Local Competition and Stochasticity for Adversarial Robustness in Deep Learning

Konstantinos P. Panousis[†] **Antonios Alexos**[‡] **Sergios Theodoridis**[§] **Sotirios Chatzis**[†]

[†]Cyprus University of Technology, Limassol, Cyprus
[‡]University of California Irvine, CA, USA
[§]National and Kapodistrian University of Athens, Athens, Greece & Aalborg University, Denmark
k.panousis@cut.ac.cy

Abstract

This work addresses adversarial robustness in deep learning by considering deep networks with stochastic local winner-takes-all (LWTA) activations. This type of network units result in sparse representations from each model layer, as the units are organized in blocks where only one unit generates a non-zero output. The main operating principle of the introduced units lies on stochastic arguments, as the network performs posterior sampling over competing units to select the winner. We combine these LWTA arguments with tools from the field of Bayesian non-parametrics, specifically the stick-breaking construction of the Indian Buffet Process, to allow for inferring the sub-part of each layer that is essential for modeling the data at hand. Then, inference is performed by means of stochastic variational Bayes. We perform a thorough experimental evaluation of our model using benchmark datasets. As we show, our method achieves high robustness to adversarial perturbations, with state-of-the-art performance in powerful adversarial attack schemes.

1 Introduction

Despite their widespread success, Deep Neural Networks (DNNs) are notorious for being highly susceptible to adversarial attacks. *Adversarial examples*, i.e. inputs comprising carefully crafted perturbations, are designed with the aim of "*fooling*" a considered model into *misclassification*. It has been shown that, even small perturbations in an original input, e.g. via an ℓ_p norm, may successfully render a DNN vulnerable; this highlights the frailness of common DNNs (Papernot et al., 2017). This vulnerability casts serious doubts regarding the confident use of modern DNNs in safety-critical applications, such as autonomous driving (Boloor et al., 2019; Chen et al., 2015), video recognition (Jiang et al., 2019), healthcare (Finlayson et al., 2019) and other real-world scenarios (Kurakin et al., 2016). To address these concerns, significant research effort has been devoted to adversarially-robust DNNs.

Adversarial attacks, and associated defense strategies, comprise many different approaches sharing the same goal; make deep architectures more *reliable* and *robust*. In general, adversarially-robust models are obtained via the following types of strategies: (i) *Adversarial Training*, where a model is trained with both original and perturbed data (Madry et al., 2017; Tramèr et al., 2017; Shrivastava et al., 2017); (ii) *Manifold Projections*, where the original data points are projected onto a different subspace, presuming that, therein, the effects of the perturbations can be mitigated (Jalal et al., 2017; Shen et al., 2017; Song et al., 2017); (iii) *Stochastic Modeling*, where some randomization of the input data and/or the neuronal activations is performed on each hidden layer (Prakash et al., 2018; Dhillon et al., 2018; Xie et al., 2017); and (iv) *Preprocessing*, where some aspects of either the data or the neuronal activations are modified to induce robustness (Buckman et al., 2018; Guo et al., 2017; Kabilan et al., 2018).

Despite these advances, most of the currently considered approaches and architectures are particularly tailored to the specific characteristics of a considered type of attacks. This implies that such a model may fail completely in case the adversarial attack patterns change in a radical manner. To overcome this challenge, we posit that we need to devise an activation function paradigm different from common neuronal activation

Proceedings of the 24[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2021, San Diego, California, USA. PMLR: Volume 130.

functions, especially ReLU.

Recently, the deep learning community has shown fresh interest in more biologically plausible models. In this context, there is an increasing body of evidence from Neuroscience that neurons with similar functions in a biological system are aggregated together in blocks, and local competition takes place therein for their activation (Local-Winner-Takes-All, LWTA, mechanism). Under this scheme, in each block, only one neuron can be active at a given time, while the rest are inhibited to silence. Crucially, it appears that this mechanism is of stochastic nature, in the sense that the same system may produce different neuron activation patterns when presented with exactly the same stimulus at multiple times (Kandel et al., 2000; Andersen et al., 1969; Stefanis, 1969; Douglas and Martin, 2004; Lansner, 2009). Previous implementations of the LWTA mechanism in deep learning have shown that the obtained sparse representations of the input are quite informative for classification purposes (Lee and Seung, 1999; Olshausen and Field, 1996), while exhibiting *automatic gain control*, *noise suppression* and *robustness to catastrophic forgetting* (Srivastava et al., 2013; Grossberg, 1982; Carpenter and Grossberg, 1988). However, previous authors have not treated the LWTA mechanism under a systematic stochastic modeling viewpoint, which is a key component in actual biological systems.

Finally, recent works in the community, e.g. Verma and Swami (2019), have explored the susceptibility of the conventional one-hot output encoding of deep learning classifiers to adversarial attacks. By borrowing arguments from coding theory, the authors examine the effect of encoding the deep learning classifier output using *error-correcting output codes* in the adversarial scenario. The experimental results suggest that such an encoding technique enhances the robustness of a considered architecture, while retaining high classification accuracy in the benign context.

Drawing upon these insights, in this work we propose a new deep network design scheme that is particularly tailored to address adversarially-robust deep learning. Our approach falls under the stochastic modeling type of approaches. Specifically, we propose a deep network configuration framework employing: (i) stochastic LWTA activations; and (ii) an Indian Buffet process (IBP)-based mechanism for learning which sub-parts of the network are essential for data modeling. We combine this modeling rationale with Error Correcting Output Codes (Verma and Swami, 2019) to further enhance performance.

We evaluate our approach using well-known benchmark datasets and network architectures. We provide related source code at: `https://github.com/konpanousis/adversarial_ecoc_lwta`. The obtained empirical evidence vouches for the potency of our approach, yielding state-of-the-art robustness against powerful benchmark attacks. The remainder of the paper is organized as follows: In Section 2, we introduce the necessary theoretical background. In Section 3, we introduce the proposed approach and describe its rationale and inference algorithms. In Section 4, we perform extensive experimental evaluations, providing insights for the behavior of the produced framework. In Section 5, we summarize the contribution of this work.

2 Technical Background

2.1 Indian Buffet Process

The Indian Buffet Process (IBP) (Ghahramani and Griffiths, 2006) defines a probability distribution over infinite binary matrices. IBP can be used as a flexible prior for latent factor models, allowing the number of involved latent features to be unbounded and inferred in a data-driven fashion. Its construction induces sparsity, while at the same time allowing for more features to emerge, as new observations appear. Here, we focus on the stick-breaking construction of the IBP proposed by Teh et al. (2007), which renders it amenable to Variational Inference. Let us consider N observations, and a binary matrix $\boldsymbol{Z} = [z_{j,k}]_{j,k=1}^{N,K}$; each entry therein, indicates the existence of feature k in observation j. Taking the infinite limit $K \to \infty$, we can construct the following hierarchical representation (Teh et al., 2007; Theodoridis, 2020):

$$u_k \sim \text{Beta}(\alpha, 1),\ \pi_k = \prod_{i=1}^{k} u_i,\ z_{j,k} \sim \text{Bernoulli}(\pi_k),\ \forall j$$

where α is a non-negative parameter, controlling the induced sparsity.

2.2 Local Winner-Takes-All

Let us assume a single layer of an LWTA-based network, comprising K LWTA blocks with U competing units therein. Each block produces an output $\boldsymbol{y}_k \in \text{one\_hot(U)},\ k = 1, \ldots, K$, given some input $\boldsymbol{x} \in \mathbb{R}^J$. Each *linear* unit in each block computes its activation $h_k^u,\ u = 1, \ldots U$, and the output of the block is decided via competition among its units. Thus, for each block, k, and unit, u, therein, the output reads:

$$y_k^u = g(h_k^1, \ldots, h_k^U) \tag{1}$$

where $g(\cdot)$ is the *competition function*. The activation of each individual unit follows the conventional inner product computation $h_k^u = \boldsymbol{w}_{ku}^T \boldsymbol{x}$, where $\boldsymbol{W} \in \mathbb{R}^{J \times K \times U}$ is

the weight matrix of the LWTA layer. In a conventional *hard* LWTA network, the final output reads:

$$y_k^u = \begin{cases} 1, & \text{if } h_k^u \geq h_k^i, \qquad \forall i = 1, \ldots, U,\ i \neq u \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

To bypass the restrictive assumption of binary output, more expressive versions of the competition function have been proposed in the literature, e.g., Srivastava et al. (2013). These generalized *hard* LWTA networks postulate:

$$y_k^u = \begin{cases} h_k^u, & \text{if } h_k^u \geq h_k^i, \qquad \forall i = 1, \ldots, U,\ i \neq u \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

This way, only the unit with the *strongest* activation produces an output in each block, while the others are inhibited to silence, i.e., the zero value. This way, the output of each layer of the network yields a sparse representation according to the competition outcome within each block.

The above schemes do not respect a major aspect that is predominant in biological systems, namely *stochasticity*. We posit that this aspect may be crucial for endowing deep networks with adversarial robustness. To this end, we adopt a scheme similar to Panousis et al. (2019). That work proposed a novel competitive random sampling procedure. We explain this scheme in detail in the following Section.

3 Model Definition

In this work, we consider a principled way of designing deep neural networks that renders their inferred representations considerably more robust to adversarial attacks. To this end, we utilize a novel stochastic LWTA type of activation functions, and we combine it with appropriate sparsity-inducing arguments from nonparametric Bayesian statistics.

Let us assume an input $\boldsymbol{X} \in \mathbb{R}^{N \times J}$ with N examples, comprising J features each. In conventional deep architectures, each hidden layer comprises nonlinear units; the input is presented to the layer, which then computes an affine transformation via the inner product of the input with weights $\boldsymbol{W} \in \mathbb{R}^{J \times K}$, producing outputs $\boldsymbol{Y} \in \mathbb{R}^{N \times K}$. The described computation for each example n yields $\boldsymbol{y}_n = \sigma(\boldsymbol{W}^T \boldsymbol{x}_n + \boldsymbol{b}) \in \mathbb{R}^K,\ n = 1, \ldots, N$, where $\boldsymbol{b} \in \mathbb{R}^K$ is a bias factor and $\sigma(\cdot)$ is a non-linear activation function, e.g. ReLU. An architecture comprises intermediate and output layers of this type.

Under the proposed stochastic LWTA-based modeling rationale, singular units are replaced by LWTA blocks, each containing a set of *competing linear* units. Thus, the layer input is now presented to each different block and each unit therein, via different weights. Letting K be the number of LWTA blocks and U the number of competing units in each block, the weights are now represented via a three-dimensional matrix $\boldsymbol{W} \in \mathbb{R}^{J \times K \times U}$.

Drawing inspiration from Panousis et al. (2019), we postulate that the local competition in each block is performed via a competitive random sampling procedure. The higher the output of a competing unit, the higher the probability of it being the winner. However, the winner is selected stochastically.

In the following, we introduce a set of discrete latent vectors $\boldsymbol{\xi}_n \in \text{one\_hot}(U)^K$, in order to encode the outcome of the local competition between the units in each LWTA block of a network layer. For each data input, $\boldsymbol{x}_n$, the non-zero entries in the aforementioned one-hot representation denotes the winning unit among the U competitors in each of the K blocks of the layer.

To further enhance the stochasticity and regularization of the resulting model, we turn to the nonparametric Bayesian framework. Specifically, we introduce a matrix of latent variables $\boldsymbol{Z} \in \{0, 1\}^{J \times K}$, to explicitly regularize the model by inferring whether the model actually needs to use each connection in each layer. Each entry, $z_{j,k}$, therein is set to one, if the j^{th} dimension of the input is presented to the k^{th} block, otherwise $z_{j,k} = 0$. We impose the sparsity-inducing IBP prior over the latent variables z and perform inference over them. Essentially, if all the connections leading to some block are set to $z_{j,k} = 0$, the block is effectively zeroed-out of the network. This way, we induce sparsity in the network architecture.

On this basis, we now define the output of a layer of the considered model, $\boldsymbol{y}_n \in \mathbb{R}^{K \cdot U}$, as follows:

$$[\boldsymbol{y}_n]_{ku} = [\boldsymbol{\xi}_n]_{ku} \sum_{j=1}^{J} (w_{j,k,u} \cdot z_{j,k}) \cdot [\boldsymbol{x}_n]_j \in \mathbb{R} \tag{4}$$

To facilitate a competitive random sampling procedure in a data-driven fashion, the latent indicators $\boldsymbol{\xi}_n$ are drawn from a posterior Categorical distribution. The concept is that the higher the output of a linear competing unit, the higher the probability of it being the winner. We yield:

$$q([\boldsymbol{\xi}_n]_k) = \text{Discrete}\left([\boldsymbol{\xi}_n]_k \Big| \text{softmax}\left(\sum_{j=1}^{J} [w_{j,k,u}]_{u=1}^{U} \cdot z_{j,k} \cdot [\boldsymbol{x}_n]_j \right) \right) \tag{5}$$

Further, we postulate that the latent variables z are drawn from Bernoulli posteriors, such that:

$$q(z_{j,k}) = \text{Bernoulli}(z_{j,k} | \tilde{\pi}_{j,k}) \tag{6}$$

These are trained by means of variational Bayes, as we

describe next, while we resort to fixed-point estimation for the weight matrices $\boldsymbol{W}$.

For the output layer of our approach, we perform the standard inner product computation followed by a softmax, while imposing an IBP over the connections, similar to the inner layers. Specifically, let us assume a C-unit penultimate layer with input $\boldsymbol{X} \in \mathbb{R}^{N\times J}$ and weights $\boldsymbol{W} \in \mathbb{R}^{J\times C}$. We introduce an auxiliary matrix of latent variables $\boldsymbol{Z} \in \{0,1\}^{J\times C}$. Then, the output $\boldsymbol{Y} \in \mathbb{R}^{N\times C}$ yields:

$$y_{n,c} = \text{softmax}\big(\sum_{j=1}^{J}(w_{j,c}\cdot z_{j,c})\cdot[\boldsymbol{x}_n]_j\big) \in \mathbb{R} \tag{7}$$

where the latent variables in $\boldsymbol{Z}$ are drawn independently from a Bernoulli posterior:

$$q(z_{j,c}) = \text{Bernoulli}(z_{j,c}|\tilde{\pi}_{j,c}) \tag{8}$$

To perform variational inference for the model latent variables, we impose a symmetric Discrete prior over the latent indicators, $[\boldsymbol{\xi}_n]_k \sim \text{Discrete}(1/U)$. On the other hand, the prior imposed over $\boldsymbol{Z}$ follows the stick-breaking construction of the IBP, to facilitate data-driven sparsity induction.

The formulation of the proposed modeling approach is now complete. A graphical illustration of the resulting model is depicted in Fig. 1a.

3.1 Convolutional Layers

Further, to accommodate architectures comprising convolutional operations, we devise a convolutional variant inspired from Panousis et al. (2019). Specifically, let us assume input tensors $\{\boldsymbol{X}_n\}_{n=1}^{N} \in \mathbb{R}^{H\times L\times C}$ at a specific layer, where H, L, C are the height, length and channels of the input. We define a set of kernels, each with weights $\boldsymbol{W}_k \in \mathbb{R}^{h\times l\times C\times U}$, where h, l, C, U are the kernel height, length, channels and *competing feature maps*, and $k = 1,\dots K$. Thus, analogously to the grouping of linear units in the dense layers, in this case, local competition is performed among *feature maps. Each kernel is treated as an LWTA block with competing feature maps*; each layer comprises multiple kernels.

We additionally consider analogous auxiliary binary latent variables $\boldsymbol{z} \in \{0,1\}^K$ to further regularize the convolutional layers. Here, we retain or omit full LWTA blocks (convolutional kernels), as opposed to single connections. This way, at a given layer of the proposed convolutional variant, the output $\boldsymbol{Y}_n \in \mathbb{R}^{H\times L\times K\cdot U}$ is obtained via concatenation along the last dimension of the subtensors:

$$[\boldsymbol{Y}_n]_k = [\boldsymbol{\xi}_n]_k\left((z_k\cdot W_k)\star\boldsymbol{X}_n\right) \in \mathbb{R}^{H\times L\times U} \tag{9}$$

where $\boldsymbol{X}_n$ is the input tensor for the n^{th} data point, and "$\star$" denotes the convolution operation. Turning to the competition function, we follow the same rationale, such that the sampling procedure is driven from the outputs of the competing feature maps:

$$q([\boldsymbol{\xi}_n]_k) = \text{Discrete}\Big([\boldsymbol{\xi}_n]_k\Big|\text{softmax}(\sum_{h',l'}[(z_k\cdot W_k)\star\boldsymbol{X}_n]_{h',l',u}\Big)$$

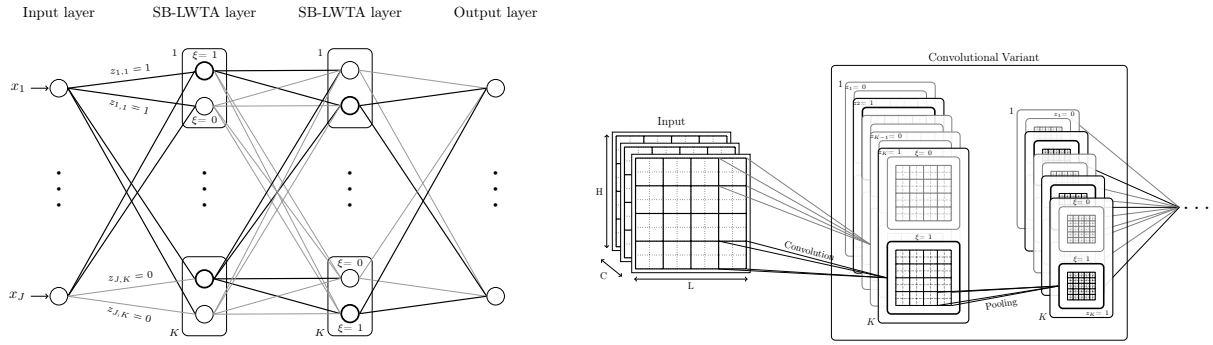
We impose an IBP prior over $\boldsymbol{z}$, while a posteriori drawing from a Bernoulli distribution, such that, $q(z_k) = \text{Bernoulli}(z_k|\tilde{\pi}_k)$. We impose a symmetric prior for the latent winner indicators $[\boldsymbol{\xi}_n]_k \sim \text{Discrete}(1/U)$. A graphical illustration of the defined layer is depicted in Fig. 1b.

3.2 Training & Inference

To train the proposed model, we resort to maximization of the Evidence Lower Bound (ELBO). To facilitate efficiency in the resulting procedures, we adopt Stochastic Gradient Variational Bayes (SGVB) (Kingma and Welling, 2014). However, our model comprises latent variables that are not readily amenable to the reparameterization trick of SGVB, namely, the discrete latent variables z and $\boldsymbol{\xi}$, and the Beta-distributed stick variables $\boldsymbol{u}$. For the former, we utilize the continuous relaxation of Discrete (Bernoulli) random variables based on the Gumbel-Softmax trick (Maddison et al., 2016; Jang et al., 2017). For the latter, we employ the Kumaraswamy distribution-based reparameterization trick (Kumaraswamy, 1980) of the Beta distribution. These reparameterization tricks are only employed during training to ensure low-variance ELBO gradients.

At inference time, we *directly draw samples* from the trained posteriors of the *winner and network subpart selection latent variables* $\boldsymbol{\xi}$ and z, respectively; this introduces *stochasticity to the network activations and architecture*, respectively. Thus, differently from previous work in the field, the arising stochasticity of the resulting model stems from two different sampling processes. On the one hand, contrary to deterministic competition-based networks presented in Srivastava et al. (2013), we implement a data-driven random sampling procedure to determine the winning units, by sampling from $q(\boldsymbol{\xi})$. In addition, we infer which subparts of the model must be used or omitted, again based on sampling from the trained posteriors $q(z)$[1].

[1] In detail, inference is performed by sampling the $q(\boldsymbol{\xi})$ and $q(z)$ posteriors a total of $S = 5$ times, and averaging the corresponding $S = 5$ sets of output logits. We have found that considering an increased $S > 5$ does not yield any further improvement.



(a) (b)

Figure 1: (a) A graphical representation of our competition-based modeling approach. Rectangles denote LWTA blocks, and circles the competing units therein. The winner units are denoted with bold contours ($\xi = 1$). Bold edges denote retained connections ($z = 1$). (b) The convolutional LWTA variant. Competition takes place among feature maps. The winner feature map (denoted with bold contour) passes its output to the next layer, while the rest are zeroed out.

4 Experimental Evaluation

We evaluate the capacity of our proposed approach against various adversarial attacks and under different setups. To this end, we follow the experimental framework of Verma and Swami (2019). Specifically, we train networks which either employ the standard one-hot representation to encode the output variable of the classifier (predicted class) or the error-correcting output strategy proposed in Verma and Swami (2019); the latter is based on Hadamard coding. We try various activation functions for the classification layer of the networks, and use either a single network or an ensemble, as described in Verma and Swami (2019). The details of the considered networks are provided in Table 1. To obtain some comparative results, the considered networks are formulated under both our novel deep learning framework and in a conventional manner (i.e., with ReLU nonlinearities and SGD training).

Table 1: A summary of the considered networks. $\mathbf{I}_k$ denotes a $k \times k$ identity matrix, while $\boldsymbol{H}_k$ a k-length Hadamard code.

Model	Architecture	Code	Output Activation
Softmax	Standard	$\mathbf{I}_{10}$	softmax
Logistic	Standard	$\mathbf{I}_{10}$	logistic
LogisticEns10	Ensemble	$\mathbf{I}_{10}$	logistic
Tanh16	Standard	$\mathbf{H}_{16}$	tanh

4.1 Experimental Setup

We consider two popular benchmark datasets, namely MNIST (LeCun et al., 2010) and CIFAR-10 (Krizhevsky, 2009). The details of the considered network architectures are provided in the Supplementary.

Table 2: Classification accuracy on MNIST.

Model	Params	Benign	PGD	CW	BSA	Rand
Softmax (U=4)	327,380	.9613	.865	.970	.950	.187
Softmax (U=2)	327,380	.992	**.935**	.990	**1.0**	.961
Logistic (U=2)	327,380	.991	.901	.990	.990	.911
LogEns10 (U=2)	**205,190**	.993	.889	.980	.970	**1.0**
Madry	3,274,634	.9853	.925	.84	.520	.351
TanhEns16	401,168	**.9948**	.929	**1.0**	**1.0**	.988

To evaluate our modeling approach, all the considered networks, depicted in Table 1, are evaluated by splitting the architecture into: (i) LWTA blocks with $U = 2$ competing units on each hidden layer, and (ii) blocks with 4 competing units. The total number of units on each layer (split into blocks of $U = 2$ or 4 units) remains the same.

We initialize the posterior parameters of the Kumaraswamy distribution to $a = K,\ b = 1$, where K is the number of LWTA blocks, while using an uninformative Beta prior, Beta$(1, 1)$. For the Concrete relaxations, we set the temperatures of the priors and posteriors to 0.5 and 0.67 respectively, as suggested in Maddison et al. (2016); we faced no convergence issues with these selections.

Evaluation is performed by utilizing 4 different benchmark adversarial attacks: (i) Projected Gradient Descent (PGD), (ii) Carlini and Wagner (CW), (iii) Blind Spot Attack (BSA) (Zhang et al. (2019)), and (iv) a random attack (Rand) (Verma and Swami, 2019). For all attacks, we adopt exactly the same experimental evaluation of Verma and Swami (2019), both for transparency and comparability. The objective function of our model used for gradient-based adversary construction is the posterior expectation of its cate-

gorical cross-entropy. To compute this expectation we draw $S = 5$ samples from the posteriors over the latent variables $\boldsymbol{\xi}$ and z.

For the PGD attack, we use a common choice for the pixel-wise distortion $\epsilon = 0.3(0.031)$ for MNIST(CIFAR-10) with 500(200) attack iterations. For the CW attack, the learning rate was set to $1e$-3, utilizing 10 binary search steps. BSA performs the CW attack with a scaled version of the input, $\alpha\boldsymbol{x}$; we set $\alpha = 0.8$. In the "Random" attack, we construct random inputs by independently and uniformly generating pixels in $(0, 1)$; we report the fraction of which that yield class probability less than 0.9, in order to assess the confidence of the classifier as suggested in Verma and Swami (2019).

4.2 Experimental Results

MNIST. We train our model for a maximum of 100 epochs, using the same data augmentation as in Verma and Swami (2019). In Table 2, we depict the comparative results for each different network and adversarial attack. Therein, we also compare to the best-performing models in Verma and Swami (2019), namely the Madry model (Madry et al., 2017), and TanhEns16. As we observe, our modeling approach yields considerable improvements over Madry et al. (2017) and TanhEns16 (Verma and Swami, 2019) in three of the four considered attacks, while imposing lower memory footprint (less trainable parameters).

Table 3: Classification accuracy on CIFAR-10.

Model	Params	Benign	PGD	CW	BSA	Rand
Tanh16(U=4)	773,600	.510	.460	.550	.600	.368
Softmax(U=2)	**772,628**	.869	.814	**.860**	**.870**	.652
Tanh16(U=2)	773,600	.872	**.826**	.830	**.830**	.765
LogEns10(U=2)	1,197,998	.882	.806	.830	.800	**1.0**
Madry	45,901,914	.871	.470	.080	0	.981
TanhEns64	3,259,456	**.896**	.601	.760	.760	**1.0**

CIFAR-10. For the CIFAR-10 dataset, we follow an analogous procedure. The obtained comparative effectiveness of our approach is depicted in Table 3. Therein, we also compare to the best-performing models in Verma and Swami (2019), namely the Madry model (Madry et al., 2017), and TanhEns64. In this case, the differences in the computational burden are more evident, since now the trained networks are based on the VGG-like architecture. Specifically, our approach requires one-two orders of magnitude less parameters than the best performing alternatives in Verma and Swami (2019). At the same time, it yields substantially superior accuracy in the context of the considered attacks, while retaining a comparable performance for the benign case.

Note that the best performing model of Verma and Swami (2019), namely TanhEns64, adopts an architecture similar to Baseline Tanh, that we compare with, yet it fits an Ensemble of 4 different such networks. This results in imposing almost 4 times the computational footprint of our approach, both in terms of memory requirements and computational times. This justifies the small improvement in accuracy in the benign case; yet, it renders our approach much more preferable in terms of the offered computation/accuracy trade-off.

Table 4: Classification accuracy for all the attacks on the MNIST dataset (with $U = 2$, wherever applicable).

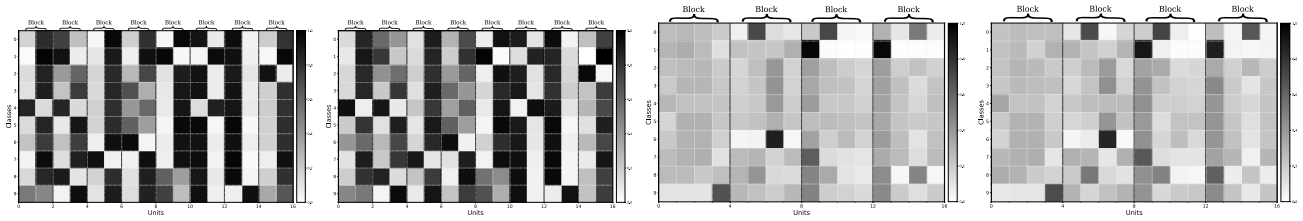
Model	Method	Benign	PGD	CW	BSA	RAND
Softmax	Baseline	.992	.082	.540	.180	.270
	LWTA$^{\max}$	.993	.302	.800	.390	.543
	LWTA$^{\max}$ & IBP	**.994**	.890	.920	.840	.361
	LWTA	.992	.900	**.990**	**1.0**	.810
	LWTA & IBP	.992	**.935**	**.990**	**1.0**	**.961**
Logistic	Baseline	.993	.093	.660	.210	.684
	LWTA$^{\max}$	.993	.388	.780	.420	.700
	LWTA$^{\max}$ & IBP	**.993**	.894	.960	.950	.230
	LWTA	.991	.856	**.990**	**.990**	**.982**
	LWTA & IBP	.991	**.901**	**.990**	**.990**	.981
LogisticEns10	Baseline	.993	.382	.880	.480	.905
	LWTA$^{\max}$	.993	.303	.920	.520	.900
	LWTA$^{\max}$ & IBP	**.994**	.860	.940	.910	.400
	LWTA	.992	.603	.900	.550	.809
	LWTA & IBP	.993	**.889**	**.980**	**.970**	**1.0**
Tanh16	Baseline	.993	.421	.790	.320	.673
	LWTA$^{\max}$	.992	.462	.910	.420	.573
	LWTA$^{\max}$ & IBP	**.994**	.898	.960	.940	.363
	LWTA	.992	.862	**.990**	**.980**	**.785**
	LWTA & IBP	.990	**.900**	**.990**	**.980**	**.785**

4.3 Further Insights

4.3.1 Ablation Study

Further, to assess the individual utility of LWTA-winner and architecture sampling in the context of our model, we scrutinize the obtained performance in both the benign case and the considered adversarial attacks. Specifically, we consider two different settings: (i) utilizing only the proposed LWTA mechanism in place of conventional ReLU activations; (ii) our full approach combining LWTA units with IBP-driven architecture sampling. The comparative results can be found in Tables 4 (MNIST) and 5 (CIFAR-10). Therein, "Baseline" corresponds to ReLU-based networks, as reported in Verma and Swami (2019). Our model's results are obtained with LWTA blocks of $U = 2$ competing units.

We begin with the MNIST dataset, where we examine two additional setups. Specifically, we employ the deterministic LWTA competition function as defined in (3) and examine its effect against adversarial attacks, both as a standalone modification (denoted as LWTA$^{\max}$) and also combined with the IBP-driven mechanism (i.e., sampling the z variables). The resulting classification performance is illustrated in the second and



(a) Benign: U=2 (b) PGD: U=2 (c) Benign: U=4 (d) PGD: U=4

Figure 2: Winning probabilities of competing units in LWTA blocks, on an intermediate layer of the Tanh16 network, for each class in the CIFAR-10 dataset. Figs. 2a and 2b depict the mean probability of activations per input class on a layer with 2 competing units, for benign and PGD test examples, respectively. Figs. 2c and 2d correspond to a network layer comprising 4 competing units. Black denotes very high winning probability, while white very low.

third row for each network in Table 4. One can observe that using deterministic LWTA activations, without stochastically sampling the winner, yields improvement, which increases with the incorporation of the IBP.

The fourth and fifth rows correspond to the *proposed* LWTA (stochastic) activations, without or with architecture sampling (via the z variables). The experimental results suggest that the stochastic nature of the proposed activation significantly contributes to the robustness of the model. It yields significant gains in all adversarial attacks compared to both the baseline, as well as to the deterministic LWTA adaptation. The performance improvement obtained by employing our proposed LWTA units can be as high as *two orders of magnitude*, in the case of the powerful PGD attack. Finally, stochastic architecture sampling via the IBP-induced mechanism further increases the robustness.

The corresponding experimental results for CIFAR-10 are provided in Table 5. Our approach yields a significant performance increase, which reaches up to *three orders of magnitude* (PGD attack, Logistic network).

Table 5: Classification accuracy for all the attacks on CIFAR-10 (with $U = 2$, wherever applicable).

Model	Method	Benign	PGD	CW	BSA	RAND
Softmax	Baseline	.864	.070	.080	.040	.404
	LWTA	.867	.804	.820	**.870**	**.701**
	LWTA & IBP	**.869**	**.814**	**.860**	**.870**	.652
Logistic	Baseline	**.865**	.006	.140	.100	.492
	LWTA	.830	.701	.720	.696	.690
	LWTA & IBP	.837	**.738**	**.800**	**.820**	**.726**
LogisticEns10	Baseline	.877	.100	.240	.140	.495
	LWTA	.879	.712	.750	.750	.803
	LWTA & IBP	**.882**	**.806**	**.830**	**.800**	**1.0**
Tanh16	Baseline	.866	.099	.080	.100	.700
	LWTA	.863	.720	.750	.750	**.800**
	LWTA & IBP	**.872**	**.826**	**.830**	**.830**	.765

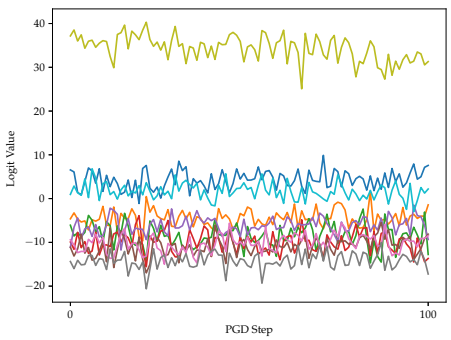
4.3.2 LWTA Behavior

Here, we scrutinize the competition patterns within the blocks of our model, in order to gain some further insights. First and foremost, we want to ensure that competition does not collapse to singular "always-winning" units. To this end, we choose a random intermediate layer of a Tanh16 network formulated under our modeling approach. We consider layers comprising 8 or 4 LWTA blocks of $U = 2$ and 4 competing units, respectively, and focus on the CIFAR-10 dataset. The probabilities of unit activations for each class are depicted in Fig. 2.

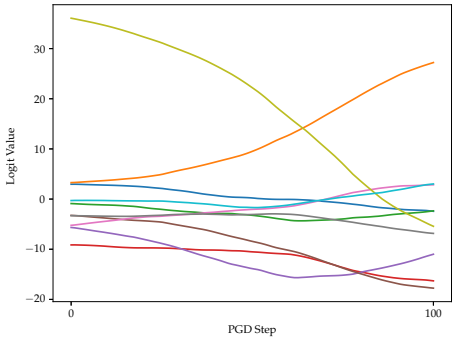
In the case of $U = 2$ competing units, we observe that the unit activation probabilities for each different setup (benign and PGD) are essentially the same (Figs. 2a and 2b). This suggests that the LWTA mechanism succeeds in encoding salient discriminative patterns in the data that are resilient to PGD attacks. Thus, we obtain networks capable of defending against adversarial attacks in a principled way.

On the other hand, in Figs. 2c and 2d we depict the corresponding probabilities when employing 4 competing units. In this case, the competition mechanism is uncertain about the winning unit in each block and for each class; average activation probability for each unit is around $\approx 25\%$. Moreover, there are several differences in the activations between the benign data and a PGD attack; these explain the significant drop in performance. This behavior potentially arises due to the relatively small structure of the network; from the 16 units of the original architecture, only 4 are active for each input. Thus, in this setting, LWTA fails to encode the necessary distinctive patterns of the data. Further results are provided in the Supplementary.

Finally, we investigate how the classifier output logit values change in the context of an adversarial (PGD) attack. We expect that this investigation may shed more light to the distinctive properties of the proposed



(a) Proposed model



(b) Conventional ReLU-based counterpart

Figure 3: Change of the output logit values under our proposed approach (3a), and a ReLU-based (3b) counterpart (PGD attack, CIFAR-10 dataset, Softmax network). The gradual, radical change of the logit values in the ReLU-based network indicates that conventional ReLU activations allow the attacker to successfully exploit gradient information. In contrast, under our proposed framework, the gradient-based attacker completely fails to do so.

framework that allow for the observed adversarial robustness. To this end, we consider the Softmax network trained on the CIFAR-10 dataset; we use a single, randomly selected example from the test set to base our attack on. In Fig. 3, we depict how the logit values pertaining to the ten modeled classes vary as we stage the PGD attack on the proposed model, as well its conventional, ReLU-based counterpart. As we observe, the conventional, ReLU-based network exhibits a gradual, yet radical change in the logit values as the PGD attack progresses; this suggests that finding an adversarial example constitutes an "easy" task for the attacker. In stark contrast, our approach exhibits varying, inconsistent, and steadily minor logit value changes as the PGD attack unfolds; this is especially true for the dominant (correct) class.

This implies that, under our approach, the gradient-based attacker is completely obstructed from successfully exploiting gradient information. This non-smooth appearance of the logit outputs seems to be due to the doubly stochastic nature of our approach, stemming from LWTA winner ($\boldsymbol{\xi}$) and network component (z) sampling. Due to this stochasticity, different sampled parts of a network may contribute to the logit outputs each time. This destroys, with high probability, the linearity with respect to the input. Similar results on different randomly selected examples are provided in the Supplementary.

4.3.3 Effect on the Decision Boundaries

We now turn our focus to the classifier decision boundaries obtained under our novel paradigm. Many recent works (Fawzi et al., 2017, 2018; Ortiz-Jimenez et al., 2020b) have shown that the decision boundaries of trained deep networks are usually highly sensitive to small perturbations of their training examples along some particular directions. These are exactly the directions that an adversary can exploit in order to stage a successful attack. In this context, Ortiz-Jimenez et al. (2020a) showed that deep networks are especially susceptible to directions of discriminative features; they tend to learn low margins in their area, while exhibiting high invariance to other directions, where the obtained margins are substantially larger.

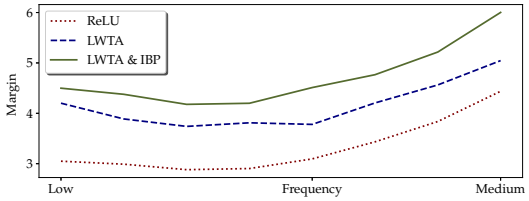


Figure 4: Mean margin over test examples of a LeNet-5 network trained on MNIST.

First, we train a LeNet-5 network on the MNIST dataset, similar to Ortiz-Jimenez et al. (2020a). We, then, extract the mean margin of the decision boundary, by using a subspace-constrained version of DeepFool (Moosavi-Dezfooli et al., 2016). This measures the margin of M samples on a sequence of subspaces; these are generated by using blocks from a 2-dimensional discrete cosine transform (2D-DCT) (Ahmed et al., 1974). Our results are depicted in Fig. 4. We train the network using our full model (dubbed "LWTA & IBP"), as well as a version keeping only the proposed LWTA activations. As we observe, our approach yields a significant increase of the margin in the low to medium

frequencies, exactly where the baseline is adversarially brittle. It is also notable that the use of the proposed IBP-driven architecture sampling mechanism is complementary to the proposed (stochastic) LWTA-type activation functions.

Subsequently, we repeat our experiments using a VGG-like architecture trained on CIFAR-10. We consider: (i) the standard, ReLU-based approach; (ii) our model using the proposed (stochastic) LWTA units but without IBP-driven architecture sampling; and (iii) the full model proposed in this work ("LWTA & IBP"). The corresponding illustrations are depicted in Fig. 5. As we observe, by using the proposed stochastic LWTA-type units, we obtain a very significant increase of the margin across the frequency spectrum. Once again, the IBP-driven architecture sampling process presents further improvements to the network, especially at the lower end of the spectrum, where it is most needed.

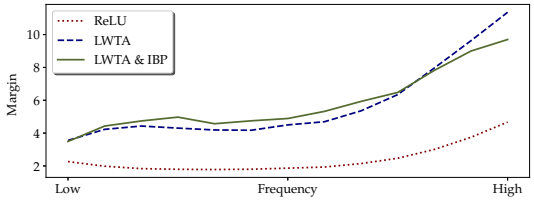


Figure 5: Mean margin over test examples on a VGG-like network trained on CIFAR-10.

Table 6: Performance of a Softmax network (Verma and Swami, 2019) under adversarial (FGSM-based) training for the MNIST dataset. Block size is $U = 2$, wherever applicable.

Method	Benign	PGD	CW	BSA	RAND
Baseline (ReLU)	**.992**	.082	.540	.180	.270
LWTA	**.992**	.900	**.990**	**1.0**	.810
LWTA & IBP	**.992**	**.935**	**.990**	**1.0**	**.961**
Baseline (ReLU) + FGSM	**.992**	.755	.820	.130	.835
LWTA + FGSM	.991	.925	**.990**	**1.0**	.960
LWTA & IBP + FGSM	.991	**.970**	**.990**	**1.0**	**.985**

4.4 Adversarial Training

Finally, it is important to analyze the behavior of our model under an adversarial training setup. Existing literature in the field has shown that conventionally-formulated networks lose some test-set performance when trained with adversarial examples. On the other hand, this sort of training renders them more robust when adversarially-attacked.

Therefore, it is interesting to examine how networks formulated under our model perform in the context of an adversarial training setup. To this end, and due to space limitations, we limit ourselves to FGSM-based (Goodfellow et al., 2015) adversarial training of the Softmax network. In the case of the MNIST dataset, FGSM is run with $\epsilon = 0.3$; for CIFAR-10, we set $\epsilon = 0.031$. On this basis, we repeat the experiments of Section 4.3.1, and assess model performance considering all the attacks therein, under the same configuration; only exception to this rule is the PGD attack, where we use 40 and 20 PGD steps to attack the networks trained on MNIST and CIFAR-10, respectively.

Table 7: Performance of a Softmax network (Verma and Swami, 2019) under adversarial (FGSM-based) training for the CIFAR-10 dataset. Block size is $U = 2$, wherever applicable.

Method	Benign	PGD	CW	BSA	RAND
Baseline (ReLU)	.864	.070	.080	.040	.404
LWTA	.867	.804	.820	**.870**	**.701**
LWTA & IBP	**.869**	**.814**	**.860**	**.870**	.652
Baseline (ReLU) + FGSM	.780	.200	.380	.340	.185
LWTA + FGSM	.820	.812	.810	**.870**	.440
LWTA & IBP + FGSM	**.830**	**.825**	**.870**	**.870**	**.790**

We depict the obtained results in Tables 6 and 7. Therein, the first three lines pertain to training by only making use of "clean" training data; the last three pertain to FGSM-based adversarial training, as described above. As we observe, in both datasets our (stochastic) LWTA activations completely outperform the ReLU-based baselines, while IBP-driven architecture sampling offers a further improvement in performance, across all attacks.

5 Conclusions

This work attacked adversarial robustness in deep learning. We introduced novel network design principles, founded upon stochastic units formulated under a sampling-based Local-Winner-Takes-All mechanism. We combined these with a network subpart omission mechanism driven from an IBP prior, which further enhances the stochasticity of the model. Our experimental evaluations have provided strong empirical evidence for the efficacy of our approach. Specifically, we yielded good accuracy improvements in various kinds of adversarial attacks, with considerably less trainable parameters. The obtained performance gains remained important under FGSM-based adversarial training.

Our future work targets novel methods for adversarial training deep networks formulated under our modeling approach. We specifically target scenarios that are not limited to the existing paradigm of gradient-based derivation of the adversarial training examples.

Acknowledgments

This work has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 872139, project aiD.

References

Ahmed, N., Natarajan, T., and Rao, K. R. (1974). Discrete cosine transfom. *IEEE Trans. Comput.*, 23(1):90–93.

Andersen, P., Gross, G. N., Lomo, T., and Sveen, O. (1969). Participation of inhibitory and excitatory interneurones in the control of hippocampal cortical output. In *UCLA forum in medical sciences*, volume 11, page 415.

Boloor, A., He, X., Gill, C., Vorobeychik, Y., and Zhang, X. (2019). Simple physical adversarial examples against end-to-end autonomous driving models. In *Proc. ICESS*, pages 1–7. IEEE.

Buckman, J., Roy, A., Raffel, C., and Goodfellow, I. J. (2018). Thermometer encoding: One hot way to resist adversarial examples. In *Proc. ICLR*.

Carpenter, G. A. and Grossberg, S. (1988). The art of adaptive pattern recognition by a self-organizing neural network. *Computer*, 21(3):77–88.

Chen, C., Seff, A., Kornhauser, A., and Xiao, J. (2015). Deepdriving: Learning affordance for direct perception in autonomous driving. In *Proc. ICCV*, pages 2722–2730.

Dhillon, G. S., Azizzadenesheli, K., Lipton, Z. C., Bernstein, J., Kossaifi, J., Khanna, A., and Anandkumar, A. (2018). Stochastic activation pruning for robust adversarial defense. *arXiv preprint arXiv:1803.01442*.

Douglas, R. J. and Martin, K. A. (2004). Neuronal circuits of the neocortex. *Annu. Rev. Neurosci.*, 27:419–451.

Fawzi, A., Moosavi-Dezfooli, S., and Frossard, P. (2017). The robustness of deep networks: A geometrical perspective. *IEEE Signal Processing Magazine*, 34(6):50–62.

Fawzi, A., Moosavi-Dezfooli, S., Frossard, P., and Soatto, S. (2018). Empirical study of the topology and geometry of deep networks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3762–3770.

Finlayson, S. G., Bowers, J. D., Ito, J., Zittrain, J. L., Beam, A. L., and Kohane, I. S. (2019). Adversarial attacks on medical machine learning. *Science*, 363(6433):1287–1289.

Ghahramani, Z. and Griffiths, T. L. (2006). Infinite latent feature models and the indian buffet process. In *Advances in neural information processing systems*, pages 475–482.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2015). Explaining and harnessing adversarial examples. In *Proc. ICLR*.

Grossberg, S. (1982). Contour enhancement, short term memory, and constancies in reverberating neural networks. In *Studies of mind and brain*, pages 332–378. Springer.

Guo, C., Rana, M., Cisse, M., and Van Der Maaten, L. (2017). Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117*.

Jalal, A., Ilyas, A., Daskalakis, C., and Dimakis, A. G. (2017). The robust manifold defense: Adversarial training using generative models. *arXiv preprint arXiv:1712.09196*.

Jang, E., Gu, S., and Poole, B. (2017). Categorical reparametrization with gumbel-softmax. In *Proc ICLR*.

Jiang, L., Ma, X., Chen, S., Bailey, J., and Jiang, Y.-G. (2019). Black-box adversarial attacks on video recognition models. In *Proc. ACM International Conference on Multimedia*, pages 864–872.

Kabilan, V. M., Morris, B., and Nguyen, A. (2018). Vectordefense: Vectorization as a defense to adversarial examples. *arXiv preprint arXiv:1804.08529*.

Kandel, E. R., Schwartz, J. H., Jessell, T. M., of Biochemistry, D., Jessell, M. B. T., Siegelbaum, S., and Hudspeth, A. (2000). *Principles of neural science*, volume 4. McGraw-hill New York.

Kingma, D. P. and Welling, M. (2014). Auto-encoding variational bayes. In Bengio, Y. and LeCun, Y., editors, *ICLR*.

Krizhevsky, A. (2009). Learning multiple layers of features from tiny images. Technical report.

Kumaraswamy, P. (1980). A generalized probability density function for double-bounded random processes. *Journal of Hydrology*, 46(1):79 – 88.

Kurakin, A., Goodfellow, I., and Bengio, S. (2016). Adversarial examples in the physical world. *arXiv preprint arXiv:1607.02533*.

Lansner, A. (2009). Associative memory models: from the cell-assembly theory to biophysically detailed cortex simulations. *Trends in neurosciences*, 32(3):178–186.

LeCun, Y., Cortes, C., and Burges, C. (2010). Mnist handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist*, 2.

Lee, D. D. and Seung, H. S. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791.

Maddison, C. J., Mnih, A., and Teh, Y. W. (2016). The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*.

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*.

Moosavi-Dezfooli, S.-M., Fawzi, A., and Frossard, P. (2016). Deepfool: A simple and accurate method to fool deep neural networks. pages 2574–2582.

Olshausen, B. A. and Field, D. J. (1996). Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609.

Ortiz-Jimenez, G., Modas, A., Moosavi-Dezfooli, S.-M., and Frossard, P. (2020a). Hold me tight! Influence of discriminative features on deep network boundaries. In *Advances in Neural Information Processing Systems 34*.

Ortiz-Jimenez, G., Modas, A., Moosavi-Dezfooli, S.-M., and Frossard, P. (2020b). Optimism in the face of adversity: Understanding and improving deep learning through adversarial robustness.

Panousis, K., Chatzis, S., and Theodoridis, S. (2019). Nonparametric Bayesian deep networks with local competition. In *Proc. ICML*, pages 4980–4988.

Papernot, N., McDaniel, P., Goodfellow, I., Jha, S., Celik, Z. B., and Swami, A. (2017). Practical black-box attacks against machine learning. In *Proc. ACM Asia conference on computer and communications security*, pages 506–519.

Prakash, A., Moran, N., Garber, S., DiLillo, A., and Storer, J. (2018). Deflecting adversarial attacks with pixel deflection. In *Proc. CVPR*, pages 8571–8580.

Shen, S., Jin, G., Gao, K., and Zhang, Y. (2017). Apegan: Adversarial perturbation elimination with gan. *arXiv preprint arXiv:1707.05474*.

Shrivastava, A., Pfister, T., Tuzel, O., Susskind, J., Wang, W., and Webb, R. (2017). Learning from simulated and unsupervised images through adversarial training. In *Proc. CVPR*, pages 2107–2116.

Song, Y., Kim, T., Nowozin, S., Ermon, S., and Kushman, N. (2017). Pixeldefend: Leveraging generative models to understand and defend against adversarial examples. *arXiv preprint arXiv:1710.10766*.

Srivastava, R. K., Masci, J., Kazerounian, S., Gomez, F., and Schmidhuber, J. (2013). Compete to compute. In *Advances in neural information processing systems*, pages 2310–2318.

Stefanis, C. (1969). Interneuronal mechanisms in the cortex. In *UCLA forum in medical sciences*, volume 11, page 497.

Teh, Y. W., Grür, D., and Ghahramani, Z. (2007). Stick-breaking construction for the indian buffet process. In *Proc. AISTATS*, pages 556–563.

Theodoridis, S. (2020). *Machine Learning: A Bayesian and Optimization Perspective*. Academic Press, 2nd edition.

Tramèr, F., Kurakin, A., Papernot, N., Goodfellow, I., Boneh, D., and McDaniel, P. (2017). Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*.

Verma, G. and Swami, A. (2019). Error correcting output codes improve probability estimation and adversarial robustness of deep neural networks. In *Advances in Neural Information Processing Systems*, pages 8643–8653.

Xie, C., Wang, J., Zhang, Z., Ren, Z., and Yuille, A. (2017). Mitigating adversarial effects through randomization. *arXiv preprint arXiv:1711.01991*.

Zhang, H., Chen, H., Song, Z., Boning, D., Dhillon, I. S., and Hsieh, C.-J. (2019). The limitations of adversarial training and the blind-spot attack. *arXiv preprint arXiv:1901.04684*.