SmartBox: Benchmarking Adversarial Detection and Mitigation Algorithms for Face Recognition

Akhil Goel, Anirudh Singh, Akshay Agarwal, Mayank Vatsa, and Richa Singh
IIIT-Delhi, India
Email: {akhil15126, anirudh15015, akshaya, mayank, rsingh}@iiitd.ac.in

Abstract

Deep learning models are widely used for various purposes such as face recognition and speech recognition. However, researchers have shown that these models are vulnerable to adversarial attacks. These attacks compute perturbations to generate images that decrease the performance of deep learning models. In this research, we have developed a toolbox, termed as SmartBox, for benchmarking the performance of adversarial attack detection and mitigation algorithms against face recognition. SmartBox is a python based toolbox which provides an open source implementation of adversarial detection and mitigation algorithms. In this research, Extended Yale Face Database B has been used for generating adversarial examples using various attack algorithms such as DeepFool, Gradient methods, Elastic-Net, and L_2 attack. SmartBox provides a platform to evaluate newer attacks, detection models, and mitigation approaches on a common face recognition benchmark. To assist the research community, the code of SmartBox is made available[1].

1. Introduction

Deep learning models have achieved state-of-the-art performance in various computer vision related tasks such as object detection and face recognition [18, 24]. However, recent studies suggest that small imperceptible perturbations can act as adversaries for these models and lead to incorrect predictions. As shown in Figure 1, imperceptible adversarial noise can be added in the original image to create perturbed images such that for humans they are exactly the same but the algorithms provide different prediction outputs compared to the original image. Majority of recently proposed face recognition algorithms are based on deep learning and we have observed that existing adversarial attacks may impact face recognition algorithms.

Existing research in adversarial perturbation can be classified into three levels: (i) attacks, (ii) attack detection, and (iii) mitigation. While some researchers are identifying the vulnerabilities of deep learning algorithms and creating newer kinds of attacks, simultaneously efforts are ongoing towards detecting these attacks and mitigating their effect. Table 1 summarizes the recent developments in all three areas.

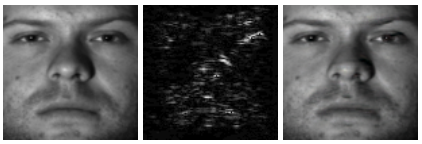

Figure 1: Original image (left), Added imperceptible noise zoomed 10 times (middle), Perturbed image (right)

Perturbations to fool the classification model can be performed either at the camera level or the processing level. Presentation attacks on face recognition system are performed at camera level, while adversarial perturbations operate at processing level. Recently, adversarial learning to fool deep learning based algorithms have gained significant attention [2]. Sharif et al. [30] have proposed a camera level perturbation algorithm by producing a pair of glasses which when worn by someone can confuse the target model and result in impersonation. Further, they also created a pair of eyeglasses using adversarial generative nets [31] that lead to misclassification.

Attack Generation Algorithms aim to perturb the input image in such a way that either the resultant image gets classified into a target class (targeted attack) or gets misclassified to any other class except the original class (untargeted attack). Attack algorithms can further be classified on the basis of amount of information they need from the trained network to work. While whitebox attacks such as Elastic-Net (EAD) [6], DeepFool [28], L_2 [5], Fast Gradient Sign Method (FGSM) [15], Projective Gradient Descent (PGD) [26], and MI-FGSM [10] have complete access and information about the trained network, blackbox attacks such as one pixel attack [32] and universal perturbations [27]

[1] http://iab-rubric.org/resources/SmartBox.html

Table 1: Summary of adversarial examples generation, detection, and mitigation algorithms implemented in the SmartBox

Type	Name	Author	Algorithm
Generation	DeepFool	Dezfooli et al. [28]	Calculates orthogonal distance from nearest separating hyperplane.
	EAD	Chen et al. [6]	Computes perturbations by minimising Elastic Net Loss.
	FGSM	Goodfellow at al. [15]	Computes gradient of the loss function w.r.t. the image vector.
	L_2	Carlini and Wagner [5]	Computes perturbations that have low distortions in L_2 metric.
Detection	Adaptive Noise Reduction	Liang et al. [23]	Applies scalar quantization and mean filter to the images.
	Artifact Learning	Feinman et al. & Gong et al. [11, 14]	Prediction based on the features learned by the network.
	Conv. Filter	Li and Li [22]	Features of convolution layers + cascaded classifier.
	PCA	Bhagoji et al. [3]	Applies PCA on input images and feeds them in a Linear SVM.
Mitigation	Adversarial Training	Szegedy et al. [33]	Trains a new model on original and adversarial training images.
	Denoising AutoEncoder	Creswell and Bharath [8]	Reconstructs original images from perturbed images.
	Randomization	Xie et al. [35]	Upsamples, downsamples and pads the input image.
	Gaussian Blur	**Proposed**	**Applies Gaussian Blur on input images.**

have no information about the trained Deep Neural Network (DNN).

Attack Detection Algorithms focus on the detection of forged examples presented to the system. The detected images could be either discarded or can be passed on to the mitigation algorithm for correct labeling. Carlini and Wagner [4] have shown the limitations of current detection algorithms. Gong et al. [14] and Hendrik et al. [19] learn a second binary classifier on internal learned features corresponding to the original training and adversarial training data. Li and Li [22], Hendrycks and Gimpel [20], and Agarwal et al. [1] applied Principal Component Analysis (PCA) on internal learned features and image pixels respectively to detect adversaries. Goswami et al. [16] propose a method that uses a classifier trained on the differences of the outputs of the intermediate layers corresponding to original and adversarial images and use it for adversarial detection. Other detection methods use statistical tests to detect adversarial images. One such method proposed by Grosse et al. [17] uses Maximum Mean Discrepancy (MMD) test that statistically differentiates adversarial images from the original images. SafetyNet, detection method proposed by Lu et al. [25], employs a detector along with the original classifier to detect adversarial images based on the information from the later layers of the network.

Attack Mitigation Algorithms: Since the realization of the severity of the problems introduced by adversarial attacks, many mitigation algorithms have been devised. Mitigation methods can be based on either modifying the input image or by changing the network. Modification of the input image can be seen as image enhancement while modification of the system can be seen as increasing the robustness towards adversaries. The mitigation methods by Das et al. [9] and Wang et al. [34] modify the input data before feeding it to the neural network. This approach removes the adversarial perturbation from the image and improves the performance of the model. Other methods such as Papernot et al. [29], Gao et al. [12], Cisse et al. [7] and Goswami et al. [16] modify the network to make it robust against adversarial attacks.

Since this is a fairly recent area, it currently lacks standardization in terms of evaluation and it is generally observed that they lack a common benchmarking approach. It is our hypothesis that the availability of a unified platform to benchmark the results can promote the research activities related to this important problem. Therefore, in this research, we have developed a Python toolbox, termed as SmartBox, to fill this gap. It contains a variety of algorithms to attack, detect, and mitigate the effects of adversarial perturbations on images. The tool also provides the functionalities to provide details analysis with respect to face identification and verification. It can analyze the results in terms of Rank 1 identification accuracy, verification accuracy, Receiver Operating Characteristic (ROC) curve, and Cumulative Match Characteristic (CMC) curve. It also has the flexibility to add new algorithms for analyzing the impact on face recognition. To the best of our knowledge, this is the first work and toolbox of its kind in adversarial learning towards face recognition.

2. SmartBox: Structure and Functionalities

SmartBox consists of all three important modules for adversarial perturbation: attack generation, detection, and mitigation. Table 1 summarizes the algorithms used for adversarial generation, detection, and mitigation in the proposed toolbox. Each of these modules is explained in detail below along with other functionalities available in SmartBox.

2.1. Attack Generation Module

SmartBox includes implementation of existing targeted and un-targeted attack algorithms to generate adversarial images from a set of input images. Their purpose is to decrease the performance of a DNN classification model. Currently, SmartBox includes the implementations of the following attacks:

EAD [6]: EAD attack is based on elastic net regularization, which uses a linear combination of L_1 and L_2 penalty functions. Let (x_{orig}, t_{orig}) and (x_{adv}, t_{adv}) be the original image-label pair and adversarial image-label pair, respectively. To craft the adversarial examples the formula used by this attack method is:

$$\min_{x_{adv}} \alpha \cdot f(x_{adv}, t_{adv}) + \beta \cdot ||x_{adv} - x_{orig}||_1 + ||x_{adv} - x_{orig}||_2^2 \quad (1)$$
$$subject\ to\ x_{adv}\ \in\ [0,1]^d$$

where, α and β are the regularization parameters. $f(x_{adv}, t)$ is the loss function which differs for targeted and un-targeted attacks. The attack method is able to generate an adversarial image that is classified to the target class t_{adv} while minimizing the elastic net loss.

DeepFool [28]: DeepFool is an un-targeted attack technique. It iteratively calculates the minimum noise to fool the system. The minimum noise is defined as the orthogonal distance from the nearest separating decision hyperplane, assuming the underlying classifier to be linear.

C&W L_2 [5]: This attack is one of the strongest adversarial attacks currently available. It can be used in targeted or un-targeted forms. It tries to find adversarial samples that have low L_2 distortion metric by considering the logits layer of the model. This method attempts to minimize following equation to generate adversarial images.

$$minimize||\frac{1}{2}(\tanh(w)+1) - x||_2^2 + c \cdot f(\frac{1}{2}(\tanh(w)+1)) \quad (2)$$

where, $f(x) = max(max\{Z(x)_i : i \neq t\} - Z(x)_t, -\kappa)$ Z is the logits layer, t is the target class, κ is the parameter that controls the confidence of mis-classification and w is the adversary that minimizes the above expression.

FGSM [15]: It computes the gradient of the loss function of the model concerning the image vector to get the direction of pixel change. FGSM perturbations can be computed by minimizing either the L_1, L_2 or L_∞ norm.

2.2. Detection Module

This module has implementations of various adversarial perturbation detection methods which aims to detect the fake images before passing them to the network. Following are the detection methods included in SmartBox:

Detection using Convolution Filter Statistics: In this method as proposed by Li and Li [22], we extract the statistical features which include normalized PCA coefficients and minimal and maximal values from the output of the convolution layers and feed it to a cascaded classifier for adversarial detection. The cascaded classifier consists of a sequence of base classifiers which are then used to detect adversarial images from a set of input images.

PCA based detection: This method, as proposed by Bhagoji et al. [3], first computes the projection matrix by applying principal component analysis on the training data. The projection matrix is then used to project images to a linear space. The features obtained through this are used to train an SVM classifier to detect adversarial images from a set of input images.

Artifacts Learning: This method follows an approach similar to the one followed by Feinman et al. [11] and Gong et al. [14]. It uses the features learned by the DNN model to distinguish between original and adversarial images. More specifically, in this method, original and adversarial training images are passed through the trained network. After that, features corresponding to desired layers are fetched and a new binary classifier (a binary neural network) is trained on these features.

Adaptive Noise Reduction: Unlike the previous method, this technique, as proposed by Liang et al. [23], is independent of the nature of noise added. Based on the entropy of the image (the amount of information), image modification techniques such as scalar quantization and smoothing spatial filters are applied. An image is classified as an adversary if modification changes its classification.

2.3. Mitigation

Mitigation module in the SmartBox has methods that attempt to mitigate the embedded perturbations in the provided image. Following are the implemented methods:

Adversarial Training: In adversarial training [33], a new model is trained using the original dataset and adversarial examples with their correct labels. The intuition behind the technique is that the new model obtained is robust to adversarial images.

Randomization: This approach is similar to the method described by Xie et al. [35]. The images are first up-sampled and then down-sampled to a random size through interpolation methods. The down-sampled images are padded with zeros. The images are padded in such a way that the size of a padded image is same as the original size of the image. From the images obtained after padding, an image is randomly selected and passed to the model. The interpolation operation in this approach "removes" the adversarial noise.

Gaussian Blurring: Gaussian Blur is an effective image processing technique that blurs and reduces intricate details of an input image by convolving the image with a kernel whose weights are derived from a Gaussian distribution, i.e.,

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2} \quad (3)$$

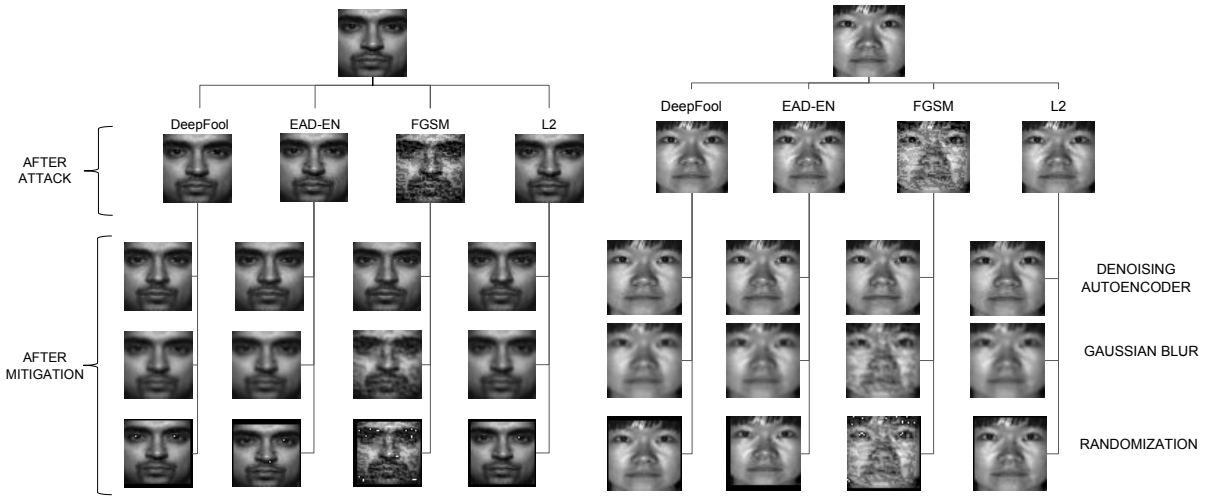

Figure 2: Illustrating the effects of adversarial generation and mitigation algorithms on two face examples (a) and (b).

Here, μ is mean, and σ is the standard deviation of the distribution. The intuition behind the proposed Gaussian blurring as a mitigation technique lies behind the fact that convolving with a kernel with reasonable deviation along the axes may result in diminishing the effect of the added perturbations.

Denoising Autoencoder: This method is similar to the one proposed by Creswell and Bharath [8] which learns the weights of a denoising autoencoder through adversarial training examples. While training, the autoencoder learns to recreate the original images from the perturbed images. In other words, an input image is "denoised" using the training autoencoder.

2.4. Other Functionalities

Apart from providing implementations of existing attacks, detection and mitigation algorithms, SmartBox also provides the following features: (i) A sample model for experiments, (ii) Provision to add a custom facial database for experiments, (iii) Functions to visualize the results using ROC and CMC curves, (iv) Support to pre-train models for adversarial training, artifact learning, and denoising autoencoders, (v) provision to choose model training/attack/detection/mitigation algorithm parameters from command-line, and (vi) Provision to select evaluation metrics to assess attack generation/detection/mitigation performance from the command line.

3. Experiments and Results

Experiments were conducted on the Extended Yale face Database B [13] [21]. The dataset consists of cropped images of 38 individuals under different illumination conditions. The dataset is randomly divided into training (80%), validation (10%), and testing (10%). Each testing probe is randomly allotted a target (for targeted attacks) and is randomly classified as genuine or an imposter (for verification). Experiments were carried out in a network with following specifications : 2 Conv layers each having 16 filters and 3x3 kernel - maxpool layer with strides set as 2 - Conv layer with 32 filters and 3x3 kernel - maxpool layer with strides set as 2 - Conv layer with 32 filters and 3x3 kernel - maxpool layer with strides set as 2 - Dense layer with 100 units - Dropout with 50% drop rate - Dense layer with 100 units - Dense layer with 1024 units - Logits layer. The model was trained using Stochastic Gradient Descent.

3.1. Attack Generation

On the extended Yale B database [13] [21] attack Generation results are summarized in Table 2. Targeted L_2, EAD-EN and EAD-L_1 are the most successful in fooling the model by bringing both Rank 1 identification and verification accuracy to 0%. Un-targeted DeepFool decreases Rank 1 identification by more than 90% and verification accuracy by around 45%.

3.2. Attack Detection

Attack detection results on several different perturbation algorithms are summarized in Table 3. Adaptive Noise Reduction performs well with network loss based attacks but fails with gradient-based attacks. In attacks such as L_2 and EAD, it scores the maximum recall value. Artifact learning consistently performs well on all attacks achieving highest detection accuracy in all the cases. Conv Filter and PCA, on the other hand, perform well with gradient-based attacks but their performance with network loss based attacks is poor.

3.3. Attack Mitigation

Attack mitigation results of proposed and existing algorithms are summarized in Table 2 and Figure 2. From

Table 2: Effect of attack generation and mitigation algorithms on face verification and identification accuracies.

Attacks	Mitigation Algorithm	Verification			Identification		
		Before Attack	After Attack	After Mitigation	Before Attack	After Attack	After Mitigation
DeepFool	Adversarial Training	96.96%	50%	**98.48%**	95.07%	3.4%	**97.34%**
	Gaussian Blur			93.56%			86.74%
	Randomization			73.48%			45.07%
	Denoising AutoEncoder			93.18%			87.50%
EAD-L1	Adversarial Training		0%	**98.86%**		0%	**97.72%**
	Gaussian Blur			92.80%			85.22%
	Randomization			68.56%			38.25%
	Denoising AutoEncoder			93.93%			85.98%
EAD-EN	Adversarial Training		0%	**99.24%**		0%	**99.24%**
	Gaussian Blur			92.04%			83.33%
	Randomization			68.56%			36.36%
	Denoising AutoEncoder			93.18%			85.60%
FGSM	Adversarial Training		67.04%	79.92%		29.92%	57.19%
	Gaussian Blur			67.42%			32.57%
	Randomization			57.95%			18.18%
	Denoising AutoEncoder			**88.25%**			**75.37%**
L2	Adversarial Training		0%	**99.62%**		0%	**99.62%**
	Gaussian Blur			92.80%			85.60%
	Randomization			70.07%			39.39%
	Denoising AutoEncoder			93.18%			85.98%

Table 3: Detection performance of adversarial detection algorithms available in SmartBox, in terms of precision, recall and accuracy (%).

Attacks	Algorithms	Precision	Recall	Accuracy
DeepFool	Adapt. Noise Reduction	77.48%	93.22%	83.06%
	Artifact Learning	**84.33%**	**95.83%**	**89.01%**
	Conv Filter	52.13%	64.77%	52.65%
	PCA	58.84%	61.74%	59.28%
EAD-L_1	Adapt. Noise Reduction	78.41%	**98.40%**	85.65%
	Artifact Learning	**89.27%**	97.72%	**92.99%**
	Conv Filter	56.77%	74.62%	58.90%
	PCA	61.06%	56.43%	60.22%
EAD-EN	Adapt. Noise Reduction	78.48%	**98.80%**	85.85%
	Artifact Learning	**86.44%**	96.59%	**90.71%**
	Conv Filter	53.79%	64.39%	54.54%
	PCA	56.86%	54.92%	56.62%
FGSM	Adapt. Noise Reduction	55.17%	46.24%	54.33%
	Artifact Learning	89.41%	**86.36%**	**88.06%**
	Conv Filter	**95.54%**	73.10%	84.84%
	PCA	95.42%	63.25%	80.11%
L_2	Adapt. Noise Reduction	78.54%	**99.20%**	86.05%
	Artifact Learning	**85.04%**	96.96%	**89.96%**
	Conv Filter	54.19%	63.63%	54.92%
	PCA	56.55%	57.19%	56.62%

the table, it is evident that adversarial training consistently performs better than the remaining algorithms, simple reason being that it has been exposed to both original and adversarial training data. Gaussian Blur effectively mitigates the perturbations produced by L_2, EAD and DeepFool, and fails on gradient-based attacks. Denoising autoencoder is effectively able to mitigate the perturbations produced by all tested attack generation algorithms. Randomization algorithm, on the other hand, performs consistently poor for all attack algorithms. This analysis is supported by the Receiver Operating Characteristic curve and Cumulative Match Characteristic curve in Figures 3 and 4.

4. Conclusion

Adversarial examples have been demonstrated to be successful in fooling various deep learning based object and face recognition algorithms. In this research, for the first time, we have developed a Python toolbox namely SmartBox consisting of multiple adversarial examples generation, detection, and mitigation algorithms. The SmartBox toolbox can be used as the benchmark to analyze the effect of an adversary on face recognition systems. In future, we plan to add the algorithms which can fool other biometrics modalities such as iris and fingerprint.

5. Acknowledgement

Agarwal is partly supported by Visvesvaraya PhD Fellowship, and Vatsa and Singh are partly supported through CAI@IIIT-Delhi. This research is also partly supported through a grant from MEITy, Government of India.

References

[1] A. Agarwal, R. Singh, M. Vatsa, and N. Ratha. Are image-agnostic universal adversarial perturbations for face recognition difficult to detect? *IEEE BTAS*, 2018.

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

[3] A. N. Bhagoji, D. Cullina, C. Sitawarin, and P. Mittal. Enhancing robustness of machine learning systems via data transformations. In *CISS*, pages 1–5, 2018.

[4] N. Carlini and D. Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. *ACM AISec*, 2017.

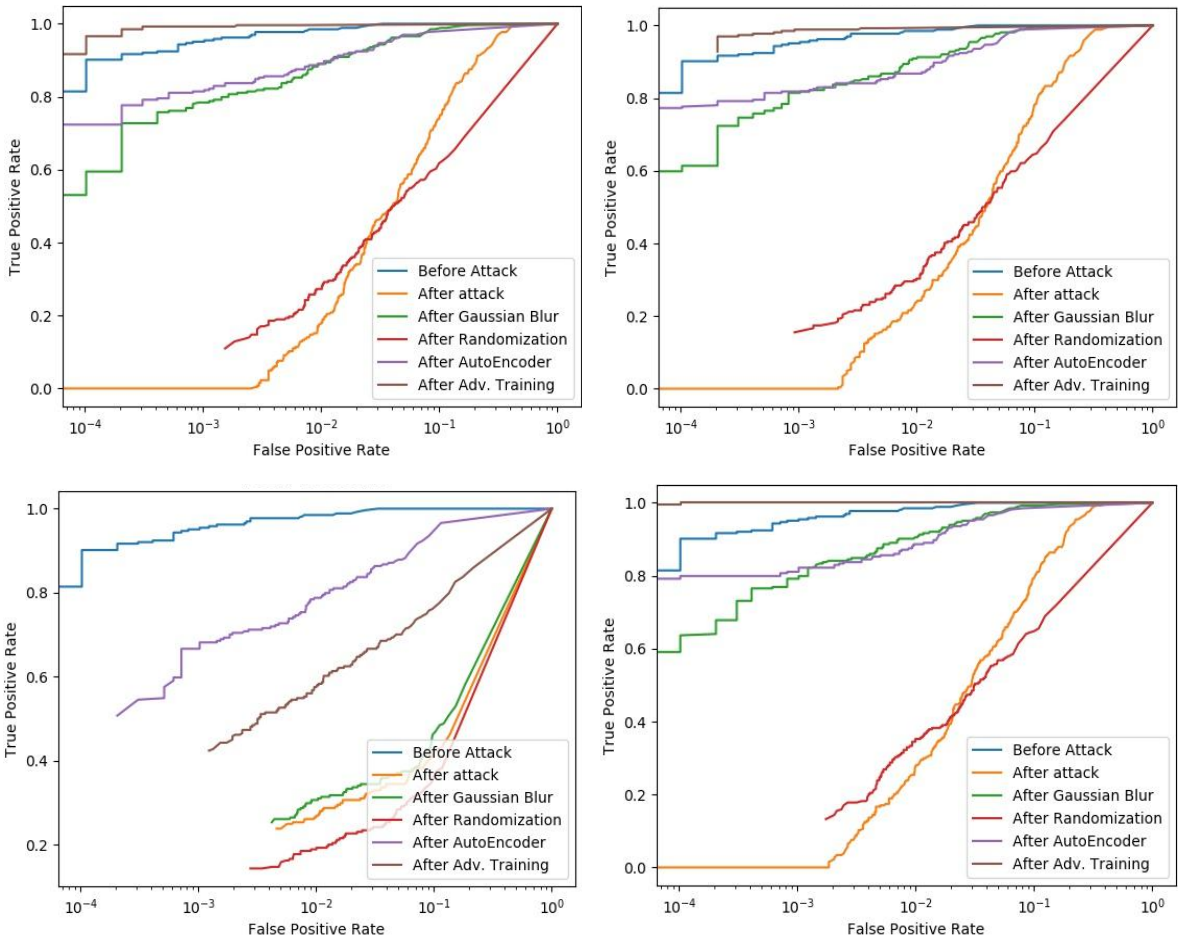

Figure 3: ROC curves summarizing the face verification performance with various attack algorithms. Attack algorithms used in top row from left to right: EAD-EN, EAD-L1; Attack algorithms used in bottom row from left to right: FGSM, L_2.

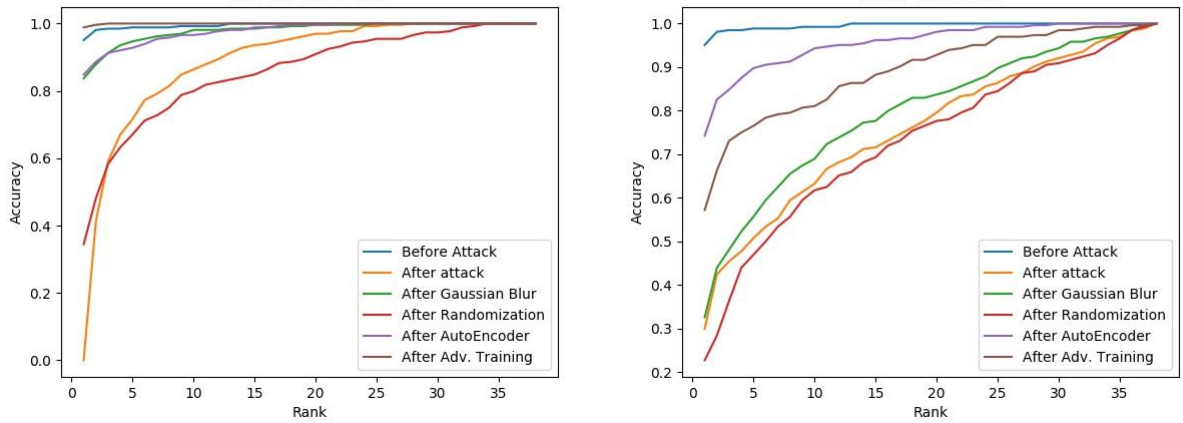

Figure 4: CMC curves summarizing face recognition under different attack algorithms. Attack algorithms used from left to right: EAD-EN, FGSM.

[5] N. Carlini and D. Wagner. Towards evaluating the robustness of neural networks. In *IEEE S&P*, pages 39–57, 2017.

[6] P.-Y. Chen, Y. Sharma, H. Zhang, J. Yi, and C.-J. Hsieh. EAD: elastic-net attacks to deep neural networks via adversarial examples. *AAAI*, 2018.

[7] M. Cisse, P. Bojanowski, E. Grave, Y. Dauphin, and N. Usunier. Parseval networks: Improving robustness to adversarial examples. In *ICML*, pages 854–863, 2017.

[8] A. Creswell and A. A. Bharath. Denoising adversarial autoencoders. *CoRR*, abs/1703.01220, 2017.

[9] N. Das, M. Shanbhogue, S.-T. Chen, F. Hohman, L. Chen, M. E. Kounavis, and D. H. Chau. Keeping the bad guys out: Protecting and vaccinating deep learning with jpeg compression. *arXiv preprint arXiv:1705.02900*, 2017.

[10] Y. Dong, F. Liao, T. Pang, H. Su, X. Hu, J. Li, and J. Zhu. Boosting adversarial attacks with momentum. *CVPR*, 2018.

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

[12] J. Gao, B. Wang, Z. Lin, W. Xu, and Y. Qi. Deepcloak: Masking deep neural network models for robustness against adversarial samples. *ICLR*, 2017.

[13] A. Georghiades, P. Belhumeur, and D. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *TPAMI*, 23(6):643–660, 2001.

[14] Z. Gong, W. Wang, and W.-S. Ku. Adversarial and clean data are not twins. *arXiv preprint arXiv:1704.04960*, 2017.

[15] I. J. Goodfellow, J. Shlens, and C. Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

[16] G. Goswami, N. Ratha, A. Agarwal, R. Singh, and M. Vatsa. Unravelling robustness of deep learning based face recognition against adversarial attacks. *AAAI*, 2018.

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

[18] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *ICCV*, pages 1026–1034, 2015.

[19] J. Hendrik Metzen, T. Genewein, V. Fischer, and B. Bischoff. On Detecting Adversarial Perturbations. *ICLR*, 2017.

[20] D. Hendrycks and K. Gimpel. Visible progress on adversarial images and a new saliency map. *CoRR*, abs/1608.00530, 2016.

[21] K. Lee, J. Ho, and D. Kriegman. Acquiring linear subspaces for face recognition under variable lighting. *TPAMI*, 27(5):684–698, 2005.

[22] X. Li and F. Li. Adversarial examples detection in deep networks with convolutional filter statistics. *ICCV*, 7, 2017.

[23] B. Liang, H. Li, M. Su, X. Li, W. Shi, and X. Wang. Detecting adversarial examples in deep networks with adaptive noise reduction. *arXiv preprint arXiv:1705.08378*, 2017.

[24] C. Lu and X. Tang. Surpassing human-level face verification performance on lfw with gaussianface. In *AAAI*, pages 3811–3819, 2015.

[25] J. Lu, T. Issaranon, and D. Forsyth. Safetynet: Detecting and rejecting adversarial examples robustly. *ICCV*, 2017.

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

[27] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard. Universal adversarial perturbations. In *CVPR*, pages 1765–1773, 2017.

[28] S. M. Moosavi Dezfooli, A. Fawzi, and P. Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *CVPR*, pages 2574–2582, 2016.

[29] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *IEEE S&P*, pages 582–597, 2016.

[30] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *CCS*, pages 1528–1540. ACM, 2016.

[31] M. Sharif, S. Bhagavatula, L. Bauer, and M. K. Reiter. Adversarial generative nets: Neural network attacks on state-of-the-art face recognition. *arXiv preprint arXiv:1801.00349*, 2017.

[32] J. Su, D. V. Vargas, and K. Sakurai. One pixel attack for fooling deep neural networks. *CoRR*, abs/1710.08864, 2017.

[33] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. *ICLR*, 2014.

[34] Q. Wang, W. Guo, K. Zhang, I. Ororbia, G. Alexander, X. Xing, X. Liu, and C. L. Giles. Learning adversary-resistant deep neural networks. *arXiv preprint arXiv:1612.01401*, 2016.

[35] C. Xie, J. Wang, Z. Zhang, Z. Ren, and A. Yuille. Mitigating adversarial effects through randomization. *ICLR*, 2018.