CLEANING ADVERSARIAL PERTURBATIONS VIA RESIDUAL GENERATIVE NETWORK FOR FACE VERIFICATION

Yuying Su, Guangling Sun, Weiqi Fan, Xiaofeng Lu, Zhi Liu

School of Communication and Information Engineering, Shanghai University, Shanghai 200444, China

ABSTRACT

Deep neural networks (DNNs) have recently achieved impressive performances on various applications. However, recent researches show that DNNs are vulnerable to adversarial perturbations injected into input samples. In this paper, we investigate a defense method for face verification: a deep residual generative network (ResGN) is learned to clean adversarial perturbations. We propose a novel training framework composed of ResGN, pre-trained VGG-Face network and FaceNet network. The parameters of ResGN are optimized by minimizing a joint loss consisting of a pixel loss, a texture loss and a verification loss, in which they measure content errors, subjective visual perception errors and verification task errors between cleaned image and legitimate image respectively. Specially, the latter two are provided by VGG-Face and FaceNet respectively and have essential contributions for improving verification performance of cleaned image. Empirical experiment results validate the effectiveness of the proposed defense method on the Labeled Faces in the Wild (LFW) benchmark dataset.

Index Terms— Deep residual generative network, Face verification, Adversarial perturbations, Joint loss

1. INTRODUCTION

Nowadays, deep neural networks (DNNs) have achieved impressive performances on various applications, such as computer vision, speech processing, natural language processing. However, recent investigations show that DNNs are fragile and can easily be confused by adversarial examples which are contaminated by elaborately designed perturbations [1]. More seriously, the adversarial examples even can pose real threats for physical world systems [2,3]. Although plenty of defensive techniques have been developed [4], making DNNs robust against adversarial examples is still an important and challenging problem.

Facial analysis system is also vulnerable to adversarial attacks. Mahmood Sharif et al [5] developed a physically realizable and inconspicuous attack through printing a pair of eyeglass frames. When worn by the attacker, the eyeglasses allow her to evade identification or to impersonate another individual. The authors further designed an adversarial generative net that outputs a physically realizable attack instance [6]. Rozsa et al. [7,8] generated adversarial examples using Fast Flip-ping Attribute (FFA) technique to change the results of facial attribute recognition so as to explore the stability of multiple deep learning approaches. Mirjalili and Ross [9] proposed a mechanism that perturbs a face image such that its gender attribute decided by a gender classifier was modified whereas its extra biometric information was retained. Shen et al. [10] generated adversarial examples for certain face image which can have high attractiveness scores but low subjective scores in the face attractiveness evaluation using deep neural network. Yukun Ma et al. [11] combined the minimum disturbance dimensions and visual concentration in order to yield the adversarial examples for face-spoofing detection.

In this paper, we attempt to design a strategy towards cleaning adversarial perturbations for defending face verification system. Specifically, we propose a framework which learns a deep residual generative network (ResGN) that completes adversarial perturbation cleaning task. The work most related with ours is APE-GAN [12] which used a typical generative adversarial network (GAN) [13] architecture. The generative network was learned to eliminate perturbation and the discriminator network was learned to improve the visual quality of the generated image. Instead of discriminator network, we use two pre-trained networks, VGG-Face and FaceNet to aid the ResGN training, which are illustrated in Fig.1. During training, only ResGN is learned and the two pre-trained networks provide feedbacks to improve the performance of the cleaned images. The optimization is to minimize a joint loss function including a pixel loss, a texture loss and a verification loss. For a pair of cleaned and legitimate images, the three losses measure content errors, subjective visual perception errors using pre-trained VGG-Face [14] and verification results errors using pre-trained FaceNet [15] respectively.

We summarize our contributions as follows:

1 We propose a novel training framework to learn ResGN, in which ResGN is used to clean adversarial perturbations.

2 We present a joint of three losses to optimize ResGN. In addition to a pixel loss, two pre-trained networks, VGG-Face and FaceNet aid the learning of ResGN by providing a texture loss and a verification loss.

3 We conduct extensive experiments to validate the effectiveness of the proposed method on the Labeled Faces in the Wild (LFW) benchmark dataset [16].

2. GENERATING ADVERSARIAL EXAMPLES

2.1. Fast Gradient Sign Method

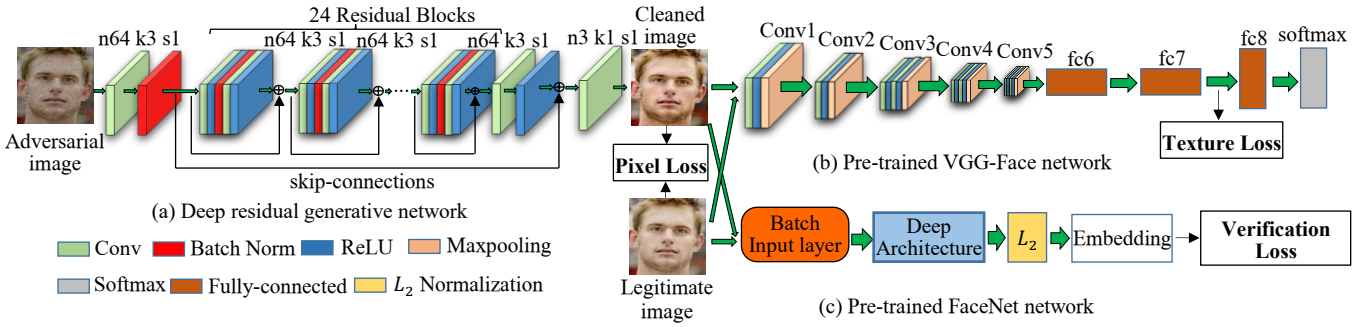


Fig.1: The proposed training framework for ResGN. The parameters of ResGN are optimized by minimizing a joint loss consisting of a pixel loss, a texture loss and a verification loss. Given an adversarial image, ResGN will output the cleaned image, the pixel loss is calculated using the cleaned image and corresponding legitimate image. Similarly, the two images are both fed into VGG-Face and FaceNet to calculate the texture loss and the verification loss respectively. A weighted summation of the three losses is the joint loss. Corresponding number of feature maps (n), kernel size (k) and stride (s) are marked for each convolutional layer in (a).

In this paper, we utilized Fast Gradient Sign Method (FGSM) [17] to craft adversarial examples for face verification. Given an input image $\boldsymbol{x}$, the adversarial example $\boldsymbol{x}^{adv}$ is obtained as follows:

$$\boldsymbol{x}^{adv} = \boldsymbol{x} + \epsilon \cdot \text{sign}(\nabla \mathcal{J}(h(\boldsymbol{x}), y_{gt})), \quad (1)$$

where $h(\boldsymbol{x})$ is a classification result on a given input image $\boldsymbol{x}$ determined by a target network, y_{gt} is the Ground Truth of $\boldsymbol{x}$, $\nabla \mathcal{J}(.,.)$ denotes the gradient of loss function $\mathcal{J}(.,.)$ and ϵ controls the amount of adversarial perturbations.

2.2 Adversarial examples for face verification

Given pairs of facial images labeled 'same person' or 'not same person', a face verification system will learn how to classify them during training; given a pair of facial images, the trained face verification system will determine whether the pair of facial images is 'same person' or 'not same person' during inference. Obviously, adversarial facial images can cause the verification system to output an incorrect verification result during inference. Dodging and impersonation are two types of attacks: given an input facial image, dodging intends to make it identified as any one different from its genuine identity; impersonation intends to make it identified as a specified identity.

Bruno López [18] released codes to craft adversarial examples using FGSM and FaceNet as target network. Given an input image and a reference image, if the two facial images are from the same person, the y_{gt} in Eq. (1) is $[1,0]$; otherwise, the y_{gt} is $[0,1]$. Both the input image and the reference image are fed into FaceNet to obtain embeddings. The Euclidean distance between the two embeddings is transformed to a *score* as follows:

$$score = \begin{cases} 0.5 + \dfrac{(d-\eta) \times 0.5}{4-\eta}, & d > \eta \\ \dfrac{0.5 \times d}{\eta}, & d < \eta \end{cases} \quad (2)$$

where d is the Euclidean distance between the two embeddings, η is the threshold value. And then $h(\boldsymbol{x})$ is formed as $[1 - score, \quad score]^T$. The threshold η is set 1.1 which can correctly classify every pair from [14]. After above mentioned conversions, both dodging and impersonation attacks can utilize Eq. (1) to produce adversarial examples for face verification.

3. PROPOSED FRAMEWORK

The training framework contains three networks, namely the ResGN, pre-trained VGG-Face and pre-trained FaceNet. During training, only ResGN is learned. As auxiliary networks, VGG-Face and FaceNet provide addition information to boost the cleaned image performance. Specifically, in addition to pixel loss evaluating the content errors, texture loss evaluating visual quality errors is provided by VGG-Face and verification loss evaluating task errors is provided by FaceNet. The three losses are jointly optimized to learn the ResGN. For the proposed training framework, its advantage over the GAN is that the training algorithm is more efficient and stable due to the ResGN learning alone. In addition, FaceNet can help the ResGN learn towards obtaining better verification performance. The three networks' components and the optimization algorithm are discussed in following subsections.

3.1. Residual generative network

Inspired by the successful application in super-resolution reconstruction [19], the ResGN is designed as shown in Fig.1 (a). It contains 24 residual blocks and all blocks have an identical layout. Each residual block has two convolutional layers with small 3×3 kernels and 64 feature maps. Batch-normalization layers and ReLU are followed by convolutional layer. The strategy of skip-connections is also used in this network. This network is trained to clean the adversarial images.

3.2. VGG-Face

The VGG-Face [14] is trained using 2.6M facial images of 2622 unique individuals. As Fig.1 (b) illustrates, the network contains five convolutional blocks, three fully connected layers and a softmax layer. Each convolutional block comprises linear operators followed by one or more non-linear layers, such as ReLU or max pooling. A stack of convolutional layers is followed by three fully connected

layers. We use 4096 dimensions feature maps obtained by the second fully connected layer for calculating the texture loss.

3.3. FaceNet

Schroff et al. [15] proposed FaceNet, which achieved the state-of-the-art face verification performance with 128 embedding dimensions. As illustrated in Fig.1(c), FaceNet includes a batch input layer and a deep convolutional network followed by L_2 normalization and outputs embedding as feature vectors. The network is trained by minimizing a triplet loss. In our work, we choose the FaceNet based on Inception Resnet V1 architecture and is trained using the MS-Celeb-1M database. The Euclidean distance of embeddings is used to calculate the verification loss for face verification task.

3.4. Optimization

To optimize the ResGN, we present an optimization formulation towards minimizing a joint loss consisting of pixel loss, texture loss and verification loss. They essentially evaluate the discrepancy between legitimate image and cleaned image from three aspects including content, visual perception and ultimate verification accuracy. Consequently, minimizing the joint loss leads to preserving the content of the legitimate image as much as possible, obtaining a high visual quality as the legitimate image, and achieving a verification performance on a par with legitimate image. We use I_C and I_L to denote cleaned image output by ResGN and legitimate image respectively.

The pixel loss $\mathcal{L}_{pixel}$ measures content errors between I_C and I_L and the loss is calculated as follows:

$$\mathcal{L}_{pixel} = \frac{1}{N_p}\sum_{j=1}^{N_p}(\boldsymbol{C}_j^{I_C} - \boldsymbol{C}_j^{I_L})^2, \quad (3)$$

where N_p is the total number of pixels in each image. $\boldsymbol{C}_j^{I_C}$ *and* $\boldsymbol{C}_j^{I_L}$ denote the color of jth pixel in RGB space corresponding to I_C and I_L respectively.

The texture loss $\mathcal{L}_{texture}$ measures subjective visual perception errors between I_C and I_L and the loss is calculated as follows:

$$\mathcal{L}_{texture} = \frac{1}{N_t}\sum_{j=1}^{N_t}(\mathcal{F}_j^{I_C} - \mathcal{F}_j^{I_L})^2, \quad (4)$$

where $\boldsymbol{\mathcal{F}}^{I_C} \in \mathcal{R}^{N_t}$ and $\boldsymbol{\mathcal{F}}^{I_L} \in \mathcal{R}^{N_t}$ denote the feature maps of $I_C$and I_L,which are available from the second fully connected layer of VGG-Face. N_t is the dimension of the feature map and equals to the node number of the second fully connected layer.

We also consider verification loss $\mathcal{L}_{verification}$ to measure verification task errors for I_C and I_L. The loss depends on FaceNet and is calculated as follows:

$$\mathcal{L}_{verification} = -log(1 - score), \quad (5)$$

where *score* is computed using Eq. (2). The 'd' in Eq. (2) refers to the Euclidean distance between two embeddings of cleaned image and legitimate image. The verification loss definition indicates that the verification result for cleaned image and legitimate image is expected to be the same person irrespective of it is dodging attack or impersonation attack.

Finally, a joint loss function is defined as a weighted summation of the three losses:

$$\mathcal{L}_{joint} = \mathcal{L}_{pixel} + 10^{-8}\cdot\mathcal{L}_{texture} + 10^{-3}\cdot\mathcal{L}_{verification}. \quad (6)$$

As hyper-parameters, the weights in Eq. (6) are set according to the training results.

The training process of ResGN is summarized in Algorithm 1.

Algorithm 1: Optimization of ResGN

Input: pairs of adversarial and legitimate images (*training set*)
Parameter: n: batch number, m: batch size, N: iteration number

Separate *training set* into n batches of size m
for $k = 1, \cdots, N$ **do**
  for $i = 1, \cdots, n$ **do**
    for $j = 1, \cdots, m$ **do**
      -Calculate the pixel loss $\mathcal{L}_{pixel}^{(i,j)}$ using Eq. (3)
      -Calculate the texture loss $\mathcal{L}_{texture}^{(i,j)}$ using Eq. (4)
      -Calculate the verification loss $\mathcal{L}_{verification}^{(i,j)}$ using Eq. (5)
      -Calculate the joint loss function $\mathcal{L}_{joint}^{(i,j)}$ using Eq. (6)
    end
    Update the ResGN by descending its stochastic gradient:
      $\nabla_\theta \frac{1}{m}\sum_{j=1}^{m}\mathcal{L}_{joint}^{(i,j)}$
      where θ represents the weights and biases of ResGN.
  end
end

4. EXPERIMENTS

4.1 Target network and dataset

FaceNet is taken as target network because of its significant performance on face verification. We choose the LFW benchmark dataset in all experiments. First, the correctly verified examples by FaceNet are selected. Then we use the codes released by [18] for implementing FGSM attack to generate adversarial examples. The attacking intensity ϵ is 0.03. From all the generated examples, a total of 4000 adversarial examples including 2000 dodging examples and 2000 impersonation examples compose training set. And a total of 2580 adversarial examples including 1286 dodging examples and 1294 impersonation examples compose testing set.

4.2 Parameter setting

We use TensorFlow [20] and a NVIDIA TITAN X GPU to train and test ResGN. We use Adam optimizer [21] during optimization. The initial learning rate is 0.0001 and 0.9 is the learning rate decay. The batch size m is 16. The iteration number N is 1000. The input image size is 160*160*3.

4.3 Experimental results

4.3.1 Experiment on verification performance of cleaned image

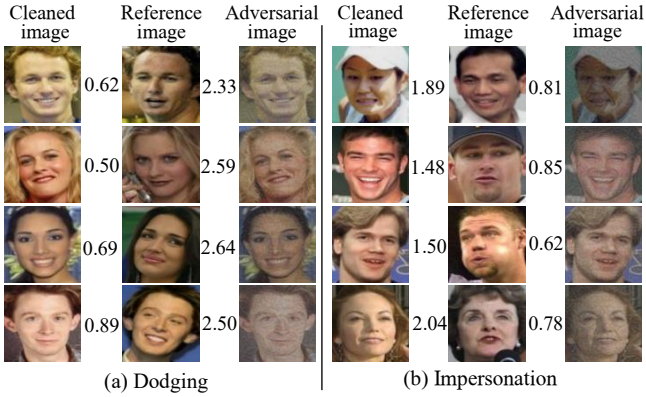
To check and verify the advantage of the joint loss, we learn seven ResGN models optimized with three single loss, three two losses combinations and the joint loss (see the left most column in table 1). During testing, any input first is processed

by ResGN and then is fed into the target network, namely FaceNet, to complete verification. The verification accuracy of 1286 dodging examples and 1294 impersonation examples cleaned by seven ResGN models are listed in table 1.

To further confirm the effectiveness of our proposed defense method, we compare it with Randomization [22]. The results in table 1 suggest the ResGN is superior to Randomization with a large margin on two types of adversarial examples and also performs better than Randomization on legitimate examples.

Tabel 1: The verification accuracy of original input examples and cleaned examples using the ResGN optimized with seven losses and Randomization. The best results are emphasized in bold. (%)

Type / Loss	Dodging	Impersonation	Legitimate
pixel	99.16	99.68	90.52
texture	99.06	99.67	97.91
verification	88.36	95.31	98.31
pixel+ texture	99.25	99.82	97.86
pixel+ verification	99.26	99.83	98.70
texture+ verification	95.99	97.57	98.83
joint	**99.45**	**99.92**	**99.18**
Randomization	65.28	61.81	97.65
Original input Example	0.84	43.21	100



(a) Dodging (b) Impersonation

Fig.2: Adversarial examples of perturbation successfully cleaned. The value between two images is a Euclidean distance computed with FaceNet network.

The bold results in Table 1 indicate that the ResGN optimized with the joint loss performs best for both attacks. In addition, the performance has achieved a significant improvement compared with their adversarial counterpart. Considering in real setting, most of the input are legitimate samples, we also test an amount of 2580 'cleaned' images that are originally legitimate images. The results show that the joint loss still outperforms other six losses and only has a slight decline compared with the original legitimate samples. Thus, in following experiments, we use the ResGN optimized with the joint loss.

Fig.2 demonstrates some instances that adversarial perturbations are successfully removed by ResGN. We can notice that not only a satisfied visual effect has been obtained for the cleaned image, but the distance between it and reference image has been reversed to obtain a correct verification result.

4.3.2 Experiment on attacking intensity

In this experiment, we produce multiple adversarial example sets using different attacking intensity in FGSM attack. And then, we learn each ResGN from each adversarial example set and use it to process all adversarial examples and legitimate examples. All processed examples are fed into the FaceNet to test the performance. We choose 0.02, 0.03 and 0.04 as attacking intensity and the results are listed in table 2.

It is discovered that the ResGN learned from higher attacking intensity is capable of effectively cleaning the perturbations injected by lower attacking intensity, yet it is not vice versa. The results imply that the ResGN learned from higher attacking intensity adversarial examples is enough to clean the adversarial examples.

Table 2: The face verification results for multiple attacking intensity adversarial examples. (%)

Test \ Train		0.02	0.03	0.04
Dodging	0.02	98.32	98.33	96.20
	0.03	5.12	99.45	98.69
	0.04	1.31	5.68	99.06
Impersonation	0.02	99.92	99.35	99.51
	0.03	42.48	99.92	99.76
	0.04	33.98	42.23	99.83

5. CONCLUSION

In this paper, we propose a novel deep residual generative network (ResGN) training framework and the optimized ResGN is used to clean adversarial example in face verification task. Given pairs of clean images and legitimate images, the joint loss comprised of a pixel loss, a texture loss provided by VGG-Face and a verification loss provided by FaceNet is minimized to learn ResGN. The empirical results validate the effectiveness of ResGN on cleaning adversarial perturbations for face verification on LFW benchmark dataset. Additionally, the proposed ResGN has a flexible adaptivity in that ResGN can incorporate with any pre-trained network applied to other face analysis task, such as face identification and facial attribute classification. In future, we will learn ResGN from more advanced adversarial examples and combine ResGN with other defensive technique to enhance security of image recognition system.

6. ACKNOWLEDGMENTS

This work is supported by Shanghai Municipal Natural Science Foundation under Grant No. 16ZR1411100 and the National Natural Science Foundation of China under Grant No. 61771301.

7. REFERENCES

[1] N. Papernot, P. McDaniel, A. Sinha and M. Wellman, "Towards the science of security and privacy in machine learning," in *arXiv preprint arXiv: 1611.03814*, 2016.

[2] A. Athalye, L. Engstrom, A. Ilyas, and K. Kwok, "Synthesizing robust adversarial examples," in *arXiv preprint arXiv: 1707.07397*, 2017.

[3] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, and D. Song, "Robust physical-world attacks on deep learning models," in *arXiv preprint arXiv: 1707.08945*, 2017.

[4] D. Meng and H. Chen, "MagNet: A two-pronged defense against adversarial examples," in *arXiv preprint arXiv: 1705.09064*, 2017.

[5] M. Sharif, S. Bhagavatula, L. Bauer and M. Reiter, "Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition," in *Proc. ACM CCS*, 2016, pp. 1528-1540.

[6] M. Sharif, S. Bhagavatula, L. Bauer and M. Reiter, "Adversarial Generative Nets: Neural network attack on state-of-the-art face recognition," in *arXiv preprint arXiv: 1801.00349*, 2017.

[7] A. Rozsa, M. Günther, E. M. Rudd, and T. E. Boult, "Are facial attributes adversarially robust?" in *arXiv preprint arXiv: 1605.05411*, 2016.

[8] A. Rozsa, M. Günther, E. M. Rudd, and T. E. Boult, "Facial attributes: Accuracy and adversarial robustness," in *arXiv preprint arXiv: 1801.02480*, 2018.

[9] V. Mirjalili and A. Ross, "Soft biometric privacy: Retaining biometric utility of face images while perturbing gender," in *Proc. Int. Joint Conf. Biometrics*, 2017, pp. 1–10.

[10] S. Shen, R. Furuta, T. Yamasaki, and K. Aizawa, "Fooling neural networks in face attractiveness evaluation: Adversarial examples with high Attractiveness score but low subjective score," in *Proc. IEEE 3rd Int. Conf. Multimedia Big Data*, Apr. 2017, pp. 66–69.

[11] Y. Ma, L. Wu, M. Jian, F. Liu and Z. Yang, "Approach to generate adversarial examples for face-spoofing detection," *Ruan Jian Xue Bao/Journal of Software* (in Chinese), pp.1-10, 2018. http://www.jos.org.cn/1000-9825/5568.html

[12] S. Shen, G. Jin, K. Gao, Y. Zhang, "APE-GAN: Adversarial perturbation elimination with GAN," in *arXiv preprint arXiv: 1707.05474,* 2017.

[13] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville and Y. Bengio, "Generative Adversarial Nets," in *arXiv preprint arXiv: 1406.2661*. 2014.

[14] O.M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," in *BMVC*, 1(3), pp. 6, 2015.

[15] F. Schroff, D. Kalenichenko and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *Proc. CVPR*, 2015, pp. 815-823.

[16] G.B. Huang, M. Ramesh, T. Berg, and E. Learned-Miller, "Labeled Faces in the Wild: A database for studying face recognition in unconstrained environments," *Workshop on Faces in 'Real-Life' Images: Detection, Alignment, and Recognition*, Oct 2008, Marseille, France. 2008.

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

[18]https://brunolopezgarcia.github.io/2018/05/09/Crafting-adversarial-faces.html.

[19] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Tejani, J. Totz, Z. Wang and W. Shi, "Photo-realistic single image super-resolution using a generative adversarial network," in *arXiv preprint arXiv: 1609.04802*, 2017.

[20] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G.S. Corrado, A. Davis, J. Dean, M. Devin, et al., "TensorFlow: Large-Scale machine learning on heterogeneous distributed systems," in *arXiv preprint arXiv: 1603.04467*, 2016.

[21] D.P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *arXiv preprint arXiv: 1412.6980*, 2014.

[22] C. Xie, J. Wang, Z. Zhang and A. L. Yuille, "Mitigating adversarial effects through randomization," in *International Conference on Learning Representations (ICLR)*, 2018.