Adversarial Learning with Margin-based Triplet Embedding Regularization

Yaoyao Zhong, Weihong Deng*
Beijing University of Posts and Telecommunications
{zhongyaoyao, whdeng}@bupt.edu.cn

Abstract

The Deep neural networks (DNNs) have achieved great success on a variety of computer vision tasks, however, they are highly vulnerable to adversarial attacks. To address this problem, we propose to improve the local smoothness of the representation space, by integrating a margin-based triplet embedding regularization term into the classification objective, so that the obtained model learns to resist adversarial examples. The regularization term consists of two steps optimizations which find potential perturbations and punish them by a large margin in an iterative way. Experimental results on MNIST, CASIA-WebFace, VGGFace2 and MS-Celeb-1M reveal that our approach increases the robustness of the network against both feature and label adversarial attacks in simple object classification and deep face recognition. The code is available at https://github.com/zhongyy/Adversarial_MTER.

1. Introduction

The Deep neural networks (DNNs) have achieved great success [20, 38, 15, 17], significantly improving the development of a variety of challenging applications such as deep face recognition [7, 36, 47, 25, 45, 9, 46, 24] and automatic driving [3, 8].

However, contradictions between the vulnerability of DNNs and the demand of security have become increasingly obvious. On one hand, DNNs are vulnerable. Previous works have discovered, with elaborate strategies, DNNs can be easily fooled by test images with imperceptible noise [42]. This type of images is named as adversarial examples. Moreover, adversarial examples are transferable in different models [31, 10], which means black-box attacks can be launched without knowing the details of target models (*e.g.* architectures, parameters and defense methods). On the other hand, the demand of security arises in safety crucial domains driven by DNNs. Adversarial examples can attack physical-world DNNs applications [21]. For instance, DNNs in an automatic vehicle system can be confused by carefully manipulated road signs [12], and DNNs in a face recognition system are susceptible to feature level adversarial attacks [35, 1, 39].

The existence of adversarial examples has given birth to a variety of researches on adversarial defenses. One straightforward defense strategy is to increase the robustness of the model by injecting adversarial examples in the training process [42, 22, 44], which is essentially a regularization of the training data augmentation. This strategy is effective in close-set classification like object classification, while may not suitable in open-set settings like deep face recognition where training categories could be in million level. Another strategy is to detect adversarial examples at inference time [27, 50, 39]. This strategy is appropriate for both open-set and close-set classification settings, while it can be easily broken in the white-box setting where the specific defense method is known [5].

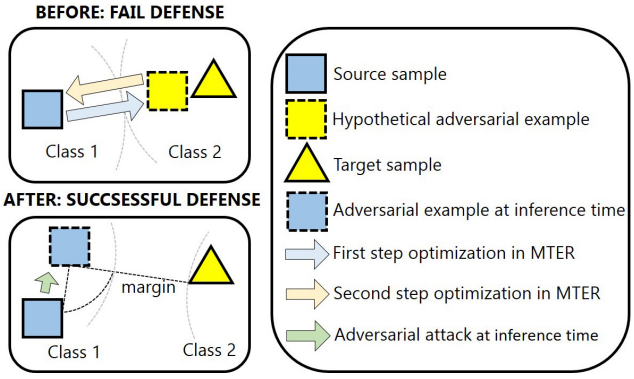

Figure 1. Schematic illustration of the defense before MTER training (top) versus after training (bottom). Arrows indicate the gradients arising from the optimization of the cost function. The same color represents the same predict class.

In this paper, we propose a margin-based triplet embedding regularization (MTER) method to train DNNs with robustness. Our intuition is that by training a model to improve the local smoothness of embedding space with fewer singular points, it will be more resistant to adversarial examples. The regularization term consists of two steps opti-

*Corresponding Author

mizations in an iterative way which first find potential perturbations in the embedding space, and then punish them by a large margin. A schematic illustration of the defense before MTER versus after MTER is shown in Figure 1. Specifically, in the embedding space, a potential attack is generated from a source to a target. We improve the robustness by encouraging the hypothetical attacks to gradually approach the source class, meanwhile move far away from all the other target classes. The result of an embedding space visualization experiment is shown in Figure 2. In the optimization, the large margin is not trivial, which strictly ensures the inter-class distance and the intra-class smoothness in the embedding space. Our contributions are as follows:

1. We propose to improve the robustness of DNNs by smoothing the embedding space, which is appropriate for DNNs trained in both open-set and close-set classification settings.

2. We introduce the large margin into adversarial learning, which further guarantees the inter-class distance and the intra-class smoothness in the embedding space, therefore improves the robustness of DNNs.

3. Experimental results on MNIST [51], CASIA-WebFace [54], VGGFace2 [4] and MS-Celeb-1M [53] demonstrate the effectiveness of our methods in simple object classification and deep face recognition.

2. Related work

2.1. Adversarial attacks

Szegedy *et al*. [42] first find that they can cause DNNs to misclassify images by a certain hardly perceptible perturbation which is generated using a box-constrained L-BFGS method. Compared with the L-BFGS attack [42], Goodfellow *et al*. [14] propose a more time-saving and practical method "fast" method (FGSM) to generate adversarial examples by performing one-step gradient update along the direction of the sign of gradient at each pixel.

Kurakin *et al*. [21] introduce a straightforward method called "basic iterative" method (BIM), to extend the "fast" method (FGSM) [14] by applying it multiple times with small step size and clip pixel values after each step to ensure the L_∞ constraint. Moreover, to generate adversarial examples of a specific desired target class, Kurakin *et al*. [21] introduce the iterative least likely method. Iterative methods could attacks DNNs with higher rate compared with the fast method in the same constraint level [21]. Similarly, another iterative attack method proposed by Moosavi-Dezfooli *et al*. [29], called Deepfool, is also reliable and efficient. With linear approximation, Deepfool try to generate the minimal perturbation in each step by moving towards the lineared decision boundary [29]. Based on Deepfool [29], Moosavi-Dezfooli *et al*. [28] propose image-agnostic adversarial attacks, which could fool DNNs with a universal perturbation on images with high probability.

Apart from the generation of adversarial examples, there are also works focus on the transferability of adversarial examples [31, 10, 49], adversarial examples in physical world [21, 12], and in specific tasks such as face recognition system [37, 13].

2.2. Defense methods

The defense methods can be classified into two categories, one is to improve the robustness of DNNs, the other one is to detect adversarial examples at inference time [27, 50, 39]. We mainly discuss the former type, which is more related to our work.

Network distillation [2, 16] is originally proposed to reduce the model size. Papernot *et al*. [32] introduce distillation as a defense method to improve the robustness by feeding back the class probability to train the original model.

Adversarial training could provide regularization to DNNs [14]. Goodfellow *et al*. [14] first propose adversarial training which could increase the robustness by injecting adversarial examples in the training process. Then adversarial training is applied and analyzed in large training dataset ImageNet [22].

Although the success of adversarial training on white-box defenses, the defense against black-box attacks is still a problem, due to the transferability of adversarial examples. To deal with the transferred black-box attacks, Tramer *et al*. [44] introduce ensemble adversarial training technique transferring one-step adversarial examples from other training models, while Na *et al*. [30] propose cascade adversarial trained transferring iterative attacks from already trained model.

Around the same time, Dong *et al*. [11] and Na *et al*. [30] minimize both the cross-entropy loss and the distance of original and adversarial embedding to improve the vanilla adversarial training. They are the most related work to ours. However, our method mainly differs from them in two aspects: (1) Our MTER method is a straightforward and thorough feature adversary which will not be limited by number of training categories, therefore is also appropriate for open-set classification. (2) We introduce the large margin into adversarial learning, which guarantees not only the intra-class smoothness but also the inter-class distance in the embedding space.

3. Margin-based Regularization

Our purpose is training a DNN to have smooth representations with fewer singular points. Therefore we consider a regularization term which exploits the vulnerability and further fix them in an iterative way.

Algorithm 1 Margin-based triplet embedding regularization (MTER)

Input:

Training set $D = \{x^{(i)}, y^{(i)} \in \{1, 2, ..., C\}\}$, model parameter θ and hyperparameter margin m, mini-batch size K.

Output:

The final model parameter θ.

Initialization at the beginning of an epoch:

// Constructing the source images queue Qs and the Qt.

Qs=Qt={}.

Random select $\frac{C}{2}$ categories in $\{1, 2, ..., C\}$ denoted as source S, the complementary set is target T.

Qs.append($\{x^i | y^i \in S\}$); Qt.append($\{x^i | y^i \in T\}$).

shuffle Qs and Qt.

Optimization in an epoch:

while Qt is not empty and Qs is not empty **do**

Take out a mini-batch Bs and Bt with $\frac{K}{2}$ samples respectively in Qs and in Qt.

$\Delta x^{(s,t)} \leftarrow$ Calculate perturbations (3) in an iterative way (5) on the batch Bs and Bt, based on the current model θ.

$\theta \leftarrow \nabla(L_{ori}$ (2) $+ R_{MTER}$ (10)) on batch Bs, Bt and $\Delta x^{(s,t)}$.

end while

3.1. Exploitation the Vulnerability

First we consider the vulnerability exploitation. We start from some notations. Let $D = \{x^{(i)}, y^{(i)}\}$ denote a labeled dataset where $x^{(i)}$ and $y^{(i)} \in \{1, 2, ..., C\}$ respectively denote an input image and the corresponding label. A DNN can be formulated in a chain

$$F_\theta{}^{(n)}(x) = f^{(n)}(...(f^{(2)}(f^{(1)}(x))), \tag{1}$$

parameterized by θ. The network is originally trained on a dataset D by cross entropy

$$L_{ori} = \arg\min_\theta H(F_\theta{}^{(n)}(x^{(i)}), y^{(i)}) = -\frac{1}{K}\sum_{i=1}^{K} \log p(y^{(i)}|x^{(i)}), \tag{2}$$

where H giving the sum of the cross entropies between the predictions $F_\theta{}^{(n)}(x^{(i)})$ and the labels $y^{(i)}$.

Given a trained DNN, a source image and a target image, denoted as $\{x^{(s)}, x^{(t)}\}$, where $y^{(s)} \neq y^{(t)}$, we could find small perturbations $\Delta x^{(s,t)}$ to the source image $x^{(s)}$ that produce an internal representation that is remarkably similar to that of the target image $x^{(t)}$ [35]. The vulnerability exploitation in embedding space can be described as:

$$\Delta x^{(s,t)} = \arg\min_{\Delta x^{(s,t)}} \left\| E(x^{(s)} + \Delta x^{(s,t)}) - E(x^{(t)}) \right\|_2^2, \tag{3}$$

subject to

$$\left\| \Delta x^{(s,t)} \right\|_\infty < \varepsilon. \tag{4}$$

$E(x)$ is the deep representation in the embedding space, which is normalized to unit length from $F_\theta{}^{(n-1)}(x)$. $F_\theta{}^{(n-1)}(x)$ is the function from the image x to its representation at the $n-1$ layer. ε limits the maximum deviation of the perturbation.

For computational efficiency, we adopt the direction defined by the gradient of the metric loss function and form adversarial perturbations in an iterative way, referred to as iterative feature target gradient sign method (IFTGSM):

$$\begin{aligned} &\Delta x_0^{(s,t)} = 0, \\ &x^{(s)} + \Delta x_{N+1}^{(s,t)} = C_{x^{(s)},\varepsilon}(x^{(s)} + \Delta x_N^{(s,t)} + \\ &\qquad sign(\nabla_{x^{(s)}+\Delta x_N^{(s,t)}} \left\| E(x^{(s)} + \Delta x_N^{(s,t)}) - E(x^{(t)}) \right\|_2^2)), \end{aligned} \tag{5}$$

where

$$C_{x,\varepsilon}(x') = \min(255, x + \varepsilon, \max(0, x - \varepsilon, x')), \tag{6}$$

the iteration is chosen heuristically $\min(\varepsilon + 4, 1.25\varepsilon)$. $\Delta x^{(s,t)}$ can also be generated using a fast method, referred to as fast feature target gradient sign method (FFTGSM). We formulate it as follows:

$$\Delta x^{(s,t)} = \varepsilon sign(\nabla_{x^{(s)}} \left\| E(x^{(s)}) - E(x^{(t)}) \right\|_2^2)). \tag{7}$$

We will use this fast attack method in the experiment on face recognition in Section 4.5.

3.2. Fix the Vulnerability

Our target is to improve the robustness of DNNs without modifying their architectures. The aforementioned vulnerability attacks a DNN by finding singular points in the internal representation space of a DNN. Considering the existence of the vulnerability, we find it is possible that we smooth the embedding space by jointly optimizing the original cross entropy and a large-margin based triplet distance constraint as a regularization term.

With a source and a target image $\{x^{(s)}, x^{(t)}\}$, consider a triplet $t := \{E(x^{(s)}), E(x^{(s)} + \Delta x^{(s,t)}), E(x^{(t)})\}$, where $\Delta x^{(s,t)}$ is the aforementioned perturbation. Ideally, for all triplets t which are generated in the training set, we would like the following constraint to be satisfied:

$$\left\| E(x^{(s)}+\Delta x^{(s,t)}) - E(x^{(s)}) \right\|_2^2 < \left\| E(x^{(s)}+\Delta x^{(s,t)}) - E(x^{(t)}) \right\|_2^2. \tag{8}$$

However, due to the first step optimization in objective (3), the actual situation at a certain moment in the training process may be:

$$\left\| E(x^{(s)}+\Delta x^{(s,t)}) - E(x^{(s)}) \right\|_2^2 > \left\| E(x^{(s)}+\Delta x^{(s,t)}) - E(x^{(t)}) \right\|_2^2. \tag{9}$$

Therefore, the vulnerability exploitation and fixing together constitute two optimization steps in the adversarial

learning, which strive to attack each other but also together improve the robustness of DNNs gradually. Formally, we define the margin-based triplet embedding regularization (MTER) as follows:

$$R_{MTER}=\frac{1}{K}\sum_{y^{(s)}\neq y^{(t)}}\max(0,m+\left\|E(x^{(s)}+\Delta x^{(s,t)})-E(x^{(s)})\right\|_2^2 - \left\|E(x^{(s)}+\Delta x^{(s,t)})-E(x^{(t)})\right\|_2^2), \tag{10}$$

where the $\Delta x^{(s,t)}$ is obtained and upgraded by objective (3), and the parameter m is the margin. In practice, we apply the vulnerability exploitation and fixing in an iterative way, which is precisely described in Algorithm 1. Parameter m controls that the similarity between the source image and the perturbed image should be much higher than that between the perturbed image and the target image. m is chosen based on the training dataset and the model capacity. We will discuss the parameter m in the following ablation study in Section 4.3.

4. Experiment

4.1. Experiment on Simple Image Classification

In this section, we first analyze the effect of the margin-based triplet embedding regularization (MTER) method on MNIST [51], a simple image classification task. We train ResNet [15] models using original training loss functions, adversarial training, and our MTER method, respectively. We test different models assuming that the adversary knows the classification algorithm, model architecture and parameters, because the reliability of a model could be demonstrated if a model is robust in the white-box setting.

We first give a brief description of the adversarial training method [22] and attack methods FGSM [14], BIM [21], FTGSM [21], ITGSM [21] which we will test and compare with our method.

Fast gradient sign method (FGSM) [14] generates adversarial examples by perturbing inputs in a manner that increases the sign of the gradients of the original loss function w.r.t. the input image $x^{(i)}$:

$$x_{adv}^{(i)}=x^{(i)}+\varepsilon sign(\nabla_{x^{(i)}}H({F_\theta}^{(n)}(x^{(i)}),y^{(i)})), \tag{11}$$

where H giving the cross entropy between the predictions ${F_\theta}^{(n)}(x^{(i)})$ and the labels $y^{(i)}$, ε limits the maximum deviation of the perturbation.

Basic iterative method (BIM) [21] is a modification of the FGSM [14] by applying it multiple times:

$$x_{adv,0}^{(i)}=x^{(i)}, x_{adv,N+1}^{(i)}= C_{x^{(i)},\varepsilon}(x_{adv,N}^{(i)}+\alpha sign(\nabla_{x_{adv,N}^{(i)}}H({F_\theta}^{(n)}(x_{adv,N}^{(i)}),y^{(i)}))), \tag{12}$$

where $\alpha=1$ is used, $C_{x^{(i)},\varepsilon}$ is referred to as equation (6), and number of iterations is $\min(\varepsilon+4,1.25\varepsilon)$.

Compared with BIM [21], iterative target gradient sign method (ITGSM) [21] leads the model to misclassify an image as another target category:

$$x_{adv,0}^{(i)}=x^{(i)}, x_{adv,N+1}^{(i)}= C_{x^{(i)},\varepsilon}(x_{adv,N}^{(i)}-\alpha sign(\nabla_{x_{adv,N}^{(i)}}H({F_\theta}^{(n)}(x_{adv,N}^{(i)}),y_t^{(i)}))), \tag{13}$$

where $y_t^{(i)}$ is the target label we would like the model to predict, $y_t^{(i)}\neq y^{(i)}$. Also, the target attack can be launched in a Fast style, referred to as fast target gradient sign method (FTGSM) [21]:

$$x_{adv}^{(i)}=x^{(i)}-\varepsilon sign(\nabla_{x^{(i)}}H({F_\theta}^{(n)}(x^{(i)}),y_t^{(i)})). \tag{14}$$

We use ResNet-18 [15] for training. The models are trained from scratch. There is no data augmentation for both datasets. $\varepsilon=0.3*255$ is applied for MNIST [51]. In ITGSM [21] attack, the least likely class is used as the target class. We use two type of original loss functions Softmax and ArcFace [9], which is a type of large-margin loss and first used in deep face recognition. The feature scale is set to 10 and angular margin is set to 0.4 in ArcFace [9]. We improve the robustness of the two type of original loss respectively using adversarial training [22] and our method. The margin m in our MTER method is set to 0.2. We follow the adversarial training implemented in Kurakin *et al.* [22], which increases the robustness by replacing half of the mini-batch clean examples with their adversarial examples into the training process. More specifically, we generate adversarial examples using FGSM [14] perturbations with respect to predicted rather than true labels following works [30, 34], to prevent "label leaking" [22] where the model tend to learn to classify adversarial examples more accurately than regular examples. The relative weight of adversarial examples in the loss is set to 0.3 following [22].

The results are shown in Table 1. As shown in the table, even though adversarial training is done with the predicted label, the label leaking phenomenon [22] still happens. Our MTER method improves the robustness of the original models using different loss functions under FGSM [14], BIM [21] and ITGSM [21] attacks. For models trained with Softmax, our method sacrifices a little performance on the clean images. while for models trained with ArcFace [9], a large margin loss function, it even improves the accuracy on clean images. Besides, it outperforms adversarial training method under BIM [21] and ITGSM [21] attacks. Even though we did not use these type of adversarial examples for data augmentation in training like adversarial training method, our method could still gain robust improvements under unknown attacks. This indicates that our method can improve the robustness of models on simple image classification like MNIST [51].

Method	Clean	FGSM [14]	BIM [21]	ITGSM [21]
Softmax loss	99.6	10.4	0.0	6.0
Adversarial training	99.6	99.9	3.1	47.2
Softmax+MTER (ours)	99.5	96.8	98.7	95.1
ArcFace Loss	99.5	28.6	1.7	24.7
Adversarial training	99.5	99.5	30.5	71.3
ArcFace Loss+MTER (ours)	99.6	96.6	98.0	95.3

Table 1. MNIST [51] test results (%) for Resnet-18 [15] models ($\varepsilon = 0.3 * 255$ at test time). The higher the accuracy is, the more robust is the target model.

4.2. Embedding Space Visualization

MNIST [51], the popular and sweet dataset is used for embedding space visualization. We use the ResNet-18 [15] by changing the original fully connected layer to two fully connected layer and then modifying the embedding dimension to 2. We retrain networks on MNIST [51] with Softmax and Softmax combined our MTER method, respectively. Then we use the clean examples of test dataset for visualization. Besides, for each class, we randomly choose a test sample and generate adversarial examples of it using BIM [21] for visualization.

The results are shown in Figure 2. The round represents the clean examples of MNIST [51] test set. The triangle represents BIM [21] adversarial examples, which we draw from $\varepsilon = 0$ to $\varepsilon = 76(\approx 0.3 \times 255)$ for one sample image per each class. We can observe that with our MTER method, the adversarial examples are close to clean examples and distributed in the original class region in the embedding space. The inter-class margin is enlarged and the intra-class smoothness is improved, which guarantee the robustness of the model.

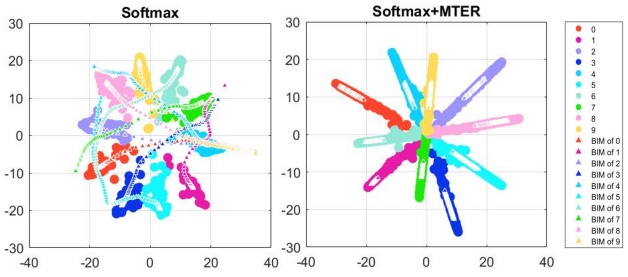

Figure 2. Embedding space visualization on ResNet-18 [15] which is modified embedding dimension to 2. Models are trained on MNIST [51] with Softmax and Softmax combined with MTER method, respectively. The round represents the clean examples of MNIST [51] test set. The triangle represents BIM [21] adversarial examples, which we draw from $\varepsilon = 0$ to $\varepsilon = 76(\approx 0.3 \times 255)$ for one sample image per each class.

4.3. Analysis on Margin m

We use MNIST [51] to conduct adversarial study [42, 14, 30, 52], to further analyze our MTER method. The only hyperparameter in our method is the margin m. So we would like to explore the influence of m and give advice on choice of it under different settings.

First we train LeNet-5 [23] and ResNet-18 [15] by varying m in $\{0.2, 0.45, 0.7, 0.95, 1.2, 1.4\}$. Then we test these models using aforementioned attack methods, *i.e.* FGSM [14], BIM [21] and ITGSM [21]. The results are illustrated in Figure 3, from which we can discover significant difference between the two type of models, LeNet-5 [23] and ResNet-18 [15]. Although the two type of models both could obtain good test accuracy on MNIST [51], the accuracy for LeNet-5 [23] and ResNet-18 [15] is 99.2% and 99.6% respectively. However, for LeNet-5 [23], along with the increase of margin m, the robustness to different attacks improves gradually and the accuracy on clean images decreases slightly. While for ResNet-18 [15], both the accuracy on clean test set and the robustness against adversarial attacks, have reached a relative high level and remain unchanged when the margin increases.

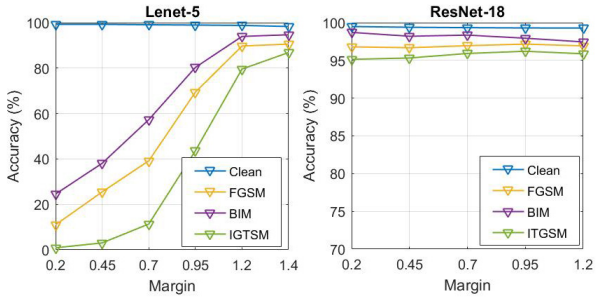

Figure 3. LeNet-5 [23] and ResNet-18 [15] trained using MTER by varying margin m in $\{0.2, 0.45, 0.7, 0.95, 1.2, 1.4\}$. The two type of models both could obtain good test accuracy on MNIST [51]. However, for LeNet-5 [23], along with the increase of margin m, the robustness to attacks improves gradually. While for ResNet-18 [15], the robustness has reached a relative high level and remains unchanged when the margin increases.

Furthermore, we fix the margin $m = 0.2$. We use this relative small margin because we would like to observe the performance of different networks under relaxed state of our method. We train models under Softmax and our MTER method respectively, using different architectures, *e.g.* LeNet-5 [23], ResNet-6 [15], ResNet-8 [15], ResNet-10 [15], ResNet-18 [15], ResNet-34 [15] and ResNet-50 [15]. We still test these trained models under the three aforementioned attacks. The results are shown in Figure 4. In the figure 4, for models trained using Softmax, the accuracy on the clean images increases when the model size becomes bigger (from LeNet-5 [23] to ResNet-50 [15]). For models trained using Softmax combined with MTER ($m = 0.2$), the accuracy on the clean images and three adversarial examples increases when the model size becomes bigger (from LeNet-5 [23] to ResNet-10 [15]), while remain relatively stable when the model size reaches a certain value

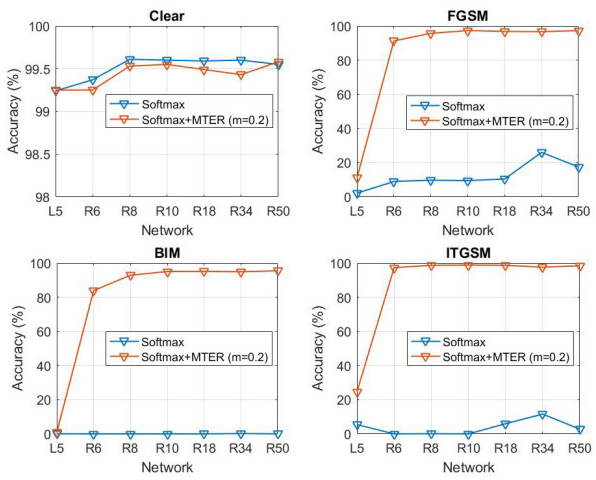

Figure 4. Accuracy on clean images, and adversarial examples generated by FGSM [14], BIM [21], and ITGSM [21] of models with different architectures. "L5","R6","R8","R10","R18","34" and "R50" on the x-axis denote LeNet-5 [23], ResNet-6 [15], ResNet-8 [15], ResNet-10 [15], ResNet-18 [15], ResNet-34 [15] and ResNet-50 [15]. For models trained using Softmax combined with MTER($m = 0.2$), the accuracy on the clean images and three adversarial examples increases when the model size becomes bigger (from LeNet-5 [23] to ResNet-10 [15]), while remain relatively stable when the model size reaches a certain value (after ResNet-10 [15])

(after ResNet-10 [15]).

We then infer that a specific classification task need a certain amount of computing power to fit the clean set and its augmentation set, *e.g.* adversarial examples. It is easy for our method to push a large model to learn both the clean images and the adversarial examples under a relative relaxed state (small m), while it will not work for a relative small model with less computing power. Therefore we recommend to increase the margin m to push the "lazy" model fighting with adversarial examples but sacrificing a little performance on the original dataset, if a small model is used and is still concerned about robustness.

4.4. Black-box Attack Analysis

We also use MNIST [51] for analysis of adversarial attacks and defense under black-box settings. We report black box attack accuracy of the adversarial examples generated from a source network and tested on another target network. In our experiment, both the source network and the target network are trained with adversarial learning methods in different architectures. Specifically, we use four models, the architecture of them are LeNet-5 [23] or ResNet-18 [15], and training methods are adversarial training [22] or our MTER method. The adversarial examples from the source models are generated by FGSM [14] or BIM [21] with $\varepsilon = 0.3 * 255$.

The results on FGSM [14] and BIM [21] are shown in Table 2 and Table 3, respectively. The row and the column denote the source and target model respectively. The LeNet-5 [23] is denoted as "L5", and ResNet-18 [15] is denoted as "R18". "ADV" is a shorthand of adversarial training [22]. "MTER-R18" means this ResNet-18 [15] model is trained supervised by Softmax combined with our MTER method. The higher the accuracy is, the more robust is the target model.

Source \ Target	ADV-L5	MTER-L5	ADV-R18	MTER-R18
ADV-L5	\	92.0	95.6	97.3
MTER-L5	85.3	\	94.5	97.7
ADV-R18	89.8	94.9	\	97.4
MTER-R18	45.2	94.2	94.1	\

Table 2. MNIST [51] test result (%) on FGSM [14] ($\varepsilon = 0.3 * 255$) adversarial examples under black box settings. The row and the column denote the source and target model respectively. The LeNet-5 [23] is denoted as "L5", and ResNet-18 [15] is denoted as "R18". "ADV" is a shorthand of adversarial training [22]. The higher the accuracy is, the more robust is the target model.

Source \ Target	ADV-L5	MTER-L5	ADV-R18	MTER-R18
ADV-L5	\	90.1	89.1	97.3
MTER-L5	79.3	\	92.8	97.5
ADV-R18	88.2	94.9	\	97.3
MTER-R18	84.0	94.6	95.6	\

Table 3. MNIST [51] test result (%) on BIM [21] ($\varepsilon = 0.3 * 255$) adversarial examples under black box settings. The row and the column denote the source and target model respectively. The LeNet-5 [23] is denoted as "L5", and ResNet-18 [15] is denoted as "R18". "ADV" is a shorthand of adversarial training [22]. The higher the accuracy is, the more robust is the target model.

As shown in the Table 2 and Table 3, if the source models and the architecture of targets models are both equal, the accuracy of target models trained using MTER method is higher than that of adversarial training method [22]. This phenomenon indicates our MTER method show better robust performance under black box attack scenario, on both FGSM [14] and BIM [21] adversarial examples, even if the adversarial training have used FGSM [14] examples for augmentation. Besides, we find that the ResNet-18 [15] models are more robust than LeNet-5 [23] models, while adversarial examples generated from LeNet-5 [23] are more aggressive and have better transferability than those of ResNet-18 [15].

4.5. Experiment on Deep Face Recognition

In a deep face recognition system, an adversary may try to disguise a face as an authorized user. We simulate this scenario using state-of-art face recognition models and test our MTER method. Deep face recognition is a open-set problem, which indicates the training identities and the test identities are usually different. We don't directly classify an identity by end-to-end classification probability, but use a DNN as a deep feature extracter and compare deep features

to distinguish faces.

Training datasets. In the experiment, the training datasets are CASIA-WebFace [54], VGGFace2 [4] and MS1M-IBUG [53]. The CASIA-WebFace [54] dataset is the first widely used large-scale training dataset in deep face recognition, containing 0.49M images from 10,575 celebrities. VGGFace2 [4] is a large-scale dataset containing 3.31M images from 9131 celebrities. There are diverse and abundant images in VGGFace2 [4], which have large variations in pose, age, illumination, ethnicity and profession. MS1M-IBUG [53] (referred to as MS1M [53]) is a refined version of MS-Celeb-1M dataset [53], which is public available and widely used. The original MS-Celeb-1M dataset [53] contains about 10k celebrities with 10M images. MS1M [53] is refined by Deng *et al*. [9] to decrease the noise and finally contains 3.8M images of 85,164 celebrities.

Network settings. For all the embedding networks, we adopt the ResNet-50 [15], but make changes as [9] which apply the "BN [19]-Dropout [40]-FC-BN" sturcture to get the final 512-D embedding feature. For image preprocessing, the images are cropped and aligned to the normalized 112×112 face following [9]. In the training process, the original models are supervised by an effective loss function ArcFace [9], which has been widely accepted by industry. The feature scale is set to 64 and angular margin is set to 0.5 for CASIA-WebFace [54] and MS1M [53], and 0.3 for VGGFace2 [4] following the original paper [9]. Different from the object classification experiment, the face models are finetuned with ArcFace combined with our MTER method. The face models finetuned with MTER method have fine convergence speed and usually converge in no more than 3 epoches. We set m to 0.2, 1.2 and 1.4 for CASIA-WebFace [54], VGGFace2 [4] and MS1M [53], respectively.

Figure 5. The first row is the five target identities, the second row is five attackers which are randomly selected in all the 13233 attackers.

Recognition performance. We test the recognition performance of all the models on LFW [18] and YTF [48]. LFW [18] contains 13233 face images from 5749 different identities. We follow the unrestricted with labeled outside data protocol on LFW [18] and test on the 3000 positive (same identity) and 3000 (different identity) negative pairs. YTF [48] is a database of face video collected from YouTube, which consists of 3,425 videos of 1,595 different people. Each video varies from 48 to 6,070 frames, with an average length as 181.3 frames. We follow the unrestricted with labeled outside data protocol on all the test datasets.

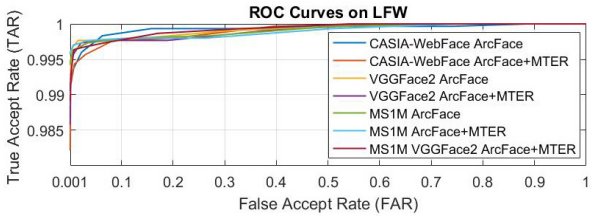

Figure 6. ROC curves of different models on LFW [18]. We define the distance threshold of a network for attacking (or distinguishing a positive and a negative pair) to have a low false accept rate (FAR = $1e-3$) on LFW.

Robustness performance. To simulate the face disguise scenario, we select five person as target identities, as shown in the first row of Figure 5. Then we use the 13233 face images in LFW [18] as attackers to disguise another five target person respectively, which construct a 13233×5 attack matrix to simulate random attacks. We test the robustness under two attack settings:(1) feature attacks and (2) label attacks. The feature attack is more practical in face recognition, while we use the label attack for demonstrating the effectiveness of our method in label defense of close-set settings which is a rarity in deep face recognition.

First we define the feature attacks settings. The attacks are launched from attackers to disguise targets. Specifically, the attack goal is to get face embedding representations of attackers closer to those of targets than the distance threshold of a face recognition system. Next, we define the threshold of a DNN in our simulation. Using positive and negative pairs of LFW [18], we compute the Euclidean distance of normalized deep features to get ROC curves, as shown in Figure 6. Then we identify distance thresholds for judging a pair is positive or negative. Since we would like to compare the adversarial robustness of the trained models like real-world applications, we define the distance threshold for attacking (or distinguishing a positive and a negative pair) to have a low false accept rate (FAR = $1e-3$). We will generate attacks (3) using IFTGSM (5) and FFTGSM (7). Then we define the evaluation criteria to measure the robustness of models. The attack goal is to get face embedding representations of attackers closer to those of targets than the Euclidean distance threshold. The defense goal is to keep the distance between the representations of attackers and targets larger than the threshold. Therefore, an attack is defined as a hit if the embedding distance between the attacker and target is lower than the threshold. We use the average hit rate of the five targets to report the robustness performance of the trained models. The lower is the average hit rate, the stronger is robustness of the model.

Training Method	Training Set	FFTGSM(ε=10)	IFTGSM(ε=5)	IFTGSM(ε=10)	ITGSM [21](ε=10)	LFW	YTF
ArcFace [9]	CASIA-WebFace [54]	99.3	100.0	100.0	98.3	99.5	95.6
ArcFace [9]+adv.	CASIA-WebFace [54]	5.1 (↓ 94.2)	5.7 (↓ 94.3)	49.5 (↓ 50.5)	0.8 (↓ 97.5)	99.4	94.7
ArcFace [9]+MTER	CASIA-WebFace [54]	2.0 (↓ 97.3)	3.5 (↓ 96.5)	27.4 (↓ 72.6)	0.1 (↓ 98.2)	99.5	94.8
ArcFace [9]	VGGFace2 [4]	98.3	100.0	100.0	100.0	99.7	97.7
ArcFace [9]+adv.	VGGFace2 [4]	3.1 (↓ 95.2)	5.5 (↓ 94.5)	63.8 (↓ 36.2)	0.1 (↓ 99.9)	99.5	97.2
ArcFace [9]+MTER	VGGFace2 [4]	4.6 (↓ 93.7)	6.1 (↓ 93.9)	35.6 (↓ 64.4)	0.1 (↓ 99.9)	99.6	97.5
ArcFace [9]	MS1M [53]	99.8	100.0	100.0	69.2	99.7	97.0
ArcFace [9]+adv.	MS1M [53]	45.4 (↓ 54.4)	20.1 (↓ 79.9)	62.6 (↓ 37.4)	0.1 (↓ 69.1)	99.6	96.2
ArcFace [9]+MTER	MS1M [53]	4.1 (↓ 95.7)	7.9 (↓ 92.1)	61.4 (↓ 38.6)	0.0 (↓ 69.2)	99.8	96.9
ArcFace [9]+MTER	MS1M [53]+VGGFace2 [4]	9.6 (↓ 90.2)	6.2 (↓ 93.8)	19.5 (↓ 80.5)	0.1 (↓ 69.1)	99.5	96.8

Table 4. The average hit rate of models trained on CASIA-WebFace [54], VGGFace2 [4] and MS1M [53] supervised by ArcFace loss [9], ArcFace [9]+adv., and ArcFace [9]+MTER, respectively. Attacks are launched from attackers to disguise targets in the feature level using FFTGSM, IFTGSM, and in the label level using ITGSM [21]. The lower is the hit rate, the stronger is robustness of models.

Finally we introduce the label attacks settings. Although the training identities are different from the test ones, we could use the predicted identity of the targets as their labels and let the attackers to launch attacks towards the predicted labels. Meanwhile, a hit is defined as the predicted label of an attacker is the same as that of the target. We also use the average hit rate of the five targets to report the robustness performance. ITGSM [21] will be used to generate label attacks. We will not report result of FTGSM [21] attacks because we find the this method often fails to attack face models.

Results. The results of defense performance against feature and label level attacks are listed in Table 4. The hit rates of original models are close to 100 percent under settings where 13233 different attackers disguise targets. This may indicate that the state-of-art face models are indeed highly vulnerable to adversarial attacks, and an arbitrary attackers would have high probability to disguise another identity. While with our MTER method, the hit rate decrease significantly, which indicates that our method improve the robustness of the state-of-art face models in both open-set and close-set settings and prevent the face disguise feature attacks to a certain degree. Besides, We discover that the robust performance of our method on MS1M [53] is less significant than that on CASIA-WebFace [54] and VGGFace2 [4]. Therefore we recommend to use the large datasets with less identities to finetune the models with large identities to get better robustness performance, *e.g.* finetune the original model trained on MS1M [53] using VGGFace2 [4]. To further evaluate our method, we also compare with a strong baseline by finetuning the original models and incorporating adversarial examples generated using IFTGSM (5). The result shows that our method further benefits from additional embedding regularization, which indicates that incorporating adversarial examples in the training process could improve robustness, while how to optimize with them is also crucial.

The results of face recognition performance on original models and robust models are shown in Table 5. We also list the state-of-art models in face recognition community.

Training Method	Training Set	LFW	YTF
DeepFace [43]	4M	97.35	91.4
FaceNet [36]	200M	99.63	95.1
VGG Face [33]	2.6M	98.95	97.3
DeepID2+ [41]	0.3M	99.47	93.2
Center Face [47]	0.7M	99.28	94.9
Noisy Softmax [6]	WebFace+	99.18	94.88
Triplet Loss [36]	WebFace [54]	98.70	93.4
L-Softmax Loss [26]	WebFace [54]	99.10	94.0
Softmax+Center Loss [47]	WebFace [54]	99.05	94.4
SphereFace [25]	WebFace [54]	99.42	95.0
CosFace [45]	WebFace [54]	99.33	96.1
ArcFace [9]	MS1MV2 (5.8M)	99.83	98.02
ArcFace [9]	WebFace [54]	99.5	95.6
ArcFace [9]+adv.	WebFace [54]	99.4	94.7
ArcFace [9]+MTER	WebFace [54]	99.5	94.8
ArcFace [9]	VGGFace2 [4]	99.7	97.7
ArcFace [9]+adv.	VGGFace2 [4]	99.5	97.2
ArcFace [9]+MTER	VGGFace2 [4]	99.6	97.5
ArcFace [9]	MS1M [53]	99.7	97.0
ArcFace [9]+adv.	MS1M [53]	99.6	96.2
ArcFace [9]+MTER	MS1M [53]	99.8	96.9
ArcFace [9]+MTER	MS1M [53]+VGGFace2 [4]	99.5	96.8

Table 5. The accuracy on LFW [18] and YTF [48]. The state-of-art models in face recognition community are listed in the first cell. Other cells are our models used in the face disguise experiment.

We could also observe that, the accuracy of robust models on LFW [18] and YTF [48] decreased slightly, which indicates that we may sacrifice a certain degree of recognition performance for the improvement of adversarial robustness.

5. Conclusion

We have proposed a margin-based triplet embedding regularization (MTER) method to improve the robustness of DNNs. Experiments on MNIST [51], CASIA-WebFace [54], VGGFace2 [4] and MS1M [53] have demonstrated the effectiveness of our method in simple object classification and deep face recognition.

6. Acknowledgments

This work was partially supported by the National Natural Science Foundation of China under Grant Nos. 61573068 and 61871052.

References

[1] M. Gunther A. Rozsa and T. E. Boult. Lots about attacking deep features. In *IJCB*, 2017. 1

[2] Jimmy Ba and Rich Caruana. Do deep nets really need to be deep? In *NIPS*, 2014. 2

[3] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao, and Karol Zieba. End to end learning for self-driving cars. *arXiv:1604.07316*, 2016. 1

[4] Q. Cao, L. Shen, W. Xie, O. M. Parkhi, and A. Zisserman. Vggface2: A dataset for recognising faces across pose and age. In *International Conference on Automatic Face and Gesture Recognition*, 2018. 2, 7, 8

[5] Nicholas Carlini and David Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *ACM Workshop*, 2017. 1

[6] Binghui Chen, Weihong Deng, and Junping Du. Noisy softmax: Improving the generalization ability of dcnn via postponing the early softmax saturation. In *CVPR*, 2017. 8

[7] Yuheng Chen, Yuheng Chen, Xiaogang Wang, and Xiaoou Tang. Deep learning face representation by joint identification-verification. In *NIPS*, 2014. 1

[8] F. Codevilla, M. Miiller, A. López, V. Koltun, and A. Dosovitskiy. End-to-end driving via conditional imitation learning. In *ICRA*, 2018. 1

[9] Jiankang Deng, Jia Guo, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. *CVPR*, 2019. 1, 4, 7, 8

[10] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9185–9193, 2018. 1, 2

[11] Yinpeng Dong, Hang Su, Jun Zhu, and Fan Bao. Towards interpretable deep neural networks by leveraging adversarial examples. *arXiv:1708.05493*, 2017. 2

[12] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust physical-world attacks on deep learning visual classification. In *CVPR*, 2018. 1, 2

[13] Akshay Agarwal Richa Singh Mayank Vatsa Gaurav Goswami, Nalini Ratha. Unravelling robustness of deep learning based face recognition against adversarial attacks. In *AAAI*, 2018. 2

[14] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015. 2, 4, 5, 6

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1, 4, 5, 6, 7

[16] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv:1503.02531*, 2015. 2

[17] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. *arXiv:1709.01507*, 2017. 1

[18] Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical Report 07-49, University of Massachusetts, Amherst, October 2007. 7, 8

[19] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv:1502.03167*, 2015. 7

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 1

[21] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *ICLR Workshop*, 2017. 1, 2, 4, 5, 6, 8

[22] Alexey Kurakin, Ian J. Goodfellow, and Samy Bengio. Adversarial machine learning at scale. In *ICLR*, 2017. 1, 2, 4, 6

[23] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, November 1998. 5, 6

[24] Shan Li and Weihong Deng. Deep facial expression recognition: A survey. *arXiv:1804.08348*, 2018. 1

[25] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. Sphereface: Deep hypersphere embedding for face recognition. In *CVPR*, 2017. 1, 8

[26] Weiyang Liu, Yandong Wen, Zhiding Yu, and Meng Yang. Large-margin softmax loss for convolutional neural networks. In *ICML*, 2016. 8

[27] Jan Hendrik Metzen, Tim Genewein, Volker Fischer, and Bastian Bischoff. On detecting adversarial perturbations. In *ICLR*, 2017. 1, 2

[28] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *CVPR*, 2017. 2

[29] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: A simple and accurate method to fool deep neural networks. In *CVPR*, 2016. 2

[30] Taesik Na, Jong Hwan Ko, and Saibal Mukhopadhyay. Cascade adversarial machine learning regularized with a unified embedding. In *ICLR*, 2018. 2, 4, 5

[31] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami. Practical black-box attacks against deep learning systems using adversarial examples. In *ASIACCS*, 2017. 1, 2

[32] N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami. Distillation as a defense to adversarial perturbations against deep neural networks. In *SP*, 2016. 2

[33] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, et al. Deep face recognition. In *BMVC*, 2015. 8

[34] Andrew Slavin Ross and Finale Doshi-Velez. Improving the adversarial robustness and interpretability of deep neural networks by regularizing their input gradients. In *AAAI*, 2018. 4

[35] Sara Sabour, Yanshuai Cao, Fartash Faghri, and David J. Fleet. Adversarial manipulation of deep representations. In *ICLR*, 2016. 1, 3

[36] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, 2015. 1, 8

[37] 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 *CCS*, 2016. 2

[38] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556*, 2014. 1

[39] Qing Song, Yingqi Wu, and Lu Yang. Attacks on state-of-the-art face recognition using attentional adversarial attack generative network. *arXiv:1811.12026*, 2018. 1, 2

[40] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014. 7

[41] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deeply learned face representations are sparse, selective, and robust. In *CVPR*, 2015. 8

[42] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, and I. Goodfellow. Intriguing properties of neural networks. In *ICLR*, 2014. 1, 2, 5

[43] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Deepface: Closing the gap to human-level performance in face verification. In *CVPR*, 2014. 8

[44] Florian Tramer, Alexey Kurakin, Nicolas Papernot, Dan Boneh, and Patrick McDaniel. Ensemble adversarial training: Attacks and defenses. In *ICLR*, 2018. 1, 2

[45] Hao Wang, Yitong Wang, Zheng Zhou, Xing Ji, Dihong Gong, Jingchao Zhou, Zhifeng Li, and Wei Liu. Cosface: Large margin cosine loss for deep face recognition. In *CVPR*, 2018. 1, 8

[46] Mei Wang and Weihong Deng. Deep face recognition: A survey. *arXiv:1804.06655*, 2018. 1

[47] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In *ECCV*, 2016. 1, 8

[48] Lior Wolf, Tal Hassner, and Itay Maoz. Face recognition in unconstrained videos with matched background similarity. In *CVPR*, 2011. 7, 8

[49] Lei Wu, Zhanxing Zhu, Cheng Tai, and Weinan E. Understanding and enhancing the transferability of adversarial examples. *arXiv:1802.09707*, 2018. 2

[50] Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. In *NDSS*, 2018. 1, 2

[51] Y. Bengio Y. LeCun, L. Bottou and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, 1998. 2, 4, 5, 6, 8

[52] Ziang Yan, Yiwen Guo, and Changshui Zhang. Deep defense: Training dnns with improved adversarial robustness. In *NIPS*, 2018. 5

[53] Yandong Guo, Lei Zhang, Yuxiao Hu, Xiaodong He, and Jianfeng Gao. Ms-celeb-1m: A dataset and benchmark for large-scale face recognition. In *ECCV*, 2016. 2, 7, 8

[54] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z. Li. Learning face representation from scratch. *arXiv:1411.7923*, 2014. 2, 7, 8