46.36/46.24	-
VGG16 78.57/48.40/44.77/44.68	30	**79.81**/46.54/43.30/43.17	**80.43**/45.53/44.30/44.24	77.05/45.17/44.00/43.91	**80.10**/46.31/43.84/43.81
	60	**78.78**/45.66/43.50/43.30	**80.26**/46.33/43.48/43.51	77.13/45.18/44.33/44.21	**79.56**/46.24/44.34/44.29
	70	78.20/45.37/43.03/42.93	**79.86**/46.43/43.99/43.94	75.22/45.06/43.80/43.87	**79.56**/46.74/**44.85/44.70**
	80	77.03/44.55/43.26/43.25	**79.72**/46.96/43.80/43.75	73.53/43.74/43.02/42.89	**79.38**/46.06/43.37/43.36
	90	72.10/42.53/42.08/41.98	**79.83**/46.43/44.44/44.36	69.38/42.50/41.21/41.20	**79.54**/45.66/43.80/43.76
DenseNet-BC 76.01/45.23/44.26/44.26	30	74.42/44.72/43.75/43.76	74.68/44.37/43.57/43.40	-	73.86/43.95/43.04/43.08
	60	73.16/43.77/42.84/42.70	73.24/44.00/42.97/42.88	-	66.33/38.15/37.57/37.54
	90	63.15/37.10/36.68/36.68	65.19/37.55/36.85/36.85	-	-
(b) One-Shot Pruning on CIFAR-100 w/ Stop-E					
ResNet18 50.50/21.71/21.10/21.10	30	**52.31/22.10/21.50/21.46**	**52.55/22.25/21.70/21.75**	**52.34/22.22/21.54/21.55**	**52.23/21.92/21.40/21.41**
	60	**51.98/22.16/21.41/21.31**	50.18/**22.05/21.59/21.53**	**51.23**/21.70/21.06/21.04	46.02/19.56/18.73/18.75
	70	**51.69/21.78/21.29/21.32**	49.87/21.69/20.96/20.97	49.49/21.30/20.61/20.56	-
	80	**51.25**/21.36/20.94/20.82	**52.54**/21.60/20.87/20.67	48.05/21.08/20.19/20.25	-
	90	47.65/20.54/19.86/19.68	**51.76/21.92/21.34/21.31**	44.39/20.41/19.94/19.80	-
VGG16 44.44/19.29/18.81/18.84	30	**46.46/19.51/19.11/19.01**	**45.91/19.77/19.21/19.21**	**44.58**/18.84/18.39/18.35	**46.49/19.30/18.85**/18.79
	60	**46.38/19.64/19.16/19.10**	**46.72/19.58/19.15/19.15**	42.71/18.05/17.29/17.31	**45.56**/19.12/18.57/18.60
	70	**46.16/19.46/18.88/18.86**	**47.04/19.78/19.08/19.09**	40.94/**18.12**/17.53/17.51	44.33/18.79/18.39/18.36
	80	**44.74/19.53**/18.81/18.72	**46.54/19.53/18.92/18.86**	38.61/17.94/17.38/17.45	39.59/17.80/17.29/17.17
	90	42.42/18.61/18.30/18.21	**45.73**/18.61/18.80/**18.87**	33.96/16.68/16.38/16.34	26.55/14.59/14.34/14.32

Table 6 (continued)

Network	$p\%$	LUP	GUP	FP	NS
(c) One-Shot Pruning on Tiny-ImageNet w/ Stop-C					
ResNet18 41.94/16.04/17.10/14.43	30	**42.72**/15.56/14.96/**14.87**	**43.18/16.70**/15.83/**15.82**	**42.68**/15.74/14.94/**14.91**	41.89/15.84/15.28/**15.23**
	60	**42.28**/15.98/15.50/**15.51**	**42.80/16.84**/16.10/**16.12**	40.88/**16.34**/15.91/**15.87**	37.92/14.39/14.10/14.11
	90	40.32/**16.43**/16.14/**16.11**	**42.21/18.11/17.32/17.26**	36.79/15.13/14.39/14.43	-
DenseNet121 49.48/19.90/19.70/19.65	30	48.86/**20.83/20.11/20.03**	48.19/**21.30/20.51/20.52**	-	46.43/**20.57/19.76/19.71**
	60	48.63/**20.90/20.01/19.98**	48.96/**20.78/19.95/19.92**	-	40.82/17.43/16.48/16.51
	90	45.72/**20.61**/18.84/18.75	46.99/18.99/18.65/18.65	-	-

Accuracy higher than the base model is in bold

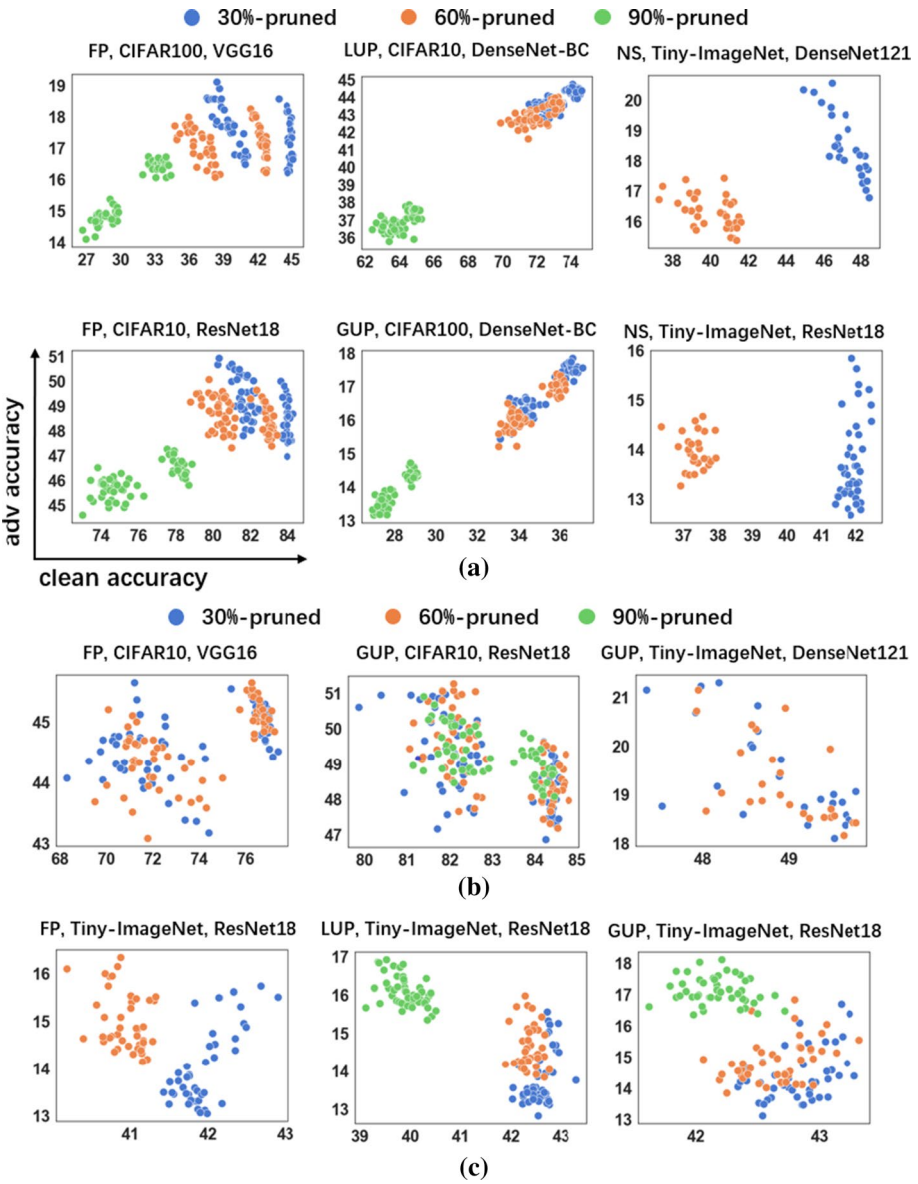

Fig. 8 Trade-off between clean and adversarial accuracy of pruned robust networks at the final phase of adversarial retraining

pruning ratio increases. FP in general cannot preserve model performance, whereas NS cannot retain performance at high pruning ratios. DenseNet-BC is too small with limited capacity, and thus its performance significantly decreases after pruning.

Fig. 8b shows successful pruning cases where both accuracies barely decrease as the pruning ratio increases. Fig. 8c shows an interesting phenomenon that, as the pruning ratio gets higher, the clean accuracy decreases while the adversarial accuracy increases,

but the sum accuracy does not decrease at all. The phenomenon is particularly notable on Tiny-ImageNet when trained on ResNet18, as shown in the figure. We think it is because ResNet18 has limited capacity to express a robust decision boundary on a complicated dataset like Tiny-ImageNet. Therefore with adversarial training, the adversarial accuracy increases at the sacrifice of clean accuracy.

Hence we argue that when discussing model robustness, it is not fair to only state adversarial accuracies. Clean and adversarial accuracies are two sides of a coin, the combination of which together describe the model performance.

Experiments of L_2-regularization

See Table 7 and Fig. 9

We also tried other means to impose sparser weights distribution in a large network in the hope of enhancing robustness, such as imposing L_2-regularization during adversarial training. The method is also known as weight decay. In the default setting of the paper, we use an SGD optimizer with weight decay $= 5 \times 10^{-4}$. In the supplementary experiments, we conduct adversarial training with weigh decay $= \alpha \times 5 \times 10^{-4}$ respectively.

The results and weights distribution under different weight decays are shown in Table 7 and Fig. 9. From Fig. 9, we can tell weight decay indeed results in sparser weights distributions. Note that different from adversarial pruning methods in the paper, weight decay does not actually yield smaller networks as most parameters are not zeros. Although the weights distributions are close to zero, the models trained by L_2-regularization do not have improved robustness, sharing a similar conclusion with (Goodfellow et al. 2015). Hence L_2 -regularization is not a successful pruning method for robust networks.

Table 7 Clean testing accuracy/ PGD-10 adversarial testing accuracy (in %) of adversarially trained networks with weight decay

Adversarially Trained w/ Weight Decay on CIFAR-10						
Weight Decay	α	1	2	3	4	5
ResNet18	clean acc	82.84	80.94	**82.91**	82.03	80.49
	adv acc	**50.87**	48.84	49.10	49.24	48.45
VGG16	clean acc	78.57	**79.74**	78.77	77.31	75.80
	adv acc	**48.04**	46.16	45.94	45.73	44.66

The higher the weight decay, the lower the robustness. Highest accuracy is in bold

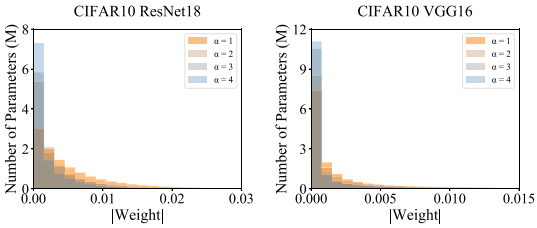

Fig. 9 Weights distribution of adversarially trained networks with different weight decays

Experiments of Adversarial Lottery Ticket

Training Hyperparameters Search

See Fig. 10

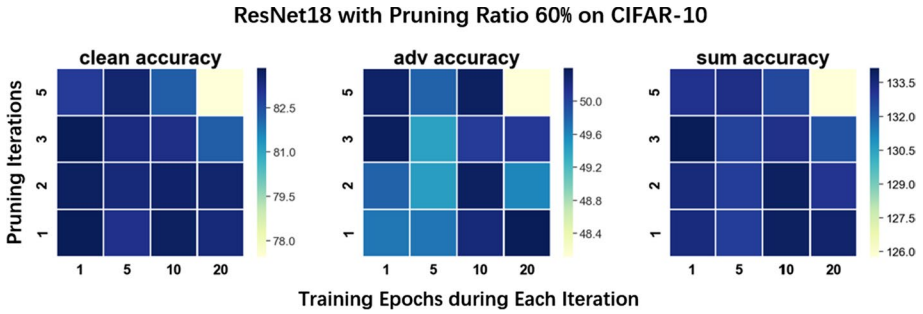

Fig. 10 Influence of pruning iterations and re-training epochs on ResNet18. Left to right figures show the accuracies on the clean testing set, adversarial testing set and both, after adversarially training 'winning ticket' for a total of 120 epochs (with iterative pruning process included)

To decide the hyperparameters in validating the lottery ticket hypothesis on robust networks, we employ grid search and investigate the effect of pruning iterations and re-training epochs at each iteration on the performance of the 'winning ticket' found after pruning. Fig. 10 shows a typical case of a network pruned for a variety of numbers of iterations and re-training for different numbers of epochs.

From the middle of Fig. 10, we can tell that one-shot pruning leads to lower robustness, which is not recognized in (Li et al. 2020). But a large iteration number (> 3) along with large re-training epochs (> 10) would also hurt the model's performance. We found that **1** epoch of re-training per iteration for 3 or 4 iterations would lead to a small network with better performance. Hence we choose these hyperparameters to search for 'winning tickets' in our experiments. We actually use less re-training epochs than searching the 'winning tickets' in a natural setting (Frankle and Carbin 2019). Since adversarial training is costly, it is beneficial to prune the network at an early stage.

Experiments of Inverse Weight Inheritance

Training Acceleration

See Table 8

Table 8 FLOPs ratios of inverse weights inheritance (IWI) to adversarially training from scratch

Inverse Weights Inheritance: FLOPs w/ Stop-E				
Network	$p\%$	CIFAR-10	$p\%$	CIFAR-100
ResNet18	80	77.48%	60	74.19%
	90	67.38%	80	57.54%
	95	59.10%	90	49.01%
VGG16	80	73.50%	60	87.32%
	90	66.84%	80	66.80%
	95	59.73%	90	54.40%

A lower ratio indicates the model is trained with less computation cost. $p\%$ denotes the parameter pruning ratio of the 'winning ticket' in IWI

Beyond performance enhancement, inverse weights inheritance also accelerates the adversarial training process, mainly due to the lower expense of adversarially training a small network and less training epochs required after the large network obtaining inherited weights. Crafting adversarial examples and re-training the model are costly on large networks, for both the forward and backward propagation are quite expensive. If such a process can be substituted by small networks, it will reduce a great deal of computation overhead.

To verify that inverse weights inheritance indeed accelerates adversarial training, we calculate the computation budget of the overall training process by floating-point operations (FLOPs). Both the forward and backward propagation are considered. We compute the FLOPs for adversarially training the base network from scratch, and the FLOPs using inverse weights inheritance for training, and report the relative ratios of the latter to the former. From Table 8, we can tell our method saves computation cost while improving the overall performance, and the higher the pruning ratio, the more computation cost saved.

Weights Distribution

See Fig. 11

For weights distribution of networks trained by IWI, we provide Fig. 11 as a supplement for Fig. 4 of the paper of different pruning ratio. The conclusion is consistent with that in the paper that IWI finds sparse weights distribution for large networks with improved performance.

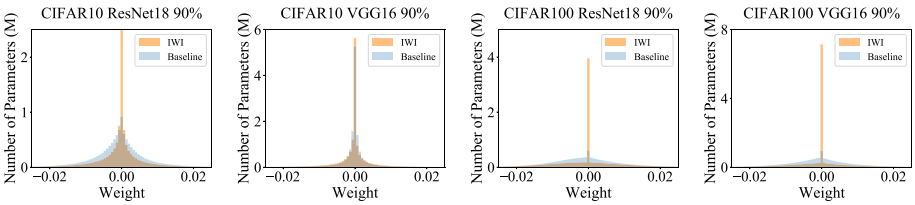

Fig. 11 Weights distribution of large networks trained by inverse weights inheritance (IWI) and adversarially trained with random initialization (Baseline). Net XX% denotes a large network trained by inheriting the weights of a 'winning ticket' with XX% weights pruned

Funding This work was partially supported by NSF China (61902245, 62032020, 62136006, 61901261), and the Science and Technology Innovation Program of Shanghai (19YF1424500, 19YF1424200).

Availability of data and material The data and material will be available once the paper is published.

Code availability The code will be available once the paper is published.

Declarations

Conflicts of interest The authors declare that they have no conflict of interest.

References

Alzantot, M., Sharma, Y., Elgohary, A., Ho, B. J., Srivastava, M., & Chang, K. W. (2018). Generating Natural Language Adversarial Examples. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, pp 2890–2896.

Arora, S., Ge, R., Neyshabur, B., & Zhang, Y. (2018). Stronger generalization bounds for deep nets via a compression approach. In: 35th International Conference on Machine Learning, vol 1, pp 390–418.

Balda, E. R., Behboodi, A., Koep, N., & Mathar, R. (2019). Adversarial Risk Bounds for Neural Networks through Sparsity based Compression. In: 23rd International Conference on Artificial Intelligence and Statistics, vol 108, pp 3816–3825.

Bartoldson, B. R., Morcos, A. S., Barbu, A., & Erlebacher, G. (2020). The Generalization-Stability Tradeoff in Neural Network Pruning. 33rd Advances in Neural Information Processing Systems.

Bastani, O., Ioannou, Y., Lampropoulos, L., Vytiniotis, D., Nori, A., & Criminisi, A. (2016). Measuring neural net robustness with constraints. In: Advances in neural information processing systems, pp 2613–2621.

Cosentino, J., Zaiter, F., Pei, D., & Zhu, J. (2019). The search for sparse, robust neural networks. In: 33rd Conference on Neural Information Processing Systems, NeurIPS 2019.

Dhillon, G. S., Azizzadenesheli, K., Lipton, Z. C., Bernstein, J., Kossaifi, J., Khanna, A., & Anandkumar, A. (2018). Stochastic activation pruning for robust adversarial defense. arXiv preprint arXiv:180301442.

Dinh, T., Wang, B., Bertozzi, A. L., & Osher, S. J. (2020). Sparsity meets robustness: channel pruning for the feynman-kac formalism principled robust deep neural nets. arXiv preprint arXiv:200300631.

Fawzi, A., Moosavi-Dezfooli, S. M., & Frossard, P. (2016). Robustness of classifiers: From adversarial to random noise. In: 29th Advances in Neural Information Processing Systems, pp 1632–1640.

Fawzi, A., Fawzi, O., & Frossard, P. (2018). Analysis of classifiers' robustness to adversarial perturbations. *Machine Learning, 107*(3), 481–508.

Frankle, J., & Carbin, M. (2019). The lottery ticket hypothesis: Finding sparse, trainable neural networks. In: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019.

Goldblum, M., Fowl, L., Feizi, S., & Goldstein, T. (2020). Adversarially robust distillation. In: Thirty-Fourth AAAI Conference on Artificial Intelligence.

Goodfellow, I. J., Shlens, J., & Szegedy, C. (2015). Explaining and harnessing adversarial examples. In: 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7–9, 2015, Conference Track Proceedings.

Gui, S., Wang, H., Yu, C., Yang, H., Wang, Z., & Liu, J. (2019). Model compression with adversarial robustness: A unified optimization framework. In: Advances in Neural Information Processing Systems, pp 1283–1294.

Guo, M., Yang, Y., Xu, R., Liu, Z., & Lin, D. (2020). When nas meets robustness: In search of robust architectures against adversarial attacks. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp 631–640.

Guo, Y., Zhang, C., Zhang, C., & Chen, Y. (2018). Sparse dnns with improved adversarial robustness. In: Advances in neural information processing systems, pp 242–251.

Han, S., Pool, J., Tran, J., & Dally, W. (2015). Learning both weights and connections for efficient neural network. In: Advances in neural information processing systems, pp 1135–1143.

Han, S., Mao, H., & Dally, W. J. (2016). Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. In: 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2–4, 2016, Conference Track Proceedings.

Hassibi, B., & Stork, D. G. (1993). Second order derivatives for network pruning: Optimal brain surgeon. In: Advances in neural information processing systems, pp 164–171.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016, pp 770–778.

Hein, M., & Andriushchenko, M. (2017). Formal guarantees on the robustness of a classifier against adversarial manipulation. In: Advances in Neural Information Processing Systems, pp 2266–2276.

Huang, G., Liu, Z., van der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017, pp 2261–2269.

Keskar, N. S., Nocedal, J., Tang, P. T. P., Mudigere, D., & Smelyanskiy, M. (2017). On large-batch training for deep learning: Generalization gap and sharp minima. In: 5th International Conference on Learning Representations.

Kurakin, A., Goodfellow, I. J., & Bengio, S. (2017). Adversarial machine learning at scale. In: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24–26, 2017, Conference Track Proceedings.

Lang, J. (ed) (2018) Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13–19, 2018, Stockholm, Sweden, ijcai.org.

LeCun, Y., Denker, J. S., & Solla, S. A. (1990). Optimal brain damage. In: Advances in neural information processing systems, pp 598–605.

Li, B., Wang, S., Jia, Y., Lu, Y., Zhong, Z., Carin, L., & Jana, S. (2020). Towards practical lottery ticket hypothesis for adversarial training. arXiv preprint arXiv:200305733.

Li, H., Kadav, A., Durdanovic, I., Samet, H., & Graf, H. P. (2017). Pruning filters for efficient convnets. In: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24–26, 2017, Conference Track Proceedings.

Liu, Z., Li, J., Shen, Z., Huang, G., Yan, S., & Zhang, C. (2017). Learning efficient convolutional networks through network slimming. In: IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22–29, 2017, pp 2755–2763.

Liu, Z., Sun, M., Zhou, T., Huang, G., & Darrell, T. (2019). Rethinking the value of network pruning. In: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6–9, 2019.

Luo, J. H., Wu, J., & Lin, W. (2017). Thinet: A filter level pruning method for deep neural network compression. In: IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017, pp 5068–5076.

Madaan, D., Shin, J., & Hwang, S. J. (2019). Adversarial neural pruning with latent vulnerability suppression. arXiv e-prints.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., & Vladu, A. (2018). Towards deep learning models resistant to adversarial attacks. In: 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30–May 3, 2018, Conference Track Proceedings.

Miyato, T., Dai, A. M., & Goodfellow, I. (2017). Adversarial training methods for semi-supervised text classification. In: 5th International Conference on Learning Representations.

Moosavi-Dezfooli, S. M., Fawzi, A., & Frossard, P. (2016). Deepfool: a simple and accurate method to fool deep neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2574–2582.

Morcos, A. S., Yu, H., Paganini, M., & Tian, Y. (2019). One ticket to win them all: generalizing lottery ticket initializations across datasets and optimizers. In: 32nd Advances in Neural Information Processing Systems.

Petzka, H., Kamp, M., Adilova, L., Boley, M., & Sminchisescu, C. (2020). Relative Flatness and Generalization in the Interpolation Regime. arXiv e-prints.

Rakin, A. S., He, Z., Yang, L., Wang, Y., Wang, L., & Fan, D. (2019). Robust sparse regularization: Simultaneously optimizing neural network robustness and compactness. arXiv preprint arXiv:190513074.

Salman, H., Yang, G., Zhang, H., Hsieh, C. J., & Zhang, P. (2019). A convex relaxation barrier to tight robustness verification of neural networks. In: Advances in Neural Information Processing Systems, pp 9832–9842.

Sehwag, V., Wang, S., Mittal, P., & Jana, S. (2019). Towards compact and robust deep neural networks. arXiv e-prints.

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In: 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7–9, 2015, Conference Track Proceedings.

Wang, B., Shi, Z., & Osher, S. (2019). Resnets ensemble via the feynman-kac formalism to improve natural and robust accuracies. In: Advances in Neural Information Processing Systems, pp 1655–1665.
Wang, L., Ding, G. W., Huang, R., Cao, Y., & Lui, Y. C. (2018). Adversarial robustness of pruned neural networks. In: Workshop of 6th International Conference on Learning Representations.
Wen, W., Wu, C., Wang, Y., Chen, Y., & Li, H. (2016). Learning structured sparsity in deep neural networks. In: Advances in neural information processing systems, pp 2074–2082.
Xu, H., & Mannor, S. (2012). Robustness and generalization. *Machine Learning, 86*(3), 391–423.
Yang, Y., Zhang, G., Xu, Z., & Katabi, D. (2019). Me-net: Towards effective adversarial robustness with matrix estimation. In: Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA, pp 7025–7034.
Ye, J., Lu, X., Lin, Z., & Wang, J. Z. (2018). Rethinking the smaller-norm-less-informative assumption in channel pruning of convolution layers. In: International Conference on Learning Representations.
Ye, S., Xu, K., Liu, S., Cheng, H., Lambrechts, J. H., Zhang, H., Zhou, A., Ma, K., Wang, Y., & Lin, X. (2019). Adversarial robustness vs model compression, or both? In: International Conference on Computer Vision.
Zhang, H., Yu, Y., Jiao, J., Xing, E. P., Ghaoui, L. E., & Jordan, M. I. (2019). Theoretically principled trade-off between robustness and accuracy. In: Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9–15 June 2019, Long Beach, California, USA, pp 7472–7482.
Zhang, T., & Zhu, Z. (2019). Interpreting adversarially trained convolutional neural networks. In: Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA, pp 7502–7511.
Zhao, Y., Shumailov, I., Mullins, R., & Anderson, R. (2018). To compress or not to compress: Understanding the interactions between adversarial attacks and neural network compression. arXiv preprint arXiv: 181000208.