Article

Robust CNN Compression Framework for Security-Sensitive Embedded Systems

Jeonghyun Lee and Sangkyun Lee

School of Cybersecurity, Korea University, Seoul 02841, Korea; nomar0107@korea.ac.kr

* Correspondence: sangkyun@korea.ac.kr

Abstract: Convolutional neural networks (CNNs) have achieved tremendous success in solving complex classification problems. Motivated by this success, there have been proposed various compression methods for downsizing the CNNs to deploy them on resource-constrained embedded systems. However, a new type of vulnerability of compressed CNNs known as the adversarial examples has been discovered recently, which is critical for security-sensitive systems because the adversarial examples can cause malfunction of CNNs and can be crafted easily in many cases. In this paper, we proposed a compression framework to produce compressed CNNs robust against such adversarial examples. To achieve the goal, our framework uses both pruning and knowledge distillation with adversarial training. We formulate our framework as an optimization problem and provide a solution algorithm based on the proximal gradient method, which is more memory-efficient than the popular ADMM-based compression approaches. In experiments, we show that our framework can improve the trade-off between adversarial robustness and compression rate compared to the existing state-of-the-art adversarial pruning approach.

Keywords: model compression; adversarial robustness; weight pruning; adversarial training; distillation; embedded system; secure AI

Citation: Lee, J.; Lee, S. Robust CNN Compression Framework for Security-Sensitive Embedded Systems. *Appl. Sci.* **2021**, *11*, 1093. <https://doi.org/10.3390/app11031093>

Academic Editor: Kyungtae Kang
Received: 31 October 2020
Accepted: 22 January 2021
Published: 25 January 2021

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

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

In the past few years, convolutional neural networks (CNNs) have achieved great success in many applications including image classification and object detection. Despite the success, the excessively large amount of learning parameters and the vulnerability for the adversarial examples [1–8] are making it difficult to deploy CNNs especially on resource-constrained environments such as smartphones, automobiles, and wearable devices. To overcome this drawback, various model compression methods have been proposed, where many are based on weight pruning [9–17]. Weight pruning generates sparse learning weights by solving an optimization problem with sparsity constraints on the weights, and then the actual compression is accomplished by removing zero weights from a trained model. Although their approach is quite simple, state-of-the-art weight pruning methods [16,17] achieve a high compression rate with little drop in accuracy.

On the other hand, it has been reported that even the state-of-the-art CNNs are vulnerable to adversarial attacks [1–8]. Adversarial attacks are accomplished by using perturbed inputs which cause misclassification where modification is nearly imperceptibly small. Such perturbation can be easily produced by exploiting the gradient information of the target neural network [1,4,6]. Furthermore, some works show that adversary can even generate adversarial examples without knowing anything about the target neural network [5]. Adversarial training [1,6] has been proposed as a countermeasure to adversarial attacks bringing robustness to neural networks against adversarial inputs. This method trains a classifier not only with training examples but also with adversarial examples generated actively by the defender for known types of adversarial perturbations. In particular, projected gradient descent attack [6]-based adversarial training is known to provide high robustness

against the first-order adversary [1,4,6]. However, it has been shown that adversarial training requires a significantly large capacity of the neural network to achieve high accuracy on both original and adversarial examples [6].

Recently, the vulnerability of the compressed neural network is raised as an issue [18]. As shown in Madry et al. [6] the adversarial robustness of compressed neural networks is hard to achieve due to the lack of its architectural capacity. This prevents the compressed neural network from being deployed to a trust-sensitive domain. Despite the seriousness of this problem, only a few methods have been proposed [19,20]. One notable technique is to consider adversarial robustness and model compression at the same time. Ye et al. [19] and Gui et al. [20] formulated an optimization problem by combining adversarial training with pruning and solved it with the alternating direction method of multiplier (ADMM) framework. These works demonstrated that considering weight pruning and adversarial training concurrently can show a better trade-off between robustness and compression rate than considering them separately. However, the ADMM framework requires two auxiliary tensors each of which has the same size as the learning parameters tensor of a CNN: this leads to a heavy memory burden for a resource-constrained environment. In this paper, we show that the joint optimization of pruning and adversarial training can be solved more memory efficiently using the proximal gradient method (PGM) without any auxiliary tensors.

Furthermore, we found that consistently providing information about the pretrained original network during adversarial training can improve the robustness of the resulting compressed network. With this intuition, we propose a novel robust pruning framework that jointly uses pruning and knowledge distillation [21] within the adversarial training procedure. Knowledge distillation is a technique to transfer the information of a network (teacher) to another network (student) by minimizing the gap between the SoftMax outputs of the two networks. In our framework, we use a pretrained original network as the teacher and provide its SoftMax output to a student network being compressed. We summarize our contribution as follows:

- We propose a new robust weight compression framework for CNNs that uses pruning and knowledge distillation jointly within the adversarial training procedure. Our method is described as an optimization problem which deals with pruning, knowledge distillation, and adversarial training concurrently.
- We show that our optimization problem can be solved with the proximal gradient method. Although the popular ADMM approach can also solve our optimization problem, it must keep two auxiliary tensors during optimization which can be a burden for a memory-constrained environment. Our proximal gradient-based approach solves the optimization problem without using any auxiliary tensor.
- In experiments, we demonstrated that the knowledge distillation in our framework improves the adversarial robustness of the compressed CNNs. In addition, our method showed a better trade-off between adversarial robustness and compression rate compared to the state-of-the-art methods [15,19,22].

2. Related Works

2.1. Adversarial Attacks

Adversarial attacks try to find allowable perturbations to change the prediction result of the target network. In the image classification domain, the set of allowable perturbations is generally defined by bounding the ℓ_p norm of perturbation to satisfy an imperceptibility constraint. Such perturbation can be generated by exploiting the information of the target network. According to the amount of this information, adversarial attacks are categorized into the black-box and white-box attacks. A black-box attack assumes a weak adversary who does not have any information about the target model. In this situation, the adversary must rely on query access for chosen input data [5] or the transferability of adversarial examples [2,3]. In a white-box setting, an adversary can access the details of the target model such as the structure, the parameters, the training dataset, etc. Based on the strong

assumption, most white-box attack methods [1,4,6] exploit the first-order information of the target model to generate sophisticated perturbations. In this paper, we focus on the white-box attacks because it is important to study such attacks to implement effective defenses.

2.2. Adversarial Training

Adversarial training is a simple and intuitive learning strategy to enhance the robustness of a neural network against adversarial attacks. It generates adversarial examples using a first-order white-box attack [1,4,6] while training a neural network so that the network will correctly classify not only the training examples but also the generated examples. Adversarial training with a single-step attack such as the fast gradient sign method (FGSM) [1] is known to suffer from so-called label leaking [23] caused by the correlation between perturbation and true label. To prevent label leaking and to generate strong adversarial examples, Madry et al. [6] proposed projected gradient descent (PGD) attack-based adversarial training.

2.3. Weight Pruning

Weight pruning is a model compression technique to make unimportant learning weights to the zero value resulting in sparse weights, and thereby to remove redundant connections or components from a neural network. According to the unit of pruning, weight pruning is categorized into element-wise pruning and filter-wise pruning.

In their early stage, pruning methods focused on element-wise pruning that generates irregular sparsity patterns. To set the values of redundant weights to zero, element-wise pruning [9] measures the importance of weights usually by their absolute values. Han et al. [10] showed that this simple pruning process can be effectively combined with weight quantization and Huffman coding to achieve further compression.

Filter-wise pruning is getting more interest since it is more adequate for GPU acceleration as well as compressing convolution filters in CNNs. Some primary works prune the filters of CNN by measuring their importance by ℓ_2 norm [13] or by the number of effects on activation map [12]. Based on these works, several advanced filter pruning methods [14–17,24] have been proposed by varying the ways of measuring the importance of each filter and the composition of the pruning procedure.

2.4. Knowledge Distillation

The main idea of the knowledge distillation [21] is to transfer the knowledge of a trained teacher network to a student network by training the student network using the input and the SoftMax output of the teacher. In the early stage, it is usually applied for model compression and achieved by transferring the knowledge of an over-parameterized teacher model to a smaller student model. Bucila et al. [25] primarily used this strategy with unlabeled synthesized data to transfer the knowledge of a large ensemble teacher. Hinton et al. [21] formally defined the knowledge distillation loss with temperature and showed that distillation is effective for transferring knowledge with the original training dataset.

Distillation also can be used as a defense to adversarial examples. The defensive distillation [22] achieves adversarial robustness by applying distillation on student and teacher models which have the same structure. However, it has been shown that the defense can be easily broken [4].

Many methods have been proposed to improve the effectiveness of distillation. Distillation with boundary support samples [26] tries to improve the generalization performance of a student model by conducting the distillation with the adversarial examples near the decision boundary. Distillation with teacher assistant [27] fills the gap between student and teacher models by using intermediate models called teacher assistants.

2.5. Adversarially Robust Model Compression

To preserve the robustness of the compressed model, adversarial pruning can be applied in most cases which combines the ideas of adversarial training and pruning. Ye et al. [19] and Gui et al. [20] formulated an objective which includes both adversarial training and sparsity constraints, and showed that applying adversarial training and pruning concurrently generated better robustness than applying them separately. Xie et al. [28] used blind adversarial training [29] during adversarial pruning which generated adversarial examples dynamically during adversarial training to reduce the sensitivity to the budget of adversarial examples. Madaan et al. [30] proposed a new pruning criterion to reduce the vulnerability of latent space represented by the difference between the activation map of adversarial example and its original input.

Some works also considered the adversarial robustness of different types of compression to pruning. Bernhard et al. [31] observed that the change of adversarial robustness according to the different levels of quantization. Lin et al. [32] proposed a defensive quantization method that reduced the sensitivity to the input of the neural network. Goldblum et al. [33] used knowledge distillation to transfer the robustness of an over-parameterized model to a predefined smaller model.

3. Methods

The main objective of our suggested method is to preserve the adversarial robustness of CNNs during the pruning procedure. An adversarially robust CNN should demonstrate high generalization performance on both original and adversarial inputs. One existing approach to generate such a CNN is adversarial pruning, which is the combination of adversarial training and pruning. However, adversarial pruning alone is not enough to achieve the goal since the decision boundary of the original network is quickly collapsed during the initial stage of the pruning procedure due to the decrease of network capacity, which results in a large decrease in generalization performance on the original inputs. To solve this problem, we propose a novel robust pruning framework that combines adversarial pruning with knowledge distillation. Using the combination, we can provide information of the decision boundary of the original network consistently during adversarial pruning.

In this section, we first describe our definition of the adversary, and then formulate our entire framework as a single optimization problem showing that it can be solved efficiently by the proximal gradient method without using any auxiliary tensors.

3.1. The Attack Model

Before describing our proposed method, we first elaborate on the attack model. For the purpose, let us define the SoftMax output of a CNN with weight parameter $w \in \mathbb{R}^p$ as $f(\cdot; w)$. Let the data pairs $\{(x_i, y_i)\}_{i=1}^n$ be a training dataset. Here, $x_i \in \mathbb{R}^d$ is an input and $y_i \in \{0, 1\}^k$ is the corresponding one-hot encoded true label. Then, the training procedure of CNN can be described as the following optimization problem.

$$w^* \in \arg \min_{w \in \mathbb{R}^p} \frac{1}{n} \sum_{i=1}^n \mathcal{L}(f(x_i; w), y_i). \quad (1)$$

Here, $\mathcal{L}$ is the cross-entropy loss [34] that indicates the gap between the SoftMax output and the true label. For the given discrete probability distribution p and q , the cross-entropy loss is defined as follows:

$$\mathcal{L}(q, p) = - \sum_k p_k \log q_k.$$

The objective of the adversary is changing the prediction result of the trained CNN by adding an imperceptible perturbation on the input image, which can be generated by both targeted attack and untargeted attack. In the targeted attack, the adversary generates perturbation that minimizes the cross-entropy between the SoftMax output and the pre-

defined target label that is different from the true label. Given input data pair (x, y) and target label y_t , the targeted attack can be described as follows:

$$\min_{\delta \in \Delta} \mathcal{L}(f(x + \delta; w^*), y_t), \quad \text{such that } y_t \neq y. \quad (2)$$

Since the effectiveness of the targeted adversarial attack varies depending on the chosen target label, most robust pruning literature [19,20,30] focus on the untargeted attack for experimenting with adversarial examples, and we take the same approach. In untargeted adversarial attack, we generate adversarial examples by maximizing the cross-entropy between the SoftMax output and the true label:

$$\max_{\delta \in \Delta} \mathcal{L}(f(x + \delta; w^*), y). \quad (3)$$

Also, we suppose a white-box setting where the adversary has full knowledge about the target CNN. In this case, the adversary can solve (2) and (3) by exploiting the gradient of the target CNN.

3.2. Adversarial Pruning with Distillation

Adversarial training is a type of robust optimization procedure which can be stated by the following min-max problem:

$$w_{den}^* \in \arg \min_{w \in \mathbb{R}^p} \frac{1}{n} \sum_{i=1}^n \max_{\delta \in \Delta} \mathcal{L}(f(x_i + \delta; w), y_i). \quad (4)$$

To solve the inner maximization problem of (4), we consider the projected gradient descent (PGD) attack method [6] with an ℓ_∞ -norm feasible set. For a given data pair (x, y) , the PGD attack is defined as follows:

$$x^{t+1} = \Pi_{\mathcal{B}(x, \epsilon)}(x^t + \alpha \cdot \text{sgn}(\nabla_{x^t} \mathcal{L}(f(x^t + \delta; w), y))). \quad (5)$$

Here, $\Pi_{\mathcal{B}(x, \epsilon)}$ is a projection operation to the ℓ_∞ -norm ball around x defined as $\mathcal{B}(x, \epsilon) := \{x + \delta : \|\delta\|_\infty \leq \epsilon\}$. Let us note that uniformly distributed random noise is added to x in the initial stage of the PGD attack to prevent the label leaking problem [23]. The solution of (4) which we denote as w_{den}^* is generally non-sparse since there is no sparse constraint on this optimization problem. By adding a sparse regularization term to (4), we can obtain the objective of adversarial pruning,

$$w_{spa}^* \in \arg \min_{w \in \mathbb{R}^p} \frac{1}{n} \sum_{i=1}^n \max_{\delta \in \Delta} \mathcal{L}(f(x_i + \delta; w), y_i) + \lambda \|w\|_0, \quad (6)$$

where $\lambda > 0$ is a hyperparameter to control the sparsity of w .

Generally, the solution of (1), denoted by w^* , is used as initial weights for solving (6). Here, our question is how we effectively preserve the accuracy of w^* on original inputs during adversarial pruning procedure. The accuracy on the original inputs is largely dropped during the adversarial pruning procedure since the one-hot encoded label y_i in (6) does not contain any information about the decision boundary of w^* .

To consistently provide the information of w^* during pruning, we combine the knowledge distillation idea with adversarial pruning. In our method, the pretrained network works as a teacher and provides SoftMax output $f^t(\cdot; w^*)$ on original input during adversarial pruning procedure. The proposed objective is formulated as follows:

$$\min_{w \in \mathbb{R}^p} \frac{1}{n} \sum_{i=1}^n (1 - \alpha) \mathcal{L}(f(x_i + \delta; w), y_i) + \alpha t^2 \mathcal{L}(f^t(x_i + \delta; w), f^t(x_i; w^*)) + \lambda \|w\|_0. \quad (7)$$

Here, δ is the solution of (3) and t is a distillation hyperparameter [21]. The t^2 is multiplied in front of the second term to prevent the shirking of gradient problem [21]. The second term in (7) is distillation loss which indicates the cross-entropy between SoftMax output of the currently pruned model $f(\cdot; w)$ and the teacher model $f(\cdot; w^*)$. The overall formulation of (7) can be interpreted as the linear combination of the adversarial pruning loss (6) and the distillation loss. By solving (7), we can obtain a sparse but robust solution that approximates the decision boundary of w^* . Our framework can be extended for filter pruning by replacing the third regularizer term with the number of non-zero filters as follows:

$$\min_{w \in \mathbb{R}^p} \frac{1}{n} \sum_{i=1}^n (1 - \alpha) \mathcal{L}(f(x_i + \delta; w), y_i) + \alpha t^2 \mathcal{L}(f^t(x_i + \delta; w), f^t(x_i; w^*)) + \lambda \sum_{g=1}^G \mathbb{1}[\|w_g\|_2 \neq 0].$$

Here, G is the number of filters and w_g is the weight vector of g th filter.

3.3. Optimization

Most of the adversarial pruning approaches use the alternative direction method of multiplier (ADMM) method to solve the resulting optimization problem, for example, Ye et al. [19] and Gui et al. [20]. However, by construction, the ADMM requires using two additional tensors to the learning weights during optimization, which can be preventive on a resource-constrained environment with limited memory. Here, we suggest another algorithm based on the proximal gradient method to solve our proposed optimization problem (7) which does not require such auxiliary tensors. For simplicity, we denote the linear combination of two cross-entropy loss in (7) by $\mathcal{L}_{\text{APD}}$:

$$\mathcal{L}_{\text{APD}}(w) = \frac{1}{n} \sum_{i=1}^n (1 - \alpha) \mathcal{L}(f(x_i + \delta; w), y_i) + \alpha t^2 \mathcal{L}(f^t(x_i + \delta; w), f^t(x_i + \delta; w^*)). \quad (8)$$

Here, APD stands for adversarial pruning with distillation. Then we can rewrite (7) as

$$\min_{w \in \mathbb{R}^p} \mathcal{L}_{\text{APD}}(w) + \lambda \|w\|_0. \quad (9)$$

By applying a second order Taylor approximation on w_k and Hessian approximation with $\nabla^2 \mathcal{L}_{\text{APD}}(w_k) \approx \frac{1}{\eta_k} I_{p \times p}$ for a $\eta_k > 0$ to (9), we obtain the following formulation:

$$\mathcal{L}_{\text{APD}}(w) \approx \mathcal{L}_{\text{APD}}(w_k) + \nabla \mathcal{L}_{\text{APD}}(w_k)^\top (w - w_k) + \frac{1}{2\eta_k} \|w - w_k\|^2.$$

Here, $I_{p \times p}$ indicates the identity matrix where the shape is $p \times p$. Based on this successive approximation result, the weight update can be formulated as follows:

$$w_{k+1} = \arg \min_{w \in \mathbb{R}^p} \mathcal{L}_{\text{APD}}(w_k) + \nabla \mathcal{L}_{\text{APD}}(w_k)^\top (w - w_k) + \frac{1}{2\eta_k} \|w - w_k\|^2 + \lambda \|w\|_0.$$

By removing the redundant parts of the above weight update equation, we can obtain

$$w_{k+1} = \arg \min_{w \in \mathbb{R}^p} \frac{1}{2\eta_k} \left(2\eta_k \nabla \mathcal{L}_{\text{APD}}(w_k)^\top w + \|w\|^2 - 2w^\top w_k \right) + \lambda \|w\|_0.$$

We can rewrite the above equation as follows:

$$w_{k+1} = \arg \min_{w \in \mathbb{R}^p} \frac{1}{2\eta_k} \left(\|w\|^2 - 2w^\top (w_k - \eta_k \nabla \mathcal{L}_{\text{APD}}(w_k)) \right) + \lambda \|w\|_0.$$

By adding a constant $\|w_k - \eta_k \nabla \mathcal{L}_{\text{APD}}(w_k)\|^2$, we can obtain

$$w_{k+1} = \arg \min_{w \in \mathbb{R}^p} \frac{1}{2\eta_k} \left(\|w\|^2 - 2w^\top (w_k - \eta_k \nabla \mathcal{L}_{\text{APD}}(w_k)) + \|w_k - \eta_k \nabla \mathcal{L}_{\text{APD}}(w_k)\|^2 \right) + \lambda \|w\|_0.$$

Then, we can get the following equation:

$$w_{k+1} = \arg \min_{w \in \mathbb{R}^p} \frac{1}{2\eta_k} \|w - (w_k - \eta_k \nabla \mathcal{L}_{\text{APD}}(w_k))\|^2 + \lambda \|w\|_0.$$

This is exactly the form of proximal operator which is described as

$$w_{k+1} = \text{prox}_{\eta_k \lambda \| \cdot \|_0} (w_k - \eta_k \nabla \mathcal{L}_{\text{APD}}(w_k)).$$

For each element, proximal operator with ℓ_0 regularization term can be computed as

$$(w_{k+1})_i = \begin{cases} (w_k - \eta_k \nabla \mathcal{L}_{\text{APD}}(w_k))_i, & |(w_k - \eta_k \nabla \mathcal{L}_{\text{APD}}(w_k))_i| > \sqrt{\lambda} \\ 0, & |(w_k - \eta_k \nabla \mathcal{L}_{\text{APD}}(w_k))_i| \leq \sqrt{\lambda} \end{cases}.$$

It is simply the thresholding operation which sets the updated weight parameter smaller than $\sqrt{\lambda}$ to zero. Let us note that by controlling the value of λ , we can explicitly manipulate the sparsity of network. The entire process of our method is described at Algorithm 1.

Algorithm 1: Adversarial Pruning with Distillation (APD)

Input: a distillation temperature t , a learning rate for the student η_s , a learning rate for the teacher η_t , the train dataset $\{(x_i, y_i)\}_{i=1}^n$ where $x_i \in \mathbb{R}^d$, $y_i \in \{0, 1\}^k$, the ℓ_∞ bound ϵ for imperceptibility;
Initialize the student weight vector $w_s \in \mathbb{R}^p$;
Initialize the teacher weight vector $w_t \in \mathbb{R}^p$;
while w_t not converged **do**
 Sample a data pair (x, y) from the train dataset;
 Compute $\mathcal{L}(f(x; w_t), y)$;
 Weight Update: $w_t \leftarrow w_t - \eta_t \nabla \mathcal{L}(f(x; w_t), y)$;
end
while w_s not converged **do**
 Sample a data pair (x, y) from the train dataset;
 For each pixel of x , generate a uniformly random noise
 $\epsilon = (\epsilon_1, \dots, \epsilon_d) \sim \mathcal{U}(-\epsilon, \epsilon)$;
 $x_{adv} \leftarrow x + \epsilon$;
 while x_{adv} not converged **do**
 Update: $x_{adv} = \Pi_{B(x, \epsilon)}(x_{adv} + \alpha \cdot \text{sgn}(\nabla_{x_{adv}} \mathcal{L}(f(x_{adv}, w_s), y)))$;
 end
 Compute the teacher SoftMax output: $f^t(x; w_t)$;
 Compute the student SoftMax output: $f^t(x_{adv}; w_s)$;
 Compute $\mathcal{L}_{\text{APD}}(w_s)$ with (8);
 Update weight: $w_s = \text{prox}_{\eta_s \lambda \| \cdot \|_0} (w_s - \eta_s \nabla \mathcal{L}_{\text{APD}}(w_s))$;
end
return w_s ;

4. Experiments

To demonstrate that our method improves the adversarial robustness of the pruned network, we applied our method on three popular CNNs: LeNet [35] with the MNIST dataset, and VGG16 [36], ResNet18 [37] with the CIFAR10 dataset [38]. The MNIST dataset consists of 28×28 gray-scaled images with 60,000 trainset and 10,000 testset.

The CIFAR10 dataset has 32×32 color images with 50,000 trainset and 10,000 testset. As in Han et al. [10], we used the term “compression rate” to indicate the ratio of the number of zeros to the number of entire weight parameters in a CNN. We denoted the test accuracy on the original images as “original accuracy” and the test accuracy on the adversarial images as “adversarial accuracy”. As in other literature [19,20,33], we consider that the robustness of the model is improved when both the original accuracy and the adversarial accuracy are improved. Otherwise, we consider a model with a higher mean value of the original and adversarial accuracy to be more robust. Given the time spent on the adversarial training for the large networks, we set the number of iterations of projected gradient descent (PGD) attack to 5 for the adversarial training of VGG16 and ResNet18. In this case, we evaluated the adversarial accuracy on both 10 iterations of PGD attack (denoted by PGD10) and 5 iterations of PGD attack (denoted by PGD5). We followed the parameters of Ye et al. [19] for the rest of the PGD attack parameters, which are strong enough to make the adversarial accuracy of the naturally trained LeNet, VGG16, and ResNet18 close to zero. The implementation of our method is available as open source (<https://github.com/JEONGHYUN-LEE/APD>).

4.1. The Effect of Knowledge Distillation

We compared the result of adversarial pruning (denoted by AP) (6) and our method (denoted by APD) (7) to show the effectiveness of the knowledge distillation, for both element-wise pruning and filter pruning. In this comparison, we set the value of α in (7) to 1 to maximize the effect of the SoftMax output of the teacher network. Also, we set the temperature t of the knowledge distillation to 10 for the MNIST dataset, and 100 for the CIFAR10 dataset for a similar reason.

4.1.1. Element-Wise Pruning

Generally, the element-wise pruning [9,10] can achieve higher sparsity with only a few accuracy drops compared to the filter pruning [11–15]. Therefore, we tested the element-wise pruning on the relatively high compression rates ($\times 2$, $\times 3$, $\times 4$) compared to the filter pruning [39]. As in Ye et al. [19], we applied the same sparsity for every convolution layer in the target neural network. For instance, if the compression rate of a given network is determined to $\times 2$, we set the fraction of zero weights in every layer of this network equal to 0.5. With this pruning scheme, we compared the element-wise pruning result of our method (7) with adversarial pruning (6). Both methods were optimized with proximal gradient descent. With this comparison, we demonstrated how much improvement was achieved by the knowledge distillation of our method. The results on MNIST and CIFAR10 are summarized at Tables 1 and 2, respectively.

A popular small network LeNet [35] is enough to achieve a high accuracy on the MNIST dataset. Our baseline LeNet, trained by the original training process achieves the original accuracy of 99.34% and the adversarial accuracy of 0%. With LeNet, our method (APD) showed a large improvement in both original accuracy and adversarial accuracy over the adversarial pruning (AP). In the compression rate of $\times 2$, APD improved the original accuracy by 1.01 and the adversarial accuracy by 2.28% over AP. In the relatively high compression rate of $\times 3$ and $\times 4$, APD achieved a larger improvement in both original accuracy and adversarial accuracy. In particular, the amount of improvement in the adversarial accuracy achieved by APD in the compression rate of $\times 3$ and $\times 4$ was over than 20%. Compared to the baseline performance, APD achieved the compression rate of $\times 4$ with the adversarial accuracy of 94.25% while reducing the original accuracy by about 1%.

We also applied APD and AP to the two CNNs, VGG16 [36] and ResNet18 [37] with the CIFAR10 dataset. Achieving high adversarial robustness on the CIFAR10 dataset is more challenging since it requires a higher architectural capacity of the CNN compared to the MNIST dataset. Our baseline VGG16 achieved the original accuracy of 92.99% and the adversarial accuracy of 0%. Despite the difficulty, APD showed an improvement with VGG16 in the entire compression rates. For instance, in the compression rate of $\times 4$, APD improved the original accuracy by 0.88% and the adversarial accuracy against both PGD5

and PGD10 by more than 1% over AP. Though ResNet18 consists of fewer parameters than VGG16 (11 M vs. 138 M), the generalization performance of ResNet18 for the CIFAR10 dataset is higher than that of VGG16. The baseline ResNet18 showed the original accuracy of 94.40% and the adversarial accuracy of 0.03%. With ResNet18, APD improved the original accuracy and adversarial accuracy against both PGD5 and PGD10 by more than 2% over AP in the entire compression rates. Based on those results, we can conclude that consistently providing the SoftMax output of the baseline CNN with the knowledge distillation improves the adversarial robustness of the element-wise pruning solution.

Table 1. Summary of element-wise pruning results of APD (ours) and AP on MNIST.

Network (Dataset)	Comp Rate	Method	Org Accuracy (%)	Adv Accuracy (%)
LeNet (MNIST)	$\times 2$	AP	97.82	92.83
		APD	98.83	95.11
	$\times 3$	AP	90.94	72.71
		APD	98.55	94.51
	$\times 4$	AP	94.45	71.63
		APD	98.48	94.25

Table 2. Summary of element-wise pruning results of APD (ours) and AP on CIFAR10.

Network (Dataset)	Comp Rate	Method	Org Acc (%)	PGD5	PGD10
				Adv Acc (%)	Adv Acc (%)
VGG16 (CIFAR10)	$\times 2$	AP	81.67	49.20	40.91
		APD	82.44	50.63	42.24
	$\times 3$	AP	80.72	48.70	40.87
		APD	81.77	50.30	42.31
	$\times 4$	AP	79.69	48.77	40.97
		APD	80.57	49.93	42.06
ResNet18 (CIFAR10)	$\times 2$	AP	85.13	51.27	42.45
		APD	87.56	54.65	45.55
	$\times 3$	AP	84.67	51.79	42.64
		APD	86.87	54.40	45.31
	$\times 4$	AP	84.65	51.25	42.55
		APD	86.73	54.23	45.61

4.1.2. Filter Pruning

The filter pruning [11–15] generates the sparse patterns more adequate for GPU acceleration compared to the element-wise pruning [9,10]. However, the sparsity that the filter pruning can achieve is often lower than that of element-wise pruning [39]. Therefore, we set the smaller compression rates of $\times 1.5$, $\times 2$, and $\times 2.5$ than those of the element-wise pruning. As with element-wise pruning, we set the same sparsity for each convolution layer. We compared our method (APD) with the adversarial pruning (AP) to show the effectiveness of the knowledge distillation on the filter pruning. The results on MNIST and CIFAR10 are summarized at Tables 3 and 4, respectively.

With LeNet, APD improved both original accuracy and adversarial accuracy in the entire compression rates. For instance, in the largest compression rate of $\times 2.5$, APD improves the original accuracy by 0.36% and the adversarial accuracy by 1.44%. The improvement

on the original accuracy tends to be smaller than the improvement on the adversarial accuracy since the original accuracy is already closed to that of the baseline network. APD also showed an improvement in both accuracy measures on the CIFAR10 dataset. With VGG16, APD improved the original accuracy significantly in high compression rate. For instance, in the compression rate of $\times 2.5$, the original accuracy is improved by 5.23%. The adversarial accuracy against both PGD5 and PGD10 attacks is also improved by APD. In the compression rate of $\times 2.5$, the adversarial accuracy increases by 2.09% against PGD5 attacks and 0.6% against PGD10 attacks. With ResNet18, APD also showed a consistent improvement on both original accuracy and adversarial accuracy in the entire compression rates. For instance, in the largest compression rate of $\times 2.5$, APD improves the original accuracy by about 2% and adversarial accuracy by about 1% against both PGD5 and PGD10. Those results imply that the knowledge distillation in our method improves the adversarial robustness of the filter pruning solution.

Table 3. Summary of filter-wise pruning results of APD (ours) and AP on MNIST.

Network (Dataset)	Comp Rate	Method	Org Accuracy (%)	Adv Accuracy (%)
LeNet (MNIST)	$\times 1.5$	AP	98.91	95.26
		APD	99.18	96.32
	$\times 2$	AP	98.79	94.95
		APD	99.17	96.21
	$\times 2.5$	AP	98.68	94.58
		APD	99.04	96.02

Table 4. Summary of filter-wise pruning results of APD (ours) and AP on CIFAR10.

Network (Dataset)	Comp Rate	Method	Org Acc (%)	PGD5	PGD10
				Adv Acc (%)	Adv Acc (%)
VGG16 (CIFAR10)	$\times 1.5$	AP	79.91	49.02	41.10
		APD	81.01	50.18	42.62
	$\times 2$	AP	73.69	47.11	40.56
		APD	76.88	48.61	41.22
	$\times 2.5$	AP	69.30	45.10	39.61
		APD	74.53	47.19	40.21
ResNet18 (CIFAR10)	$\times 1.5$	AP	84.57	51.42	42.35
		APD	86.70	54.29	45.42
	$\times 2$	AP	83.37	51.27	42.90
		APD	85.55	53.32	45.59
	$\times 2.5$	AP	82.09	51.65	43.21
		APD	84.02	52.54	44.63

4.2. The Convergence Behavior

To investigate the effect of the knowledge distillation on the convergence behavior of the adversarial pruning, we traced both original accuracy and adversarial accuracy of AP and APD on every epoch. The results on the epoch 0 indicate the initial performance of the currently pruned model where the weight parameters were initialized with the baseline model. We focused on the original accuracy of the early stage of the optimization

to show how well APD preserved the original accuracy of the baseline model during the adversarial pruning.

4.2.1. Element-Wise Pruning

We traced both original accuracy and adversarial accuracy of AP and APD with the element-wise pruning scheme in the compression rate of $\times 2$, $\times 3$ and $\times 4$. The results are described at Figure 1. Let us note that the adversarial accuracy is measured against PGD10. APD achieved a significant improvement in the original accuracy in the early stage of optimization with LeNet, VGG16, and ResNet18. With LeNet, the original accuracy of AP fell to lower than 20% on the first epoch whereas the original accuracy of APD was maintained above 90% across the entire optimization process. With VGG16, the original accuracy of both AP and APD was dropped on the first epoch. However, the amount of decrease in the original accuracy on the first epoch of APD was less than that of AP. For instance, in the compression rate of $\times 4$, the original accuracy on the first epoch of APD was higher than that of AP by about 20%. Moreover, with LeNet and VGG16, APD improved the convergence behavior of both original accuracy and adversarial accuracy compared to AP. For instance, in the compression rate of $\times 3$ with VGG16, APD only required 40 epochs for the average value of the original accuracy and the adversarial accuracy to reach 61.00% (the maximum average value achieved by AP), whereas AP required 46 epochs to achieve that. With ResNet18, APD reduced the drop of original accuracy on the first epoch by about 10% across the entire compression rates though the improvement in the convergence behavior of both original accuracy and adversarial accuracy is smaller than that of other networks.

4.2.2. Filter Pruning

We also traced both original accuracy and adversarial accuracy of AP and APD with the filter pruning scheme in the compression rate of $\times 1.5$, $\times 2$, and $\times 2.5$. The results are described at Figure 2. APD improved the overall convergence behavior of the filter pruning. With LeNet, APD reduced the drop of the original accuracy on the first epoch about 5%. With VGG16, the improvement in the first epoch was more significant. For instance, in the compression rate of $\times 1.5$, APD reduced the drop of the original accuracy on the first epoch by about 20%. Mitigating the drop of original accuracy in the first epoch led to an improvement in the overall convergence behavior. For instance, in the compression rate of $\times 1.5$ with LeNet, APD required 49 epochs for the average value of the original accuracy and the adversarial accuracy to reach 96.63% (the maximum average value achieved by AP), whereas AP required 86 epochs to achieve that. In the compression rate $\times 1.5$ with VGG16, APD required 33 epochs for the average value of both accuracies to reach 54.46% (the maximum average value achieved by AP), whereas AP required 59 epochs to achieve that. With ResNet18, APD also reduced the drop of original accuracy in the initial stage of pruning but the amount of improvement decreased in the high compression rate.

4.3. Comparison with the State-of-the-Art Methods

To show the relative benefit of our method (denoted as APD) compared to other state-of-the-art methods, we also compared APD to Defensive Distillation [22] (denoted as DD), Filter Pruning via Geometric Median [15] (denoted as FPGM), and Ye et al. [19]. The results are summarized at Table 5.

(a) LeNet Accuracy Trajectory on the Element-wise Pruning

(b) VGG16 Accuracy Trajectory on the Element-wise Pruning

(c) ResNet18 Accuracy Trajectory on the Element-wise Pruning

Figure 1. The original accuracy and the adversarial accuracy of AP and APD (ours) with respect to the epoch of the element-wise pruning procedure for (a) LeNet, (b) VGG16, and (c) ResNet18. The left of each row is the result in the compression rate of $\times 2$, the middle of each row is the result in the compression rate of $\times 3$, and the right side of each row is the result in compression rate of $\times 4$. The blue line means the original accuracy and the red line indicates the adversarial accuracy. The solid line is the result of APD and the dashed line is the result of AP.

Table 5. Summary of filter-wise pruning results of APD (ours) and other state-of-the-art methods.

Network (Dataset)	Type	Method	Comp Rate	Org Acc (%)	Adv Acc (%)
VGG16 (CIFAR10)	Pruning	FPGM	$\times 1.3$	93.13	16.17
	Pruning + Defense	APD	$\times 1.5$	81.01	42.62
LeNet (MNIST)	Defense	DD	$\times 1$	93.15	86.57
	Pruning + Defense	APD	$\times 2$	99.17	96.21
LeNet (MNIST)	Pruning + Defense	Ye et al.	$\times 2$	99.01	95.44
	Pruning + Defense	APD	$\times 2$	99.17	96.21
	Pruning + Defense	Ye et al.	$\times 4$	98.87	94.77
	Pruning + Defense	APD	$\times 4$	98.88	94.90
	Pruning + Defense	Ye et al.	$\times 8$	98.07	89.95
	Pruning + Defense	APD	$\times 8$	98.08	91.06
ResNet18 (CIFAR10)	Pruning + Defense	Ye et al.	$\times 2$	81.83	48.00
	Pruning + Defense	APD	$\times 2$	82.09	48.03

DD is a well-known defense strategy that generates a robust model by using knowledge distillation. It trains a teacher model with a high temperature value in a modified SoftMax output and then applies knowledge distillation to a student model whose architecture is the same as that of the teacher model. We compared the original accuracy and the adversarial accuracy of APD and DD with LeNet in the compression rate of $\times 2$. For DD, we set the temperature t as 40 and the number of epochs as 100. In comparison, APD showed about 6% higher original accuracy and 10% higher adversarial accuracy than DD.

FPGM is a SOTA filter pruning method that effectively prunes the redundant filters by measuring the Geometric Median [40] of each filter. To show that the pruning method only is not enough to generate sparse but robust solutions, we compared our pruned VGG16 with the compression rate of $\times 1.5$ to FPGM's pruned VGG16 with the compression rate of $\times 1.3$. APD showed 26.45% higher adversarial accuracy and 12.12% lower original accuracy compared to FPGM. The mean value of the original and the adversarial accuracy of APD is 61.82 and that of FPGM is 54.65. This result demonstrates that the model generated by the pruning method alone is vulnerable to adversarial attack.

Ye et al. is a SOTA robust pruning method. To solve the adversarial pruning (6) problem using alternative direction method of multipliers (ADMM), the method introduced two additional tensors for auxiliary parameters and Lagrangian multipliers. The size of those two tensors is exactly the same as the size of the weight parameters and therefore, it requires two times more memory than the memory required to store the weight parameters during the optimization procedure. On the other hand, APD solves our optimization problem (7) with the proximal gradient descent, which does not require any auxiliary tensor. We compared the result of APD and Ye et al with LeNet and ResNet18. VGG16 was excluded in this comparison since the exact values of the original accuracy and the adversarial accuracy with VGG16 are not available in the original paper of Ye et al. We set the compression rates to $\times 2$, $\times 4$, and $\times 8$ for LeNet, and $\times 2$ for ResNet18. With LeNet, APD slightly improved both original accuracy and adversarial accuracy over Ye et al. in entire compression rates. With ResNet18, APD improved the original accuracy by 0.26% and the adversarial accuracy by 0.03% compared to Ye et al. The adversarial robustness of APD appears to be similar to that of Ye et al.; however, APD requires far less memory than Ye et al. and therefore will be more suitable for generating robust models in memory-constrained environments as we discuss in the next section.

(a) LeNet Accuracy Trajectory on the Filter Pruning

(b) VGG16 Accuracy Trajectory on the Filter Pruning

(c) ResNet18 Accuracy Trajectory on the Filter Pruning

Figure 2. The original accuracy and the adversarial accuracy of AP and APD (ours) with respect to the epoch of the filter pruning procedure for (a) LeNet, (b) VGG16, and (c) ResNet18. The left of each row is the result in the compression rate of $\times 1.5$, the middle of each row is the result in the compression rate of $\times 2$, and the right side of each row is the result in compression rate of $\times 2.5$. The blue line means the original accuracy and the red line indicates the adversarial accuracy. The solid line is the result of APD and the dashed line is the result of AP.

4.4. Computational and Space Complexity

To show the computational and memory efficiency of APD in comparison to other methods, here we provide a short analysis without big O notations. The most dominant part of the training procedure of CNNs in terms of computational complexity is the forward and backward operations. For a given network and input data, we denoted the amount of computation for a forward as F and the amount of computation for a backward as B . In addition, we supposed that the number of iterations for training given network is I_T and the number of iterations for generating adversarial example as I_A . Then, the computational complexity of most of the pruning methods such as FPGM is $I_T \times (F + B)$. DD contains additional forward operations for generating the SoftMax output of the teacher network resulting in $I_T \times (2F + B)$. A relatively large increase of computational complexity for APD and Ye et al. is inevitable since the adversarial training requires an iterative adversarial attack for every iteration. Considering this, the computational complexity of Ye et al. is $I_T \times (F + B + I_A \times F)$, where APD requires $I_T \times (2F + B + I_A \times F)$ since it contains both adversarial training and knowledge distillation.

On the other hand, the most dominant part of the space complexity of the training procedure is the number of learning parameters. To describe the space complexity, let us denote the number of weights of the given network as P . FPGM requires no additional parameter and therefore its complexity is P . The space complexity of DD and APD are $2P$ since they require a teacher and a student network to perform knowledge distillation. Ye et al. requires two additional parameters for ADMM and a large $3P$ space complexity in result. Compared to Ye et al., the analysis shows that APD requires far less memory with the cost of an additional forward step.

4.5. Effectiveness of Knowledge Distillation on Other Attack Methods

To test our method on the other adversarial attacks, we evaluated the adversarial accuracy of our PGD-based trained LeNet (MNIST) against Fast Gradient Sign Method (FGSM) attack [1] and Carlini–Wagner (CW) ℓ_2 attack [4]. For FGSM attack, we set the attack radius ϵ to 0.3. For CW attack, we used ℓ_2 bounded perturbation and set the maximum iterations to 1000. The baseline LeNet showed the original accuracy of 99.41% and the adversarial accuracy of 1.08% against FGSM and 0.48% against CW. The results are described in Table 6. The APD showed higher original accuracy and adversarial accuracy against both FGSM and CW ℓ_2 attacks compared to AP in the entire compression rates. In particular, the improvement on the adversarial accuracy against CW ℓ_2 attack is significant. Those results imply that our PGD-based approach is also effective on the other attack methods.

Table 6. Summary of AP and APD results against FGSM and CW ℓ_2 attacks on the MNIST dataset.

Network (Dataset)	CompRate	Method	Org Acc (%)	FGSM Adv Acc (%)	CW ℓ_2 Adv Acc (%)
LeNet (MNIST)	$\times 1.5$	AP	98.91	97.62	70.23
		APD	99.18	98.27	91.01
	$\times 2$	AP	98.79	97.48	69.77
		APD	99.17	98.28	88.19
	$\times 2.5$	AP	98.68	97.38	77.11
		APD	99.04	98.15	93.30

5. Conclusions

The adversarial robustness of the compressed CNNs is essential for deploying them to the real-world embedded systems. In this paper, we proposed a robust model compression framework for CNNs. Our framework used the knowledge distillation to improve the result of the existing adversarial pruning approach. In several experiments, our framework

showed a significant improvement in the trade-off between the compression rate and the adversarial robustness on the two datasets, MNIST and CIFAR10. We found that the amount of improvement of our framework tends to decrease in the high compression rate. We expect that this phenomenon is due to the large gap in the architectural capacity between the teacher network and the student network. We hope that this phenomenon will be mitigated through future works.

Author Contributions: conceptualization, J.L. and S.L.; methodology, J.L. and S.L.; validation, J.L.; writing—original draft preparation, J.L.; writing—review and editing, S.L.; supervision, S.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2018R1D1A1B07051383), and by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2020-0-01749) supervised by the IITP (Institute of Information & Communications Technology Planning & Evaluation).

Data Availability Statement: The data presented in this study are openly available: MNIST (<http://yann.lecun.com/exdb/mnist/>) and CIFAR10 (<https://www.cs.toronto.edu/~kriz/cifar.html>).

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

References

- Goodfellow, I.J.; Shlens, J.; Szegedy, C. Explaining and Harnessing Adversarial Examples. *arXiv* **2015**, arXiv:1412.6572.
- Papernot, N.; McDaniel, P.; Goodfellow, I. *Transferability in Machine Learning: From Phenomena to Black-Box Attacks Using Adversarial Samples*; Technical Report; Pennsylvania State University: State College, PA, USA, 2016.
- Papernot, N.; McDaniel, P.; Goodfellow, I.; Jha, S.; Celik, Z.B.; Swami, A. Practical Black-Box Attacks against Machine Learning. In Proceedings of the Asia Conference on Computer and Communications Security, New York, NY, USA, 2–6 April 2017.
- Carlini, N.; Wagner, D. Towards Evaluating the Robustness of Neural Networks. In Proceedings of the 2017 IEEE Symposium on Security and Privacy (SP), San Jose, CA, USA, 22–26 May 2017.
- Nitin Bhagoji, A.; He, W.; Li, B.; Song, D. Practical Black-box Attacks on Deep Neural Networks using Efficient Query Mechanisms. In *European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2018.
- Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; Vladu, A. Towards Deep Learning Models Resistant to Adversarial Attacks. *arXiv* **2018**, arXiv:1706.06083.
- Laidlaw, C.; Feizi, S. Functional Adversarial Attacks. In *Advances in Neural Information Processing Systems*; Curran Associates Inc.: Vancouver, BC, Canada, 2019.
- Huang, Z.; Zhang, T. Black-Box Adversarial Attack with Transferable Model-based Embedding. *arXiv* **2020**, arXiv:1911.07140.
- Han, S.; Pool, J.; Tran, J.; Dally, W.J. Learning both Weights and Connections for Efficient Neural Networks. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 1135–1143.
- Han, S.; Mao, H.; Dally, W.J. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. *arXiv* **2016**, arXiv:1510.00149.
- Wen, W.; Wu, C.; Wang, Y.; Chen, Y.; Li, H. Learning Structured Sparsity in Deep Neural Networks. *Adv. Neural Inf. Process. Syst.* **2016**, *29*, 2074–2082.
- He, Y.; Zhang, X.; Sun, J. Channel Pruning for Accelerating Very Deep Neural Networks. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017.
- Li, H.; Kadav, A.; Durdanovic, I.; Samet, H.; Graf, H.P. Pruning Filters for Efficient ConvNets. *arXiv* **2017**, arXiv:1608.08710.
- He, Y.; Kang, G.; Dong, X.; Fu, Y.; Yang, Y. Soft Filter Pruning for Accelerating Deep Convolutional Neural Networks. *arXiv* **2018**, arXiv:1808.06866.
- He, Y.; Liu, P.; Wang, Z.; Hu, Z.; Yang, Y. Filter Pruning via Geometric Median for Deep Convolutional Neural Networks Acceleration. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019.
- Lin, M.; Ji, R.; Wang, Y.; Zhang, Y.; Zhang, B.; Tian, Y.; Shao, L. HRank: Filter Pruning using High-Rank Feature Map. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 14–19 June 2020.
- Chin, T.W.; Ding, R.; Zhang, C.; Marculescu, D. Towards Efficient Model Compression via Learned Global Ranking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 14–19 June 2020.
- Wang, L.; Ding, G.W.; Huang, R.; Cao, Y.; Lui, Y.C. Adversarial Robustness of Pruned Neural Networks. In *ICLR Workshop Submission*; OpenReview.net: Vancouver, BC, Canada, 2018.

19. Ye, S.; Lin, X.; Xu, K.; Liu, S.; Cheng, H.; Lambrechts, J.H.; Zhang, H.; Zhou, A.; Ma, K.; Wang, Y. Adversarial Robustness vs. Model Compression, or Both? In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea, 27–28 October 2019; pp. 111–120.
20. Gui, S.; Wang, H.; Yu, C.; Yang, H.; Wang, Z.; Liu, J. Model Compression with Adversarial Robustness: A Unified Optimization Framework. In *Advances in Neural Information Processing Systems*; Curran Associates Inc.: Vancouver, BC, Canada, 2019.
21. Hinton, G.; Vinyals, O.; Dean, J. Distilling the Knowledge in a Neural Network. *arXiv* **2014**, arXiv:1503.02531.
22. Papernot, N.; McDaniel, P.; Wu, X.; Jha, S.; Swami, A. Distillation as a Defense to Adversarial Perturbations against Deep Neural Networks. In Proceedings of the IEEE Symposium on Security and Privacy (SP), San Jose, CA, USA, 23–25 May 2016.
23. Kurakin, A.; Goodfellow, I.; Bengio, S. Adversarial Machine Learning at Scale. *arXiv* **2017**, arXiv:1611.01236.
24. Lee, N.; Ajanthan, T.; Torr, P.H.S. SNIP: Single-shot Network Pruning based on Connection Sensitivity. *arXiv* **2019**, arXiv:1810.02340.
25. Bucila, C.; Caruana, R.; Niculescu-Mizil, A. Model compression. In Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, 20–23 August 2006.
26. Heo, B.; Lee, M.; Yun, S.; Choi, J.Y. Knowledge Distillation with Adversarial Samples Supporting Decision Boundary. *Proc. Aaai Conf. Artif. Intell.* **2019**, *33*, 3771–3778. [[CrossRef](#)]
27. Mirzadeh, S.I.; Farajtabar, M.; Li, A.; Levine, N.; Matsukawa, A.; Ghasemzadeh, H. Improved Knowledge Distillation via Teacher Assistant. *Proc. Aaai Conf. Artif. Intell.* **2020**, *34*, 5191–5198. [[CrossRef](#)]
28. Xie, H.; Qian, L.; Xiang, X.; Liu, N. Blind Adversarial Pruning: Balance Accuracy, Efficiency and Robustness. *arXiv* **2020**, arXiv:2004.05914.
29. Xie, H.; Xiang, X.; Liu, N.; Dong, B. Blind Adversarial Training: Balance Accuracy and Robustness. *arXiv* **2020**; arXiv:2004.05914.
30. Madaan, D.; Shin, J.; Hwang, S.J. Adversarial Neural Pruning with Latent Vulnerability Suppression. *arXiv* **2020**, arXiv:1908.04355.
31. Bernhard, R.; Moellic, P.A.; Dutertre, J.M. Impact of Low-bitwidth Quantization on the Adversarial Robustness for Embedded Neural Networks. In Proceedings of the International Conference on Cyberworlds (CW), Kyoto, Japan, 2–4 October 2019.
32. Lin, J.; Gan, C.; Han, S. Defensive Quantization: When Efficiency Meets Robustness. *arXiv* **2019**, arXiv:1904.08444.
33. Goldblum, M.; Fowl, L.; Feizi, S.; Goldstein, T. Adversarially Robust Distillation. *Proc. Aaai Conf. Artif. Intell.* **2020**, *34*, 3996–4003.
34. Cox, D. The Regression Analysis of Binary Sequences. *J. R. Stat. Soc. Ser. (Methodological)* **1958**, *20*, 1958. [[CrossRef](#)]
35. Lecun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [[CrossRef](#)]
36. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2015**, arXiv:1409.1556.
37. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
38. Krizhevsky, A. *Learning Multiple Layers of Features from Tiny Images*; Technical Report, University of Toronto, Toronto, ON, Canada, 2009.
39. Renda, A.; Frankle, J.; Carbin, M. Comparing Rewinding and Fine-tuning in Neural Network Pruning. *arXiv* **2020**, arXiv:2003.02389.
40. Fletcher, P.T.; Venkatasubramanian, S.; Joshi, S. Robust statistics on Riemannian manifolds via the geometric median. In Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition, Anchorage, AK, USA, 23–28 June 2008; pp. 1–8.