Making Neural Networks Robust in Resource Constrained Settings

Nikhil Paliwal[7009915] Arpit Jadon[7010483] Akshay Dodwadmath[7010036]

Universität des Saarlandes, 66121 Saarbrücken, Germany
{nipa00002, arja00001, akdo00001}@stud.uni-saarland.de

Abstract

Neural networks have achieved impressive performance in many applications but their ever increasing size has led to significant computational and storage overheads. Moreover, compression of the networks usually leads to deterioration either in benign or robust accuracy. There has been recent interest in learning to compress models without hurting their robustness due to adversarial attacks, in addition to maintaining accuracy. However, the models proposed so far fail to maintain all of these criteria. We propose a novel approach for model compression that combines a state of the art pruning technique with self-supervised pre-training based on contrastive learning framework. We show that our model is able to maintain both benign and robust accuracy while achieving a high level of network compression.

1. Introduction

In the real-world deployment of neural network models, there is vulnerability against adversarial attacks. In such applications, it is often common to have limited resources (lack of computation and specialized data). This scenario is against the common notion for robustness, that is the model needs to be large, which consequently demands more data and computational resources. This makes acquiring robustness more challenging in a resource constrained environment. Overall, within safety-critical and resource-constrained scenarios, there are multiple hurdles for a neural network model to pass, such as lack of data, computational resources, and need for robustness from adversarial attacks. We aim to tackle these problems with self-supervised pre-training and network pruning while maintaining the robustness of the model against test-time adversarial attacks such as Projected Gradient Descent (PGD) [10] and black box attacks. Ours will be the first work exploring the robustness capability while combining self-supervised training and neural network pruning.

We propose to pre-train a classification model on CIFAR-10 dataset with the adversarial learning method in a self-supervised setting [8] . Then we further utilize robust [13] pruning method for compressing the model while maintaining the robustness against attacks such as PGD [10]. Further, we show that we are able to converge towards a robust and efficient model (considering data, computation, and storage). Moreover, by combining pruning with self-supervised pre-training , we are able to preserve both benign and robust accuracy. Finally, we show the success of our project by means of various experimental results. Our code is publicly available at https://github.com/akshaydodwadmath/Project_MLCysec.

2. Related Work

2.1. Robust Neural Network Pruning

Neural Network pruning as a model compression technique has been proposed extensively for facilitating deep network implementations on resource constrained application systems. Wijayanto et al. [15] primarily focuses on a specific compactness strategy called quantization. Wen et al. [14] use a column pruning framework and Zhang et al. [18] combine column pruning with an alternating direction method of multipliers (ADMM). Recently, there has been interest in developing pruning techniques which preserve robustness. LWM based pruning heuristic was used to obtain empirical adversarial robustness by Sehwag et al. [12]. Another recent work by Ye et al. [16] and Gui et al. [4] combine the LWM based pruning heuristic with an ADMM pruning framework to achieve better empirical robustness along with pruning. Our work is also aimed to maintain robustness while pruning.

2.2. Self-Supervised Learning [SSL]

Self-supervised learning [2, 6, 9] recently became popular for representation learning in deep neural networks. SSL techniques use unlabeled data to train a model by self-generating the labels from the data itself. Increased attention towards these approaches is justified by the difficulty in gathering clean labeled data. One of the popular SSL approach named SimCLR [2] got a lot of attention recently.

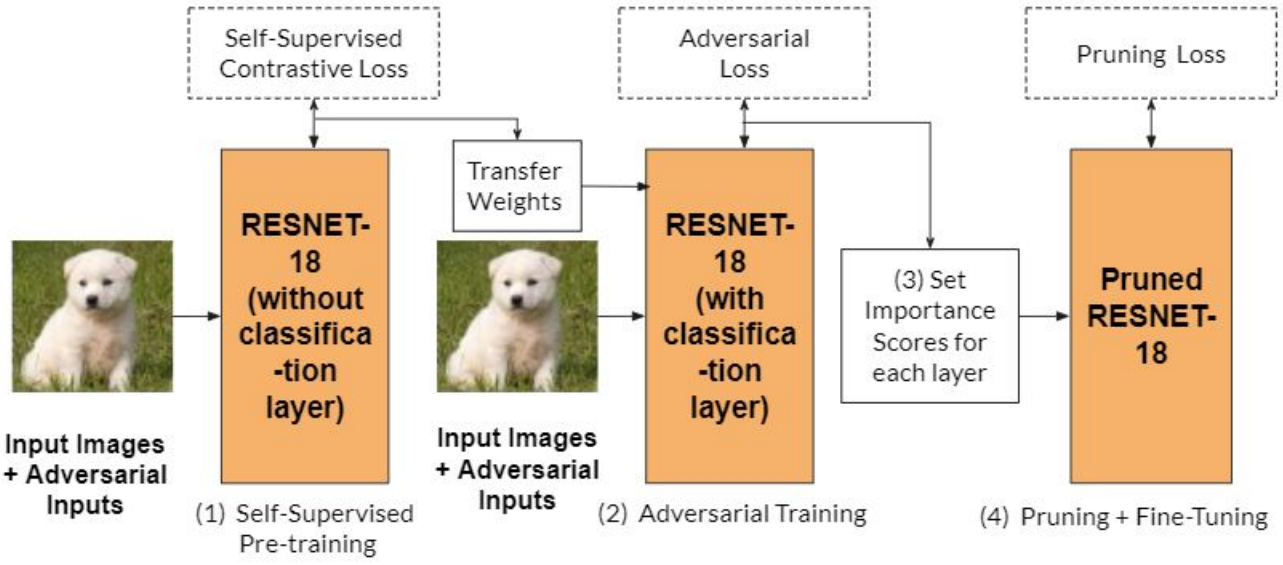

Figure 1. Overview of our method. We perform the following: (1) Pre-training using Self-Supervised Robust Contrastive loss.(2) Supervised adversarial training using Adversarial Loss. (3) Set importance scores for each of the trained layers. (4) Perform pruning using a pruning loss and finally adversarial fine-tuning of pruned weights.

Using SimCLR, authors managed to match the top-1 accuracy of a supervised ResNet-50 [7] on ImageNet. In such a constrastive learning framework, positive samples are generated by data augmentations and negative samples are taken from instances of other classes. The aim is to bring positive samples closer than negative samples in latent feature space: $d\{x, x_{pos}\} < d\{x, x_{neg}\}$. It is done by a constrastive loss function:

$$\mathcal{L}_{\mathrm{con},\theta,\pi}\left(x,\{x_{\mathrm{pos}}\},\{x_{\mathrm{neg}}\}\right) := -\log \frac{\sum_{\{\mathrm{z_{pos}}\}} \exp(\mathrm{sim}(\mathrm{z},\{\mathrm{z_{pos}}\})/\tau)}{\sum_{\{\mathrm{z_{pos}}\}} \exp(\mathrm{sim}(\mathrm{z},\{\mathrm{z_{pos}}\})/\tau)+\sum_{\{\mathrm{z_{neg}}\}} \exp(\mathrm{sim}(\mathrm{z},\{\mathrm{z_{neg}}\})/\tau)} \quad (1)$$

where x denotes data point, z is latent vector in latent feature space and $sim(.)$ denotes cosine similarity between two vectors. Few works have addressed model robustness using self-supervised pre-training [3] while others have been trying to establish correlation between self-supervised learning and pruning techniques [1]. However, there are not a lot of work that target model pruning while preserving model robustness using self-supervised learning.

2.3. Baseline Models

Our work is based on two models. The first is HYDRA, proposed by Sehwag et. al. [13] , a pruning technique that maintains the robustness of the neural network. They were able to achieve robustness with the importance score initialization (later used for pruning networks) based on pre-trained weights. The second model we use as reference was proposed by Kim et. al. [8]. They make use of Robust Contrastive Learning(RoCL) adversarial framework for neural models to make them robust to attacks without utilizing labels.

Table 1. Final accuracy (clean and robust) scores of our method against baselines on the CIFAR-10 dataset.

Method	Clean Acc(%)	Robust Acc(%)
Baseline 1-RoCL (Not pruned)	77.85	31.45
Baseline 2-HYDRA (90% Pruned)	69.84	41.74
Our method (90%Pruned)	76.66	**48.68**

3. Methods

We argue that the pre-trained weights obtained after training a neural network using the adversarial robust self-supervised framework have inherent robustness and could complement pruning techniques aware of robust training objective. Thereby, we propose our four step method combining the ideas. An overview of our method is shown in Figure 1.

3.1. Self-Supervised Adversarial Training Using Robust Contrastive Learning [RoCL]

The most commonly followed approach to accomplish adversarial robustness is adversarial learning. Adversarial

Table 2. Detailed performance scores across different stages. We observe that as natural training progresses, the clean accuracy increases while the robust accuracy decreases. This can be attributed to the fact that the model has never seen adversarial examples but only clean examples during training. With adversarial training, the robust accuracy achieves the best score in the fine-tuning stage (after a little setback in the pruning stage). The models were selected based on the best top-1 robust accuracy. Note that in the pruning stage, 90% parameters are pruned here.

Method	Supervised Training		Pruning		Fine-tuning	
	Clean Acc(%) (top-1/top-5)	Robust Acc(%) (top-1/top-5)	Clean Acc(%) (top-1/top-5)	Robust Acc(%) (top-1/top-5)	Clean Acc(%) (top-1/top-5)	Robust Acc(%) (top-1/top-5)
Baseline 2 (w/o adv training)	36.15/86.17	8.46/63.69	79.82/99.13	0.08/55.79	88.64/99.65	0.01/52.99
Baseline 2	68.17/97.29	40.55/88.35	68.75/97.19	39.43/87.82	69.84/97.35	41.74/89.06
Baseline 1 + Baseline 2	67.98/97.31	42.31/89.68	75.00/98.27	46.12/91.17	75.10/98.25	47.21/91.85
Our method	**76.78/98.47**	**49.21/92.29**	**75.91/98.40**	**47.33/91.91**	**76.66/98.57**	**48.68/92.04**

learning trains the model with perturbed samples to maximize the loss. However, most of these conventional adversarial learning approaches utilize class labels to generate adversarial samples that are supposed to trick the model in making a false prediction.

Robust contrastive learning (RoCL) makes use of self-supervised learning and thus, unlike most adversarial learning methods, it does not use class labels to generate adversarial samples. Further, RoCL specifically uses a constrastive self-supervised learning framework [2] to train a neural network for adversarial robustness.

In our approach, we train ResNet18 using RoCL[1] on CIFAR-10 data against instance-wise adversarial attack (l_∞ constraint). Since, this is a self-supervised setup, no labels are used in this attack. The perturbation is generated using an instance wise attack, which is a modified version of PGD with a contrastive loss function. Generation of perturbed adversarial examples require maximization of the self-supervised contrastive loss for discriminating between the instances. Following equation depicts how an adversarial sample is generated,

$$t(x)^{adv} = t(x) + \alpha * sign(\nabla_{t(x)} \mathcal{L}_{\text{con},\theta,\pi}) \tag{2}$$

where $t(x)$ is the transformed image using stochastic augmentation, $t(x)^{adv}$ is the generated adversarial sample, and α is the step size of the attacks.

RoCL generates instance-wise adversarial examples from stochastically augmented samples. To summarize, the overall idea is to generate perturbations on the augmented samples such that the contrastive loss is maximized and the instance-level classifier gets confused about the correct class of the perturbed samples. Following this approach, RoCL is able to obtain accuracies comparable to supervised adversarial learning methods without even using any class labels during training. The complete robust constrastive learning objective is defined below in Equation 4.

$$\mathcal{L}_{\text{RocL},\theta,\pi} := \mathcal{L}_{\text{con},\theta,\pi}\left(t, \left\{t', t^{adv}\right\}, \left\{t_{\text{neg}}\right\}\right) \tag{3}$$

$$\mathcal{L}_{\text{total}} := \mathcal{L}_{\text{RocL},\theta,\pi} + \lambda \mathcal{L}_{\text{con},\theta,\pi}\left(t^{adv}, \left\{t'\right\}, \left\{t_{\text{neg}}\right\}\right) \tag{4}$$

where t and $t^{'}$ are the stochastic augmentations while t^{adv} is the adversarial perturbation corresponding to an augmented sample. Also, here t_{neg} refers to the negative instances of t. With this step the performance of our method is boosted without need of any external data.

3.2. Supervised adversarial training

The robust pre-trained weights, trained in self-supervised fashion in the previous step are utilized to initialize our model for supervised training. This provides a better starting point to our model parameters for training. The training objective in this stage is to minimize the expected adversarial loss along with the natural loss as given by,

$$\theta_{\text{trained}} = \underset{\theta}{\text{argmin}} \underset{(x,y)\sim\mathcal{D}}{E} [\mathcal{L}(\theta, x, y)) + \beta \mathcal{L}(\theta, PGD(x), y))] \tag{5}$$

where β defines the trade-off between robust and natural (from clean sample) loss. The adversarial samples are generated using PGD attack [10], the examples of which we show in Figure 2.

3.3. Adversarial Pruning

It has been shown by work of Zhang et. al [17] that models need to be larger for adversarial robustness. Pruning neural networks can reduce the size efficiently, but it has been done in isolation from robust training. The heuristic based approaches for pruning such as lowest weight magnitude (LWM) [5] have incurred huge performance loss with adversarial training. We adopt the idea proposed by Sehwag et. al [13], HYDRA (the pruning technique), in which pruning is defined as empirical risk minimization problem. The

[1]We use the following implementation for RoCL - https://github.com/Kim-Minseon/RoCL

Table 3. Accuracy (clean and robust) scores with 99% pruned parameters on CIFAR-10. Our results are consistent even with more rigorous pruning.

Method	Supervised Training		Pruning		Fine-tuning	
	Clean Acc(%) (top-1/top-5)	Robust Acc(%) (top-1/top-5)	Clean Acc(%) (top-1/top-5)	Robust Acc(%) (top-1/top-5)	Clean Acc(%) (top-1/top-5)	Robust Acc(%) (top-1/top-5)
Baseline 2 (HYDRA)	71.43/97.62	42.82/89.62	61.95/95.49	34.07/85.11	66.62/96.57	38.02/87.08
Our method	**76.64/98.57**	**49.15/92.16**	**66.26/96.77**	**37.57/87.61**	**68.37/96.85**	**40.22/88.35**

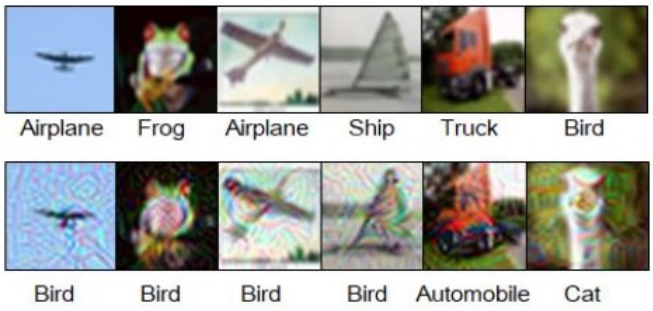

Figure 2. Example images generated using PGD attack. Original images along with the original labels are shown in the top row, and the adversarial images along with the predicted labels are shown in the bottom row.

parameter θ in Eq. 5 is replaced by $\theta_{trained}$ and the loss is minimized w.r.t mask m .

Since the mask are binary weights and can not be optimized directly, the importance scores based optimization is used [11]. The importance scores s are scaled, initialized from pre-trained weights, and included in loss as $\mathcal{L}(\theta_{trained}, s, x, y)$. During the forward pass only top-k (k being the remaining model parameters after pruning) weights (with highest score magnitude) are used to make prediction. However, during backward pass, all the scores are updated with gradients. This adversarial robust pruning will provide a compact model for utilization in a resource-constrained environment, while preserving robustness.

3.4. Adversarial Fine-tuning

A binary pruning mask m is generated from updated importance score by assigning 1 to top-k percentile scores and rest to 0. The generated pruning mask is then used in the fine-tuning step, by minimizing objective function defined in Eq. 5 with $\theta = \theta_{trained} \odot m$. The fine-tuning step is necessary to recover the performance loss due to pruning of model parameters.

Table 4. Black box attack

Method	Robust Acc(%)
Baseline 2 (w/o adv training)	10.01
Baseline 2	16.83
Our method (w/o adv training)	13.08
Baseline 1 + Baseline 2	17.99
Our method	**18.80**

4. Experiments

For our experiments, we use CIFAR-10 dataset. We consistently use ResNet18 architecture for classification task. The adversarial samples are generated with following attack settings: perturbation ϵ is set to $8/255$, the step-size α is kept at $2/255$, and the number of attack iterations are set to 10. We adopt the standard, cross-entropy loss for training for the classification task. We report the achieved accuracy with benign images (Clean accuracy) as well as adversarial examples (Robust accuracy) for every experiment performed on the test dataset of CIFAR-10.

Training Encoder Network with Robust Contrastive Learning: Our first step is to train an encoder network using RoCL on the CIFAR-10 dataset against the instance-wise adversarial attack (l_∞ constraint). The training is done by minimzing the final contrastive loss function mentioned in the Equation 4 while keeping the regularization parameter, $\lambda = 1/256$. Apart from that, rest of the optimization steps are similar to SimCLR [2]. The attack parameters used here are modified to match across experiments as defined previously. The overall training was done for 300 epochs. Note that since this is a self-supervised setup, no labels are used here. We put a simple linear (fully-connected) layer as the classifier on top of obtained ResNet18 feature extractor (encoder network) and consider it as our first baseline, Baseline 1 (RoCL). However, this is not a pruned model and cannot be considered as a solution to our resource-constraint problem. Performance of baseline 1 on the test set of CIFAR-10 is shown in Table 1.

Robust neural network pruning: We initialized a ResNet18 model with pre-train robust weights from previous step (i.e., robust constrastive training). Then we further

Table 5. We show drop in performance due to pruning without considering adversarial robustness into the objective function (as used in HYDRA). With adversarially trained weight initialization there is a significant difference in performance (without adversarial training).

Method	Supervised Training		Pruning		Fine-tuning	
	Clean Acc(%) (top-1/top-5)	Robust Acc(%) (top-1/top-5)	Clean Acc(%) (top-1/top-5)	Robust Acc(%) (top-1/top-5)	Clean Acc(%) (top-1/top-5)	Robust Acc(%) (top-1/top-5)
Baseline 2 (w/o adv training)	36.15/86.17	8.46/63.69	79.82/99.13	0.08/55.79	88.64/99.65	0.01/52.99
Our method (w/o adv training)	65.16/96.27	40.67/88.19	82.10/99.30	1.09/70.64	91.77/99.74	0.03/20.98

continue our method with supervised training, pruning, and fine-tuning. We train each stage for 25 epochs and 90% pruning ratio (only 10% parameters remaining). We also train ResNet18 from scratch - without utilizing robust pre-train weights with HYDRA[2] and consider as Baseline 2. This baseline is pertinent to our problem, since this also focuses on adversarial robust pruning. The model weights are selected based on best robust accuracy on validation set.

Table 1 compares our proposed method's performance against the two strong baselines. Our method achieves a high clean accuracy and the best robust accuracy compared to baseline 1 and 2. Moreover, among the two pruned networks (i..e, ours and baseline 2-HYDRA), our method outperforms HYDRA in both, clean and robust accuracy. Table 2 goes into a detailed performance analysis of the model across three different stages (since top-1 accuracy became very low, we also show top-5 accuracy for ease of comparison). In Table 2, we also include different variants of baselines such as HYDRA method (i.e., baseline 2) without adversarial training and replacing supervised training by Baseline 1 model, denoted as Baseline 1 + Baseline 2. Some variants are very close to our method, regardless we observe that our model outperform all other baseline methods in all three stages. This success is attributed to the robust weight initialization using the weights obtained from step 1 (Sec. 3.1), providing better start for model optimization and faster convergence. This difference between our method and other baselines can be observed in first column of Table 2. The adversarial aware training objective in step 3 (Sec. 3.3) maintain the robustness of our method. In the first row of table 2, Baseline 2 (without adversarial training), we observe a drastic decrease in robustness, because it is not preserved in the pruning stage unlike our method.

Robustness in harsh pruning: In further experiments, we increase the pruning ratio to 99% (1% parameters remain). In Table 3, we show that our method still outperforms baselines consistently in all three stages. Moreover, the performance is comparable to baseline 2 performance at 90% pruning, establishing strong superiority.

Black box attack: To test our method in realistic scenario, where adversary does not have access to the model, we create adversarial samples with an auxiliary model. We train a ResNet20 model on CIFAR-10 and use PGD attack to generate adversarial examples on test set. Table 4 shows the robust accuracy of different baseline variants and our method. We still observe better performance by our method in this new type of unseen attack.

Robustness drop without adversarial training: In this experiment we want to show the importance of adversarial training in different stages of our proposed approach. We train baseline and our method without including the adversarial samples in objective functions (of supervised, pruning and fine-tuning stages). As shown in Table 5 the robustness plummets irrespective of robust weight initialization across all stages. This confirms the importance of adversarial training for robustness.

5. Conclusion

In this work, we proposed a solution to achieve a robust and compact model that can be deployed in a resource-constrained environment. In all 4 steps of our approach, we include natural (on clean samples) and robust (on adversarial samples) training loss for parameter optimization. To boost the robustness, we utilize robust contrastive learning and transfer the weights to initialize our model before supervised learning. We learn the pruning mask by incorporating importance scores in the robust objective function, using which we are able to preserve the robustness. We compare our method with strong baselines and their variants for thorough study. Because of our elegant 4 step method, solving different resource-constraint problems at each stage, we achieve best performance throughout all our experiments. We test our method with different pruning ratios, i.e., 0% after supervised training stage, 90% and, 99% and our method outperforms others at every stage. Moreover, with different types of attacks, i.e., white-box (PGD) and black-box, our observations remain consistent. With extensive experiments and solid reasoning of our method, we have established strong dominance over baseline and variants, providing a reliable method for deployment.

References

[1] Tianlong Chen, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Michael Carbin, and Zhangyang Wang. The

[2] https://github.com/inspire-group/hydra

lottery tickets hypothesis for supervised and self-supervised pre-training in computer vision models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16306–16316, 2021. 2

[2] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020. 1, 3, 4

[3] Tianlong Chen, Sijia Liu, Shiyu Chang, Yu Cheng, Lisa Amini, and Zhangyang Wang. Adversarial robustness: From self-supervised pre-training to fine-tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 699–708, 2020. 2

[4] S. Gui, H. N.Wang, H. Yang, C. Yu, Z.Wang, and J. Liu. Model compression with adversarial robustness: A unified optimization framework. *Advances in Neural Information Processing Systems, pages 1283–1294*, 2019. 1

[5] Song Han, Jeff Pool, John Tran, and William J. Dally. Learning both weights and connections for efficient neural network. *ArXiv*, abs/1506.02626, 2015. 3

[6] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020. 1

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015. 2

[8] Minseon Kim, Jihoon Tack, and Sung Ju Hwang. Adversarial self-supervised contrastive learning. *arXiv preprint arXiv:2006.07589*, 2020. 1, 2

[9] Nikos Komodakis and Spyros Gidaris. Unsupervised representation learning by predicting image rotations. In *International Conference on Learning Representations (ICLR)*, 2018. 1

[10] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017. 1, 3

[11] Vivek Ramanujan, Mitchell Wortsman, Aniruddha Kembhavi, Ali Farhadi, and Mohammad Rastegari. What's hidden in a randomly weighted neural network? *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11890–11899, 2020. 4

[12] V. Sehwag, S. Wang, P. Mittal, and S. Jana. Towards compact and robust deep neural networks. *arXiv preprint arXiv:1906.06110*, 2019. 1

[13] Vikash Sehwag, Shiqi Wang, Prateek Mittal, and Suman Jana. Hydra: Pruning adversarially robust neural networks. *arXiv preprint arXiv:2002.10509*, 2020. 1, 2, 3

[14] Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. *Advances in neural information processing systems, , pages 2074–2082*, 2016. 1

[15] A. W. Wijayanto, J. J. Choong, K. Madhawa, and T. Murata. Towards robust compressed convolutional neural networks. *2019 IEEE International Conference on Big Data and Smart Computing (BigComp), pages 1–8. IEEE,*, 2019. 1

[16] S. Ye, K. Xu, S. Liu, H. Cheng, J.-H. Lambrechts, H. Zhang, A. Zhou, K. Ma, Y. Wang, and X. Lin. Adversarial robustness vs. model compression, or both. *The IEEE International Conference on Computer Vision (ICCV), volume 2, 2019*, 2019. 1

[17] 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 *ECCV*, 2018. 3

[18] 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. 1