Defensive Tensorization

Adrian Bulat *,[1]
adrian@adrianbulat.com
Jean Kossaifi *,[2]
jean.kossaifi@gmail.com
Sourav Bhattacharya[1]
sourav.b1@samsung.com
Yannis Panagakis[3]
yannisp@di.uoa.gr
Timothy Hospedales[1,4]
t.hospedales@ed.ac.uk
Georgios Tzimiropoulos[1,5]
g.tzimiropoulos@qmul.ac.uk
Nicholas D Lane[1,6]
nic.lane@samsung.com
Maja Pantic[7]
m.pantic@imperial.ac.uk

[1] Samsung AI Centre
Cambridge, UK
[2] NVIDIA
Santa Clara, USA
[3] University of Athens
Athens, Greece
[4] University of Edinburgh
Edinburgh, UK
[5] Queen Mary University London
London, UK
[6] University of Cambridge
Cambridge, UK
[7] Imperial College London
London, UK

Abstract

We propose defensive tensorization, an adversarial defence technique that leverages a latent high-order factorization of the network. The layers of a network are first expressed as factorized tensor layers. Tensor dropout is then applied in the latent subspace, therefore resulting in dense reconstructed weights, without the sparsity or perturbations typically induced by the randomization. Our approach can be readily integrated with any arbitrary neural architecture and combined with techniques like adversarial training. We empirically demonstrate the effectiveness of our approach on standard image classification benchmarks. We validate the versatility of our approach across domains and low-precision architectures by considering an audio classification task and binary networks. In all cases, we demonstrate improved performance compared to prior works.

1 Introduction

The popularity of DNNs in production-ready systems has raised a serious security concern as DNNs were found to be susceptible to a wide range of adversarial attacks [1, 13, 17, 22, 30, 34], where small and imperceptible perturbations of the input data lead to incorrect predictions by the networks. This shortcoming is an obstacle in wide-scale adoption of DNN, especially when such models become part of security and safety-related solutions [2]. Consequently,

* - denotes equal contribution

© 2021. The copyright of this document resides with its authors.
It may be distributed unchanged freely in print or electronic forms.

a large volume of work attempts to design robust networks [12, 20, 32, 41, 44]. However, advances in designing robust DNNs have been followed with stronger perturbation schemes that defeat such defences [4].

Most defenses that rely on randomization apply randomized transformations either to the input, e.g. [51], or within the network, e.g., on the activations [12] or on the weights directly [47]. However, all these approaches typically introduce artifacts (e.g. sparsity in the weights or activations) and can be defeated by carefully crafted attacks [4]. In this paper, we take a different approach to randomization. We first parametrize the network using tensor factorization, effectively introducing a latent subspace spanning the weights. We then apply randomization in that latent subspace, enabling us to create models that are robust to adversarial attacks, without modifying directly the weights, activations or inputs. In summary, we make the following contributions:

- We propose an adversarial defence technique that relies on a latent randomized tensor parametrization of each layer and can be seamlessly integrated within any architecture.
- We thoroughly evaluate our method's robustness against various adversarial attacks and show that it consistently and significantly improves over the current state of the art especially when combined with adversarial training.
- We empirically demonstrate that our method successfully hardens the models against these attacks for both quantized and real-valued nets.
- We validate our strategy across domains on both image and audio-based classification.

2 Related work

2.1 Adversarial attacks

First, we review a few of the most popular adversarial attacks alongside the current defense strategies employed. Given a data sample, e.g., an image $\mathcal{X}$, an adversary will try to find a *small* perturbation, often *imperceptible* to a human eye, but that, added to the input sample, will cause it to be misclassified by the target model, with high confidence. Mathematically, the attacker generates a perturbation Δ bounded in terms of some ℓ_p norm, i.e. $\|\Delta\|_p \leq \varepsilon$, typically with $p = 2$ or $p = \infty$. The adversarial sample is obtained by adding the perturbation to the input sample $\mathcal{X}_{adv} = \mathcal{X} + \Delta$.

Several ways of obtaining this adversarial perturbations have been proposed [4, 7, 17]. Among them, *black-box* attacks, consider the network as a black-box in which the attacker has no information regarding its architecture or the gradients. *White-box* attacks on the contrary, assume complete access to the network architecture and all its parameters. Moreover, attacks can either be *untargeted*, in which case the goal is simply to make the network predict *any* wrong label, or *targeted*, in which case the aim is to force the network to predict a specific label, independently from the input sample. Next, we introduce the main white-box attacks used in this paper.

Fast Gradient Sign Method (FGSM) is a single-step, gradient based technique, introduced by Goodfellow et al. [17] to generate ℓ_∞-bounded adversarial perturbations as follows:

$$\mathcal{X}_{adv} = \boldsymbol{X} + \varepsilon \cdot \text{sgn}(\nabla_{\mathcal{X}} \mathcal{L}(\theta, \mathcal{X}, y)) \tag{1}$$

with θ the parameters of the target network. While the single gradient-step nature of FGSM makes it better for transferability attacks, it can also lead to a suboptimal ascent direction.

Basic Iterative Method (BIM) and Projected Gradient Descend (PGD) aim to address the shortcoming of FGSM by running it for several iterations. Kurakin et al. [30] propose BIM, in which the FGSM is run for several iterations, clipping the values of the perturbation at each step to be inside the bounds. Madry et al. [34] further improve upon this by prepending BIM with a random start and replacing clipping with a projection onto the acceptable set:

$$\mathcal{X}_{adv}^{t+1} = \Pi_{\mathcal{X}+S}(\mathcal{X}_{adv}^{t} + \alpha \cdot \text{sgn}\left(\nabla_{\mathcal{X}} \mathcal{L}(\theta, X_{adv}^{t}, y)\right), \tag{2}$$

where α is the step size and $\Pi_{\mathcal{X}+S}$ is a projection operation forcing the generated adversarial samples to be in the ℓ_p ball S around $\boldsymbol{X}$. A model resilient to PGD attacks is considered to be reasonably resistant to all first order attacks [34].

2.2 Adversarial defences

Despite recent advances, developing robust neural networks remains an open, challenging problem [4]. Current defense strategies typically attempt to either detect the adversarial samples and denoise them, or inject adversarial samples during training. The latter is known as adversarial training [17, 30, 34] and is considered the most resilient defense technique. As adversarial training degrades the accuracy on the clean data, recent methods attempt to balance-out the trade-off between standard and robust accuracy via boosted loss functions [49], early stopping [40] or by analyzing a plethora of architectural and training aspects [19]. Beyond this, other defense strategies were proposed. For example, feature squeezing Xu et al. [53] hardens the model by reducing the complexity of the data representation, causing the adversarial perturbations to disappear due to lower sensitivity. Guo et al. [20] proposes a set of five transformations, that applied to the image to increase the robustness to adversarial attacks of a given model. However, even the combination of all these transformations was shown to be vulnerable to carefully tuned attacks [4]. Samangouei et al. [41] introduces the so-called *Defense-GAN* technique. The main idea is to project the samples into the manifold of a generator before classifying them. Similarly, Song et al. [44] uses a PixelCNN instead of a generative model. [52] introduces a series of denoising blocks that perform feature denoising using the non-local means or other filters. [35] proposes to increase the outer-class distance by forcing the features for each class to lie inside a convex polytope that is maximally separated from the ones of other classes. [54] attempt to scramble the structure of the adversarial noise by randomly dropping pixels out of the input image and then reconstructing them using matrix estimation methods. Despite the variety of recently proposed defence strategies, in [4] the authors show that most of the existing defense techniques rely on one form of gradient obfuscation proposing both a method to detect such class of defences and to defeat them. In parallel, a series of methods propose certified defenses [8, 18, 56]. However, the provided guarantees to not match the empirical result offered by other techniques.

As opposed to all the aforementioned works which either manipulate the data samples [20, 41, 44] or introduce stochasticity on the activations [12] or weights [47] of each layer, we propose a novel defense strategy that leverages tensor factorization of the weights in order to apply randomization in that latent space, before reconstructing the weights. The approach is introduced in details in section 3.

2.3 Tensor methods in deep learning and tensor dropout

Tensors are high dimensional generalizations of matrices [26]. Recently, tensor decompositions have found a surge of applications in deep learning, mainly focusing on network

compression and acceleration. By parametrizing layers of neural networks using tensor decomposition, or even whole networks [27], the number of parameters can be reduced with little to no loss of performance, and in some cases the operations can be done more efficiently [3, 24, 31, 36].

In some cases, tensor decompositions can exhibit high computational cost and possibly low convergence rates when applied to massive data. To accelerate computation, and enable them to scale, several randomized tensor decompositions have been developed. In this way, CP decomposition can be done by selecting randomly elements from the original tensor [5], or using randomization to solve the problem on one or several smaller tensors before projecting back the result to the original space [15, 42, 46]. Wang et al. [48] proposed a fast yet provable randomized CP decomposition using FFT to perform tensor contraction. Randomization approaches have also be explored for fast approximation of other tensor decompositions, e.g., Tucker decomposition via sketching [45, 57] and tensor ring using tensor random projections [55].

These methods are orthogonal to our approach and can be combined with it. As opposed to the aforementioned works, which focus on compression or efficiency, we explore tensor factorization methods in the context of adversarial defense, proposing to combine a factorized reparametrization of convolutional layers combined with tensor dropout that significantly hardens the model, increasing its robustness to a wide range of adversarial attacks. Our method is generic and can be applied to both real-valued networks and binary ones. In addition, it is orthogonal to the existing defence methods and can be combined with existing defenses such as adversarial training. The method is introduced in details in section 3.

3 Defensive tensorization

Our defense leverages a randomized higher-order factorization method, which is used as the basis for our defense. Typically, defensive methods relying on randomization do so by introducing sparsity in either the weights or the input activation tensors of the layers of the deep neural neural network. For instance, Dhillon et al. [12] sparsify the input tensor, by stochastically pruning some of the activations and scaling up the remaining ones. Wang et al. [47] apply sparsification to the weights directly using dropout both during training and testing.

However, all these approaches degrade performance by setting some of the activations to zero, and, while rescaling the non-zero entries can mitigate the issue, increasing the sparsity (and therefore the efficiency of the defense) translates into large losses in performance. By contrast, we propose to rely on a latent parametrization of the layers using tensor decomposition. Intuitively, a latent subspace spanning the weights is learnt, along with a non-linear projection to and from that subspace. The sparsity inducing randomization is applied in the latent space. Upon projection, the resulting weights are dense and yet preserve the robustness against adversarial attacks. This allows us to built models that are both more robust to adversarial methods than existing works, while preserving high classification accuracy.

Notation: We denote vectors (1^{st}order tensors) as small bold letters $\boldsymbol{v}$, matrices (2^{nd}order tensors) as bold capital letters $\mathbf{M}$ and tensors, which generalize the concept of matrices for orders (number of dimensions) higher than 2, in capital calligraphic letters $\mathcal{X}$. The *n–mode* product is defined, for a given tensor $\mathcal{X} \in \mathbb{R}^{D_0 \times D_1 \times \cdots \times D_N}$ and a matrix $\mathbf{M} \in \mathbb{R}^{R \times D_n}$, as the tensor $\mathcal{T} = \mathcal{X} \times_n \mathbf{M} \in \mathbb{R}^{D_0 \times \cdots \times D_{n-1} \times R \times D_{n+1} \times \cdots \times D_N}$, with: $\mathcal{T}_{i_0,i_1,\cdots,i_n} = \sum_{k=0}^{D_n} \mathbf{M}_{i_n,k} \mathcal{X}_{i_0,i_1,\cdots,i_n}$.

Latent high-order parametrization of the network: We introduce tensor factorization in the context of deep neural networks. Note that this method is independent of the dimen-

sionality of the input but we introduce it here, without loss of generality, for the case of a 4 dimensional kernel of 2D convolutions. Specifically, we consider a deep neural network composed of L layers convolutional layers, interlaced with non-linearities Φ_l, $l \in [1 \,..\, L]$. Let's consider a convolutional layer $l \in [1 \,..\, L]$, taking as input an activation tensor $\mathcal{X}_l$ and parametrized by a weight tensor $\mathcal{W}_l \in \mathbb{R}^{F,C,H,W}$, where F, C, H, W correspond respectively to number of Filters (e.g. output channels), input Channels, Height and Width of the convolutional kernel. The output of that layer, after applying non-linearity, will be $\Phi(\mathcal{X}_l \star \mathcal{W}_l)$.

We introduce a latent parametrization of the weight kernel $\mathcal{W}_l$ by expressing it as a low-rank tensor, in this paper using a Tucker decomposition [26]. In other words we express $\mathcal{W}_l$ in a latent subspace as a core tensor $\mathcal{G}_l$. The mapping to and from this subspace is done via factor matrices $\mathbf{U}_l^F, \mathbf{U}_l^C, \mathbf{U}_l^H$ and $\mathbf{U}_l^W$: $\mathcal{W}_l = \mathcal{G}_l \times_0 \mathbf{U}_l^F \times_1 \mathbf{U}_l^C \times_2 \mathbf{U}_l^H \times_3 \mathbf{U}_l^W$.

Tensor dropout: In addition to the above deterministic decomposition, we apply tensor dropout [25] to each layer, effectively randomizing the rank of the decomposition. To do so, we introduce diagonal sketching matrices $\mathbf{M}_F, \mathbf{M}_C, \mathbf{M}_H$ and $\mathbf{M}_W$, the diagonal entries of which are i.i.d. and follow a Bernoulli distribution parametrized by probability $\theta \in [0, 1]$. Specifically, we samples random vectors $\boldsymbol{\lambda}^F \in \mathbb{R}^O, \boldsymbol{\lambda}^C \in \mathbb{R}^C, \boldsymbol{\lambda}^H \in \mathbb{R}^H$ and $\boldsymbol{\lambda}^W \in \mathbb{R}^W$, the entries of which are i.i.d. following a Bernoulli distribution parametrized by probability θ. We can then define the sketching matrices as $\mathbf{M}_O = \text{diag}(\boldsymbol{\lambda}_F), \mathbf{M}_C = \text{diag}(\boldsymbol{\lambda}_C), \mathbf{M}_H = \text{diag}(\boldsymbol{\lambda}_H)$ and $\mathbf{M}_W = \text{diag}(\boldsymbol{\lambda}_W)$.

This randomization is then applied not directly to the weight tensor $\mathcal{W}$, but rather in the low-rank subspace, effectively randomizing the *rank* of the convolutional kernel:

$$\tilde{\mathcal{W}}_l = \underbrace{(\mathcal{G}_l \times_0 \mathbf{M}_F \times \cdots \times_3 \mathbf{M}_W)}_{\text{randomized core } \hat{\mathcal{G}}_l} \times_0 \underbrace{\left(\mathbf{U}_l^F \mathbf{M}_F^\top\right) \times \cdots \times_3 \left(\mathbf{U}_l^W \mathbf{M}_W^\top\right)}_{\text{randomized factors } \hat{\mathbf{U}}_l^F, \hat{\mathbf{U}}_l^I, \hat{\mathbf{U}}_l^H, \hat{\mathbf{U}}_l^W} \tag{3}$$

This stochastic reduction of the rank can be done without affecting performance thanks to the over-parametrization of deep networks, which, while crucial for learning [14, 43], create large amounts of redundancies. In addition, since $\mathcal{G}_l \times_0 \mathbf{M}_F \times_0 \left(\mathbf{U}_l^F \mathbf{M}_F^\top\right) = \mathcal{G}_l \times_0 \left(\mathbf{U}_l^F \mathbf{M}_F^\top \mathbf{M}_F\right)$, and $\mathbf{M}_F, \mathbf{M}_I, \mathbf{M}_H$ and $\mathbf{M}_W$ are idempotent, eq. 3 can be simplified to:

$$\tilde{\mathcal{W}}_l = (\mathcal{G}_l \times_0 \mathbf{M}_F \times \cdots \times_3 \mathbf{M}_W) \times_0 \mathbf{U}_l^F \times \cdots \times_3 \mathbf{U}_l^W = \tilde{\mathcal{G}}_l \times_0 \mathbf{U}_l^F \times \cdots \times_3 \mathbf{U}_l^W \tag{4}$$

In other words, we sketch the core tensor, then project it back using the original factors. Importantly, the randomization terms from the above equation, $\tilde{\mathcal{G}} = \mathcal{G} \times_0 \mathbf{M}_F \times \cdots \times_N \mathbf{M}_W$ are never explicitly computed using actual tensor contractions. Instead, the elements are directly sampled from the core and the corresponding factors, which is much more computationally effective. For the binary case, we plug in eq. (4) into the binarization procedure.

The randomization being done in the latent subspace, it induces no sparsity, unlike pruning or dropout based methods and the reconstructed weights are dense. Since the weights are learnt end-to-end with randomization on the latent cores, the network cannot rely on any single latent component for prediction, thus learning intrinsically more robust representations. The result is a network that is naturally more robust to perturbations in the inputs.

4 Experimental setting and implementation details

Datasets: We conducted experiments on two widely used datasets for image and audio classification, CIFAR-10 and Speech Command [50]. **CIFAR-10 [29]** is a widely used image

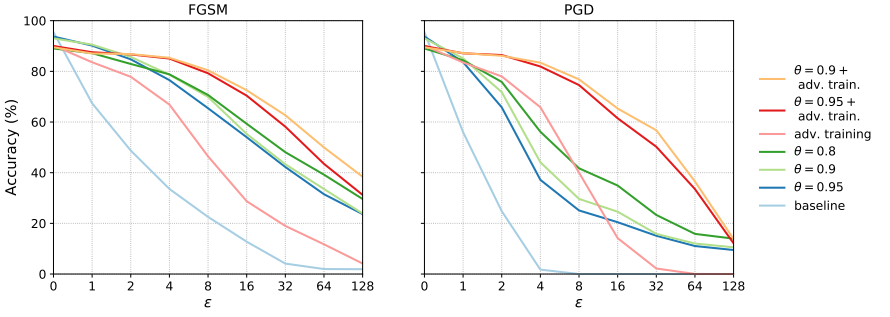

Figure 1: FGSM (left) and PGD (right) attacks on the CIFAR-10 dataset for various values of ε with and without adversarial training. Notice that our method alone surpasses the strong adversarial training defence. When combined together the robustness is increased even further.

classification dataset consisting of 60,000 images (50,000 training and 10,0000 testing) of size 32×32px in 10 classes, equally represented. We only used random horizontal flipping for data augmentation. See supplementary material for details on Speech Command.

Training the model All our CIFAR-10 [29] experiments were conducted using a ResNet-18 [21] architecture. The network was trained for 350 epochs using SGD with momentum (0.9) and a starting learning rate of 0.1 that was dropped at epoch 150 and 250 by a factor of 0.1. The weight decay was set to $1e-6$. In order to accelerate the training process the models with $\theta < 1$ were initialized from a pretrained model that was trained without stochasticity ($\theta = 1$). The binary counterpart models were trained following the method proposed by Rastegari et al. [33] using the same optimizer and learning scheduler as for the real-valued ones. The binarization and re-parametrization follows the same procedure as for CIFAR-10. See supplementary material for details on the speech recognition experiments. Training was done on a single 1080Ti GPU.

Attacking the model: For FGSM, we run the attack for various values of $\varepsilon = \{1, 2, 4, 8, 16, 32, 64, 128\}$ (for an image range $[0 \,..\, 255]$) across the entire validation/testing set averaging the results over 10 runs. For the iterative methods BIM and PGD, we follow Kurakin et al. [30] and Song et al. [44] setting the step size to 1 and the number of iterations to $\lfloor \min(\varepsilon + 4, 1.25\varepsilon) \rfloor$.

Threat Model: we assume that the attacker has access to everything (e.g. the architecture of the network, its weights, inputs, outputs, training process and gradients, etc) except the random seed used for sampling the Bernoulli random variables.

Attack	ε	Baseline	Tensor Dropout Rate θ		
			0.95	0.9	0.8
Clean *(no attack)*		**95.3**	94.5	93.0	90.1
FGSM	**2**	48.7	84.9	**86.8**	83.4
	8	22.5	65.4	69.9	**71.5**
	16	12.7	54.0	56.0	**60.3**
BIM	**2**	23.0	60.2	69.5	**71.8**
	8	0.0	26.6	33.1	**45.5**
	16	0.0	27.0	33.0	**42.4**
PGD	**2**	22.9	64.4	72.3	**76.2**
	8	0.0	27.0	28.1	**42.9**
	16	0.0	22.4	27.4	**34.3**

Table 1: **Real-valued network performance on CIFAR-10** for FGSM, BIM and PGD attacks.

All of our models were implemented using PyTorch [37] and trained on a single Titan X GPU. The latent, randomized tensor factorization was implemented using TensorLy [28]. For the adversarial attacks we used the

Attack	ε	Baseline	Tensor Dropout Rate θ					
			0.95	**0.9**	**0.8**	**0.95** + adv.	**0.9** + adv.	**0.8** + adv.
Clean *(no attack)*		**76.2**	68.9	67.2	62.6	60.4	58.6	56.7
FGSM	**2**	21.8	65.8	64.5	58.8	58.5	58.5	54.8
	8	8.7	57.6	56.5	54.2	57.1	56.2	52.6
	16	4.5	47.9	48.9	48.3	53.2	54.2	51.3
BIM	**2**	11.0	56.6	55.7	54.4	56.9	58.0	53.9
	8	0.0	40.6	44.1	44.7	53.4	53.6	51.2
	16	0.0	37.6	41.8	43.0	50.2	52.6	50.6
PGD	**2**	11.6	57.5	58.3	55.3	56.8	57.4	55.2
	8	0.0	36.7	38.6	41.8	54.5	53.8	51.8
	16	0.0	26.6	29.0	32.0	50.0	49.8	49.5

Table 2: **Real-valued network performance on CIFAR-100** for FGSM, BIM and PGD attacks. + adv indicated training with both θ and adversarial training.

FoolBox 2.0 [39] and AutoAttack [11] packages.

5 Results

In this section, we empirically demonstrate the robustness property of our proposed method against adversarial examples by extensively evaluating it on CIFAR-10 and comparing it against existing state of the art defense techniques. Moreover, we show that our approach can be combined with adversarial training based techniques leading to further robustness gains. All the experiments are run using the experimental setup described in Section 4.

Robustness to adversarial attacks: When evaluated against FGSM attacks on CIFAR-10, our method is significantly more robust than the baseline approach, especially for high values of ε (see Table 1 and Fig. 1). Furthermore, the results presented in Table 1 show that for lower values of $\theta = 0.8$, our network significantly decreases the typical high attack success-rate achieved by strong iterative attack algorithms such as BIM and PGD. We note however that against newer attack sets, such AutoAttack [11], our model is noticeable less robust. We leave further exploration of this area within the tensors framework for future work.

Attack	ε	Baseline	Tensor Dropout Rate θ		
			0.95	**0.9**	**0.8**
AA	**8**	0.0	30.0	35.4	37.5

Table 3: **Real-valued network performance on CIFAR-10** using AutoAttack [11]

Comparison to the State of the Art: To better understand the performance of our method, we compare it against existing state of theart defense techniques such as: DQ [32] in which the authors attempt to reduce the propagation of the adversarial attacks inside the network by controlling the Lipschitz constant, features squeezing [53] that simple reduces the dimensionality of the search space by controlling the color bit depth of each pixel while applying spatial smoothing and finally, adversarial training in two variants: R+FGSM as proposed in [32] and PGD [34]. Where the later (i.e. adversarial training) is considered to

be one of the strongest defence techniques developed. For feature-squeezing we used 5 bits for image color reduction combined with a 2×2 median filter. For adversarial training as in [30, 34] we sample the number of steps (for PGD) and ε randomly. As the results from Table 4 show, our method consistently outperforms existing defense strategies for various attacks (FGSM, BIM, PGD), across different values of $\varepsilon = \{2, 8, 16\}$. Furthermore, when combined with adversarial training our method can further increase its resilience to attacks.

Method	clean	**FGSM** $\varepsilon = 2/8/16$	**PGD** $\varepsilon = 2/8/16$
Normal	95.4	48.7/22.6/12.9	22.9/0/0
DQ [32]	**95.9**	68/53/42	62/4/0
Feature squeezing [53]	94.1	61/35/27	64/2/0
Adv. training+R FGSM [32]	91.6	81/52/38	84/43/11
Adv. training PGD [34]	86.6	74/46/31	76/44/20
Ours ($\theta = 0.8$)	90.1	83.4/71.5/60.3	76.2/42.9/34.3
Ours ($\theta = 0.95$) **+ Adv. training PGD**	89.5	**86.5/81.4/70.1**	**84.9/75.4/59.8**

Table 4: **Comparison against various defense methods** on CIFAR10. Our method significantly outperforms other state of theart methods, especially when combined with adversarial training.

Defending against black-box attacks While we focus mostly on white box attacks, we also show that our approach increases robustness against black-box attacks, following the same standard setting as Mustafa et al. [35]. Our method is more robust in all cases Table 5.

One thing to notice is the relative difference in performance: our method starts with a slightly lower performance on the clean set but is much less affected by the adversarial attacks.

Method	Clean	**FGSM** $\varepsilon = 2/8/16$	**BIM** $\varepsilon = 2/8/16$	**PGD** $\varepsilon = 2/8/16$
Baseline	95.4	94.2/87.8/79.1	93.0/84.7/77.5	94.0/82.3/59.9
Ours	89.5	87.4/87.1/83.3	87.4/84.5/83.8	87.6/85.2/83.4

Table 5: **Defending against Black-box attacks** [35]

Defending against omniscient attackers: Another interesting question is whether our defense strategy would still work against an *omniscient* attacker, i.e., an attacker with access to the full un-randomized weights. We trained a network for $\theta = 0.9$. Then, during training, we first generate an adversarial example using the full, un-randomized weights (i.e. $\theta = 1$) and test it using the stochasticity (i.e. $\theta = 0.9$). As can be seen in Table 6c, our network is still robust against these attacks, despite the attacker having full access to the weights. Similar behaviour can be observed for other values of θ. Note that this is an extreme scenario and in general, the weights could be stored separately (and safely) with the network getting, at each time, the randomly reconstructed weights.

Defending against adaptive attacks: Given that the proposed method hardens the models against adversarial attacks via a randomisation performed in a latent subspace, herein we analyse its resilience against adaptive methods typically capable of overcoming randomization-based approaches. In particular, following [4] at each iteration of gradient descent, for each convolutional layer, instead of taking a step in the direction of $\nabla_x f(x)$ we move in the direction of $\sum_{i=1}^{k} \nabla_x f(x)$ where each pass has the weights randomized in the latent space using our approach. While in [4] the authors use $k = 10$ we try with up to $k = 20$ but without noticing any significant increase in the success rate of the attack. The PGD attack itself was run for 500 iterations as in [4]. The experiment reported was conducted on CIFAR 10 using our best

(a) **Robustness of our method against BPDA on CIFAR10**.

Attack / ε	2	8	16
BPDA [4]	83.3	54.9	43.8

(b) **Robustness against attacks on CIFAR10 for methods applying the randomization on the subspace (ours) vs directly on the activations**. The network used and attack settings are aligned with the ones used in Wang et al. [47].

Method	**Clean**	**FGSM**	**BIM**	**PGD**
Ours	85.9	60.0	42.9	43.8
[47]	83.4	41.7	32.3	35.2

(c) **Robustness against attacker with access to the un-randomized weights**. FGSM, BIM and PGD attacks with $\varepsilon \in \{2, 8, 16\}$ are computed using the full (un-randomized) weights (i.e. $\theta = 1$) and used against the **same** network with the **same** weights but with $\theta = 0.9$.

ε	**FGSM**	**PGD**	**BIM**
2	91.1	82.5	81.9
8	85.4	74.1	80.4
16	82.4	53.9	80.1

Table 6: Robustness in various scenarios on CIFAR10.

model. As the results from Table 6a show, our method is capable of offering a reasonable degree of robustness even against such attack that probe wherever the defense technique mask or shatter the gradients.

6 Ablation studies

To further validate our findings we test out approach on two different scenarios: on fully binarized networks (Section 6) and audio classification task (see supplementary material). Furthermore, we empirically validate the importance of performing the randomization in a latent subspace (Section 6) and that of using a tensorized form of the algorithm over its matrixised version (Section 6). See supplementary material for results on speech recognition.

Importance of randomization in a latent subspace Here, we demonstrate the importance of incorporating the stochasticity in the latent subspace rather than in the parameter space directly. We compare our method against the so-called *Defensive dropout* of [47], in which the authors proposed to apply dropout directly to the activation of the first fully connected layer at test time. Since their method requires the presence of multiple fully connected layers, we apply our randomized tensorization directly on their architecture. The results of this comparison can be seen in Table 6b in the same setting and the same epsilon as in the original paper [47]. Notice that our approach consistently outperforms the defensive dropout.

Matrix vs. Tensor decomposition Tensor methods have the ability to leverage the multilinear structure in the data, weight and activations. This structure is typically discarded by matrix based approaches, thus loosing information that we would want to preserve and leverage when building robust neural networks.

It can be noted that the matrix case is a special case of our approach. Specifically, in Equation 3, this can be obtained by setting $M_I = M_H = M_W = U_l^I = U_l^H = U_l^W = \mathbf{I}$. The equality can then be rewritten in term of the mode-1 unfolding to obtain the matrix case. To demonstrate the advantage of our tensor-based approach compared to the matrix

Method	**Clean**	**FGSM** (2/8/16)	**BIM** (2/8/16)	**PGD** (2/8/16)
Ours	94.5	84.9/65.4/54.0	60.6/26.6/27.0	64.4/27.0/22.4
Matrix	93.7	76.9/52.8/40.9	44.6/17.5/18.2	50.0/16.9/15.0

Table 7: **Robustness against attacks on CIFAR10 for the tensor decomposition (ours) and matrix case**.

case, we ran an additional experiment on CIFAR-10, using a ResNet-18 architecture for both the matrix and tensor version of our method, for the same value θ. As the results from Table 7 show, the tensor decompositions offers consistent gains over the matrix one.

Attacking binary neural networks. Network binarization is the most extreme case of quantization, where the weights and features are represented using a single bit [6, 10, 38]. The typical approach quantizes the network using the *sign* function [9], however this introduces high quantization errors that hinder the learning process. To alleviate this, a real-valued scaling factor is introduced by Rastegari et al. [38]. In this work we binarize the network following Rastegari et al. [38]: $\mathcal{I} * \mathcal{W} = (\text{sgn}(\mathcal{I}) \circledast \text{sgn}(\mathcal{W})) \odot \mathcal{K}\boldsymbol{\alpha}$, where $\mathcal{I} \in \mathbb{R}^{c \times w_{in} \times h_{in}}$ and $\mathcal{W} \in \mathbb{R}^{c \times h \times w}$ denote the input and the weight of the L-th convolutional layer, $\boldsymbol{\alpha} \in \mathbb{R}^{c \times 1 \times 1}$ represent the weight scaling factor and $\mathcal{K} \in \mathbb{R}^{1 \times h_{out} \times w_{out}}$ the input scaling factor. Both $\boldsymbol{\alpha}$ and $\mathcal{K}$ are computed analytically as proposed by Rastegari et al. [38]. While it was previously thought that such binarized networks are more resilient to adversarial attacks [16, 23, 33] than their real-valued counterpart, in this work we confirm the recent findings of [32] by showing that in fact it is the opposite, i.e., binary networks are more susceptible to adversarial attacks. Typically, during the training phase the derivative of the quantization function (sgn) is approximated using a STE (e.g., an identity function clipped to $[-1, 1]$ in this case). The same estimator can be used during the attacking phase and it often leads to a high rate of success of the attacks. Interestingly, the results in Table 8 show that we can go one step further by approximating the derivative of the *sgn* function using $\tanh(x)$ and $\tanh(0.75x)$ respectively. The use of these approximations make the binary networks become more sensitive to the attacks. When evaluated on a binary network on the CIFAR-10 dataset, as for the real-valued models, our method shows significant improvements across the entire range of values and attacks tested (see Table 8 and Fig. 1). We note that for the binary case, since such networks have a lower representational capacity, we set $\theta = 0.99$.

Attack	ε	Baseline			Ours
		Id	$\tanh(x)$	$\tanh(0.75x)$	θ=0.99
Clean *(no attack)*		83.7	83.7	83.7	80.0
FGSM	**2**	36.6	34.5	34.1	**76.9**
	8	6.9	6.1	5.8	**65.0**
	16	4.3	3.4	3.0	**58.7**
BIM	**2**	37.0	34.7	35.1	**66.3**
	8	0.0	0.0	0.0	**46.4**
	16	0.0	0.0	0.0	**44.0**
PGD	**2**	41.7	38.7	39.8	**67.5**
	8	0.1	0.0	0.0	**47.9**
	16	0.0	0.0	0.0	**41.5**

Table 8: **Binary network performance on CIFAR-10** for FGSM, BIM and PGD attacks. Our approach is significantly more robust, especially against iterative attacks.

7 Conclusion

Defensive tensorization is a novel adversarial defence technique that leverages a latent high order factorization of the network. Tensor dropout is applied in the latent subspace, directly on the factorized weights parametrizing each layer, resulting in dense reconstructed weights, without the sparsity or perturbations typically induced by randomization. We empirically demonstrate that this approach makes the network significantly more robust to adversarial attacks. Contrarily to a widely spread belief, we observe that binary networks are *more* sensitive to adversarial attacks than their real-valued counter-part. We show that our method significantly improves robustness in the face of adversarial attacks for both binary and real-valued networks. We demonstrate this empirically for both image and audio classification.

References

[1] Naveed Akhtar and Ajmal Mian. Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access*, 6:14410–14430, 2018.

[2] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

[3] Marcella Astrid and Seung-Ik Lee. Cp-decomposition with tensor power method for convolutional neural networks compression. *CoRR*, abs/1701.07148, 2017.

[4] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*, 2018.

[5] Casey Battaglino, Grey Ballard, and Tamara G Kolda. A practical randomized cp tensor decomposition. *SIAM Journal on Matrix Analysis and Applications*, 39(2):876–901, 2018.

[6] Adrian Bulat and Georgios Tzimiropoulos. Binarized convolutional landmark localizers for human pose estimation and face alignment with limited resources. In *ICCV*, 2017.

[7] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57. IEEE, 2017.

[8] Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, pages 1310–1320. PMLR, 2019.

[9] Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. Binaryconnect: Training deep neural networks with binary weights during propagations. In *NIPS*, 2015.

[10] Matthieu Courbariaux, Itay Hubara, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1. *arXiv*, 2016.

[11] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International conference on machine learning*, pages 2206–2216. PMLR, 2020.

[12] Guneet S Dhillon, Kamyar Azizzadenesheli, Zachary C Lipton, Jeremy Bernstein, Jean Kossaifi, Aran Khanna, and Anima Anandkumar. Stochastic activation pruning for robust adversarial defense. *arXiv preprint arXiv:1803.01442*, 2018.

[13] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Computer Vision and Pattern Recognition*, 2018.

[14] Simon S Du and Jason D Lee. On the power of over-parametrization in neural networks with quadratic activation. In *ICML*, 2018.

[15] N Benjamin Erichson, Krithika Manohar, Steven L Brunton, and J Nathan Kutz. Randomized cp tensor decomposition. *arXiv preprint arXiv:1703.09074*, 2017.

[16] Angus Galloway, Graham W Taylor, and Medhat Moussa. Attacking binarized neural networks. *arXiv preprint arXiv:1711.00449*, 2017.

[17] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[18] Sven Gowal, Krishnamurthy Dj Dvijotham, Robert Stanforth, Rudy Bunel, Chongli Qin, Jonathan Uesato, Relja Arandjelovic, Timothy Mann, and Pushmeet Kohli. Scalable verified training for provably robust image classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4842–4851, 2019.

[19] Sven Gowal, Chongli Qin, Jonathan Uesato, Timothy Mann, and Pushmeet Kohli. Uncovering the limits of adversarial training against norm-bounded adversarial examples. *arXiv preprint arXiv:2010.03593*, 2020.

[20] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens van der Maaten. Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117*, 2017.

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition*, 2016.

[22] Sandy Huang, Nicolas Papernot, Ian Goodfellow, Yan Duan, and Pieter Abbeel. Adversarial attacks on neural network policies. *arXiv preprint arXiv:1702.02284*, 2017.

[23] Elias B Khalil, Amrita Gupta, and Bistra Dilkina. Combinatorial attacks on binarized neural networks. *arXiv preprint arXiv:1810.03538*, 2018.

[24] Yong-Deok Kim, Eunhyeok Park, Sungjoo Yoo, Taelim Choi, Lu Yang, and Dongjun Shin. Compression of deep convolutional neural networks for fast and low power mobile applications. *ICLR*, 05 2016.

[25] Arinbjörn Kolbeinsson, Jean Kossaifi, Yannis Panagakis, Adrian Bulat, Animashree Anandkumar, Ioanna Tzoulaki, and Paul M Matthews. Tensor dropout for robust learning. *IEEE Journal of Selected Topics in Signal Processing*, 15(3):630–640, 2021.

[26] Tamara G. Kolda and Brett W. Bader. Tensor decompositions and applications. *SIAM REVIEW*, 51(3):455–500, 2009.

[27] Jean Kossaifi, Adrian Bulat, Georgios Tzimiropoulos, and Maja Pantic. T-net: Parametrizing fully convolutional nets with a single high-order tensor. In *Computer Vision and Pattern Recognition*, 2019.

[28] Jean Kossaifi, Yannis Panagakis, Anima Anandkumar, and Maja Pantic. Tensorly: Tensor learning in python. *The Journal of Machine Learning Research*, 20(1):925–930, 2019.

[29] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[30] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*, 2016.

[31] Vadim Lebedev, Yaroslav Ganin, Maksim Rakhuba, Ivan V. Oseledets, and Victor S. Lempitsky. Speeding-up convolutional neural networks using fine-tuned cp-decomposition. In *ICLR*, 2015.

[32] Ji Lin, Chuang Gan, and Song Han. Defensive quantization: When efficiency meets robustness. *arXiv preprint arXiv:1904.08444*, 2019.

[33] Xuanqing Liu, Yao Li, Chongruo Wu, and Cho-Jui Hsieh. Adv-bnn: Improved adversarial defense through robust bayesian neural network. *arXiv preprint arXiv:1810.01279*, 2018.

[34] 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.

[35] Aamir Mustafa, Salman Khan, Munawar Hayat, Roland Goecke, Jianbing Shen, and Ling Shao. Adversarial defense by restricting the hidden space of deep neural networks. In *International Conference on Computer Vision*, 2019.

[36] Alexander Novikov, Dmitry Podoprikhin, Anton Osokin, and Dmitry Vetrov. Tensorizing neural networks. In *Neural Information Processing Systems*, 2015.

[37] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*, 2017.

[38] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *ECCV*, 2016.

[39] Jonas Rauber, Wieland Brendel, and Matthias Bethge. Foolbox v0. 8.0: A python toolbox to benchmark the robustness of machine learning models. *arXiv preprint arXiv:1707.04131*, 5, 2017.

[40] Leslie Rice, Eric Wong, and Zico Kolter. Overfitting in adversarially robust deep learning. In *International Conference on Machine Learning*, pages 8093–8104. PMLR, 2020.

[41] Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-gan: Protecting classifiers against adversarial attacks using generative models. *arXiv preprint arXiv:1805.06605*, 2018.

[42] Nicholas D Sidiropoulos, Evangelos E Papalexakis, and Christos Faloutsos. Parallel randomly compressed cubes: A scalable distributed architecture for big tensor decomposition. *IEEE Signal Processing Magazine*, 31(5):57–70, 2014.

[43] Mahdi Soltanolkotabi, Adel Javanmard, and Jason D Lee. Theoretical insights into the optimization landscape of over-parameterized shallow neural networks. *IEEE Transactions on Information Theory*, 2018.

[44] Yang Song, Taesup Kim, Sebastian Nowozin, Stefano Ermon, and Nate Kushman. Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. *arXiv preprint arXiv:1710.10766*, 2017.

[45] Charalampos E Tsourakakis. Mach: Fast randomized tensor decompositions. In *Proceedings of the 2010 SIAM International Conference on Data Mining*, pages 689–700. SIAM, 2010.

[46] Nico Vervliet, Otto Debals, Laurent Sorber, and Lieven De Lathauwer. Breaking the curse of dimensionality using decompositions of incomplete tensors: Tensor-based scientific computing in big data analysis. *IEEE Signal Processing Magazine*, 31(5): 71–79, 2014.

[47] Siyue Wang, Xiao Wang, Pu Zhao, Wujie Wen, David Kaeli, Peter Chin, and Xue Lin. Defensive dropout for hardening deep neural networks under adversarial attacks. In *Proceedings of the International Conference on Computer-Aided Design*, page 71. ACM, 2018.

[48] Yining Wang, Hsiao-Yu Tung, Alexander J Smola, and Anima Anandkumar. Fast and guaranteed tensor decomposition via sketching. In *Neural Information Processing Systems*, 2015.

[49] Yisen Wang, Difan Zou, Jinfeng Yi, James Bailey, Xingjun Ma, and Quanquan Gu. Improving adversarial robustness requires revisiting misclassified examples. In *International Conference on Learning Representations*, 2020.

[50] Pete Warden. Speech Commands: A Dataset for Limited-Vocabulary Speech Recognition. *arXiv e-prints*, art. arXiv:1804.03209, Apr 2018.

[51] Cihang Xie, Jianyu Wang, Zhishuai Zhang, Zhou Ren, and Alan Yuille. Mitigating adversarial effects through randomization. In *International Conference on Learning Representations*, 2018.

[52] Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan L Yuille, and Kaiming He. Feature denoising for improving adversarial robustness. In *Computer Vision and Pattern Recognition*, 2019.

[53] Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. *arXiv preprint arXiv:1704.01155*, 2017.

[54] Yuzhe Yang, Guo Zhang, Dina Katabi, and Zhi Xu. Me-net: Towards effective adversarial robustness with matrix estimation. *ICML*, 2019.

[55] Longhao Yuan, Chao Li, Jianting Cao, and Qibin Zhao. Randomized tensor ring decomposition and its application to large-scale data reconstruction. *arXiv preprint arXiv:1901.01652*, 2019.

[56] Huan Zhang, Hongge Chen, Chaowei Xiao, Sven Gowal, Robert Stanforth, Bo Li, Duane Boning, and Cho-Jui Hsieh. Towards stable and efficient training of verifiably robust neural networks. *arXiv preprint arXiv:1906.06316*, 2019.

[57] Guoxu Zhou, Andrzej Cichocki, and Shengli Xie. Decomposition of big tensors with low multilinear rank. *arXiv preprint arXiv:1412.1885*, 2014.