CSTAR: Towards Compact and Structured Deep Neural Networks with Adversarial Robustness

Huy Phan[1], **Miao Yin**[1], **Yang Sui**[1], **Bo Yuan**[1], **Saman Zonouz**[2]
[1]Department of Electrical and Computer Engineering, Rutgers University
[2]Schools of Cybersecurity and Privacy, Georgia Institute of Technology
{huy.phan, miao.yin, yang.sui}@rutgers.edu,
bo.yuan@soe.rutgers.edu, saman.zonouz@gatech.edu

Abstract

Model compression and model defense for deep neural networks (DNNs) have been extensively and individually studied. Considering the co-importance of model compactness and robustness in practical applications, several prior works have explored to improve the adversarial robustness of the sparse neural networks. However, the structured sparse models obtained by the existing works suffer severe performance degradation for both benign and robust accuracy, thereby causing a challenging dilemma between robustness and structuredness of compact DNNs.

To address this problem, in this paper, we propose CSTAR, an efficient solution that simultaneously impose **C**ompactness, high **ST**ructuredness and high **A**dversarial **R**obustness on the target DNN models. By formulating the structuredness and robustness requirement within the same framework, the compressed DNNs can simultaneously achieve high compression performance and strong adversarial robustness. Evaluations for various DNN models on different datasets demonstrate the effectiveness of CSTAR. Compared with the state-of-the-art robust structured pruning, CSTAR shows consistently better performance. For instance, when compressing ResNet-18 on CIFAR-10, CSTAR achieves up to 20.07% and 11.91% improvement for benign accuracy and robust accuracy, respectively. For compressing ResNet-18 with $16\times$ compression ratio on Imagenet, CSTAR obtains 8.58% benign accuracy gain and 4.27% robust accuracy gain compared to the existing robust structured pruning.

Introduction

Deep neural networks (DNNs) have obtained widespread popularity in various important intelligent systems, such as autonomous driving, smart home, computer-aided diagnosis, and so on. Despite their current unprecedented prosperity, DNNs are still facing two severe problems, namely *efficiency challenge* and *security challenge*, which can potentially hinder their future success in many practical applications. Specifically, the efficiency challenge refers to the high energy/storage/timing costs when deploying the modern large-scale DNN models in the field, especially on resource-constrained embedded and mobile devices. Meanwhile, the security challenge mainly refers to the vulnerability of DNN models to the attacks launched by the malicious adversary (Zang et al. 2020; Xie et al. 2020b; Phan et al. 2020, 2022b; Shi et al. 2022; Phan et al. 2022a), especially to the imperceptible and high-damage *adversarial attacks*.

To address these emerging challenges and further promote and democratize the use of DNNs, the machine learning community has conducted tremendous research activities to improve the execution efficiency and security of neural networks. In particular, *model compression* and *adversarial defense*, as two key techniques that can efficiently reduce the model size and enhance the model robustness, have been extensively studied in the existing literature. More specifically, various types of DNN compression approaches, e.g., pruning (Han et al. 2015; Sui et al. 2021) and quantization (Gong et al. 2014), as well as many different DNN defense methods, e.g., adversarial training (Madry et al. 2017; Wong, Rice, and Kolter 2020) and input denoising (Prakash et al. 2018), have been developed in both academia and industry.

Co-exploring Model Compactness & Robustness. Considering the co-importance of compactness and robustness for the DNNs used in the practical applications, an attractive and natural thought is to develop a method that can generate neural networks with high compactness *and* strong robustness simultaneously. Motivated by this idea, several prior works have explored to examine and further improve the robustness of the compressed DNNs. In particular, because of the popularity of *pruning* in many model compression tasks, most of the existing compactness/robustness co-exploration efforts (Sehwag et al. 2020, 2019; Ye et al. 2019; Vemparala et al. 2021) focus on efficient approaches to generate the robust pruned DNN models. To be specific, (Ye et al. 2019) demonstrates that the model robustness and compactness can co-exist for neural networks via concurrent adversarial training and weight pruning. Considering the pruning criterion in (Sehwag et al. 2019) is designed to preserve benign accuracy instead of improving robust accuracy, (Sehwag et al. 2020) further proposes robust training-aware pruning to enhance both.

The Dilemma Between Robustness and Structuredness of Compact DNNs. Although the state-of-the-art robust pruning works can simultaneously enable high sparsity and strong robustness for the target DNN models, all of their best performance comes from performing *unstructured pruning*. As extensively reported and observed in the prior literature (Han et al. 2016; Liu et al. 2015; Zhu et al.

2020; Deng et al. 2021), the unstructured sparse models, though can indeed exhibit high accuracy and high compression ratio, cannot bring considerable speedup on the off-the-shelf CPU/GPU. To mitigate this challenge, based on the observation that the structured pruning, e.g., channel/-filter pruning, can enable the measurable acceleration, prior efforts have also explored the robustness-aware structured pruning to strike for simultaneous robustness, compactness and structuredness of DNN models. Unfortunately, as reported in (Sehwag et al. 2019, 2020; Ye et al. 2019), ***the existing robust structured sparse models suffer severe performance degradation as compared to their unstructured counterparts with respect to both benign and robust accuracy.*** Consequently, the challenging dilemma between robustness and structuredness of the compact DNNs has not been well addressed yet.

Low-rank Robustness: A Structured Method Towards Compact & Robust DNNs. From the perspective of model compression, structured pruning is not the only way to produce structured compact models. *Low-rank tensor decomposition* (Denton et al. 2014; Kim et al. 2015; Yang, Krompass, and Tresp 2017; Pan et al. 2019; Yin et al. 2021a, 2022), as another type of popular model compression technique, can also ensure that the compressed model exhibits the desired high structuredness for the practical speedup. To be specific, different from structured pruning that removes the entire channels/filters to enable the structured sparsity, low-rank tensor decomposition explores the structure-level model redundancy and factorizes the original dense model into multiple tensor cores. Because those decomposed tensor cores are still in the dense format and their involved computations are entirely based on the general matrix multiplication (GEMM), considerable acceleration can be observed and measured on the off-the-shelf CPU/GPU (Kim et al. 2015; Zhang et al. 2015b,a). Evidently, such hardware-friendly low-rankness-based compactness, if can further coexist with the adversarial robustness, will be attractive for compact and robust DNN model design.

Questions to be Answered. Despite the promising potentials of robust tensor decomposed models, several important questions need to be answered and addressed. To be specific, what is the suitable perspective to connect and unify the low-rankness and robustness of a DNN model? Once we can formulate a unified problem, what is the efficient approach to simultaneously impose low-rankness and robustness with a high compression ratio, benign accuracy, and robust accuracy? Also, considering the conventional tensor decomposition requires a tedious manual rank selection process, which will be very time-consuming if integrated with defense methods such as adversarial training, is there any efficient automatic tensor rank determination scheme?

Technical Preview and Contributions. To answer these questions and deliver the promise of robust low-rank DNN models, in this paper we study and develop an efficient solution, namely CSTAR, that can simultaneously impose the desired high low-rankness-based **C**ompactness, high **ST**ructuredness and high **A**dversarial **R**obustness on the target DNN models. By formulating the low-rankness and robustness requirement within the same framework as well as determining the tensor ranks in a global way, the compressed DNNs can achieve high compression performance and strong adversarial robustness at the same time. Overall, the contributions of this paper are summarized as follows:

- We propose a framework that formulates the low-rankness and adversarial robustness requirement to a unified constrained optimization problem. Based on that, we then propose an adversarial training process that can gradually impose the desired low-rankness on the DNN model, thereby simultaneously ensuring high compression performance and adversarial robustness.
- We propose a low-cost automatic rank selection scheme that can select the desired tensor ranks for each layer in a very convenient way. Such selection can be dynamically adjusted during the compression procedure, thereby avoiding the tedious manual rank selection and meanwhile ensuring high performance.
- We evaluate the performance of the CSTAR solution for various DNN models on different datasets. Compared with the state-of-the-art robust structured pruning methods, our proposed approach shows consistently better performance. For instance, when compressing ResNet-18 on the CIFAR-10 dataset, CSTAR can achieve up to 20.07% and 11.91% increase for benign accuracy and robust accuracy, respectively. On the Imagenet dataset, for compressing ResNet-18 with $16\times$ compression ratio, CSTAR can obtain 8.58% benign accuracy gain and 4.27% robust accuracy gain as compared to the existing robust structured pruning method.

Background and Preliminaries

Tensor Decomposition for Model Compression

Tensor decomposition, as a type of low-rank decomposition approach, has been widely used in many DNN model compression works. According to tensor theory, there exist various types of tensor decomposition approaches, such as Tucker (Tucker 1963), CP (Harshman 1970), Tensor Train (Oseledets 2011) and Tensor Ring (Zhao et al. 2016). Without loss of generality, in this paper, we adopt Tucker-2 decomposition (Kim et al. 2015) as the underlying low-rank tensor method for model compression.

Tensor Contraction. Tucker-2 decomposition can be compactly represented via using tensor contraction. In general, tensor contraction can be performed between any two tensors that have at lease one matched dimension. For instance, consider two tensors $\boldsymbol{\mathcal{A}} \in \mathbb{R}^{d_1 \times d_2 \times n}$ and $\boldsymbol{\mathcal{B}} \in \mathbb{R}^{n \times d_3 \times d_4}$, the output tensor $\boldsymbol{\mathcal{C}}$ of size $\mathbb{R}^{d_1 \times d_2 \times d_3 \times d_4}$ after the tensor contraction can be calculated as follow:

$$\boldsymbol{\mathcal{C}}_{(a_1,a_2,b_1,b_2)} = \boldsymbol{\mathcal{A}} \times_1^3 \boldsymbol{\mathcal{B}} = \sum_{i=1}^{n} \boldsymbol{\mathcal{A}}_{(a_1,a_2,i)} \boldsymbol{\mathcal{B}}_{(i,b_1,b_2)}. \quad (1)$$

Tucker-2 Tensor Decomposition. With the notation of tensor contraction, given the weight tensor of convolutional layer $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{O \times I \times K \times K}$ as well as the corresponding input tensor $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{I \times H \times W}$ and output tensor $\boldsymbol{\mathcal{Y}} \in \mathbb{R}^{O \times H' \times W'}$, $\boldsymbol{\mathcal{W}}$ can be factorized via Tucker-2 decomposition as:

$$\boldsymbol{\mathcal{W}} = \boldsymbol{\mathcal{U}}^1 \times_2^2 \boldsymbol{\mathcal{U}}^2 \times_2^2 \boldsymbol{\mathcal{G}}, \tag{2}$$

where $\boldsymbol{\mathcal{U}}^1 \in \mathbb{R}^{O \times R_1}$, $\boldsymbol{\mathcal{U}}^2 \in \mathbb{R}^{I \times R_2}$, and $\boldsymbol{\mathcal{G}} \in \mathbb{R}^{R_1 \times R_2 \times K \times K}$. R_1 and R_2 denote the Tucker-2 tensor ranks. Then, the output tensor $\boldsymbol{\mathcal{Y}} = \boldsymbol{\mathcal{U}}^1 \times_1^2 \boldsymbol{\mathcal{T}}^2$ can be calculated via tensor contraction-based computations:

$$\boldsymbol{\mathcal{T}}^1 = \boldsymbol{\mathcal{U}}^2 \times_1^1 \boldsymbol{\mathcal{X}}, \quad \boldsymbol{\mathcal{T}}^2 = \sum_{p=1}^{K} \sum_{q=1}^{K} \boldsymbol{\mathcal{G}}_{(:,:,p,q)} \times_1^2 \boldsymbol{\mathcal{T}}^1 \tag{3}$$

where $\boldsymbol{\mathcal{T}}^1 \in \mathbb{R}^{R_2 \times H \times W}$, and $\boldsymbol{\mathcal{T}}^2 \in \mathbb{R}^{R_1 \times H' \times W'}$ are the intermediate results. Notice that tensor contraction has been well supported by PyTorch and TensorFlow platforms via using `torch.tensordot` and `tf.tensordot`.

Adversarial Training for DNN Defense

Projected Gradient Decent (PGD). PGD is a very popular adversarial examples generation approach (Madry et al. 2017) because of its high-quality and fast generation process. In general, PGD first calculates the gradient w.r.t. the benign input data x and then updates the input data in the direction that maximizes the loss function $\mathcal{L}$. The resulting adversarial example x_{adv} is then projected to the maximum allowed perturbation Δ via using projection operator $\mathbf{\Pi}$. Such process is repeated k times to obtain a strong example:

$$\boldsymbol{\mathcal{X}}_{adv}^{N+1} = \mathbf{\Pi}_{\Delta}(\boldsymbol{\mathcal{X}}_{adv}^{N} + \epsilon \cdot \text{sign}(\nabla \mathcal{L}(\boldsymbol{\mathcal{W}}, \boldsymbol{\mathcal{X}}_{adv}^{N}, \boldsymbol{y})), \tag{4}$$

where $\boldsymbol{\mathcal{X}}_{adv}^{N+1}$ is the set of adversarial examples at the N-th iteration, ϵ is the step size, and $\boldsymbol{y}$ is set of the labels.

Adversarial Training. Adversarial training (AT) (Madry et al. 2017) is one of the most effective methods to protect DNNs against adversarial attacks. The key idea of adversarial training is to generate adversarial examples during the training process to make the model can better fit for adversarial examples. In general, adversarial training aims to solve the following min-max optimization problem:

$$\min_{\boldsymbol{\mathcal{W}}} \max_{\delta \in \boldsymbol{\Delta}} \mathcal{L}(\boldsymbol{\mathcal{W}}, \boldsymbol{\mathcal{X}} + \delta, \boldsymbol{y}), \tag{5}$$

where $\boldsymbol{\mathcal{X}}$ denotes the set of benign inputs, and $\mathcal{L}$ is the loss function. Here the inner maximization problem can be solved by the above described PGD.

Related Works

Model Compression with Adversarial Training. To date several prior works have explored the efficient integration of model compression to the adversarial training process. To be specific, (Sehwag et al. 2019; Ye et al. 2019; Sehwag et al. 2020; Vemparala et al. 2021; Xie et al. 2020a; Guo et al. 2018; Madaan, Shin, and Hwang 2020; Rakin et al. 2019; Özdenizci and Legenstein 2021) investigate the robustness-aware pruning to achieve high model sparsity and adversarial robustness. Also, (Fu et al. 2021; Lin, Gan, and Han 2019) propose several approaches to develop low-bit precision robust models. In addition, (Gui et al. 2019) studies the adversarial robust model with the combination of sparsity, low bit-precision and matrix factorization. However, most of these works are built either on unstructured sparsity or complicated hybrid-precision schemes (e.g, 4-8 bits), which can only be supported by the specialized hardware, thereby severely limiting their practical deployment on the off-the-shelf CPUs/GPUs. To date, only (Sehwag et al. 2019; Ye et al. 2019; Sehwag et al. 2020) provide the structured pruning solutions for adversarial robust DNN models that can exhibit practical speedup. However, the corresponding benign and robust accuracy is not satisfied.

Low-rank Matrix & Tensor Decomposition. Similar to structured pruning, low-rank decomposition is another powerful structuredness-ensuring compression approach that can bring considerable speedup. In general, low-rank decomposition can be realized via either matrix decomposition or tensor decomposition. **Matrix decomposition** (Tai et al. 2015; Li and Shi 2018; Xu et al. 2019; Yang et al. 2020; Idelbayev and Carreira-Perpinán 2020; Xu et al. 2019) uses SVD to factorize the large matrix to small matrix components. In particular, when the objective is the 4-D weight tensor of convolutional layer, it has to first reshapes the 4-D tensor to 2-D matrix, and then performs decomposition to obtain small 2-D matrices. ***Evidently, such flattening strategy can not fully exploit the inherent spatial information of the 4-D weight tensor, thereby causing limited performance.*** Motivated by these limitations, **Tensor decomposition**, such as Tucker (Kim et al. 2015), Tensor Train (Oseledets 2011) and Tensor Ring (Zhao et al. 2016), ***which can directly factorize the original high-order tensor in the high-dimensional space***, have been studied to facilitate DNN compression. Because tensor decomposition can fully exploit and leverage the rich spatial correlation and information of weight tensor in the high-order tensor space, it can bring impressive compression performance (Tai et al. 2015; Kim et al. 2015; Li and Shi 2018; Wang et al. 2018)

Rank Selection. Determining the proper rank is very critical for low-rank decomposed DNN models. To date, most of the existing low-rank DNN compression approaches (Kim et al. 2015; Zhao et al. 2016; Yang, Krompass, and Tresp 2017; Xu et al. 2019; Idelbayev and Carreira-Perpinán 2020; Yin et al. 2021b) select the rank in a manual way. However, because each layer needs to be assigned with one or more rank values, such heuristic searching is a very tedious and time-consuming procedure. Recently, some works have begun to study efficient automatic rank determination. In (Li et al. 2021; Hajimolahoseini et al. 2021), progressive search-based methods are proposed to identify proper low matrix rank (Hajimolahoseini et al. 2021) and low tensor rank (Li et al. 2021). However, this strategy is performance-limited because it still partially needs manual selection (Hajimolahoseini et al. 2021) or costly iterative searching and sampling (Li et al. 2021). In addition, (Gusak et al. 2019; Ye et al. 2018) propose to automatically select the rank in a layer-wise way. However, such strategy is not the ideal solution because of the inevitable high cost: (Gusak et al. 2019) needs expensive Bayesian calculation plus setting additional auxiliary hyper-parameters during the searching, and (Ye et al. 2018) needs many rounds of enumeration to search for the proper rank values. Moreover, determining the rank

layer by layer, in principle, is not a promising solution since it may not bring the globally optimal rank determination for the entire model.

Advantages of Our Proposed CSTAR. Compared with the above discussed related works, CSTAR enjoys the following benefits:

- **Practical Measurable Speedup.** Compared with most existing works on joint compression and adversarial training via unstructured pruning (Ye et al. 2019; Sehwag et al. 2020), low-bit precision (Fu et al. 2021; Lin, Gan, and Han 2019) or hybrid combination (Gui et al. 2019), CSTAR can bring practical speedup on the off-the-shelf CPUs/GPUs because of its strong model structuredness provided by low-rankness (see Section 14).
- **Low-Cost Automatic Rank Selection.** The rank selection of CSTAR is naturally integrated to the compression process and automatically learned from data, and hence it has very low cost without any expensive enumeration or searching operation that prior works (Li et al. 2021; Hajimolahoseini et al. 2021; Gusak et al. 2019) need. Meanwhile, the ranks for all the layers are jointly determined in a global way (see Appendix).
- **High Benign and Robust Accuracy.** Compared with the state-of-the-art structured pruning-based adversarial training methods (Ye et al. 2019; Sehwag et al. 2020) that can also bring practical speedup, CSTAR shows very significant performance improvement in terms of benign accuracy and robust accuracy across different models on various datasets (see Section 14).

Our Proposed Method

Problem Formulation. CSTAR aims to co-explore adversarial robustness and low-rankness on DNN models to achieve high compression ratio, high benign accuracy, and high robust accuracy with practical speedup simultaneously. To achieve this goal, a suitable perspective that can properly connect and unify the adversarial robustness and low-rankness should be identified. Considering adversarial training is currently the most important and popular adversarial defense approach, we formulate our goal as the following optimization problem:

$$\begin{aligned} &\min_{\boldsymbol{\mathcal{W}}_i} \max_{\delta\in\boldsymbol{\Delta}} \mathcal{L}(\boldsymbol{\mathcal{W}}, \boldsymbol{\mathcal{X}}+\delta, \boldsymbol{y}), \\ &\text{s.t.} \quad \texttt{rank}(\boldsymbol{\mathcal{W}}_i) \le \mathbf{t}_i, \quad i=1,\dots,N, \end{aligned} \tag{6}$$

where N denotes the number of layers, $\texttt{rank}(\cdot)$ is the function that calculates the Tucker-2 ranks $\mathbf{r}=[r_1, r_2]$ of the input tensor, and $\mathbf{t}=[t_1, t_2]$ denotes the target ranks that can satisfy the compression ratio. Here $\mathbf{r} \le \mathbf{t}$ operation means element-wise comparison.

Challenges to be Addressed. In general, solving problem 6 is non-trivial yet challenging. More specifically, two critical issues need to be resolved:

Challenge #1: *How should we optimize problem 6 that is constrained by the non-differentiable* $\texttt{rank}(\boldsymbol{\mathcal{W}}_i) \le \boldsymbol{t}_i$*?*

Analysis. A straightforward way is to directly decompose the pre-trained $\{\boldsymbol{\mathcal{W}}_i\}$ into the suitable low-rank format that satisfies the rank constraint, and then fine-tune the entire decomposed model. Though this direct strategy can indeed bring the desired low-rankness, such straightforward and explicit decomposition on the high-accuracy model will probably cause considerable approximation error. Consider that the common pre-trained CNN models typically lack sufficient low-rank property, the corresponding approximation error and its resulting benign and robust accuracy loss, consequently, will be too huge to be recovered after fine-tuning.

Algorithm 1: Tucker-2 proj. for solving Eqn. 12

```
def decompose(W, r1, r2):
#Input: Weight tensor W, ranks r1, r2
#Ouput: Decomposed U1, U2, G tensors
  W=W.flatten(start_dim=2)
  t1=W.flatten(start_dim=1)
  U1,sigma1,V1=torch.svd(t1)
  t2=W.permute(1,0,2).flatten(start_dim=1)
  U2,sigma2,V2=torch.svd(t2)
  G=torch.tensordot(U2,W,dims=([0],[1]))
  G=torch.tensordot(U1,G,dims=([0],[1]))
  U1,U2,G=U1[:,:r1],U2[:,:r2],G[:r1,:r2]
  return U1, U2, G
def recover(U1, U2, G):
#Input: Decomposed U1, U2, G tensors
#Output: Recovered low-rank W tensor
  W=torch.tensordot(U2,G,dims=([1],[1]))
  W=torch.tensordot(U1,W,dims=([1],[1]))
  O,I,KK=W.size()
  K=int(KK ** 0.5)
  return W.view(O, I, K, K)
```

Our Proposed Solution. To address this challenge, ***we propose to gradually impose the desired low-rankness on the DNN model during the adversarial training procedure.*** To be specific, the to-be-compressed model is first trained in the full-rank format but gradually exhibits low-rank characteristic as well as adversarial robustness. Our rationale for this indirect training strategy lies in the belief that the smooth and "soft" transition from full-rank to low-rank format, intuitively, can better preserve benign and robust accuracy than the direct and "hard" decomposition. Motivated by this philosophy, we re-formulate problem 6 via adding the low-rankness requirement as a **soft regularization term**:

$$\begin{aligned} &\min_{\boldsymbol{\mathcal{W}}_i, \boldsymbol{\mathcal{Z}}_i\in\mathcal{S}} \max_{\delta\in\boldsymbol{\Delta}} \mathcal{L}(\boldsymbol{\mathcal{W}}_i, \boldsymbol{\mathcal{X}}+\delta, \boldsymbol{y}), \\ &\text{s.t.} \quad \boldsymbol{\mathcal{W}}_i = \boldsymbol{\mathcal{Z}}_i, \quad i=1,\dots,N, \end{aligned} \tag{7}$$

where $\boldsymbol{\mathcal{Z}}_i$ is the introduced auxiliary variables and $\mathcal{S} = \{\boldsymbol{\mathcal{W}}_i \mid \texttt{rank}(\boldsymbol{\mathcal{W}}_i) \le \mathbf{t}_i\}$. Then, we further reformulate the problem via using the augmented Lagrangian:

$$\begin{aligned} &\min_{\boldsymbol{\mathcal{W}}_i, \boldsymbol{\mathcal{Z}}_i\in\mathcal{S}} \max_{\boldsymbol{\mathcal{M}}_i, \delta\in\boldsymbol{\Delta}} \mathcal{L}(\boldsymbol{\mathcal{W}}_i, \boldsymbol{\mathcal{X}}+\delta, \boldsymbol{y}) \\ &+\frac{\rho}{2}\sum_{i=1}^{N}\left(\|\boldsymbol{\mathcal{W}}_i - \boldsymbol{\mathcal{Z}}_i + \boldsymbol{\mathcal{M}}_i\|_F^2 + \|\boldsymbol{\mathcal{M}}_i\|_F^2\right), \end{aligned} \tag{8}$$

where $\boldsymbol{\mathcal{M}}_i$ are dual multipliers and $\rho > 0$ is the augmented parameter. After that, problem 8 can be solved via Dykstra's alternating projection method (Bauschke and Borwein

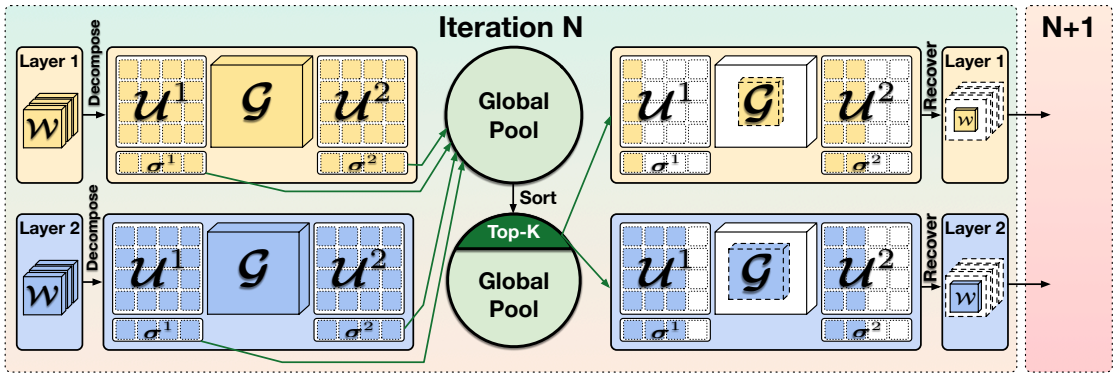

Figure 1: Proposed singular value-based automatic tensor rank selection. Here the rank selection can vary as training progresses.

1994), a primal-dual optimization algorithm that is well-suited for multi-term minimization. To be specific, the iterative updates are performed as:

Update $\{\mathcal{W}_i\}$ using SGD. In this step, we optimize the sub-problem as:

$$\min_{\{\mathcal{W}_i\}} \underbrace{\max_{\delta\in\mathbf{\Delta}} \mathcal{L}(\{\mathcal{W}_i\}, \mathcal{X}+\delta, \mathcal{Y})}_{\text{Adversarial Training}} + \underbrace{\frac{\rho}{2}\sum_{i=1}^{N} \|\mathcal{W}_i - \mathcal{Z}_i + \mathcal{M}_i\|_F^2}_{\text{Low-rank Regularization}}. \tag{9}$$

The first term in Eq. 9 is to find adversarial examples, and it can be realized via using PGD described in Sec. . The second term is to gradually impose low-rank property onto $\{\mathcal{W}_i\}$. Since both terms are differentiable, we can iteratively solve Eq. 9 using stochastic gradient descent (SGD) with learning rate α:

$$\begin{aligned}\mathcal{W}_i \leftarrow \mathcal{W}_i &- \alpha\cdot\nabla_{\mathcal{W}_i}\left[\mathcal{L}(\mathcal{W}_i, \mathcal{X}+\delta, \mathcal{Y})\right] \\ &- \alpha\cdot\rho\left(\|\mathcal{W}_i - \mathcal{Z}_i + \mathcal{M}_i\|\right).\end{aligned} \tag{10}$$

Update $\{\mathcal{Z}_i\}$ using Projection. In this step, $\{\mathcal{Z}_i\}$ can be updated as:

$$\mathcal{Z}_i \leftarrow \underset{\mathcal{Z}_i\in\mathcal{S}}{\operatorname{argmin}} \ \|\mathcal{W}_i - \mathcal{Z}_i + \mathcal{M}_i\|_F^2. \tag{11}$$

More specifically, problem 11 can be solved via projecting $\mathcal{Z}_i$ onto set $\mathcal{S}$ as:

$$\mathcal{Z}_i \leftarrow \mathbf{Prj}(\mathcal{W}_i + \mathcal{M}_i), \tag{12}$$

where the projection truncates the ranks of $\mathcal{W}_i + \mathcal{M}_i$ to the desired target ranks $\mathbf{t}_i$. Algorithm 1 describes the details of this low-rank projection.

Challenge #2: *How should we properly select the tensor ranks in a convenient way?*

Analysis. Because 1) the rank values directly determine the model capacity, computational cost, and memory footprint; and 2) different layers exhibit different low-rankness characteristics, a proper rank selection scheme is very critical and important to achieve a high-performance low-rank DNN model. Unfortunately, as analyzed in Section , to date the rank selection for the tensor decomposed models are still performed in a costly and time-consuming way. In particular, consider each low-rank layer in modern DNNs may have multiple to-be-set tensor ranks, e.g., Tucker-2 format needs to configure two rank values per layer, the existing rank selection scheme will become even more expensive.

Algorithm 2: The overall procedure of CSTAR

Input: Dataset $\mathcal{D}$, pre-trained weight $\mathcal{W}$, comp. ratio r, low-rank reg. steps T_1, fine-tune steps T_2.
Output: Fine-tuned $\{\mathcal{U}_i^1, \mathcal{U}_i^2, \mathcal{G}_i\}$.
$\{\mathcal{Z}_i\} \leftarrow \{\mathcal{W}_i\}$; $\{\mathcal{M}_i\} \leftarrow \mathbf{0}$;
for $t = 1$ ***to*** T_1 **do** ▷ *low-rank regularization*
  update $\{\mathcal{W}_i\}$ via Eq. 10;
  update $\{\mathcal{Z}_i\}$ via Eq. 12 and Alg. 1;
  $\mathcal{M}_i \leftarrow \mathcal{M}_i + \mathcal{W}_i - \mathcal{Z}_i$;
$\{\mathcal{U}_i^1, \mathcal{U}_i^2, \mathcal{G}_i\} \leftarrow$ `decompose`$(\{\mathcal{W}_i\})$ via Alg. 1;
for $t = 1$ ***to*** T_2 **do** ▷ *fine-tune*
  $\mathcal{X}, \mathcal{Y} \leftarrow$ `sample_batch`$(\mathcal{D})$;
  $\mathcal{X}_{adv} \leftarrow$ `PGD`$(\mathcal{X}, \mathcal{Y})$ via Eq. 4;
  $\hat{\mathcal{Y}} \leftarrow$ `conv`$(\{\mathcal{U}_i^1, \mathcal{U}_i^2, \mathcal{G}_i\}, \mathcal{X}_{adv})$ via Eq. 3;
  loss $\leftarrow$ `cross_entropy`$(\hat{\mathcal{Y}}, \mathcal{Y})$;
  `update`$(\{\mathcal{U}_i^1, \mathcal{U}_i^2, \mathcal{G}_i\}$, loss);

Our Proposed Solution. To overcome this challenge, we propose a low-cost automatic rank selection scheme for low-rank adversarial robust DNN models. Our key idea is to leverage the singular value information to globally select the suitable tensor rank for each individual layer. To be specific, considering mathematically the magnitude of each singular value strongly represents the importance of the corresponding rank component, e.g., with the sorted singular values σ_i^1 for $\mathcal{U}^1$, $\mathcal{U}_k^1 = \sigma_1^1 u_1^1 v_1^{1T} + ... + \sigma_k^1 u_k^1 v_k^{1T}$ is the best rank-k approximation for $\mathcal{U}^1$. Hence we can sort all the singular values from the entire model and select the largest ones and their corresponding rank components under the target compression budget. As illustrated in Fig. 1, this global comparison and selection strategy can determine the suitable tensor rank values for different layers in an automatic way, thereby significantly facilitating the overall compression workflow.

Comp. Ratio	**LWM** Filter Pruning	**LWM** Column Pruning	**RAP** Filter Pruning	**RAP** Column Pruning	**CSTAR** Low-Rank Decomp.
			VGG-16 (Pre-train 80.08 / 44.10)		
2×	73.69 / 44.05	77.31 / 44.96	76.85 / 46.32	79.25 / 45.30	**79.71 / 47.23**
4×	63.76 / 38.93	70.46 / 41.54	71.48 / 42.59	75.55 / 43.97	**79.64 / 47.06**
8×	54.30 / 34.45	63.69 / 38.35	62.48 / 37.58	69.54 / 41.52	**79.23 / 46.42**
16×	45.77 / 30.37	55.29 / 35.01	53.72 / 33.72	60.55 / 37.36	**77.82 / 46.13**
32×	40.17 / 27.35	47.94 / 31.71	43.78 / 29.34	52.62 / 34.03	**75.82 / 45.43**
64×	10.00 / 10.00	40.01 / 29.35	10.00 / 10.00	45.08 / 32.49	**70.77 / 43.13**
			ResNet-18 (Pre-train 83.23 / 49.78)		
2×	83.62 / 45.47	83.57 / 44.91	82.68 / 48.85	82.51 / 49.45	**83.86 / 50.06**
4×	79.10 / 46.49	80.54 / 46.79	79.78 / 46.78	81.81 / 46.76	**83.22 / 49.48**
8×	70.16 / 42.41	75.23 / 44.46	71.82 / 43.17	76.89 / 45.57	**82.30 / 48.81**
16×	61.31 / 37.12	68.55 / 40.86	64.39 / 38.01	70.18 / 41.78	**80.90 / 47.36**
32×	51.59 / 32.43	59.20 / 36.60	54.31 / 33.32	62.68 / 37.15	**77.73 / 46.47**
64×	42.36 / 28.52	50.92 / 32.16	46.97 / 29.41	52.01 / 32.48	**72.08 / 44.39**

Table 1: Benign accuracy / PGD-50 robust accuracy (%) on CIFAR-10 with Standard AT.

A.T.	**Friendly AT (Zhang et al. 2020)**		**TRADES (Zhang et al. 2019)**		**MART (Wang et al. 2019)**	
C.R.	**RAP**	**CSTAR**	**RAP**	**CSTAR**	**RAP**	**CSTAR**
	Pre-train 85.86 / 38.24		**Pre-train 78.45 / 47.43**		**Pre-train 73.97 / 45.61**	
2×	84.94 / 39.34	**85.94 / 40.14**	78.47 / **46.92**	**80.00** / 44.81	75.40 / 44.49	**76.02 / 47.98**
4×	83.24 / 37.10	**85.86 / 40.29**	75.13 / 44.27	**80.20 / 44.91**	71.23 / 44.60	**77.08 / 44.87**
8×	79.06 / 33.89	**85.49 / 39.68**	69.82 / 39.58	**79.48 / 44.94**	65.43 / 43.04	**76.36 / 44.99**
16×	74.06 / 29.01	**85.02 / 38.70**	65.72 / 36.00	**78.99 / 44.38**	57.74 / 38.57	**74.57 / 46.25**
32×	63.41 / 22.62	**83.57 / 37.72**	56.47 / 30.75	**77.49 / 42.59**	50.00 / 35.58	**71.94 / 46.72**
64×	54.30 / 20.59	**81.32 / 35.44**	49.76 / 29.67	**73.11 / 42.44**	42.40 / 31.76	**66.33 / 44.95**

Table 2: Performance of VGG-16 on CIFAR-10 using different adversarial training approaches. Here C.R. means compression ratio, and A.T. means adversarial training.

Noticed that here the desired singular values have already been available after low-rank projection (see Line 6 and 8 in Alg. 1), so it is naturally integrated to the entire compression procedure with very low computing overhead. Also, as showed in Fig. 1, such a sorting-and-selection process can be performed in each epoch to dynamically adjust the rank selection as training progresses.

Fine-tuning. Upon the iterative update finishes, the model has been gradually imposed on the desired low-tensor-rank property, and then we can decompose it with the automatically selected ranks and fine-tune it in the low-rank format. Algorithm 2 summarizes the CSTAR procedure.

Experiments

To demonstrate the effectiveness of CSTAR, we evaluate various model architectures with multiple adversarial training objectives on different datasets. To be specific, we evaluate the benign and robust accuracy of VGG-16/19, ResNet-18/34, DenseNet-121, GoogleNet and MobileNet-V2. In addition, several adversarial training approaches, including the Standard PGD (Madry et al. 2017), Friendly AT (Zhang et al. 2020), TRADES (Zhang et al. 2019), MART (Wang et al. 2019), and Fast AT (Wong, Rice, and Kolter 2020) are adopted to show the generality of our approach. The models are compressed on CIFAR-10/100 and ImageNet datasets with different compression ratios ranging from $2\times$ to $64\times$. On CIFAR-10/100 L_∞ is selected with $\Delta = 8/255$, and PGD with step size $\alpha = 2/255$ serves to generate adversarial examples. Here the number of PGD iterations for training and testing are 10 and 50, respectively. Other hyper-parameter settings can be found in the Appendix.

Comparison with Robust Aware Pruning (RAP) on CIFAR-10/100. Table 1 shows our comparison with the RAP approach (Ye et al. 2019). For fair comparison, we perform compression on the same pre-train models with the same number of training epochs. In addition, we also report the performance of Least Weight Magnitude (LWM)-based approach (Sehwag et al. 2019) in this table. It is seen that our proposed approach consistently outperforms RAP with different compression ratios on both benign accuracy and robust accuracy. In particular, with $64\times$ compression, our CSTAR can achieve up to 20.07% and 11.91% improvement for benign accuracy and robust accuracy, respectively. For the results of CIFAR-100 dataset, see the Appendix.

Comparison with HYDRA on CIFAR-10. HYDRA (Sehwag et al. 2020) is AN importance-score based adversarial pruning approach. Notice that because 1) HYDRA does not release code for their **structured pruning** version; and 2) HYDRA only reports very limited data point (VGG-16 on CIFAR-10) for its performance with structured pruning, in Table 5 we only list the benign and robust accuracy of our approach with the C.Rs. ($2\times$ and $10\times$) that HYDRA reports. From this table it is seen that our method shows very significant performance improvement over HYDRA. With $10\times$ C.R., our CSTAR approach can achieve 60.99% of benign accuracy and 28.48% robust accuracy increase.

Stronger Baseline: RAP or HYDRA? Comparing Table

Model Arch.	Pre-Train	Methods / Compression Ratios					
		RAP 8x	CSTAR 8x	RAP 16x	CSTAR 16x	RAP 32x	CSTAR 32x
DenseNet-121	80.91 / 45.36	71.74 43.66	**78.87 / 48.07**	57.42 / 36.23	**74.91 / 45.38**	36.79 / 26.33	**63.12 / 37.71**
GoogleNet	81.85 / 48.18	73.28 / 42.74	**79.59 / 47.34**	62.03 / 36.67	**76.16 / 44.13**	49.27 / 30.49	**62.06 / 35.29**
VGG-19	74.64 / 43.33	65.94 / 40.79	**73.60 / 45.95**	57.35 / 35.95	**73.88 / 45.87**	47.45 / 31.73	**72.69 / 45.59**
ResNet-34	82.64 / 47.85	79.05 / 47.34	**83.52 / 50.35**	73.93 / 44.62	**81.95 / 48.69**	66.45 / 40.13	**79.63 / 48.24**
MobileNetV2	84.62 / 45.15	62.98 / 38.56	**83.03 / 41.54**	50.70 / 33.19	**66.30 / 41.72**	35.96 / 26.42	**63.88 / 40.18**

Table 3: Performance of benign / PGD-50 robust accuracy (%) on CIFAR-10 using a wide range of model architectures with Standard AT. Here C.R. means compression ratio.

C.R.	Top-1 Accuracy			Top-5 Accuracy		
	RAP	CSTAR	Δ	RAP	CSTAR	Δ
	Pre-train 50.75 / 26.57			Pre-train 75.25 / 51.56		
2×	43.40 / 22.22	**51.36 / 25.75**	+7.96 / +3.53	68.69 / 45.16	**75.28 / 51.42**	+6.59 / +6.26
4×	35.29 / 17.58	**41.11 / 21.05**	+5.82 / +3.47	60.32 / 37.71	**66.60 / 43.28**	+6.28 / +5.57
6×	29.40 / 14.50	**34.92 / 17.35**	+5.52 / +2.85	53.06 / 31.95	**59.92 / 37.41**	+6.86 / +5.46
8×	23.24 / 11.72	**30.29 / 14.84**	+7.05 / +3.12	45.31 / 26.88	**54.60 / 33.10**	+9.29 / +6.22
10×	19.09 / 9.72	**28.91 / 14.10**	+9.82 / +4.38	39.41 / 23.14	**52.82 / 31.86**	+13.41 / +8.72
16×	9.34 / 5.05	**17.92 / 9.32**	+8.58 / +4.27	22.93 / 13.20	**37.83 / 22.22**	+14.90 / +9.02

Table 4: Performance of benign / robust accuracy (%) for ResNet-18 on ImageNet dataset using Fast AT (Wong, Rice, and Kolter 2020). Here C.R. means compression ratio.

C.R.	HYDRA	CSTAR	Δ
2×	52.90 / 38.00	**80.00 / 44.81**	+27.10 / +6.81
10×	18.30 / 16.70	**79.29 / 45.18**	+60.99 / +28.48

Table 5: HYDRA vs CSTAR on CIFAR-10 using TRADES. Here C.R. means compression ratio.

C.R.	VGG-16 / CPU		ResNet-18 / GPU	
	RAP	CSTAR	RAP	CSTAR
P.T.	9.98ms	9.98ms	0.0801ms	0.0801ms
2×	9.98ms	6.75ms/↓1.48×	0.0801ms	0.0562ms/↓1.43×
4×	9.98ms	4.38ms/↓2.28×	0.0801ms	0.0437ms/↓1.83×

Table 6: Inference time per image of executing CSTAR models for CIFAR-10 data compared with unstructured pruning.

1 and Table 5, it is seen that with the same (2×) or even higher (16×) compression ratio, RAP shows much higher benign and robust accuracy than HYDRA on the same pre-trained VGG-16 model. Such phenomenon shows that the least weight magnitude-based structured pruning seems more suitable for obtaining adversarial robustness than the importance score-based structured pruning. In addition, considering HYDRA only reports its **structured pruning** performance for VGG-16 on CIFAR-10; therefore, in our following experiments involved with other models and dataset, we select RAP (Column pruning) as the baseline.

Generalization Across Different Adversarial Training Approaches. To demonstrate the generality of CSTAR, we evaluate its performance for compressing VGG-16 model via using other three popular adversarial training methods (FAT, TRADES and MART). As shown in Table 2, our solution can consistently outperform RAP method with different compression ratios on both benign and robust accuracy.

Generalization Across Different Model Architectures. We also evaluate the generality of CSTAR on different CNN architectures. As shown in Table 3, our CSTAR solution achieves both higher benign and robust accuracy than RAP across multiple models sizes and architectures.

Performance on ImageNet Dataset. Table 4 shows the performance of our approach for compressing ResNet-18 on ImageNet dataset. It is seen that in this challenging task CSTAR achieves consistently better performance than RAP. For instance, CSTAR can obtain 8.58% benign accuracy gain and 4.27% robust accuracy gain with $16\times$ C.R..

Runtime Speedup on CPU/GPU. We also measure the speedup of executing CSTAR model on the off-the-shelf hardware. Table 6 reports the inference time with batch size = 1 for the original and compressed models on both CPU (AMD Ryzen 9 5900HX) and GPU (NVIDIA GeForce RTX 3090). It is seen that the low-rank structuredness of CSTAR indeed brings practical speedup.

Conclusion

This paper proposes CSTAR, an efficient approach that can simultaneously impose high low-rankness-based compactness, high structuredness, and high adversarial robustness on the DNN models. By using a unified framework to gradually impose the low-tensor-rankness and automatically select the rank setting during the adversarial training procedure, CSTAR can make the compressed DNNs achieve high compression performance and strong adversarial robustness at the same time.

Acknowledgments

This work was partially supported by the National Science Foundation under Grant CCF-1955909, RTML, CMMI and CPS programs.

References

Bauschke, H. H.; and Borwein, J. M. 1994. Dykstra' s alternating projection algorithm for two sets. *Journal of Approximation Theory*, 79(3): 418–443.

Deng, C.; Sui, Y.; Liao, S.; Qian, X.; and Yuan, B. 2021. GoSPA: An Energy-efficient High-performance Globally Optimized SParse Convolutional Neural Network Accelerator. In *2021 ACM/IEEE 48th Annual International Symposium on Computer Architecture (ISCA)*, 1110–1123. IEEE.

Denton, E. L.; Zaremba, W.; Bruna, J.; LeCun, Y.; and Fergus, R. 2014. Exploiting linear structure within convolutional networks for efficient evaluation. *Advances in neural information processing systems*, 27.

Fu, Y.; Yu, Q.; Li, M.; Chandra, V.; and Lin, Y. 2021. Double-win quant: Aggressively winning robustness of quantized deep neural networks via random precision training and inference. In *International Conference on Machine Learning*, 3492–3504. PMLR.

Gong, Y.; Liu, L.; Yang, M.; and Bourdev, L. 2014. Compressing deep convolutional networks using vector quantization. *arXiv preprint arXiv:1412.6115*.

Gui, S.; Wang, H.; Yu, C.; Yang, H.; Wang, Z.; and Liu, J. 2019. Model compression with adversarial robustness: A unified optimization framework. *arXiv preprint arXiv:1902.03538*.

Guo, Y.; Zhang, C.; Zhang, C.; and Chen, Y. 2018. Sparse dnns with improved adversarial robustness. *Advances in neural information processing systems*, 31.

Gusak, J.; Kholiavchenko, M.; Ponomarev, E.; Markeeva, L.; Blagoveschensky, P.; Cichocki, A.; and Oseledets, I. 2019. Automated multi-stage compression of neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 0–0.

Hajimolahoseini, H.; Rezagholizadeh, M.; Partovinia, V.; Tahaei, M.; Awad, O. M.; and Liu, Y. 2021. Compressing Pre-trained Language Models using Progressive Low Rank Decomposition. *Advances in Neural Information Processing Systems*.

Han, S.; Liu, X.; Mao, H.; Pu, J.; Pedram, A.; Horowitz, M. A.; and Dally, W. J. 2016. EIE: Efficient inference engine on compressed deep neural network. *ACM SIGARCH Computer Architecture News*, 44(3): 243–254.

Han, S.; Pool, J.; Tran, J.; and Dally, W. J. 2015. Learning both weights and connections for efficient neural networks. *arXiv preprint arXiv:1506.02626*.

Harshman, R. 1970. Foundations of the PARAFAC procedure: Models and conditions for an" explanatory" multimodal factor analysis. *UCLA Working Papers in Phonetics*, 16(1): 84.

Idelbayev, Y.; and Carreira-Perpinán, M. A. 2020. Low-rank compression of neural nets: Learning the rank of each layer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8049–8059.

Kim, Y.-D.; Park, E.; Yoo, S.; Choi, T.; Yang, L.; and Shin, D. 2015. Compression of deep convolutional neural networks for fast and low power mobile applications. *arXiv preprint arXiv:1511.06530*.

Li, C.; and Shi, C. 2018. Constrained optimization based low-rank approximation of deep neural networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 732–747.

Li, N.; Pan, Y.; Chen, Y.; Ding, Z.; Zhao, D.; and Xu, Z. 2021. Heuristic rank selection with progressively searching tensor ring network. *Complex & Intelligent Systems*, 1–15.

Lin, J.; Gan, C.; and Han, S. 2019. Defensive quantization: When efficiency meets robustness. *arXiv preprint arXiv:1904.08444*.

Liu, B.; Wang, M.; Foroosh, H.; Tappen, M.; and Pensky, M. 2015. Sparse convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 806–814.

Madaan, D.; Shin, J.; and Hwang, S. J. 2020. Adversarial neural pruning with latent vulnerability suppression. In *International Conference on Machine Learning*, 6575–6585. PMLR.

Madry, A.; Makelov, A.; Schmidt, L.; Tsipras, D.; and Vladu, A. 2017. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*.

Oseledets, I. V. 2011. Tensor-train decomposition. *SIAM Journal on Scientific Computing*, 33(5): 2295–2317.

Özdenizci, O.; and Legenstein, R. 2021. Training adversarially robust sparse networks via Bayesian connectivity sampling. In *International Conference on Machine Learning*, 8314–8324. PMLR.

Pan, Y.; Xu, J.; Wang, M.; Ye, J.; Wang, F.; Bai, K.; and Xu, Z. 2019. Compressing recurrent neural networks with tensor ring for action recognition. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 4683–4690.

Phan, H.; Shi, C.; Xie, Y.; Zhang, T.; Li, Z.; Zhao, T.; Liu, J.; Wang, Y.; Chen, Y.; and Yuan, B. 2022a. RIBAC: Towards Robust and Imperceptible Backdoor Attack against Compact DNN. In *European Conference on Computer Vision*, 708–724. Springer.

Phan, H.; Xie, Y.; Liao, S.; Chen, J.; and Yuan, B. 2020. CAG: A Real-time Low-cost Enhanced-robustness High-transferability Content-aware Adversarial Attack Generator. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 5412–5419.

Phan, H.; Xie, Y.; Liu, J.; Chen, Y.; and Yuan, B. 2022b. Invisible and Efficient Backdoor Attacks for Compressed Deep Neural Networks. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 96–100. IEEE.

Prakash, A.; Moran, N.; Garber, S.; DiLillo, A.; and Storer, J. 2018. Deflecting adversarial attacks with pixel deflection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 8571–8580.

Rakin, A. S.; He, Z.; Yang, L.; Wang, Y.; Wang, L.; and Fan, D. 2019. Robust sparse regularization: Simultaneously optimizing neural network robustness and compactness. *arXiv preprint arXiv:1905.13074*.

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

Sehwag, V.; Wang, S.; Mittal, P.; and Jana, S. 2020. Hydra: Pruning adversarially robust neural networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 7.

Shi, C.; Zhang, T.; Li, Z.; Phan, H.; Zhao, T.; Wang, Y.; Liu, J.; Yuan, B.; and Chen, Y. 2022. Audio-domain position-independent backdoor attack via unnoticeable triggers. In *Proceedings of the 28th Annual International Conference on Mobile Computing And Networking*, 583–595.

Sui, Y.; Yin, M.; Xie, Y.; Phan, H.; Aliari Zonouz, S.; and Yuan, B. 2021. CHIP: CHannel Independence-based Pruning for Compact Neural Networks. *Advances in Neural Information Processing Systems*, 34.

Tai, C.; Xiao, T.; Zhang, Y.; Wang, X.; et al. 2015. Convolutional neural networks with low-rank regularization. *arXiv preprint arXiv:1511.06067*.

Tucker, L. R. 1963. Implications of factor analysis of three-way matrices for measurement of change. *Problems in measuring change*, 15: 122–137.

Vemparala, M.-R.; Fasfous, N.; Frickenstein, A.; Sarkar, S.; Zhao, Q.; Kuhn, S.; Frickenstein, L.; Singh, A.; Unger, C.; Nagaraja, N.-S.; et al. 2021. Adversarial Robust Model Compression Using In-Train Pruning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 66–75.

Wang, W.; Sun, Y.; Eriksson, B.; Wang, W.; and Aggarwal, V. 2018. Wide compression: Tensor ring nets. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 9329–9338.

Wang, Y.; Zou, D.; Yi, J.; Bailey, J.; Ma, X.; and Gu, Q. 2019. Improving adversarial robustness requires revisiting misclassified examples. In *International Conference on Learning Representations*.

Wong, E.; Rice, L.; and Kolter, J. Z. 2020. Fast is better than free: Revisiting adversarial training. *arXiv preprint arXiv:2001.03994*.

Xie, H.; Xiang, X.; Liu, N.; and Dong, B. 2020a. Blind Adversarial Training: Balance Accuracy and Robustness. *arXiv preprint arXiv:2004.05914*.

Xie, Y.; Shi, C.; Li, Z.; Liu, J.; Chen, Y.; and Yuan, B. 2020b. Real-time, universal, and robust adversarial attacks against speaker recognition systems. In *ICASSP 2020-2020 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, 1738–1742. IEEE.

Xu, Y.; Li, Y.; Zhang, S.; Wen, W.; Wang, B.; Dai, W.; Qi, Y.; Chen, Y.; Lin, W.; and Xiong, H. 2019. Trained rank pruning for efficient deep neural networks. In *2019 Fifth Workshop on Energy Efficient Machine Learning and Cognitive Computing-NeurIPS Edition (EMC2-NIPS)*, 14–17. IEEE.

Yang, H.; Tang, M.; Wen, W.; Yan, F.; Hu, D.; Li, A.; Li, H.; and Chen, Y. 2020. Learning low-rank deep neural networks via singular vector orthogonality regularization and singular value sparsification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, 678–679.

Yang, Y.; Krompass, D.; and Tresp, V. 2017. Tensor-train recurrent neural networks for video classification. In *International Conference on Machine Learning*, 3891–3900. PMLR.

Ye, J.; Wang, L.; Li, G.; Chen, D.; Zhe, S.; Chu, X.; and Xu, Z. 2018. Learning compact recurrent neural networks with block-term tensor decomposition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 9378–9387.

Ye, S.; Xu, K.; Liu, S.; Cheng, H.; Lambrechts, J.-H.; Zhang, H.; Zhou, A.; Ma, K.; Wang, Y.; and Lin, X. 2019. Adversarial Robustness vs. Model Compression, or Both? In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 111–120.

Yin, M.; Liao, S.; Liu, X.-Y.; Wang, X.; and Yuan, B. 2021a. Towards Extremely Compact RNNs for Video Recognition with Fully Decomposed Hierarchical Tucker Structure. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12085–12094.

Yin, M.; Sui, Y.; Liao, S.; and Yuan, B. 2021b. Towards efficient tensor decomposition-based dnn model compression with optimization framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10674–10683.

Yin, M.; Sui, Y.; Yang, W.; Zang, X.; Gong, Y.; and Yuan, B. 2022. HODEC: Towards Efficient High-Order DEcomposed Convolutional Neural Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12299–12308.

Zang, X.; Xie, Y.; Chen, J.; and Yuan, B. 2020. Graph universal adversarial attacks: A few bad actors ruin graph learning models. *arXiv preprint arXiv:2002.04784*.

Zhang, H.; Yu, Y.; Jiao, J.; Xing, E.; El Ghaoui, L.; and Jordan, M. 2019. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning*, 7472–7482. PMLR.

Zhang, J.; Xu, X.; Han, B.; Niu, G.; Cui, L.; Sugiyama, M.; and Kankanhalli, M. 2020. Attacks which do not kill training make adversarial learning stronger. In *International Conference on Machine Learning*, 11278–11287. PMLR.

Zhang, X.; Zou, J.; He, K.; and Sun, J. 2015a. Accelerating very deep convolutional networks for classification and detection. *IEEE transactions on pattern analysis and machine intelligence*, 38(10): 1943–1955.

Zhang, X.; Zou, J.; Ming, X.; He, K.; and Sun, J. 2015b. Efficient and accurate approximations of nonlinear convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and pattern Recognition*, 1984–1992.

Zhao, Q.; Zhou, G.; Xie, S.; Zhang, L.; and Cichocki, A. 2016. Tensor ring decomposition. *arXiv preprint arXiv:1606.05535*.

Zhu, C.; Huang, K.; Yang, S.; Zhu, Z.; Zhang, H.; and Shen, H. 2020. An efficient hardware accelerator for structured sparse convolutional neural networks on FPGAs. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 28(9): 1953–1965.