Assured Deep Learning: Practical Defense Against Adversarial Attacks (Invited Paper)

Bita Darvish Rouhani, Mohammad Samragh, Mojan Javaheripi, Tara Javidi, and Farinaz Koushanfar
University of California San Diego
bita@ucsd.edu, msamragh@ucsd.edu, mojan@ucsd.edu, tjavidi@ucsd.edu, farinaz@ucsd.edu

ABSTRACT

Deep Learning (DL) models have been shown to be vulnerable to adversarial attacks. In light of the adversarial attacks, it is critical to reliably quantify the confidence of the prediction in a neural network to enable safe adoption of DL models in autonomous sensitive tasks (e.g., unmanned vehicles and drones). This article discusses recent research advances for unsupervised model assurance against the strongest adversarial attacks known to date and quantitatively compare their performance. Given the widespread usage of DL models, it is imperative to provide model assurance by carefully looking into the feature maps automatically learned within Dl models instead of looking back with regret when deep learning systems are compromised by adversaries.

Keywords: Adversarial Deep Learning, Unsupervised Model Assurance, Real-time Defense, Reconfigurable Computing.

1 INTRODUCTION

Recent advances in adversarial attacks have shown that deep learning models are not reliable in the presence of malicious entities. Adversarial samples are generated by adding imperceptible perturbations to legitimate input samples. Such perturbations are not visible to the human cognitive system but cause the deep learning model to misclassify the input image. Figure 1 shows an example such adversarial samples.

Figure 1: An example of original input data (left) and its corresponding adversarial sample (right). The added noise is visually hard to see but can mislead the victim DL model.

ICCAD '18, November 5–8, 2018, San Diego, CA, USA

ACM ISBN 978-1-4503-5950-4/18/11…$15.00
https://doi.org/10.1145/3240765.3274525

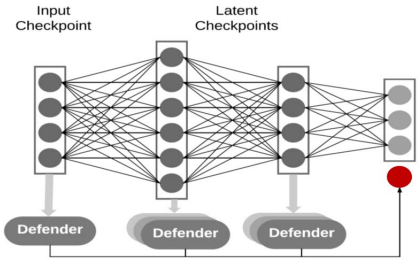

Figure 2: Overview of DeepFense countermeasure.

In response to various adversarial attack algorithms [1–3], a bulk of recent research has been focused on defensive countermeasures to detect/reject adversarial samples [4–9]. Figure 3 provides a high-level comparison of existing defense mechanisms in the context of deep learning. In particular, we compare the previous mechanisms in terms of the following characteristics:

- **Supervised/Unsupervised**: Supervised methods leverage adversarial samples generated by particular attacks to construct defensive mechanisms. Thereby, supervised methods are tailored for particular attack algorithms which makes them vulnerable to future attack methodologies. Unsupervised methods, on the contrary, do not use any adversarial samples for defense establishment. Adopting an unsupervised learning approach, in turn, ensures that the proposed defense mechanism can be well generalized to a wide class of adversarial attacks.
- **Robustness against black-box attacks**: Black-box attacks refer to scenarios where the adversary has access to the main (victim) DL model but is not aware of the deployed defense mechanism. A proposed countermeasure should be (at least) robust against black-box attacks.
- **Robustness against white-box attacks**: White-box attacks refer to scenarios where the adversary has access to both the DL model and the defense system. This is the strongest type of attack that is plausible in real-world.
- **Preserving the main DL model**: A number of defense mechanisms require modifying the structure of the main (victim) neural network or the training algorithms. Such modifications can reduce the accuracy of the victim model and add extra training/execution overhead.
- **Hardware acceleration**: Adversarial attacks mainly target real-time systems such as unmanned vehicles and drones; as such, it is critical to be able to execute the proposed defense in real-time while adhering to the hardware/power constraints of the underlying platform.

	Unsupervised Defense	Robust Against Black-box Attacks	Robust Against White-box Attacks	Preserving DL Architecture	Hardware Acceleration
DeepFense (ICCAD'18)	✓	✓	✓	✓	✓
MagNet (CCS'17)	✓	✓	x	✓	x
APE-GAN (ICLR'17)	x	✓	x	✓	x
LID (ICLR'18)	✓	✓	✓	✓	x
Ensemble Training (arXiv'17)	x	x	x	✓	x
Distillation (S&P'16)	✓	x	x	x	x
Denoising Auto-encoder (arXiv'14)	x	x	x	x	x

Figure 3: High-level comparison of existing defenses against adversarial attacks.

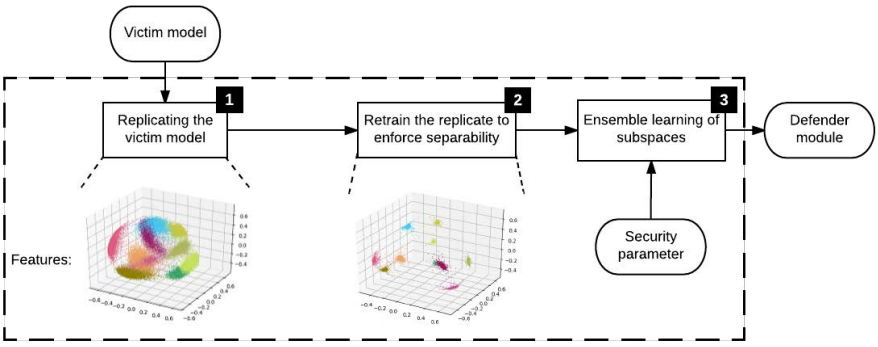

Figure 4: Block diagram of the training procedure for devising parallel checkpointing modules. Each defender module is built by minimizing the disentanglement of intermediate data features in a Euclidean space at a particular checkpoint location.

This article highlights DeepFense, the first hardware-accelerated unsupervised defense against adversarial DL attacks [10, 11]. DeepFense learns the Probability Density Function (PDF) of legitimate samples in different layers of the main neural network. For each incoming sample, DeepFense checkpoints the corresponding activation maps in each layer to detect abnormal samples. Extensive evaluations on various benchmarks in both white-box and black-box attacks corroborate the superiority of DeepFense in comparison with prior work (see Section 4).

Figure 2 shows a schematic depiction of DeepFense' proposed countermeasure. A set of defender modules are trained to checkpoint both input and latent (hidden) activation maps. Suspicious samples are classified as a "do not know" class (the red node in Figure 2). The underlying topology and/or weights of the main model is not altered. To deceive the defensive system, an adversary is required to fool all defenders simultaneously. In the following, we briefly discuss the training process of latent (Section 2) and input (Section 3) defenders.

2 TRAINING LATENT DEFENDERS

The goal of each defender (checkpointing) module is to learn the PDF of the explored sub-spaces in a particular intermediate DL feature map. DeepFense considers a Gaussian Mixture Model (GMM) as the prior probability to characterize the data distribution at each checkpoint location. It is worth noting that our proposed approach is rather generic and is not restricted to the GMM distribution.

To effectively characterize the explored sub-space as a GMM distribution, one is required to minimize the entanglement between each two Gaussian distribution (corresponding to every two different classes) while decreasing the inner-class diversity. Figure 4 illustrates the high-level block diagram of the training procedure for devising a latent checkpointing module. Training a latent defender is a one-time offline process and is performed in three steps.

1 Replicating the victim neural network and all its feature maps. An L_2 normalization layer is inserted in the desired checkpoint location. The normalization layer maps the latent feature variables, $\phi(x)$, into the Euclidean space such that the acquired data embeddings live in a d-dimensional hyper-sphere, i.e., $\|\phi(x)\|_2 = 1$. This normalization is crucial as it partially removes the effect of overfitting to particular data samples that are highly correlated with the underlying DL parameters.

2 Fine-tuning the replicated network to enforce disentanglement of data features (at a particular checkpoint location). To do so, DeepFense optimizes the defender module by incorporating the following loss function with the conventional cross entropy loss:

$$\gamma \left[\underbrace{\|C^{y^*} - \phi(x)\|_2^2}_{loss_1} - \underbrace{\Sigma_{i \neq y^*} \|C^i - \phi(x)\|_2^2}_{loss_2} + \underbrace{\Sigma_i (\|C^i\|_2 - 1)^2}_{loss_3} \right]. \quad (1)$$

Here, γ is a trade-off parameter that specifies the contribution of the additive loss term, $\phi(x)$ is the corresponding feature vector

of input sample x at the checkpoint location, y^* is the ground-truth label, and C^i denotes the center of all data abstractions ($\phi(x)$) corresponding to class i. The center values C^i and intermediate feature vectors $\phi(x)$ are trainable variables that are learned by fine-tuning the defender module.

3 Learning the statistics of the undelying GMM distribution. Once the defender model is fine-tuned in step 2, the mean and covariance of each Gaussian distribution in the GMM are obtained by querying the defender with legitimate samples. DeepFense profiles the percentage of benign samples lying within different L_2 radiuses of the learned GMM centers. DeepFense takes a user-defined security parameter ($SP \in [0-100]$) to break down the underlying space into the benign and adversarial sub-spaces. The percentile profiling is particularly used to map the selected security parameter into an L_2 threshold which is later leveraged to differentiate legitimate and adversarial samples in the test phase.

3 TRAINING INPUT DEFENDERS

DeepFense leverages dictionary learning and sparse signal recovery techniques to measure the Peak to Signal Noise Ratio (PSNR) of each incoming sample and automatically filter out atypical samples in the input space. Figure 5 illustrates the high-level block diagram of an input defender module. As shown, devising an input checkpoint model is performed in two main steps: (i) dictionary learning, and (ii) characterizing the typical PSNR per class after sparse recovery.

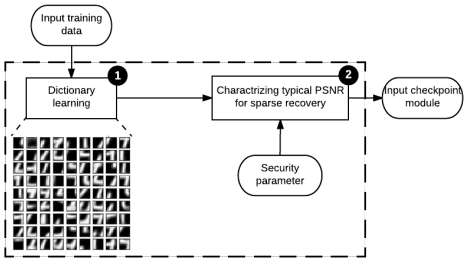

Figure 5: The input defender is devised based on robust dictionary learning techniques to automatically filter out noisy samples that highly deviate from the typical PSNR.

1 Dictionary learning. DeepFense learns a separate dictionary for each class of data by solving:

$$\underset{D^i}{argmin}\ \frac{1}{2}\|Z^i - D^iV^i\|_2^2\ +\ \beta\|V^i\|_1\ s.t.\ \|D_k^i\| = 1,\ 0 \le k \le k_{max}. \tag{2}$$

Here, Z^i is a matrix whose columns are pixels extracted from different regions of input images belonging to category i. For instance, if we consider 8×8 patches of pixels, each column of Z^i would be a vector of 64 elements. The goal of dictionary learning is to find matrix D^i that best represents the distribution of pixel patches from images belonging to class i. We denote the number of columns in D^i by k_{max}. For a certain D^i, the image patches Z^i are represented with a sparse matrix V^i, and D^iV^i is the reconstructed patches. DeepFense leverages Least Angle Regression (LAR) method to solve the Lasso problem defined in Eq. (2).

For an incoming sample, during the execution phase, the input defender module takes the output of the victim DL model (e.g., predicted class i) and uses Orthogonal Matching Pursuit (OMP) routine [12] to sparsely reconstruct the input data with the corresponding dictionary D^i. The dictionary matrix D^i contains a set of samples that commonly appear in the training data belonging to class i; As such, the input sample classified as class i should be well-reconstructed as D^iV^* with a high PSNR value, where V^* is the optimal solution obtained by the OMP routine. During the execution phase, all of the non-overlapping patches within the image are denoised by the dictionary to form the reconstructed image.

2 Characterizing typical PSNR in each category. DeepFense profiles the PSNR of legitimate samples within each class and find a threshold that covers all legitimate training samples. If an incoming sample has a PSNR lower than the threshold (i.e., high perturbation after reconstruction by the corresponding dictionary), it will be regarded as a malicious data point. In particular, PSNR is defined as:

$$PSNR = 20log_{10}(MAX_I) - 10log_{10}(MSE), \tag{3}$$

where the mean square error (MSE) is defined as the L_2 difference of the input image and the reconstructed image based on the corresponding dictionary. The MAX_I term is the maximum possible pixel value of the image (usually equivalent to 255).

4 EVALUATION

DeepFense has a superior performance compared to prior works in both white-box and black-box attack scenarios.

Resiliency against black-box attacks: Table 1 summarizes the Area Under Curve (AUC) score obtained by four different defense methodologies in a black-box setting for CIFAR10 dataset. As illustrated, DeepFense achieves significantly higher AUC score in the face of different attacks. This is particularly because DeepFense framework learns the PDF of legitimate samples within different layers of a neural network as opposed to just looking into statistical artifacts of adversarial samples. We emphasize that DeepFense defense methodology is unsupervised meaning that no adversarial samples are used to establish the defenders. For more details please refer to the original paper [10].

Table 1: Summary of the AUC score obtained by DeepFense and prior art defenses [6, 13, 14] against three attack algorithms: Fast Gradient Sign (FGS) [3], Basic Iterative Method (BIM) [15], and Carlini&WagnerL2 [2].

CIFAR10 Data	DeepFense [10]	LID [6]	BU [13]	KD [14]
FGS	0.921	0.824	0.705	0.649
BIM	0.944	0.691	0.817	0.823
Carlini&WagnerL2	0.954	0.912	0.913	0.933

Resiliency against white-box attacks: Table 2 details DeepFense performance against the state-of-the-art adaptive white-box attack proposed in [16]. As shown, while the prior works cannot withstand this type of adaptive attacks, DeepFense obtains comparably higher defense success rate.

5 CONCLUSION

This article discusses recent advances in characterizing and thwarting adversarial samples generated to mislead deep learning models.

Table 2: Comparison of DeepFense with prior art in the white-box attack scenario.

	DeepFense Methodology [10]					Prior-Art Defenses		
						Magnet [4]	Efficient Defenses [17]	APE-GAN [5]
Number of Defenders	N=1	N=2	N=4	N=8	N=16	N=16	-	-
Defense Success	43%	53%	64%	65%	66%	1%	0%	0%
Normalized Distortion	1.04	1.11	1.12	1.31	1.38	1.37	1.30	1.06
False Positive Rate	2.9%	4.4%	6.1%	7.8%	8.4%	-	-	-

We particularly study DeepFense framework which incorporates parallel checkpointing modules into the execution of neural networks. Each checkpointing module characterizes the PDF distribution of legitimate samples and raises alarm flags for the samples that deviate from the learned PDF. Experiments show the effectiveness of DeepFense in both white-box and black-box attack scenarios.

REFERENCES

[1] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossard, "Deepfool: a simple and accurate method to fool deep neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[2] N. Carlini and D. Wagner, "Towards evaluating the robustness of neural networks," in *Security and Privacy (SP), 2017 IEEE Symposium on*. IEEE, 2017.

[3] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013.

[4] D. Meng and H. Chen, "Magnet: a two-pronged defense against adversarial examples," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2017.

[5] S. Shen, G. Jin, K. Gao, and Y. Zhang, "Ape-gan: Adversarial perturbation elimination with gan," *ICLR Submission, available on OpenReview*, 2017.

[6] X. Ma, B. Li, Y. Wang, S. M. Erfani, S. Wijewickrema, M. E. Houle, G. Schoenebeck, D. Song, and J. Bailey, "Characterizing adversarial subspaces using local intrinsic dimensionality," *arXiv preprint arXiv:1801.02613*, 2018.

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

[8] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," 2016.

[9] S. Gu and L. Rigazio, "Towards deep neural network architectures robust to adversarial examples," *arXiv preprint arXiv:1412.5068*, 2014.

[10] B. Rouhani, M. Samragh, M. Javaheripi, T. Javidi, and F. Koushanfar, "Deepfense: Online accelerated defense against adversarial deep learning," 2018.

[11] B. Rouhani, M. Samragh, T. Javidi, and F. Koushanfar, "Safe machine learning and defeating adversarial attacks," 2018.

[12] J. Tropp, A. C. Gilbert *et al.*, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Transactions on Information Theory*, vol. 53, no. 12, pp. 4655–4666, 2007.

[13] K. Grosse, P. Manoharan, N. Papernot, M. Backes, and P. McDaniel, "On the (statistical) detection of adversarial examples," *arXiv preprint arXiv:1702.06280*, 2017.

[14] R. Feinman, R. R. Curtin, S. Shintre, and A. B. Gardner, "Detecting adversarial samples from artifacts," *arXiv preprint arXiv:1703.00410*, 2017.

[15] A. Kurakin, I. Goodfellow, and S. Bengio, "Adversarial examples in the physical world," *arXiv preprint arXiv:1607.02533*, 2016.

[16] N. Carlini and D. Wagner, "Magnet and" efficient defenses against adversarial attacks" are not robust to adversarial examples," *arXiv preprint arXiv:1711.08478*, 2017.

[17] V. Zantedeschi, M.-I. Nicolae, and A. Rawat, "Efficient defenses against adversarial attacks," in *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*. ACM, 2017.