Adversarial Attacks and Defense on Rendering-Autoencoder of Faces

Wenbin Teng

September 2021

Abstract

The 3D reconstruction from a 2D face image is a challenging task because of the ill-posed nature of this problem. To overcome this issue, model-based face-rendering autoencoders (MoFA) train a deep neural network in an unsupervised manner and fit a robust model to the target image. However, the face reconstruction process is vulnerable to adversarial attack, i.e. normal images added with a visually imperceptible perturbation would cause the renderer to construct a face that belongs to a different identity. In this work, we propose an iterative attack generation algorithm to distort the semantic representations so that mislead the face autoencoder to generate completely different output. Experimental results show that MoFA is even easier to attack compared with traditional classifiers. In addition, we train another deep neural network as an adversarial detector to differentiate between normal and adversarial examples in order to perform adversarial defense by reforming them into the space of normal examples. Both qualitative and quantitative experimental results on public dataset show that our adversarial attack and defense algorithm are effective.

Keywords
adversarial attack, model-based autoencoder, face recognition

1 Introduction

The 3D reconstruction of faces from the RGB image is a longstanding research problem in computer vision due to the high degree of variability of uncalibrated photos in terms of resolution and capture device [1]. Traditional monocular 3D reconstruction methods aim to perform a regression model to fit the pose, shape, expression and illumination parameters of each human faces, but these approaches can only be trained in a supervised manner and such criterion is always hard to achieve. Model-based face autoencoders (MoFA) [1] address this problem by training a deep neural network to extract each semantic information of faces (e.g. shape, appearance, pose, etc.) and fitting a 3D Morphable Model (3DMM) [2] to a given 2D image. 3DMM is a commonly used technique that could parametrize faces with shape and texture parameters. With the help of the technique, the autoencoders achieve an end-to-end training in an unsupervised manner that connects a CNN-based encoder and model-based decoder, which yield better performance and reconstruction quality compared with other methods only with a regression-based approach [3, 4, 5].

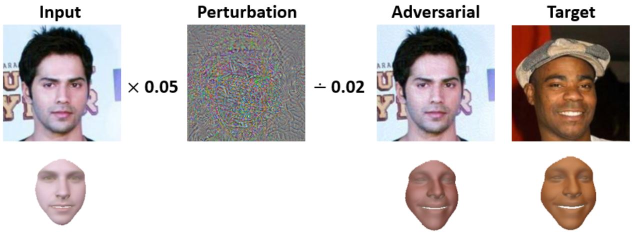


Figure 1: An adversarial attack example for our proposed method. Top row: the 2D images of original normal image, adversarial attack perturbation, adversarial and target image (left to right). Bottom row: the 3D reconstruction of normal, adversarial and target image (left to right). $\doteq$ shows that the magnitude of pixel perturbation is capped below 0.02, the details of which will be discussed in Section 3.2

Despite a more robust 3D reconstruction procedure introduced by MoFA, it is nonetheless vulnerable to adversarial attack, a non-trivial technique that optimizes the input image in order to mislead the original model to generate unexpected results. The attackers, which we refer to as perturbations, are usually imperceptible so that the systems are easily "fooled". Therefore, when perturbations attack the face autoencoders, the 3D face reconstruction would be distorted so that it visually belongs a completely different identity.

Addressing the aforementioned problem, extensive research has been studied regarding the topics including different types of attacking methods together with how to defend them. However, these works largely focus on issues with attacking discriminative models: the output score $p_c(\theta|x)$, which represents the probability of being classified as class c, would be largely different after adversarial attack $p_c(\theta|x+\eta)$, where η is the perturbation. There are very few works dealing with adversarial attacks on generative models. Some existing methods integrate the techniques of generating adversarial examples with autoencoders, optimizing perturbations by progressively aligning embeddings of attacked and target images [6, 7], but their limitations are three-fold: first, they constrain their focuses within the adversarial attack on reconstruction of images in 2D domain; second, robust results are only evaluated on small dataset like MNIST [8] and SVHN [9]; therefore the attack is not generalizable to larger in-the-wild datasets; and third, compared with MoFA, although applying similar network to extract latent representations, they also apply CNN-based decoder to reconstruct their inputs, which makes it harder to perform a successful attack. In contrast with preceding works, our paper addresses that similar attacks could also succeed in the process of 3D face rendering with model-based autoencoders and we also evaluate our proposed method on larger datasets like CelebA [10].

To generate the adversarial examples, we randomly choose a pair of original and target image; a perturbation matrix with the same dimension is initialized and a proportion of it is added to the original image. Setting all parameters fixed, the encoder of the face autoencoder is applied on both the adversarial and target image to extract the corresponding shape, appearance, expression, pose and illumination parameters. The perturbation is optimized iteratively and added with the original image so that the divergence of each set of parameters between adversarial and target image are as close as converged. To restrict the per-pixel magnitude of the perturbation, we set both a minimum and maximum cap value and add a regularization term as part of the loss function (See Eqn. (1)). Figure 1 shows an example of the original, adversarial and target images. As it could be discovered, only a tiny proportion of the generated perturbation added on the original image would cause the face autoencoder to output a 3D model completely different from that of the original input.

For adversarial detection and defense, we design another CNN-based autoencoder that is trained only on the normal examples. The well-trained autoencoder would be capable of (1) learning the space of normal examples, (2) selecting the adversarial examples by measuring the distance between input and output images, and (3) reforming the adversarial examples by moving them close to the space of normal examples. For the second point, the primary choice for measuring the distance is the reconstruction error based on L_p-norm. However, the magnitude of perturbation is optimized to be as low as possible so that the L_p-norm based distance is too small to select an adequate threshold. To solve this problem, we apply Jensen-Shannon divergence to measure the difference in their probability distribution between the input and output of the autoencoder. This divergence measures whether the two set of samples are generated with the same distribution, and divergence is shown to be significant even if the reconstruction error is not. We will show more details in Section 3.3. Furthermore, the output of the autoencoder is regarded as a defense to our attack procedure as it reforms the adversarial examples and moves them close to the space of corresponding normal examples. The qualitative results of defense will be shown in Section 4.5

In summary, we make the following contributions in this paper:

- We focus on the problem of adversarial attack on model-based face autoencoder (MoFA) so that a small perturbation would successfully mislead the network to generate 3D face reconstruction that belongs a different identity. Further analysis show that this a much easier attack than those on classifiers.

- We propose to train a simple CNN-based autoencoder to detect and defend adversarial attack on MoFA architecture. The network is still effective when encountering harder adversarial examples.

- Qualitative and quantitative results on the large in-the-wild images proves that our attack and defense algorithm is robust and reliable.

2 Related Works

2.1 Adversarial attacks on face recognition

Deep CNNs are vulnerable to adversarial attacks [11, 12, 13]. Likewise, face recognition has also presented similar property of vulnerability. For example, Sharif et al. [14] propose a perturbation optimization scheme that constrains perturbations to the eyeglass region, which fool face recognition systems; Dong et al. [15] propose an evolutionary attack algorithm, which can model the local geometry of the search directions and reduce the dimension of search space. However these works only focus on adversarial attack and adversarial example generation on face recognition in a decision-based setting; by contrast, in [16], generative adversarial networks (GANs) is utilized to generate adversarial examples with desired objective, but, like [14], the perturbation is constrained in eyeglass region and also quite perceptible. In this paper, we will focus on adversarial examples in a generative-based setting; the attacks concentrate on global region and not easily perceptible.

2.2 Model-based Face Autoencoder

The Model-based Face Autoencoder (MoFA) is an end-to-end architecture proposed by Tewari et al. [1] that combines a deep neural network as an encoder and a renderer as a decoder. The encoder applies the VGG-19 [17] or AlexNet [18] to extract the semantic information, including the shape, expression, appearance, pose and illumination features, of a 2D face image. The face renderer first parametrizes the extracted features into spatial embedding and skin reflection by fitting two 3D Morphable Models (3DMM). A camera model is applied to transform the spatial embedding from world space into camera space and further into screen space, whereas an illumination model is applied to predict the pixel color of each spatial vertex. The 3DMM plays an important part in MoFA architecture since it parametrizes the latent distribution of faces, connecting the encoder and the model-based decoder to achieve an end-to-end training. With the help of 3DMM together with the analytical and differentiable face decoder, MoFA is proved to generate more robust 3D reconstructions than other methods [19, 20, 3, 4, 5]. However, in our work, we will show that the face autoencoder is vulnerable to adversarial attack and easier to be attacked than classification networks.

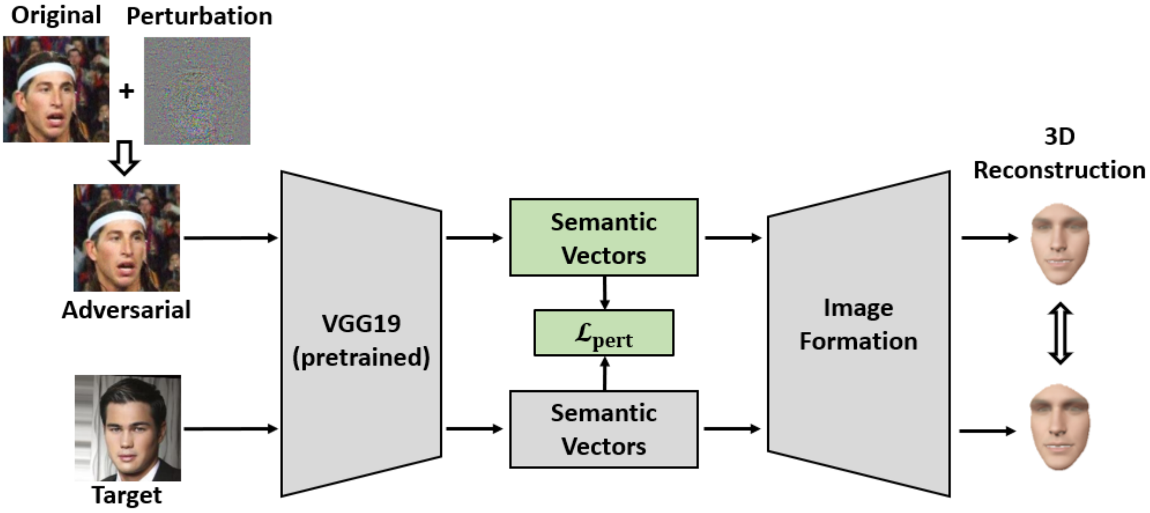


Figure 2: Visualization of adversarial attack model on face rendering autoencoder. Perturbation is optimized so that the distance between semantic vectors of adversarial and target images is minimized. The goal is to mislead the autoencoder to reconstruct the 3D model of a different target face.

3 Approach

The architecture of the full network is shown in Figure 2. In Section 3.1, we will introduce more terminology and training schemes of MoFA network. Our adversarial attack, detection and defense algorithms will be discussed in Section 3.2, 3.3 and 3.4, respectively.

3.1 Model-based Face Autoencoder

Similar to previous discussion, the MoFA architecture consists of an encoder and model-based based decoder. Given an image I, the encoder extracts a set of semantic vectors $\mathbf{z}$ that represent the shape $\boldsymbol{\alpha} \in \mathbb{R}^{80}$, expression $\boldsymbol{\delta} \in \mathbb{R}^{64}$, appearance $\boldsymbol{\beta} \in \mathbb{R}^{80}$, pose $\mathbf{T} \in \mathbb{R}^{3}$, translation $\mathbf{t} \in \mathbb{R}^{3}$ and illumination $\boldsymbol{\gamma} \in \mathbb{R}^{27}$ of each 2D face images. Therefore, the semantic vector is a dimension concatenation of the sets of vectors: $\mathbf{z} = (\boldsymbol{\alpha}, \boldsymbol{\delta}, \boldsymbol{\beta}, \mathbf{T}, \mathbf{t}, \boldsymbol{\gamma}) \in \mathbb{R}^{257}$. 3DMM connects the encoder and decoder by parameterizing a shape model $\mathbf{V}$ and texture model $\mathbf{R}$. The face renderer consists of a camera model that performs a projection from space to screen: $\mathbf{V}(\boldsymbol{\alpha}, \boldsymbol{\delta}) \rightarrow \mathbf{u}$, and illumination model calculates the color pixel: $\mathbf{R}(\beta) \rightarrow \mathbf{c}$. Then the decoder renders a 3D face reconstruction through image formulation function $\mathcal{F}$: $[\mathbf{u}, \mathbf{c}] \rightarrow I_R$. If the perturbation attack doesn't exist, MoFA will efficiently render the 3D face close to the bottom row of Figure 2.

Algorithm 1: Perturbation Attack

```
Input: Original image $\mathbf{I}$; target image $\mathbf{I}_t$
Output: Distortion: $\mathbf{d}$
initialization $\mathbf{d} = \mathbf{0}$;
while not converged do
    Obtain attacked image;
    $\mathbf{I}_a = \mathbf{I} + \epsilon \cdot \texttt{Clip}(\mathbf{d}, L, U)$;
    Extract semantic vectors;
    $\mathbf{z}_a = \texttt{encoder}(\mathbf{I}_a); \mathbf{z}_t = \texttt{encoder}(\mathbf{I}_t)$;
    Update $\mathbf{d}$ with Eqn. (1)
end
```

3.2 Adversarial Attack for MoFA

Perturbation attack initializes a small distortion $\mathbf{d}$ with the same resolution as the original image input $\mathbf{I}$ and adds it directly onto the original image to obtain the attacked image $\mathbf{I}_a$. Both the attacked image $\mathbf{I}_a$ and target image $\mathbf{I}_t$ are fed into the pre-trained face encoder to extract the corresponding set of semantic vectors $\mathbf{z}_a$ and $\mathbf{z}_t$. The loss functions are designed to optimize the pixel value of distortion so that (1) the L1 distances of corresponding semantic vectors between the attacked image and target image are minimized, and (2) the magnitude of perturbation is minimized. Given the components defined in Section 2.2, we define the perturbation attack loss as:

$$\begin{aligned}
\mathcal{L}_{\text{pert}}(\mathbf{z}_a, \mathbf{z}_t) &= \Delta(\mathbf{z}_a, \mathbf{z}_t) + \lambda_{\text{pert}} \|\mathbf{d}\|_1 \\
\Delta(\mathbf{z}_a, \mathbf{z}_t) &= \lambda_\alpha \frac{\|\boldsymbol{\alpha}_a - \boldsymbol{\alpha}_t\|_1 \sigma_\alpha}{n_\alpha} + \lambda_\delta \frac{\|\boldsymbol{\delta}_a - \boldsymbol{\delta}_t\|_1 \sigma_\delta}{n_\delta} \\
&\quad + \lambda_\beta \frac{\|\boldsymbol{\beta}_a - \boldsymbol{\beta}_t\|_1 \sigma_\beta}{n_\beta} + \lambda_T \frac{\|\mathbf{T}_a - \mathbf{T}_t\|_1}{n_T} \\
&\quad + \lambda_\gamma \frac{\|\boldsymbol{\gamma}_a - \boldsymbol{\gamma}_t\|_1}{n_\gamma}
\end{aligned} \tag{1}$$

where n_α, n_δ, n_β, n_T, n_γ are the number of parameters for each semantic vector discussed in Section 2.2. λ_{pert},λ_α, λ_δ, λ_β, λ_T, λ_γ are the corresponding weights. σ_α, σ_δ, σ_β are the standard deviation of shape, expression and appearance parameters from 3DMM. We simply apply a L1-regularization to control the mean magnitude of perturbation, but we are not able to control the individual pixel value, so the chances that the perturbations get spotted by naked eye are still very high. Therefore, we set cap values for individual pixels. The detailed algorithm for perturbation attack is illustrated in Alg. 1

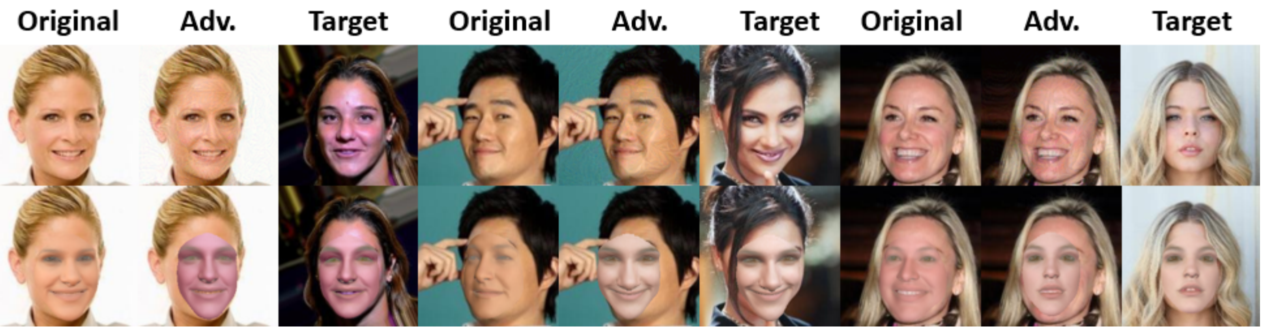


Figure 3: Qualitative comparisons on face reconstruction between original, adversarial and target images. The 1st row represents 2D images; the 2nd row represent the 3D reconstruction overlay on images.

3.3 Adversarial Detection

The adversarial detection process could be defined as a function $d : \mathbf{I} \to \{0, 1\}$ that decides whether the input image is normal or attacked. Recent work trains a binary classifier to distinguish between the normal samples against adversarial samples [21]. The limitations are two-fold: (1) the magnitude of perturbation is relatively low as we set a small cap value when generating adversarial samples, making the classifier hard to train; (2) the perturbation is not unique for different samples so that (i) the adversarial data is not generated with the same distribution, and (ii) the defender does not know the process for generating the adversarial samples so that it is unlikely to generalize.

To avoid the aforementioned problems, we train an autoencoder only on the normal samples and use the reconstruction error to measure the distance between the input and autoencoder output. The autoencoder is composed of an encoder $h : \mathbf{I} \to x$ that extracts the hidden representation of input, and a decoder $g : x \to \mathbf{I}$ that brings the hidden representation back to input space. When training the autoencoder, we optimize the reconstruction error as the loss function:

$$\mathcal{L}_{\text{rec}} = \frac{1}{N} \sum_{I \in \mathbf{I}} \|I - g \circ h(I)\|_2 \tag{2}$$

where N is the number of training samples. For any given test sample I (either normal or adversarial), the reconstruction error is defined as:

$$Err(I) = \|I - g \circ h(I)\|_p \tag{3}$$

We will conduct corresponding experiments for the level of p. Suppose the input is sampled from the same distribution of the training set, then a small reconstruction error is expected; on the other hand, if the input is an adversarial sample, a larger reconstruction error is expected.

However, for some small perturbations, the reconstruction errors are also very small, making the detector fail to differentiate between normal and adversarial examples. To overcome this problem, we utilize the features extracted with the face recognition classifier f discussed in Section 2.2. Let $f(I)$ be the output of classifier f. If I is a normal image, and the reconstruction error $\|I - g \circ h(I)\|_p$ should be close to 0, and the probability mass functions $f(I)$ and $f(g \circ h(I))$ should also be similar. On the other hand, if I is an adversarial image, then $f(I)$ and $f(g \circ h(I))$ should also be significantly different. We use the Jensen-Shannon divergence to measure the difference between the distribution of semantic vectors of input and reconstructions:

$$\text{JSD}(P\|Q) = \frac{1}{2} D_{\text{KL}}(P\|M) + \frac{1}{2} D_{\text{KL}}(Q\|M) \tag{4}$$

where

$$D_{\text{KL}}(P\|Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)} \tag{5}$$

and

$$P = f(I), Q = f(g \circ h(I)), M = \frac{1}{2}(P + Q) \tag{6}$$

Jensen-Shannon divergence measures the similarity between two probability distribution. We observe from the experiment results that even though the reconstruction error for adversarial examples is very small, they are generated from a distribution that is significantly different from that of normal examples.

3.4 Adversarial Defense

One of a naive approaches of defense is to add a random noise on the input of adversarial example. Therefore, suppose the random noise is sampled from a Gaussian distribution, the defensed example is formulated as:

$$I_{df} = \texttt{Clip}(I_a + \alpha \cdot \mathcal{N}(0, 1)) \tag{7}$$

where I_{df} is the image after defense, `Clip` is a function to set the output inside the valid range and α is a hyperparameter to scale the noise. But this method only tries to disturb the pattern of the original perturbation and its performance is not guaranteed to be robust.

Instead, the autoencoder trained in Section 3.3 could be applied to generate normal reconstructions of adversarial examples. Given a normal example, the autoencoder would generate an output that looks similar to the input. On the other hand, given an adversarial example, the autoencoder would generate an output that is close to the space of normal examples. We will visualize the 3D face rendering results before and after defense in Section 4.5.

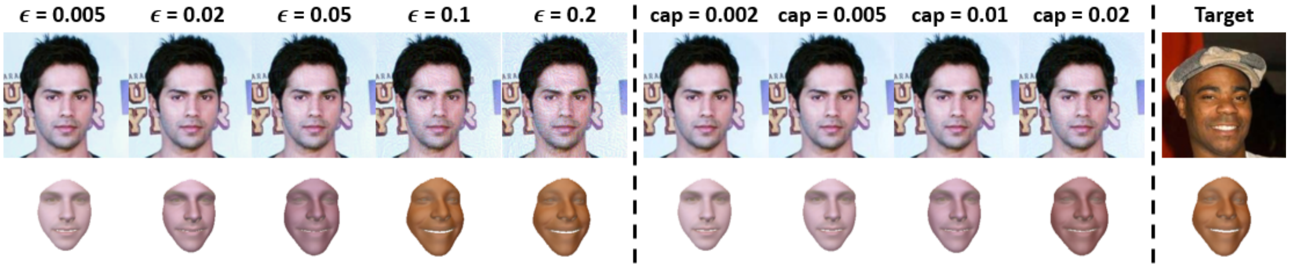


Figure 4: (Left) Adversarial examples and 3D reconstructions with different ϵ values in Eqn. (1) compared with target image. We choose 5 different ϵ values including 0.005, 0.02, 0.05, 0.1 and 0.2. From left to right, the perturbation is more and more perceivable but the 3D reconstruction becomes similar to target face. (Middle) Adversarial examples generated with different value of magnitude cap. The four examples are generated with cap value of 0.002, 0.005, 0.01 and 0.02, respectively. As noticed, the larger the magnitude value becomes, the better impersonation would be while the perturbation becomes more and more noticeable. (Right) Target image and its 3D reconstruction

Layer	Kernel	Stride	Output Size
ResNet-50	-	-	1024
Deconv-1	7×7	2	$7 \times 7 \times 256$
BN-1	-	-	$7 \times 7 \times 256$
Deconv-2	4×4	2	$14 \times 14 \times 128$
BN-2	-	-	$14 \times 14 \times 128$
Deconv-3	4×4	2	$28 \times 28 \times 128$
BN-3	-	-	$28 \times 28 \times 128$
Deconv-4	4×4	2	$56 \times 56 \times 64$
BN-4	-	-	$56 \times 56 \times 64$
Deconv-5	4×4	2	$112 \times 112 \times 32$
BN-5	-	-	$112 \times 112 \times 32$
Deconv-6	4×4	2	$224 \times 224 \times 3$
Sigmoid	-	-	$224 \times 224 \times 3$

Table 1: Specific architecture of adversarial detector, which is composed of a encoder and decoder. We apply the ResNet-50 [22]
as encoder and design a new network for decoder. More specifically, Deconv means the convolutional transpose layer and BN means the batch normalization layer. After Deconv and BN block, a ReLU activation function follows.

4 Experiment

In this section, we perform both qualitative analysis and quantitative studies to show that our adversarial attack and defense procedures are effective and robust. Based on our knowledge, our work is the first to consider the problem of adversarial attack on model-based encoder, so the comparable state-of-the-art methods are not available. However, we will conduct further experiments to show that our method yields much easier attacking strategies.

4.1 Implementation Details

MoFA. We apply Basel Face Model [23] as 3DMM. VGG-19 [17] is selected as the backbone network for MoFA encoder and the pre-trained weights are applied. The full model is trained end-to-end on the CelebA trainset [10], in which all faces are well-aligned and landmark positions are provided. During training, the Adadelta optimizer is applied with a learning rate of 0.1. A batch size of 5 is chosen and the model is trained for 200k iterations.

Adversarial attack. To optimize the perturbation matrix of our adversarial attack algorithm, we set the value of ϵ, λ_α, λ_δ, λ_β, λ_T and λ_γ in Eqn. (1) to be 0.05, 0.01, 0.01, 1.0, 0.05 and 1.5, respectively. We apply Adam algorithm [24] and learning rate is set be 0.1, and the perturbation is trained for 50 iterations.

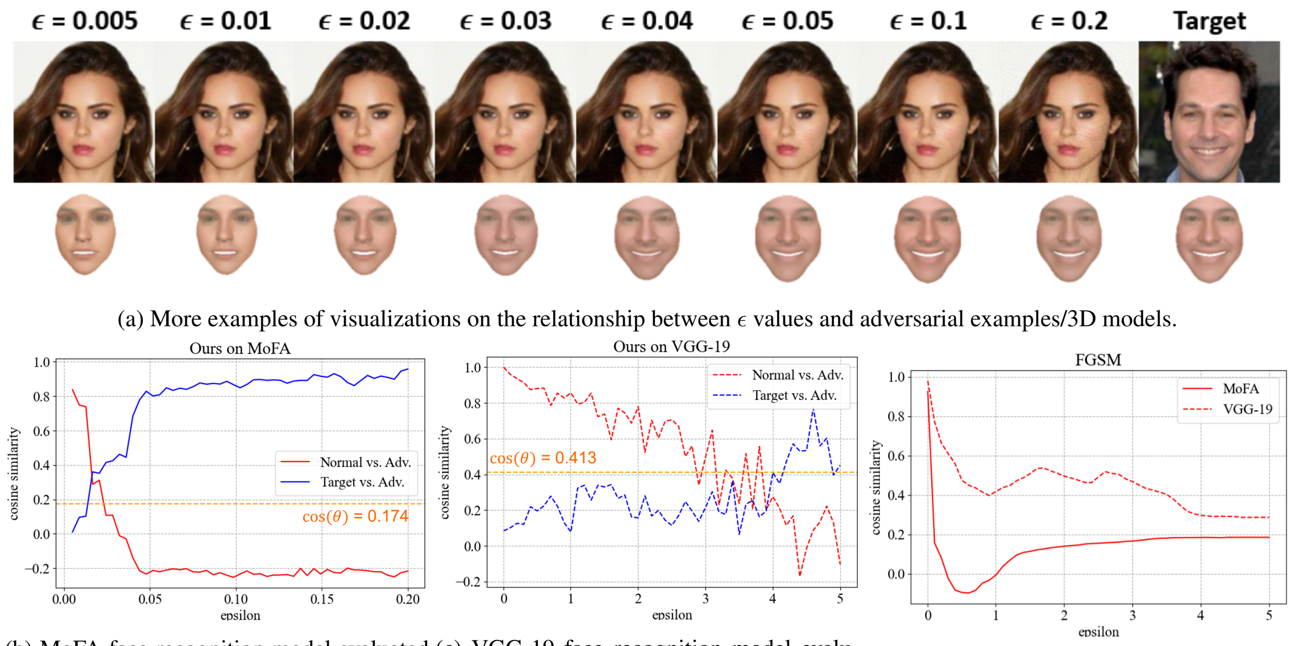
Detector. The details of our adversarial detector architecture is shown in Table 1. We fine-tuned the popular ResNet-50 [22] as the encoder and set the output size to be 1024. After that, we reshape the extracted representation into 2×2×256. Then we apply several convolutional transpose layer on the feature map and reconstruct the encoded representation back to original image. The whole detector network is trained with batch size of 128 and 30 epochs. The reconstruction loss function 2 is optimized with stochastic gradient descent with momentum, which learning rate is set to be 0.1 and decreased by 50% after every 10 epochs. The momentum is set to be 0.9.

4.2 Qualitative Analysis

Figure 3 visualizes some qualitative results of the comparison on the face reconstruction between original and adversarial examples. As it could be discovered, the 3D reconstruction of the adversarial image looks similar to that of the target image and difference between original and adversarial image is hard to perceive with naked eyes. As indicated by Algorithm 1, the appearance of face reconstruction is influenced by the value of ϵ (the proportion of perturbation added to original image) as well as $\{L, U\}$ (the minimum and maximum values of per-pixel magnitude). In Figure 4, we adjust both hyperparameters and it is not difficult to discover that as we increase the values of either ϵ or $\{L, U\}$, the perturbation will become more and more noticeable and the face reconstruction will visually looks much similar to that of the target face.

4.3 Quantitative Studies on Adversarial Attacks

To measure whether an optimized perturbation yields a successful attack to the MoFA architecture, we decided to train a face identity classifier on the semantic vectors extracted



(a) More examples of visualizations on the relationship between ϵ values and adversarial examples/3D models.

(b) MoFA face recognition model evaluated on our adversarial examples generation procedure

(c) VGG-19 face recognition model evaluated on our adversarial examples generation procedure

(d) Both MoFA and VGG-19 face recognition model evaluated on FGSM [11]

Figure 5: Cosine similarity evaluated by face recognition model with different ϵ values in Eqn. (1). The red line represents the cosine similarity between normal and adversarial images, whereas the blue line represents target and adversarial images. Solid lines are results that MoFA face recognition model is evaluated and 0.174 is the threshold selected, whereas dashed lines are results that VGG-19 face recognition model is evaluated and 0.413 is the corresponding threshold selected.

by the MoFA encoder. Although the main purpose of training the face autoencoder is to obtain a robust 3D face reconstruction, the latent representations still master some degree of discrimination [4]. Ideally, if the attack is successful, the original image and adversarial images should be verified by the classifier as different identities, whereas the target and adversarial image should be verified as the same. Therefore, we start to define the cosine similarity as a measurement of correlation between each pair of face images: $s(\mathbf{x}_i, \mathbf{x}_j), i \neq j$.

$$s(\mathbf{x}_i, \mathbf{x}_j) = \frac{\mathbf{x}_i^T \mathbf{x}_j}{\|\mathbf{x}_i\|_2 \cdot \|\mathbf{x}_j\|_2} \quad (8)$$

where s is the cosine similarity score and $\|.\|$ represents the L_2-norm. We randomly select a pair of original normal image and target face image, and evenly select 50 different ϵ values from 0.005 to 0.2. The cosine similarity scores under different ϵ values are calculated; the results are shown in Figure 5b. We discover an downward trend of cosine similarity between normal and adversarial image whereas an upward trend between target and adversarial image. When ϵ exceeds 0.05, lines in Figure 5b become flat and, shown in Figure 5a, the 3D reconstruction output barely changes.

In addition to training a face classifier on MoFA extracted features, a VGG-19 network [17] is also trained directly on the training set as a classifier for comparison. The range of selected ϵ values is expanded to [0.005, 5.0] in order to form effective comparison to MoFA. Shown in Figure 5c, our attack would change the decision of the classifier significantly when ϵ is greater than 4.0. This observation shows that MoFA architecture is much easier to attack. To further prove this statement, the FGSM method [11] is also applied to attack both MoFA and VGG-19. Shown in Figure 5d, if a small ϵ (<1) is applied on the sign of the gradient of classification loss, the cosine similarity between normal and adversarial examples of MoFA features are significantly lower than that of VGG-19.

4.4 Results on Adversarial Detection

As discussed in Section 3.3, we propose to train a separate convolutional neural network based autoencoder as our adversarial detector. Also as discussed before, the Jensen-Shannon divergence is a better protocol to evaluate the performance of the autoencoder on both normal and adversarial examples. Therefore in Table 2, we compare the detection accuracy between different protocols including L_1, L_2, L_∞ norms and Jensen-Shannon divergence. Apart from that, we also compare the contributions of different ϵ values. It could be observed from Table 2 that when smaller ϵ values are chosen, L_p-norms report accuracy scores around to 50%. This illustrates that the reconstruction error is relatively small so that it is hard to pick the threshold value. However, the Jensen-Shannon divergence could effectively measure the difference between probability distribution, and therefore, the normal and adversarial samples are effectively separated as they come from different data generation processes.

4.5 Adversarial Defense

The autoencoder trained in Section 4.4 could be applied to move the adversarial examples back to the normal sample space. In Figure 6, we visualize the 3D reconstruction

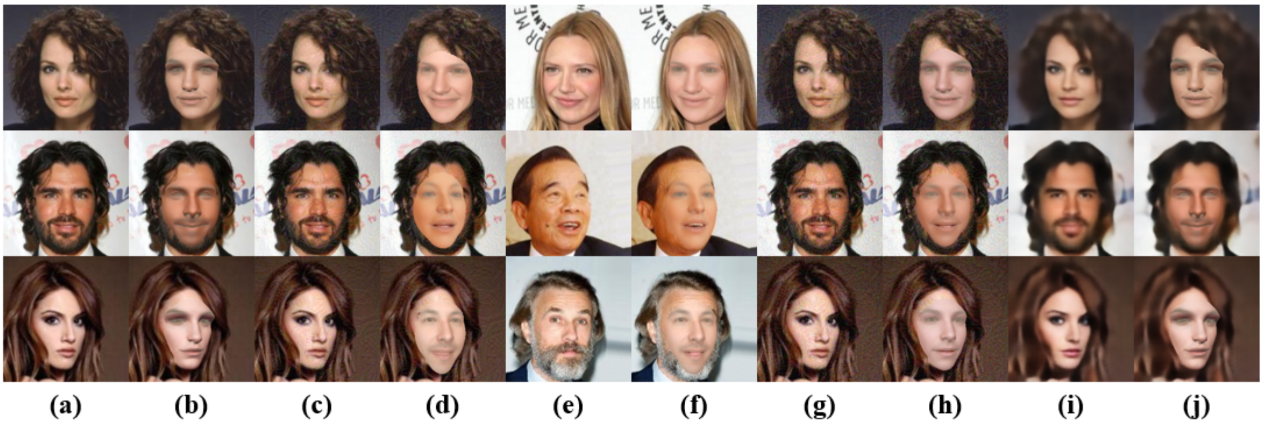


Figure 6: An illustration of the adversarial defense effect on perturbation attacks on CelebA testset [10]. (a) Original normal examples. (c) Adversarial examples. (e) Target examples. (g) Defensed examples added with Gaussian Noise. (i) Defensed examples output by CNN-based autoencoder. Columns (b), (d), (f), (h) and (j) are the MoFA generated 3D reconstructions of the 2D images in the previous column.

	L_1	L_2	L_∞	JS
$\epsilon = 0.005$	50.21%	50.09%	50.07%	71.13%
$\epsilon = 0.02$	51.03%	50.51%	50.16%	88.46%
$\epsilon = 0.05$	53.94%	52.30%	50.49%	96.32%
$\epsilon = 0.2$	74.29%	67.45%	54.24%	97.94%

Table 2: Caption

model of the adversarial examples (column d) and the defensed examples (column j). The results show that the autoencoder is capable of reforming the adversarial images into the space of normal samples so that their face-rendering results are close to those of original normal samples. Also shown in column h of Figure 6, the defense method based on random noise has the potential to recover the shape and expression characteristics, but others including pose, illumination and color are hard to impersonate. In comparison, our defensed method based on deep convolutional network is capable of restoring all semantic information of original normal faces.

Conclusions

In this paper, we propose an adversarial attack method on model-based face autoencoder through iterative perturbation optimization. We show that such attacks are easier than attacking traditional classifiers by training a face recognition network. Another CNN-based autoencoder is trained for adversarial detection and we prove with experimental results that the Jensen-Shannon divergence is the robust measurement. The same autoencoder is also shown to be effective in adversarial defense so that the attacked examples are moved close to the space of normal samples.

References

[1] Ayush Tewari, Michael Zollhoefer, Florian Bernard, Pablo Garrido, Hyeongwoo Kim, Patrick Perez, and Christian Theobalt. High-fidelity monocular face reconstruction based on an unsupervised model-based face autoencoder. *IEEE transactions on pattern analysis and machine intelligence*, 2018.

[2] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3d faces. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 187–194, 1999.

[3] Elad Richardson, Matan Sela, and Ron Kimmel. 3d face reconstruction by learning from synthetic data. In *2016 fourth international conference on 3D vision (3DV)*, pages 460–469. IEEE, 2016.

[4] Anh Tuan Tran, Tal Hassner, Iacopo Masi, and Gérard Medioni. Regressing robust and discriminative 3d morphable models with a very deep neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5163–5172, 2017.

[5] Matan Sela, Elad Richardson, and Ron Kimmel. Unrestricted facial geometry reconstruction using image-to-image translation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1576–1585, 2017.

[6] Pedro Tabacof, Julia Tavares, and Eduardo Valle. Adversarial images for variational autoencoders. *arXiv preprint arXiv:1612.00155*, 2016.

[7] Shixiang Gu and Luca Rigazio. Towards deep neural network architectures robust to adversarial examples. *arXiv preprint arXiv:1412.5068*, 2014.

[8] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[9] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[10] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

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

[12] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

[13] Florian Tramèr, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. *arXiv preprint arXiv:1705.07204*, 2017.

[14] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K Reiter. Accessorize to a crime: Real and stealthy attacks on state-of-the-art face recognition. In *Proceedings of the 2016 acm sigsac conference on computer and communications security*, pages 1528–1540, 2016.

[15] Yinpeng Dong, Hang Su, Baoyuan Wu, Zhifeng Li, Wei Liu, Tong Zhang, and Jun Zhu. Efficient decision-based black-box adversarial attacks on face recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7714–7722, 2019.

[16] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K Reiter. A general framework for adversarial examples with objectives. *ACM Transactions on Privacy and Security (TOPS)*, 22(3):1–30, 2019.

[17] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[19] Justus Thies, Michael Zollhofer, Marc Stamminger, Christian Theobalt, and Matthias Nießner. Face2face: Real-time face capture and reenactment of rgb videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2387–2395, 2016.

[20] Pablo Garrido, Michael Zollhöfer, Dan Casas, Levi Valgaerts, Kiran Varanasi, Patrick Pérez, and Christian Theobalt. Reconstruction of personalized 3d face rigs from monocular video. *ACM Transactions on Graphics (TOG)*, 35(3):1–15, 2016.

[21] Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On detecting adversarial perturbations. *arXiv preprint arXiv:1702.04267*, 2017.

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[23] Pascal Paysan, Reinhard Knothe, Brian Amberg, Sami Romdhani, and Thomas Vetter. A 3d face model for pose and illumination invariant face recognition. In *2009 Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance*, pages 296–301. Ieee, 2009.

[24] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.