Adversarial Robustness vs. Model Compression, or Both?

Shaokai Ye[1*] Kaidi Xu[2*] Sijia Liu[3] Hao Cheng[4] Jan-Henrik Lambrechts[1] Huan Zhang[6]
Aojun Zhou[5] Kaisheng Ma[1+] Yanzhi Wang[2+] Xue Lin[2+]
[1]IIIS, Tsinghua University & IIISCT, China [2]Northeastern University, USA
[3]MIT-IBM Watson AI Lab, IBM Research [4]Xi'an Jiaotong University, China
[5]SenseTime Research, China [6]University of California, Los Angeles, USA

Abstract

It is well known that deep neural networks (DNNs) are vulnerable to adversarial attacks, which are implemented by adding crafted perturbations onto benign examples. Min-max robust optimization based adversarial training can provide a notion of security against adversarial attacks. However, adversarial robustness requires a significantly larger capacity of the network than that for the natural training with only benign examples. This paper proposes a framework of concurrent adversarial training and weight pruning that enables model compression while still preserving the adversarial robustness and essentially tackles the dilemma of adversarial training. Furthermore, this work studies two hypotheses about weight pruning in the conventional setting and finds that weight pruning is essential for reducing the network model size in the adversarial setting; training a small model from scratch even with inherited initialization from the large model cannot achieve neither adversarial robustness nor high standard accuracy. Code is available at https://github.com/yeshaokai/Robustness-Aware-Pruning-ADMM.

1. Introduction

Deep learning or deep neural networks (DNNs) have achieved extraordinary performance in many application domains such as image classification [19, 39], object detection and recognition [27, 35], natural language processing [10, 34] and medical image analysis [28, 37]. Besides deployments on the cloud, deep learning has become ubiquitous on embedded systems such as mobile phones, IoT devices, personal healthcare wearables, autonomous driving [4, 11], unmanned aerial systems [6, 23], etc.

It has been well accepted that DNNs are vulnerable to adversarial attacks [14, 46, 47, 53], which raises concerns of DNNs in security-critical applications and may result in disastrous consequences. For example, in autonomous driving, a stop sign may be mistaken by a DNN as a speed limit sign; malware may escape from deep learning based detection; and in authentication using face recognition, unauthorized people may escalate their access rights by fooling the DNN.

Adversarial attacks are implemented by generating adversarial examples, i.e., adding sophisticated perturbations onto benign examples, such that adversarial examples are classified by the DNN as target (wrong) labels instead of the correct labels of the benign examples. The adversary may have white-box accesses to the DNN where the adversary has full information about the model (e.g., structure and weight parameters) [7, 8, 52, 45]; or black-box accesses where the adversary can only make queries and observe outputs [9, 22]. The black-box scenarios are of particular interest in the Machine Learning as a Service (MLaaS) paradigm, specifically in some cases where DNN models trained through the cloud platform cannot be downloaded and are accessed only through the service's API.

According to [3], defenses that cause obfuscated gradients may provide a false sense of security and can be overcome with improved attack techniques such as backward pass differentiable approximation, expectation over transformation, and reparameterization. Also pointed out in [3], *adversarial training leveraging min-max robust optimization* [33] does not have obfuscated gradients issue and can be a promising defense mechanism. Since that researchers have begun to notice the issue when designing new defenses, more defenses have been proposed including adversarial training based ones [29, 38, 42] and others [40, 48, 25].

Min-max robust optimization based adversarial training [33, 41] can provide a notion of security against all first-order adversaries (i.e., attacks that rely on gradients of the loss function with respect to the input), by modeling an universal first-order attack through the inner maximization problem while the outer minimization still representing the

*Equal contribution
[1]Institute for Interdisciplinary Information Core Technology
+ Corresponding Authors

training process. However, as noted by [33], adversarial robustness requires a significant larger architectural *capacity of the network* than that for the natural training with only benign examples. For example, we may need to quadruple a DNN model with state-of-the-art standard accuracy on MNIST for strong adversarial robustness. In addition, increasing the network capacity may provide a better trade-off between standard accuracy of an adversarially trained model and its adversarial robustness [41].

Therefore, the required large network capacity by adversarial training may limit its use for security-critical scenarios especially in resource constrained application systems. On the other hand, model compression techniques such as *weight pruning* [17, 15, 49, 20, 43] have been essential for implementing DNNs on resource constrained embedded and IoT systems. Weight pruning explores weight sparsity to prune synapses and neurons without notable accuracy degradation. References [16, 44] theoretically discuss the relationship between adversarial robustness and weight sparsity, but do not apply any active defense techniques in their research. The work [44] concludes that moderate sparsity can help with adversarial robustness in that it increases the ℓ_p norm of adversarial examples (although DNNs with weight sparsity are still vulnerable under attacks).

We are motivated to investigate whether and how weight sparsity can facilitate an active defense technique i.e., the adversarial training, by relaxing the network capacity requirement. Figure 1 characterizes the weight distribution of VGG-16 network on CIFAR dataset. We test on the original size, 1/2 size, and 1/4 size of VGG-16 network for their standard accuracy and adversarial accuracy. We have following observations: (i) Smaller model size (network capacity) indicates both lower standard accuracy and adversarial accuracy for adversarially trained model. (ii) Adversarially trained model is less sparse (fewer zero weights) than naturally trained model. Therefore, *pre-pruning before adversarial training is not a feasible solution* and *it seems harder to prune an adversarially trained model*.

This paper tries to answer the question of whether we can enjoy both the adversarial robustness and model compression together. Basically, we integrate weight pruning with the adversarial training to enable security-critical applications in resource constrained systems.

Our Contributions: We build a framework that achieves both adversarial robustness and model compression through implementing concurrent weight pruning and adversarial training. Specifically, we use the ADMM (alternating direction method of multipliers) based pruning [50, 51] in our framework due to its compatibility with adversarial training. More importantly, the ADMM based pruning is universal in that it supports both irregular pruning and different kinds of regular pruning, and in this way we can easily switch between different pruning schemes for fair comparison. Eventually, our framework tackles the dilemma of adversarial training.

We also study two hypotheses about weight pruning that were proposed for the conventional model compression setting and experimentally examine their validness for the adversarial training setting. We find that the weight pruning is essential for reducing the network model size in the adversarial setting, training a small model from scratch even with inherited initialization from the large model cannot achieve adversarial robustness and high standard accuracy at the same time.

With the proposed framework of concurrent adversarial training and weight pruning, we systematically investigate the effect of different pruning schemes on adversarial robustness and model compression. We find that irregular pruning scheme is the best for preserving both standard accuracy and adversarial robustness while pruning the DNN models.

2. Related Work

2.1. Adversarial Training

Adversarial training [33] uses a min-max robust optimization formulation to capture the notion of security against adversarial attacks. It does this by modeling an universal first-order attack through the inner maximization problem while the outer minimization still represents the training process. Specifically, it solves the optimization problem:

$$\min_{\boldsymbol{\theta}} \quad \mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\max_{\delta\in\boldsymbol{\Delta}} L(\boldsymbol{\theta}, x+\delta, y) \right] \tag{1}$$

where pairs of examples $x \in \mathbb{R}^d$ and corresponding labels $y \in [k]$ follow an underlying data distribution $\mathcal{D}$; δ is the added adversarial perturbation that belongs to a set of allowed perturbations $\boldsymbol{\Delta} \subseteq \mathbb{R}^d$ for each example x; $\boldsymbol{\theta} \in \mathbb{R}^p$ presents the set of weight parameters to be optimized; and $L(\boldsymbol{\theta}, x, y)$ is the loss function, for instance, the cross-entropy loss for a DNN.

The inner maximization problem is solved by sign-based projected gradient descent (PGD), which presents a powerful adversary bounded by the ℓ_∞-ball around x as:

$$x^{t+1} = \Pi_{x+\boldsymbol{\Delta}} \left(x^t + \alpha \operatorname{sgn}(\nabla_x L(\boldsymbol{\theta}, x, y)) \right) \tag{2}$$

where t is the iteration index, α is the step size, and $\operatorname{sgn}(\cdot)$ returns the sign of a vector. PGD is a variant of IFGSM attack [24] and can be used with random start to add uniformly distributed noise to model $\boldsymbol{\Delta}$ during adversarial training.

One major drawback of adversarial training is that it needs a significantly larger network capacity for achieving strong adversarial robustness than for correctly classifying

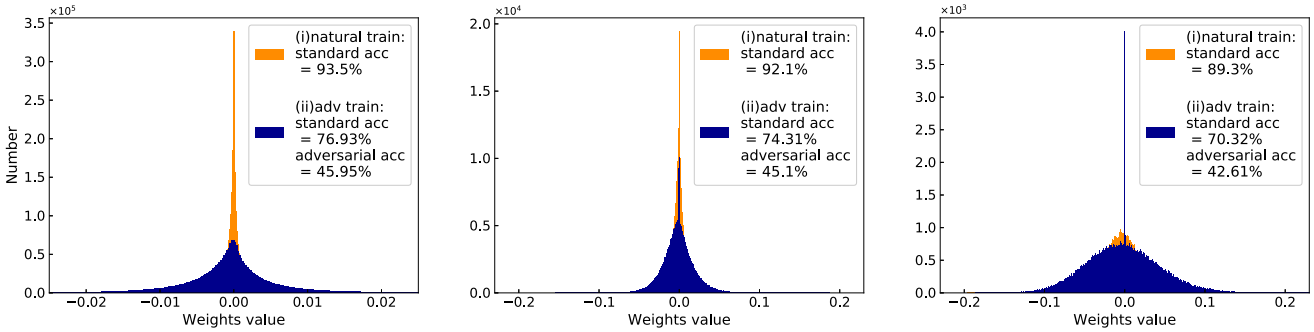


Figure 1: Weight distribution of VGG-16 network with (a) original size, (b) 1/2 size, and (c) 1/4 size on CIFAR dataset. For each size in one subfigure, the weights are characterized for (i) a naturally trained model and (ii) an adversarially trained model. The standard accuracy and adversarial accuracy are marked with the legend.

benign examples only [33]. In addition, adversarial training suffers from a more significant overfitting issue than the natural training [36]. Later in this paper, we will demonstrate some intriguing findings related to the above mentioned observations.

2.2. Weight Pruning

Weight pruning as a model compression technique has been proposed for facilitating DNN implementations on resource constrained application systems, as it explores weight sparsity to prune synapses and consequently neurons without notable accuracy degradation. There are in general the *regular pruning scheme* that can preserve the model's structure in some sense, and otherwise the *irregular pruning scheme*. Regular pruning can be further categorized as the *filter pruning scheme* and the *column pruning scheme*. Filter pruning by the name prunes whole filters from a layer. Column pruning prunes weights for all filters in a layer, *at the same locations*. Please note that some references mention channel pruning, which by the name prunes some channels completely from the filters. But essentially channel pruning is equivalent to filter pruning, because if some filters are pruned in a layer, it makes the corresponding channels of next layer invalid [20].

In this work, we implement and investigate the filter pruning, column pruning, and irregular pruning schemes in the adversarial training setting. Also, with each pruning scheme, we uniformly prune every layer by the same pruning ratio. For example, when we prune the model size (network capacity) by a half, it means the size of each layer is reduced by a half.

There are existing irregular pruning work [17, 15, 49, 50] and regular pruning work [20, 43, 51, 26, 30]. In addition, almost all the regular pruning work are actually filter pruning, except the work [43] which is the first to propose column pruning and work [51] which can implement column pruning through an ADMM based approach. In this work, we use the ADMM approach due to its potential for all the pruning schemes and its compatibility with adversarial training, as shall be demonstrated in the later section.

Researchers have also begun to reflect and make some hypotheses about the weight pruning. The lottery ticket hypothesis [12] conjectures that inside the large network, a subnetwork together with their initialization makes the pruning particular effective, and together they are termed as the "winning tickets". In this hypothesis, the original initilizaiton of the sub-network (before the large network pruning) is needed for it to achieve competitive performance when trained in isolation. In addition, the work [31] concludes that training a predefined target model from scratch is no worse or even better than applying structured (regular) pruning on a large over-parameterized model to the same target model architecture.

However, these hypotheses and findings are proposed for the general weight pruning. In this paper, we make some intriguing observations about weight pruning in the adversarial setting, which are insufficiently explained under the existing hypotheses [12, 31].

3. Concurrent Adversarial Training and Weight Pruning

In this section, we provide the framework for concurrent adversarial training and weight pruning. We formulate the problem in a way that lends itself to the application of ADMM (alternating direction method of multipliers):

$$\begin{aligned} \min_{\boldsymbol{\theta}_i} \quad & \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\max_{\delta\in\boldsymbol{\Delta}} L(\boldsymbol{\theta}, x+\delta, y)\right] + \sum_{i=1}^{N} g_i(\mathbf{z}_i), \\ s.t. \quad & \boldsymbol{\theta}_i = \mathbf{z}_i,\ i = 1,\ldots,N. \end{aligned} \tag{3}$$

Here $\boldsymbol{\theta}_i$ are the weight parameters in each layer.

$$g_i(\boldsymbol{\theta}_i) = \begin{cases} 0 & \text{if } \boldsymbol{\theta}_i \in S_i \\ +\infty & \text{otherwise} \end{cases} \tag{4}$$

is an indicator function to incorporate weight sparsity constraint (different weight pruning schemes can be defined through the set S_i). $\mathbf{z}_i$ are auxiliary variables that enable the ADMM based solution.

The ADMM framework is built on the augmented Lagrangian of an equality constrained problem [5]. For problem (3), the augmented Lagrangian form becomes

$$\begin{aligned} \mathcal{L}(\{\boldsymbol{\theta}_i\}, \{\mathbf{z}_i\}, \{\mathbf{u}_i\}) = \mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\max_{\delta\in\boldsymbol{\Delta}} L(\boldsymbol{\theta}, x+\delta, y)\right] \\ + \sum_{i=1}^{N} g_i(\mathbf{z}_i) + \sum_{i=1}^{N} \mathbf{u}_i^T(\boldsymbol{\theta}_i - \mathbf{z}_i) + \frac{\rho}{2}\sum_{i=1}^{N} \|\boldsymbol{\theta}_i - \mathbf{z}_i\|_2^2, \end{aligned} \tag{5}$$

where $\{\mathbf{u}_i\}$ are Lagrangian multipliers associated with equality constraints of problem (3), and $\rho > 0$ is a given augmented parameter. Through formation of the augmented Lagrangian, the ADMM framework decomposes problem (3) into two subproblems that are solved iteratively:

$$\{\boldsymbol{\theta}_i^k\} = \arg\min_{\{\boldsymbol{\theta}_i\}} \mathcal{L}(\{\boldsymbol{\theta}_i\}, \{\mathbf{z}_i^{k-1}\}, \{\mathbf{u}_i^{k-1}\}), \tag{6}$$

$$\{\mathbf{z}_i^k\} = \arg\min_{\{\mathbf{z}_i\}} \mathcal{L}(\{\boldsymbol{\theta}_i^k\}, \{\mathbf{z}_i\}, \{\mathbf{u}_i^{k-1}\}), \tag{7}$$

where k is the iteration index. The Lagrangian multipliers are updated as $\mathbf{u}_i^k := \mathbf{u}_i^{k-1} + \rho(\boldsymbol{\theta}_i^k - \mathbf{z}_i^k)$.

The first subproblem (6) is explicitly given by

$$\min_{\boldsymbol{\theta}_i} \mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\max_{\delta\in\boldsymbol{\Delta}} L(\boldsymbol{\theta}, x+\delta, y)\right] + \frac{\rho}{2}\sum_{i=1}^{N} \|\boldsymbol{\theta}_i - \mathbf{z}_i^k + \mathbf{u}_i^k\|_2^2. \tag{8}$$

The first term in the objective function of (8) is a min-max problem. Same as solving the adversarial training problem in Section 2.1, here we can use the PGD adversary (2) with T iterations and random start for the inner maximization problem. The inner problem is tractable under an universal first-order adversary [33]. The second convex quadratic term in (8) arises due to the presence of the augmented term in (5). Given the adversarial perturbation $\boldsymbol{\delta}$, we can apply the stochastic gradient decent algorithm for solving the overall minimization problem. Due to the non-convexity of the loss function, the global optimality of the solution is not guaranteed. However, ADMM could offer a local optimal solution when ρ is appropriately chosen since the quadratic term in (8) is strongly convex as $\rho > 0$, which stabilizes the convergence of ADMM [21].

On the other hand, the second subproblem (7) is given by

$$\underset{\{\mathbf{z}_i\}}{\text{minimize}} \quad \sum_{i=1}^{N} g_i(\mathbf{z}_i) + \frac{\rho}{2}\sum_{i=1}^{N} \|\boldsymbol{\theta}_i^{k+1} - \mathbf{z}_i + \mathbf{u}_i^k\|_2^2. \tag{9}$$

Note that $g_i(\cdot)$ is the indicator function defined by S_i, thus this subproblem can be solved analytically and optimally [5]. The optimal solution is

$$\mathbf{z}_i^{k+1} = \boldsymbol{\Pi}_{S_i}(\boldsymbol{\theta}_i^{k+1} + \mathbf{u}_i^k), \tag{10}$$

where $\boldsymbol{\Pi}_{S_i}(\cdot)$ is Euclidean projection of $\boldsymbol{\theta}_i^{k+1} + \mathbf{u}_i^k$ onto S_i.

Algorithm 1 Concurrent Adversarial Training and Weight Pruning

Input: dataset $\mathcal{D}$, ADMM iteration number K, PGD step size α, PGD iteration number T, augmented parameter ρ, and sets S_i's for weight sparsity constraint.
Output: weight parameters $\boldsymbol{\theta}$.
for $k = 1, 2, \ldots, K$ **do**
  Sample batch (x, y) from $\mathcal{D}$
  ▷ Solve Eq (8) over $\boldsymbol{\theta}$
  for $t = 1, 2, \ldots, T$ **do**
    Solve the inner max by Eq (2)
  end for
  Apply Adam optimizer on the outer min of Eq (8) to obtain $\{\boldsymbol{\theta}_i^k\}$
  Solve Eq (9) using Eq (10) to obtain $\{\mathbf{z}_i^k\}$
  $\mathbf{u}_i^k := \mathbf{u}_i^{k-1} + \rho(\boldsymbol{\theta}_i^k - \mathbf{z}_i^k)$.
end for

3.1. Definitions of S_i for Weight Pruning Schemes

This subsection introduces how to use the weight sparsity constraint $\boldsymbol{\theta}_i \in S_i$ to implement different weight pruning scheme. For each weight pruning scheme, we first provide the exact form of $\boldsymbol{\theta}_i \in S_i$ constraint and then provide the explicit form of the solution (10). Before doing that, we reduce $\boldsymbol{\theta}_i$ back into the four dimensional tensor form as $\boldsymbol{\theta}_i \in R^{N_i \times C_i \times H_i \times W_i}$, where N_i, C_i, H_i, and W_i are respectively the number of filters, the number of channels in a filter, the height of a filter, and the width of a filter.

Filter pruning

$$\boldsymbol{\theta}_i \in S_i := \{\boldsymbol{\theta}_i \mid \|\boldsymbol{\theta}_i\|_{n=0} \le \alpha_i\}. \tag{11}$$

Here, $\|\boldsymbol{\theta}_i\|_{n=0}$ means the number of filters containing non-zero elements. To obtain the solution (10) with such constraint, we firstly calculate $O_n = \|(\boldsymbol{\theta}_i^{k+1} + \mathbf{u}_i^k)_{n,:,:,:}\|_F^2$, where $\|\cdot\|_F$ denotes the Frobenius norm. We then keep α_i largest values in O_n and set the rest to zeros.

Column pruning

$$\boldsymbol{\theta}_i \in S_i := \{\boldsymbol{\theta}_i \mid \|\boldsymbol{\theta}_i\|_{c,h,w=0} \leq \beta_i\}. \tag{12}$$

Here, $\|\boldsymbol{\theta}_i\|_{c,h,w=0}$ means the number of elements at the same locations in all filters in the ith layer containing non-zero elements. To obtain the solution (10) with such constraint, first we calculate $O_c = \|(\boldsymbol{\theta}_i^{k+1} + \mathbf{u}_i^k)_{:,c,h,w}\|_F^2$. We then keep β_i largest values in O_c and set the rest to zeros.

Irregular pruning

$$\boldsymbol{\theta}_i \in S_i := \{\boldsymbol{\theta}_i \mid \|\boldsymbol{\theta}_i\|_0 \leq \gamma_i\}. \tag{13}$$

In this special case, we only constrain the number of non-zero elements in the ith layer filters i.e., in $\boldsymbol{\theta}_i$. To obtain the solution (10), we keep γ_i largest magnitude elements in $\boldsymbol{\theta}_i$ and set the rest to zeros.

Algorithm 1 summarizes the framework of concurrent adversarial training and weight pruning.

In addition to ADMM based weight pruning, we also show results of post-pruning in Appendix A, with and without retraining respectively.

4. Weight Pruning in the Adversarial Setting

In this section, we examine the performance of weight pruning in the adversarial setting. We obtain intriguing results contradictory from those [12, 31] in the conventional model compression setting. Here we specify the proposed framework of concurrent adversarial training and weight pruning by the filter pruning scheme, which is a common pruning choice to facilitate the implementation of sparse neural networks on hardware. Other pruning schemes will be investigated in the experiment section. In Table 1, we summarize all the networks tested in the paper with their model architectures specified by the width scaling factor w.

4.1. Weight Pruning vs Training from Scratch

An ongoing debate about pruning is whether weight pruning is actually needed and why not just training a small network from scratch. To answer this question, the work [31] performs a large amount of experiments to find that (i) training a large, over-parameterized model is often not necessary to obtain an efficient final model, and (ii) the meaning of weight pruning lies in searching the architecture of the final pruned model. In another way, if we are given with a predefined target model, it makes no difference whether we reach the target model from pruning a large, over-parameterized model or we train the target model from scratch. We also remark that the above conclusions from [31] are made while performing regular pruning.

Although the findings in [31] may hold in the setting of natural training, the story becomes different in the setting of adversarial training. Tables 2, 3, and 4 demonstrate the *natural test accuracy / adversarial test accuracy* of natural training, adversarial training, and concurrent adversarial training and weight pruning for different datasets and networks. Let us take Table 2 as an example. When we naturally train a network of size $w = 1$, we have 98.25% natural test accuracy and 0% adversarial test accuracy. When we adversarially train the network of size $w = 1$, both natural test accuracy and adversarial test accuracy become 11.35%, which is still quite low. It demonstrates that the network of size $w = 1$ does not have enough capacity for strong adversarial robustness. In order to promote the adversarial robustness, we need to adversarially train the network with size of $w = 4$ at least. Surprisingly, by leveraging our concurrent adversarial training and weight pruning on the network of size $w = 4$, we can obtain a much smaller pruned model with the target size of $w = 1$ but achieve competitive natural test accuracy / adversarial test accuracy (96.22% / 89.41%) compared to the adversarially trained model of size $w = 4$. To obtain a network of size $w = 1$ with the highest natural and adversarial test accuracy, we should apply the proposed framework on the network of size $w = 8$. Similar observations hold for Tables 3 and 4.

In summary, the value of weight pruning is essential in the adversarial training setting: it is possible to acquire a network of small model size (by weight pruning) with both high natural test accuracy and adversarial test accuracy. By contrast, one may lose the natural and adversarial test accuracy if the adversarial training is directly applied to a small-size network that is not acquired from weight pruning.

4.2. Pruning to Inherit Winning Ticket or Else?

In the natural training (pruning) setting, the lottery ticket hypothesis [12] states that the meaning of weight pruning is in that the small sub-network model can inherit the initialization (the so-called "winning ticket") from the large model. Or in another way, the weight pruning is meaningful only in that it provides effective initialization to the final pruned model.

To test whether or not the lottery ticket hypothesis is valid in the adversarial setting, we perform adversarial training under the similar experimental setup as [12]. The natural/adversarial test accuracy results are summarized in Table 5, where the result in cell w_1-w_2 ($w_1 > w_2$) denotes the accuracy of an adversarially trained model of size w_2 using the inherited initialization from an adversarially trained model of size w_1. No pruning is used in Table 5. For example, cell 4-2 in Table 5 only yields 11.35%/11.35% accuracy. Recall from Table 2 that if we use our proposed framework of concurrent adversarial training and weight pruning to prune from a model with size 4 to a small model with size

Table 1: Network structures used in our experiments. FC, M, and A mean fully connected layer, max-pooling layer, and average-pooling layer, respectively. Other numbers denote the numbers of filters in convolutional layers. We use w to denote the scaling factor of a network. Each layer is equally scaled with w.

MNIST	2*w, 4*w, FC(196*w, 64*w), FC(64*w,10)
CIFAR LeNet	6*w, 16*w, FC(400*w, 120*w), FC(120*w, 84*w), FC(84*w,10)
CIFAR VGG	4*w,4*w,M,8*w,8*w,M,16*w,16*w,16*w,M,32*w,32*w,32*w,M,32*w,32*w,32*w,M,A,FC(32*w,10)
CIFAR ResNet	b*w, where b denotes $1/16$ of the size of ResNet18 [19]

Table 2: **Natural test accuracy/adversarial test accuracy** (in %) on **MNIST** of [column ii] naturally trained model with different size w, [column iii] adversarially trained model with different size w, [columns iv–vii] concurrent adversarial training and weight pruning from a large size to a small size.

w	nat baseline	adv baseline	1	2	4	8
1	98.25/0.00	11.35/11.35	-	-	-	-
2	98.72/0.00	11.35/11.35	11.35/11.35	-	-	-
4	99.07/0.00	98.15/91.38	96.22/89.41	97.68/91.77	-	-
8	99.20/0.00	98.85/93.51	**97.31/92.16**	**98.31/93.93**	98.87/94.27	-
16	99.31/0.00	99.02/94.65	96.19/87.79	98.07/89.95	**98.87/94.77**	**99.01/95.44**

of 2, we can have high accuracy $97.68\%/91.77\%$ in cell 4-2 of Table 2. Our results suggest that the weight pruning in the adversarial setting is out of the scope of the lottery ticket hypothesis.

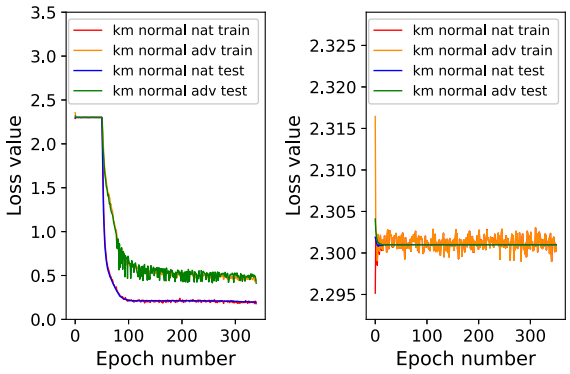


Figure 2: The adversarial training loss of kaiming_normal initialization with Adam optimizer trained from scratch on MNIST by LeNet with size of $w = 1$ using different random seeds. The left subfigure is the only successful case we found and the right subfigure represents a common case.

Moreover, to further explore the relationship between initialization and model capacity in adversarial training, we conduct additional experiment. Seven different initialization methods are compared to train the smallest LeNet model ($w = 1$) with 300 epochs using Adam, SGD and CosineAnnealing [32] on MNIST. We repeat this experiment 10 times with different random seed and report the average accuracy in Table B1. As suggested by Table 2, adversarial training from scratch failed as $w = 1, 2$. In all studied scenarios, we only find two exceptions: a) Adam with uniform initialization and b) Adam with kaiming_normal in which 1 out of 10 trials succeeds (the losses are drawn in Figure 2). Even for these exceptions, the corresponding test accuracy is much worse than that of the smallest model obtained from concurrent adversarial training and weight pruning in Table 2. We also find that the accuracy 11.35% corresponds to a saddle point that the adversarial training meets in most of cases. Our results in Table B1 suggest that without concurrent adversarial training and weight pruning, it becomes extremely difficult to adversarially train a small model from scratch even using different initialization schemes and optimizers.

4.3. Possible Benefit of Over-Parameterization

It is clear from Sec. 4.1 and 4.2 that in the adversarial setting, pruning from a large model is useful, which yields benefits in both natural test accuracy and adversarial robustness. By contrast, these advantages are not provided by adversarially training a small model from scratch. Such intriguing results could be explained from the *benefit of over-parameterization* [54, 1, 2], which shows that training neural networks possibly reaches the global solution when the number of parameters is larger than that is statistically required to fit the training data. In the similar spirit, in adversarial training setting, the larger, over-parameterized models lead to good convergence while adversarially trained small models are stuck at the saddle points frequently. These two observations have motivated us to propose a framework that can benefit from larger models during adversarial training and at the same time reduce the models' size. As a result, the remained weights preserve

Table 3: **Natural test accuracy/adversarial test accuracy** (in %) on **CIFAR10 by LeNet** of [column ii] naturally trained model with different size w, [column iii] adversarially trained model with different size w, [columns iv–vii] concurrent adversarial training and weight pruning from a large size to a small size.

w	nat baseline	adv baseline	1	2	4	8
1	74.84/0.01	10.00/10.00	-	-	-	-
2	78.41/0.07	55.03/33.29	50.3/31.33	-	-	-
4	83.36/0.19	65.01/36.30	**53.30/32.41**	62.77/34.52	-	-
8	85.12/0.55	72.80/37.67	52.27/31.91	62.22/35.42	70.50/37.92	-
16	87.22/0.93	74.91/38.65	51.28/31.30	**62.10/35.55**	**70.59/37.93**	**71.93/39.00**

Table 4: **Natural test accuracy/adversarial test accuracy** (in %) on **CIFAR10 by ResNet** of [column ii] naturally trained model with different size w, [column iii] adversarially trained model with different size w, [columns iv–vii] concurrent adversarial training and weight pruning from a large size to a small size.

w	nat baseline	adv baseline	1	2	4	8
1	84.23/0.00	57.16/34.40	-	-	-	-
2	87.05/0.00	71.16/42.45	64.53/37.90	-	-	-
4	91.93/0.00	77.35/44.99	64.36/37.78	73.21/43.14	-	-
8	93.11/0.00	77.26/47.28	**64.52/38.01**	**73.36/43.17**	78.12/45.49	-
16	94.80/0.00	82.71/49.31	64.17/37.99	71.80/42.86	**78.85/47.19**	**81.83/48.00**

Table 5: **Natural test accuracy/adversarial test accuracy** (in %) on **MNIST** for validating the lottery ticket hypothesis in the adversarial setting.

w	1	2	4	8
2	11.35/11.35	-	-	-
4	11.35/11.35	11.35/11.35	-	-
8	11.35/11.35	97.36/90.19	98.64/94.66	-
16	11.35/11.35	11.35/11.35	98.42/91.63	98.96/95.49

adversarial robustness.

5. Pruning Schemes and Transfer Attacks

In this section, we examine the performance of the proposed concurrent adversarial training and weight pruning under different pruning schemes (i.e., filter/column/irregular pruning) and transfer attacks. The proposed framework is tested on CIFAR10 using VGG-16 and ResNet-18 networks, as shown in Figure 3. As we can see, the natural and adversarial test accuracy decrease as the pruned size decreases. Among different pruning schemes, the irregular pruning performs the best while the filter pruning performs the worst in both natural and adversarial test accuracy. That is because in addition to weight sparsity, filter pruning imposes the structure constraint, which restricts the pruning granularity compared to the irregular pruning. Moreover, irregular pruning preserves the accuracy against different pruned sizes. The reason is that the weight sparsity is beneficial to mitigate the overfitting issue [17], and the adversarial training suffers a more significant overfitting than the natural training [36].

In Table C1, we evaluate the performance of our PGD adversary based robust model against C&W ℓ_∞ attacks. As we can see, the concurrent adversarial training and weight pruning yields the pruned model robust to transfer attacks. In particular, the pruned model is able to achieve better adversarial test accuracy than that of the original model prior to pruning (baseline).

Furthermore, we design a cross transfer attack experiment. Consider the baseline models in Table 2, when $w = 1, 2$, the models are not well-trained so we generate adversarial examples by PGD attack from baseline model with $w = 4, 8, 16$ and apply them to test the pruned models. In Table 6, the results show that even the worst case in each pruned model, the adversarial test accuracy is also higher than that of the pruned models in Table 2. The results imply that the model is most vulnerable against adversarial examples generated by itself, regardless of the size of the model.

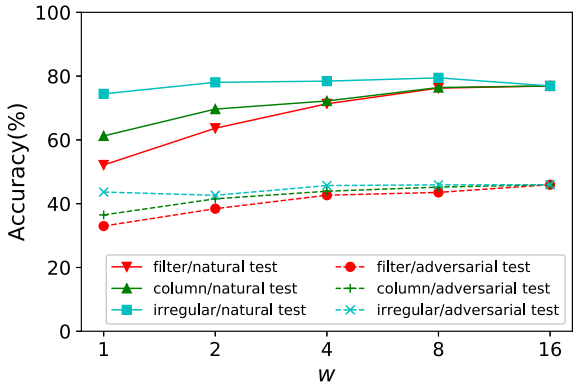
6. Supplementary Details of Experiment Setup

We use LeNet for MNIST, and LeNet, VGG-16 and ResNet-18 for CIFAR10. The LeNet models used here follow the work [33]. Batch normalization (BN) is applied in VGG-16 and ResNet-18. More details about the network structures are listed in Table 1.

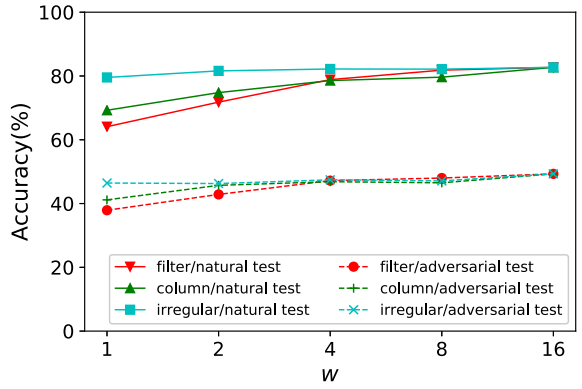
To solve the inner max problem in (1), we set PGD adversary iterations as 40 and 10, step size α as 0.01 and

Table 6: Adversarial test accuracy (in %) on MNIST against transfer attack from baseline models (row) when $w \in \{4, 8, 16\}$ to pruned models (column) $m - n$ which means pruned from original model with $w = m$ to small model with $w = n$.

w	16-1	8-1	4-1	16-2	8-2	4-2	16-4	8-4	16-8
4	91.70	93.77	**91.43**	94.95	95.45	**93.55**	97.26	96.56	97.56
8	92.13	**93.47**	92.06	94.40	**94.30**	94.15	96.21	**95.27**	96.46
16	93.05	94.3	93.07	**94.37**	95.72	94.75	**95.61**	96.31	**95.37**



(a) VGG-16



(b) ResNet-18

Figure 3: Natural and adversarial test accuracy of the proposed framework of concurrent adversarial training and weight pruning on CIFAR10. Filter, column, and irregular pruning schemes are applied in the proposed framework respectively. Weight pruning is performed from size of $w = 16$ to sizes of $w = 1, 2, 4, 8$. The solid lines denote natural accuracy when pruning from the size of $w = 16$ to different sizes, and the dashed lines denote the corresponding adversarial accuracy.

$2/255$, the ℓ_∞ bound as 0.3 and $8/255$ for MNIST and CIFAR respectively, and all pixel values are normalized in $[0, 1]$. We use Adam with learning rate 1×10^{-4} to train our LeNet for 83 epochs as suggested by the released code of [33]. During pruning we set $\rho = 1 \times 10^{-3}$ and $K = 30$ for Algorithm 1. Moreover, there are controversial on the baselines of CIFAR and we do the following to ensure our baselines are strong enough:

1. We follow the suggestions by [31] to train our models with a larger learning rate 0.1 as initial learning rate.
2. We train all models in CIFAR with 300 epochs and divide the learning rate by 10 times at epoch 80 and epoch 150 following the [33].
3. Liu *et al.* [31] suggests that models trained from scratch need fair training time to compare with pruned models. Therefore, we double the training time if the loss is still descent at the end of the training.
4. Since there is always a trade off between natural accuracy and adversarial accuracy, we report accuracy when the models achieve the lowest average loss for both natural and adversarial images on test dataset.

Hence, we believe that in our setting, we have fair baselines for training from scratch.

7. Conclusion

Min-max robust optimization based adversarial training can provide a notion of security against adversarial attacks. However, adversarial robustness requires a significant larger capacity of the network than that for the natural training with only benign examples. This paper proposes a framework of concurrent adversarial training and weight pruning that enables model compression while still preserving the adversarial robustness and essentially tackles the dilemma of adversarial training. Furthermore, this work studies two hypotheses about weight pruning in the conventional setting and finds that weight pruning is essential for reducing the network model size in the adversarial setting, and that training a small model from scratch even with inherited initialization from the large model cannot achieve adversarial robustness and high standard accuracy at the same time. This work also systematically investigates the effect of different pruning schemes on adversarial robustness and model compression.

Acknowledgments

This work is partly supported by the National Science Foundation CNS-1932351, Institute for Interdisciplinary Information Core Technology (IIISCT) and Zhongguancun Haihua Institute for Frontier Information Technology.

References

[1] Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. Learning and generalization in overparameterized neural networks, going beyond two layers. *arXiv preprint arXiv:1811.04918*, 2018. 6

[2] Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. *arXiv preprint arXiv:1811.03962*, 2018. 6

[3] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International Conference on Machine Learning*, pages 274–283, 2018. 1

[4] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016. 1

[5] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1):1–122, 2011. 4

[6] Adrian Carrio, Carlos Sampedro, Alejandro Rodriguez-Ramos, and Pascual Campoy. A review of deep learning methods and applications for unmanned aerial vehicles. *Journal of Sensors*, 2017, 2017. 1

[7] Hongge Chen, Huan Zhang, Pin-Yu Chen, Jinfeng Yi, and Cho-Jui Hsieh. Attacking visual language grounding with adversarial examples: A case study on neural image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2587–2597, 2018. 1

[8] Pin-Yu Chen, Yash Sharma, Huan Zhang, Jinfeng Yi, and Cho-Jui Hsieh. Ead: elastic-net attacks to deep neural networks via adversarial examples. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 1

[9] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pages 15–26. ACM, 2017. 1

[10] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning (ICML)*, pages 160–167. ACM, 2008. 1

[11] Daniel J Fagnant and Kara Kockelman. Preparing a nation for autonomous vehicles: opportunities, barriers and policy recommendations. *Transportation Research Part A: Policy and Practice*, 77:167–181, 2015. 1

[12] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2019. 3, 5

[13] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010. 11

[14] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *2015 ICLR*, arXiv preprint arXiv:1412.6572, 2015. 1

[15] Yiwen Guo, Anbang Yao, and Yurong Chen. Dynamic network surgery for efficient dnns. In *Advances In Neural Information Processing Systems*, pages 1379–1387, 2016. 2, 3

[16] Yiwen Guo, Chao Zhang, Changshui Zhang, and Yurong Chen. Sparse dnns with improved adversarial robustness. In *Advances in Neural Information Processing Systems*, pages 240–249, 2018. 2

[17] Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems*, pages 1135–1143, 2015. 2, 3, 7

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015. 11

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1, 6

[20] Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1389–1397, 2017. 2, 3

[21] Mingyi Hong, Zhi-Quan Luo, and Meisam Razaviyayn. Convergence analysis of alternating direction method of multipliers for a family of nonconvex problems. *SIAM Journal on Optimization*, 26(1):337–364, 2016. 4

[22] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. In *International Conference on Machine Learning*, pages 2142–2151, 2018. 1

[23] Arbaaz Khan and Martial Hebert. Learning safe recovery trajectories with deep neural networks for unmanned aerial vehicles. In *2018 IEEE Aerospace Conference*, pages 1–9. IEEE, 2018. 1

[24] Alexey Kurakin, Ian J Goodfellow, and Samy Bengio. Adversarial machine learning at scale. In *International Conference on Learning Representations*, 2016. 2

[25] Ji Lin, Chuang Gan, and Song Han. Defensive quantization: When efficiency meets robustness. *ICLR*, 2019. 1

[26] Ji Lin, Yongming Rao, Jiwen Lu, and Jie Zhou. Runtime neural pruning. In *Advances in Neural Information Processing Systems*, pages 2181–2191, 2017. 3

[27] Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 936–944. IEEE, 2017. 1

[28] Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen

Ghafoorian, Jeroen Awm Van Der Laak, Bram Van Ginneken, and Clara I Sánchez. A survey on deep learning in medical image analysis. *Medical image analysis*, 42:60–88, 2017. 1

[29] Xuanqing Liu, Yao Li, Chongruo Wu, and Cho-Jui Hsieh. Adv-BNN: Improved adversarial defense through robust bayesian neural network. In *International Conference on Learning Representations*, 2019. 1

[30] Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2736–2744, 2017. 3

[31] Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. Rethinking the value of network pruning. In *International Conference on Learning Representations*, 2019. 3, 5, 8

[32] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 6

[33] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018. 1, 2, 3, 4, 7, 8

[34] Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60, 2014. 1

[35] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016. 1

[36] Ludwig Schmidt, Shibani Santurkar, Dimitris Tsipras, Kunal Talwar, and Aleksander Madry. Adversarially robust generalization requires more data. In *Advances in Neural Information Processing Systems*, pages 5019–5031, 2018. 3, 7

[37] Xiaoshuang Shi, Manish Sapkota, Fuyong Xing, Fujun Liu, Lei Cui, and Lin Yang. Pairwise based deep ranking hashing for histopathology image classification and retrieval. *Pattern Recognition*, 81:14–22, 2018. 1

[38] Chuanbiao Song, Kun He, Liwei Wang, and John E. Hopcroft. Improving the generalization of adversarial training with domain adaptation. In *International Conference on Learning Representations*, 2019. 1

[39] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, volume 4, page 12, 2017. 1

[40] Guanhong Tao, Shiqing Ma, Yingqi Liu, and Xiangyu Zhang. Attacks meet interpretability: Attribute-steered detection of adversarial samples. In *Advances in Neural Information Processing Systems*, pages 7728–7739, 2018. 1

[41] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. Robustness may be at odds with accuracy. In *International Conference on Learning Representations*, 2019. 1, 2

[42] Huaxia Wang and Chun-Nam Yu. A direct approach to robust deep learning using adversarial networks. In *International Conference on Learning Representations*, 2019. 1

[43] Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. In *Advances in neural information processing systems*, pages 2074–2082, 2016. 2, 3

[44] Kai Y Xiao, Vincent Tjeng, Nur Muhammad Mahi Shafiullah, and Aleksander Madry. Training for faster adversarial robustness verification via inducing relu stability. In *International Conference on Learning Representations*, 2019. 2

[45] Kaidi Xu, Hongge Chen, Sijia Liu, Pin-Yu Chen, Tsui-Wei Weng, Mingyi Hong, and Xue Lin. Topology attack and defense for graph neural networks: An optimization perspective. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence (IJCAI-19)*, 2019. 1

[46] Kaidi Xu, Sijia Liu, Gaoyuan Zhang, Mengshu Sun, Pu Zhao, Quanfu Fan, Chuang Gan, and Xue Lin. Interpreting adversarial examples by activation promotion and suppression. *arXiv preprint arXiv:1904.02057*, 2019. 1

[47] Kaidi Xu, Sijia Liu, Pu Zhao, Pin-Yu Chen, Huan Zhang, Quanfu Fan, Deniz Erdogmus, Yanzhi Wang, and Xue Lin. Structured adversarial attack: Towards general implementation and better interpretability. In *International Conference on Learning Representations (ICLR)*, 2019. 1

[48] Ziang Yan, Yiwen Guo, and Changshui Zhang. Deep defense: Training dnns with improved adversarial robustness. In *Advances in Neural Information Processing Systems*, pages 417–426, 2018. 1

[49] Tien-Ju Yang, Yu-Hsin Chen, and Vivienne Sze. Designing energy-efficient convolutional neural networks using energy-aware pruning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5687–5695, 2017. 2, 3

[50] Tianyun Zhang, Shaokai Ye, Kaiqi Zhang, Jian Tang, Wujie Wen, Makan Fardad, and Yanzhi Wang. A systematic dnn weight pruning framework using alternating direction method of multipliers. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 184–199, 2018. 2, 3

[51] Tianyun Zhang, Kaiqi Zhang, Shaokai Ye, Jiayu Li, Jian Tang, Wujie Wen, Xue Lin, Makan Fardad, and Yanzhi Wang. Adam-admm: A unified, systematic framework of structured weight pruning for dnns. *arXiv preprint arXiv:1807.11091*, 2018. 2, 3

[52] Pu Zhao, Sijia Liu, Yanzhi Wang, and Xue Lin. An admm-based universal framework for adversarial attacks on deep neural networks. In *2018 ACM Multimedia Conference on Multimedia Conference*, pages 1065–1073. ACM, 2018. 1

[53] Pu Zhao, Kaidi Xu, Tianyun Zhang, Makan Fardad, Yanzhi Wang, and Xue Lin. Reinforced adversarial attacks on deep neural networks using admm. In *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 1169–1173. IEEE, 2018. 1

[54] Difan Zou, Yuan Cao, Dongruo Zhou, and Quanquan Gu. Stochastic gradient descent optimizes over-parameterized deep relu networks. *arXiv preprint arXiv:1811.08888*, 2018. 6