Creating a Defense against Face De-Identification

REINOUT H. A. VOS, University of Twente, The Netherlands

Face recognition systems are used to identify a person's face by detecting patterns in their facial features. It is possible to throw off the FR system by deliberately *obfuscating* the facial patterns the FR system looks for, while at the same time keeping the image unchanged to the human eye. Due to the large range of potential adversarial attack methods, the goal of this research is to improve robustness of FR systems by training the model using obfuscated images as well as regular images, instead of just regular images.

Additional Key Words and Phrases: Identity obfuscation, de-identification, adversarial attack, face recognition.

1 INTRODUCTION

Uploading a face image on social media entails much more than one might think. While it appears to be an innocent act of sharing a photo with friends, the social media platform or a third party, could be "scraping" your face for their gain, using a face recognition (FR) system. For example, ClearView AI [8] has gathered a database containing billions of photos posted on various social media platforms without anyone's knowledge, by developing their own FR model.

Because these FR systems are trained to identify a person's face by detecting patterns in their facial features, they are vulnerable to any obfuscations made to an image designed to mask these patterns. Additionally, this method keeps the visual change to the photo to a minimum, such that the photo still looks the same to the human eye. These adapted images can be referred to as *de-identified* images, *obfuscated* images or *adversarial attacks*. However, these obfuscation methods can also be used maliciously to compromise beneficial FR systems, like photo tagging in social media or automated border control.

Furthermore, obfuscation methods can have a wide ranging complexity: from simply changing only one pixel in an image[15] (this is not effective however), to using a Generative Adversarial Network (GAN) to generating a "mask" which is added to the image[3]. Due to this large range of potential adversarial attack methods that can be used to fool widely used FR systems, it is essential to make FR systems more robust to small perturbations of their inputs[14]. Therefore, the goal of this research is to improve robustness of FR systems by training the model using obfuscated images as well as regular images, instead of just regular images.

Achieving this will improve upon recent research, in which it is shown that two different, regular images of the same person can accurately be verified by FR systems. However, when the FR system is given a regular image and an obfuscated image, or the two images are both obfuscated before being inputted into the FR system, it can easily be fooled [9, 16].

TScIT 37, February 3, 2023, Enschede, The Netherlands

1.1 Research Questions

This leads to the following research question:
How will training an FR model using obfuscated images affect the ability of said model to correctly verify a given obfuscated face image?

In order to help answer this question, or to help solidify the findings in this research, these questions will be answered as well:

- How does a popular FR model trained with only regular images perform against a specific adversarial attack?
- If an FR system is trained with images generated by one obfuscation method, is it more robust to other methods as well?

2 BACKGROUND

2.1 Related Work

As it is essential to keep ahead of the developing obfuscation methods, in order to protect the FR systems, there is plenty of research being done in this field. In the paper written by Goodfellow et al.[5] , the possibility of using adversarial images to improve the robustness of FR systems is first introduced.

The principle of incorporating adversarial images in the training stage has been research extensively since then. These are some notable developments in the field:

Defense-GAN[11] proposes a defense mechanism based on a GAN: trained on unaltered images, one model is trained to detect whether a given image is adversarial or not. The other model is trained to generate these inputted images with the goal to prevent the first model from labelling it as adversarial.

Akhtar et al. [1] propose, amongst other defense methods, a way to effectively train their model using a combination of unaltered images in combination with images containing adversarial perturbations generated with multiple algorithms. This model is then used to detect whether or not a given image is perturbed.

Parseval Networks [2] has developed another defense method by training a neural network using gradient descent and adversarial images. As a result, the system matches the state-of-the-art in terms of accuracy, while being more robust to adversarial attacks.

PixelDefend [13] combines adversarial training with 'feature squeezing', a form of pre-processing: Before the classification, all images are reduced in color range and smoothed, leading to an increase in robustness. In the same paper, another defense method developed by Warde-Farley & Goodfellow [6] is explored. This method is called 'label smoothing' and softens the ground-truth labels in the training data in an attempt to prevent overfitting. However, the paper proved that this method is only effective against simple attacks, making it less robust than the other approaches.

2.2 Obfuscation Methods

There are two approaches when it comes to generating obfuscated images. The first approach is to generate a specific perturbation for each image, the other approach is to train a model which generates an obfuscated image for any given image.

Optimisation method

The optimisation method is based on an optimization algorithm: For a given image, a lower-dimensional representation of the image is created, called an embedding. This embedding is then used to explore the latent space. The latent space is a representation of a dataset in which items, in this case images, resembling each other are positioned closer to one another. The optimization algorithm generates an embedding in the latent space that differs from the original embedding as much as possible.

Fawkes, Adversarial Privacy-preserving Filter (APF) and One Person One Mask (OPOM)[12, 16, 17] are models designed to generate obfuscated images using a 'cloaking' algorithm. For a given image, it computes a small set of perturbations in the form of minor pixel changes, to shift the embedding of the image in latent space. The goal is to move the embedding of the original image to the latent space of a another image of a different person, making the classifier give the image the incorrect label or in the case of verification, move the embedding as far away from the original image as possible. In the case of Fawkes, the algorithm will add image-specific cloak patterns throughout the user's images in order to make it even more difficult to identify/verify the images.

Learning method

AdvFaces and the method developed by Kelly et al. [3, 9] both generate obfuscated images by training a model. AdvFaces uses a GAN approach, whereas Kelly et al. focus on training a model to move an image's embedding in latent space as much as possible, similar to models like Fawkes. The advantage of the learning method is that, once a model is trained, it can be easily shared and is able to generate obfuscated images faster compared to the optimisation method.

2.3 ArcFace

ArcFace [4] is a state-of-the-art FR system, obtaining higher verification accuracy than similar systems. The reason for this is the ArcFace Loss function used to train the model. As shown in figure 1, Softmax, another loss function, is able to seperate the identities, however the boundary regions are unclear. ArcFace, on the other hand, is able to make this clear by penalizing intra-class distance during training, making the predictions more accurate.

A problem with ArcFace, however, is that it is not trained on any obfuscated images. This makes the model vulnerable to obfuscation attacks, as is displayed in the AdvFaces paper [3].

3 PROPOSED SYSTEM

In this research we will take ArcFace as a basis, and re-train the model with the aim that this 'finetuned' version of ArcFace will be more robust against adversarial attacks compared to the standard model. we will re-train ArcFace on a regular dataset, as well as an obfuscated version of the same dataset. The reason for taking two datasets instead of just the one obfuscated dataset is to prevent the model from being overfitted; the model should still be able to correctly verify normal images.

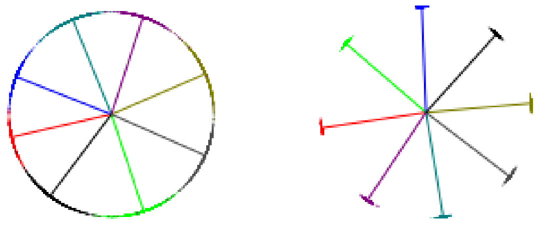


Fig. 1. Figure taken from ArcFace paper [4] used to illustrate the difference between Softmax and ArcFace loss on 8 identities with 2D feature representation.

The first step for re-training the ArcFace model is to reshape the final layers to have the same number of outputs as the number of classes in the datasets we are using. This is due to the fact that ArcFace is trained as a classifier (suited for identification) on a different dataset, containing a different number of identities. In order to use the finetuned model for verification, we will remove the classification layer after training.

The second step is to create a suitable optimizer: While re-training the model, during each epoch the model's weights need to be modified and the loss function needs to be minimized. An optimizer is a function that modifies the attributes of the network, such as weights and biases. This will help in reducing the overall loss and improve the accuracy [7]. We will use the Adam optimizer [10] because it is known as a benchmark for optimization algorithms.

The final step is to define a loss function, we will use Cross Entropy Loss as this is what ArcFace uses, and run the training function:

Algorithm 1 Pseudocode for training function

```
 1: for each $epoch$ do
 2:     for each $batch \in \mathcal{D}ataset1$ do
 3:         - set images' gradients to zero
 4:         - forward images through ArcFace
 5:         - forward images through classifier
 6:         - compute CrossEntropyLoss
 7:         - perform BackPropagation
 8:         - perform optimization step
 9:     end for
10:
11:     for each $batch \in \mathcal{D}ataset2$ do
12:         ...                      ▷ repeat the same steps for 2nd dataset
13:     end for
14: end for
```

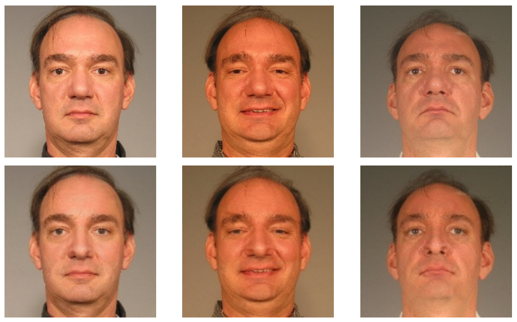


Fig. 2. Top row: Some example images of the FRGC dataset. Bottom row: Obfuscated counterparts (using AdvFaces).

3.1 Training datasets

Trough-out this research we will be using several normal datasets, as well as obfuscated ones. To re-train the ArcFace model, we will be using two datasets:

- The Face Recognition Grand Challenge (FRGC) dataset. This dataset contains roughly 20000 images of 482 different identities. The images themselves are all the same size; cropped to fit the face, have a similar, single-color background and are centered. The identities are all photographed in portrait mode, but facial expressions differ. Also, the lighting is variable.
- The second dataset is generated by obfuscating each image in the FRGC dataset, using the AdvFaces [3] obfuscation method. As shown in Fig. 2, to the human eye the images remain mostly unchanged (when looking closely some differences can be spotted). However, these images will provide a lot of information gain to the model, as structurally they have changed as much as possible due to the obfuscation.

3.2 Evaluation Metrics

In order to measure the performance of a model, we can compute the Equal Error Rate (EER) using these formulas (Eq. 1, 2):

$$FalseAcceptanceRate = \frac{\#FalseAcceptances}{\#VerificationAttempts} \tag{1}$$

$$FalseRejectionRate = \frac{\#FalseRejections}{\#VerificationAttempts} \tag{2}$$

The False Acceptance Rate (FAR) and the False Rejection Rate (FRR) can be used to compute the Equal Error Rate:

$$EqualErrorRate = \frac{FAR + FRR}{2} \tag{3}$$

The Equal Error Rate (EER) indicates that the amount of false acceptances is equal to the amount of false rejections. Therefore, the lower the equal error rate value, the higher the accuracy of the system.

Each time a model is tested on a validation dataset, a corresponding histogram is generated. The histogram plots the distributions of the genuine and impostor comparison scores. This graph acts as a visualization of the EER, so combined they can be used to compare the performances of different models.

Fig. 3. Some example images of the PUT dataset.

Another metric for measuring performance is the Attack Success Rate (ASR).

$$AttackSuccessRate = \frac{\#(ComparisonScores < \tau)}{Total\#Comparisons} \tag{4}$$

Here, τ is the decision threshold at which the FRR of the corresponding FR system is minimal under the constraint that the FAR < 0.1%.

3.3 Validation datasets

Once the model has been re-trained, we will use several validation datasets to help answer the research questions:

- Another part of the FRGC dataset containing roughly 3000 images of 86 identities, of which none of the identities are in the training dataset.
- Two datasets generated by obfuscating each image in the FRGC validation dataset using the Fawkes [12] obfuscation method, as well as the one create by Kelly et al. [9].
- A dataset similar to the FRGC dataset, but with new identities, backgrounds, facial expressions and some images captured from different angles. This dataset is called the PUT dataset.
- A dataset generated by obfuscating each image in the PUT dataset using the AdvFaces [3] obfuscation method.

4 EXPERIMENTS

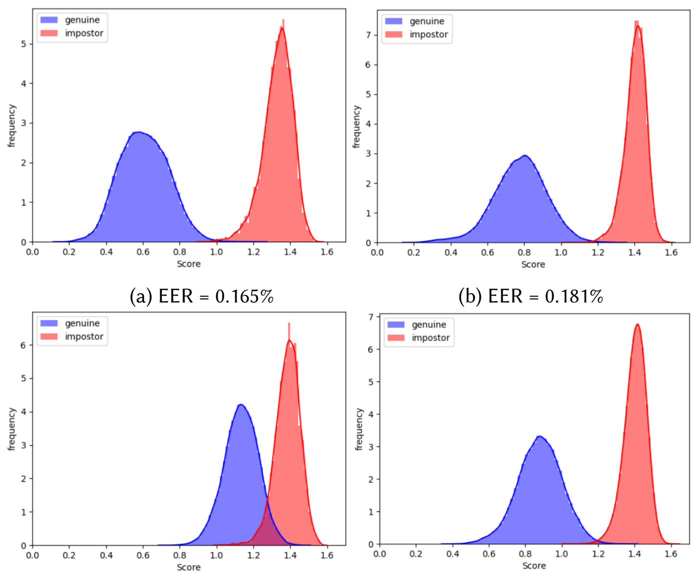
We re-train the the model for 10 epochs, with a batch size of 32 and a learning rate of 0.0001. The values of these parameters are mainly found using a trial-and-error approach, i.e. changing values until the performance no longer increases. Once the finetuned model is finished, we can compare its performance with the regular ArcFace model.

4.1 Results of Additional Training

In this experiment we first obtain a baseline by testing the performance of the regular ArcFace model on the FRGC validation dataset and the obfuscated FRGC validation dataset. Additionally, in order to test how the model handles the problem of comparing a regular image to an obfuscated image, we use both of the datasets together to intertwine regular images with obfuscated images.

Looking at Fig. 4, as the ArcFace model is trained on regular images only, it can verify regular images with extreme accuracy (Fig. 4a). However, once the model is tested on obfuscated images it becomes apparent how effective the AdvFaces obfuscation method is, due to the significant increase in EER (Fig. 4c).

Afterwards, we run the finetuned model on the same datasets. Compared to the regular ArcFace model, even though a minor bit



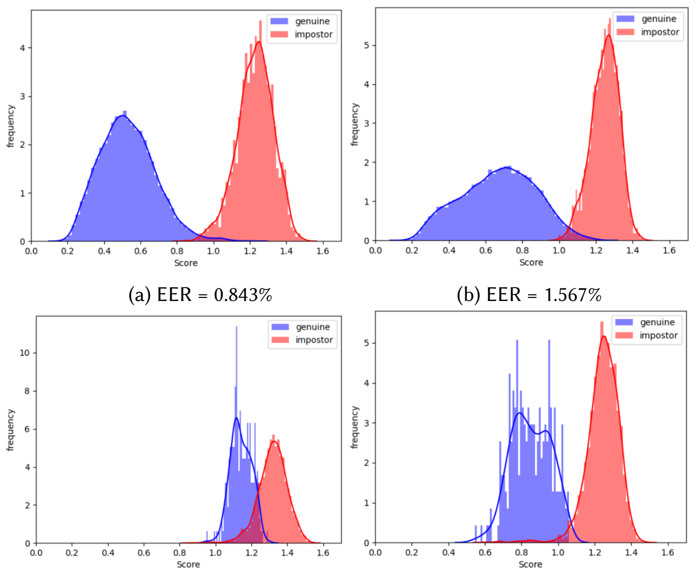
(a) EER = 0.165%

(b) EER = 0.181%

(c) EER = 5.769%

(d) EER = 0.353%

Fig. 4. Performances of ArcFace model (left column) and re-trained model (right column) on FRGC validation datasets. Top row: normal vs. normal. Bottom row: normal vs. obfuscated.



(a) EER = 0.843%

(b) EER = 1.567%

(c) EER = 9.623%

(d) EER = 1.771%

Fig. 5. Performances of ArcFace model (left column) and re-trained model (right column) on PUT validation datasets. Top row: normal vs. normal. Bottom row: normal vs. obfuscated.

of performance is lost on the regular FRGC dataset (Fig. 4a, b), the results show a drop in EER (Fig. 4c, d). Additionally, by using the obfuscated FRGC dataset, the AdvFaces method obtains an Attack Success Rate of 91.0% on the ArcFace model, whereas the finetuned model obtained an ASR of 0.55%, indicating the proposed training method is effective.

4.2 Testing robustness on new data

The ArcFace [4] model is trained extensively on a dataset containing millions of images, leading to high robustness. Since the datasets we use are comparatively much smaller in size, the model has been limited in its training data therefore it is generally more difficult to obtain the same level of accuracy. Following this, we test whether the finetuned model has retained its original robustness. Similar to the previous experiment, we use the PUT dataset, the obfuscated PUT dataset, then we combine the two datasets to test regular vs obfuscated image scenarios. In Fig. 5 the results of the ArcFace model compared to the finetuned model are shown.

The first observation is that both models are less successful on the PUT datasets (Fig. 5) compared to the FRGC datasets (Fig. 4). It is especially surprising that the ArcFace model is performing significantly worse on the regular PUT dataset (Fig. 5a) compared to the regular FRGC dataset (Fig. 4a). An explanation for this could be the fact that the PUT dataset contains more variety in their face images than the FRGC dataset, including photos taken under pose, instead of frontally. Comparing these straight angles with side angles could be more difficult for the model.

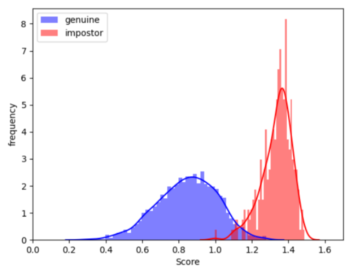
Looking at Fig. 5b, d, f, in each histogram the EER has increased compared to performances on the FRGC datasets (Fig. 4b, d, f). This shows that the finetuned model has lost some of the robustness, being less competent in verifying new types of face images.

When comparing the ArcFace model (Fig. 5a, c) against the finetuned model (Fig. 5b, d), the results show that the finetuned model is considerably more capable of verifying obfuscated images as well verifying normal images against obfuscated images. Using AdvFaces again to create the obfuscated PUT dataset, an Attack Success Rate of 76.7% on the ArcFace model is obtained. On the finetuned model, the ASR was 7.9%. This shows that the proposed model was effective to some degree, but compared to the ASR of the FRGC dataset (experiment 1), it is clear some robustness has been lost. However, we hypothesize using bigger datasets for training could have lead to increased performances on the PUT datasets.

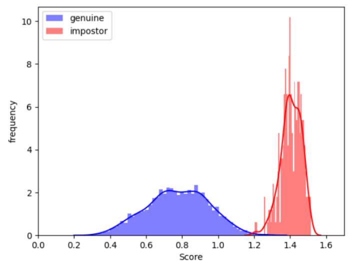
4.3 Testing robustness on new obfuscation methods

The previous two experiments have shown that an FR model can verify obfuscated images relatively well when it is trained on obfuscated data, especially compared to a model trained on regular images only. As another test of robustness we experiment on two different obfuscation methods to find out whether the finetuned model can still perform accurately. For this experiment we use an optimisation-based obfuscation method, namely Fawkes [12], and a learning-based obfuscation method, the method developed by Kelly et al. [9].

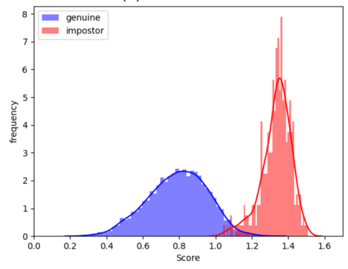
Looking at the results (Fig. 6, 7) the finetuned model significantly outperforms the ArcFace model in both cases. This is confirmed by the Attack Success Rates we obtained from the corresponding obfuscation methods. Firstly, the Fawkes method obtained an ASR of 90.7% on the ArcFace model and an ASR of 0.65% on the finetuned



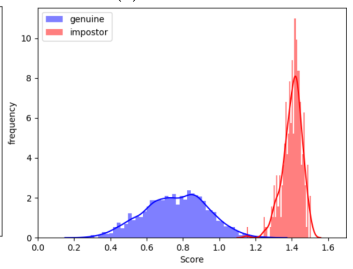
(a) EER = 2.537%



(b) EER = 0.319%

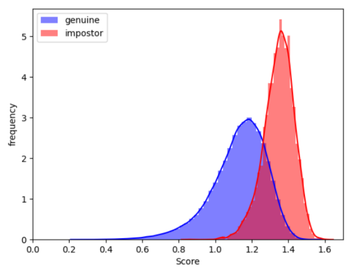


(c) EER = 1.748%

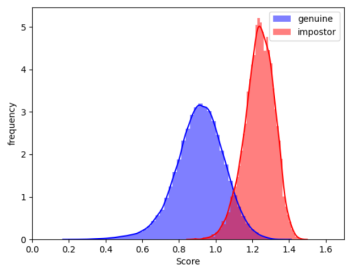


(d) EER = 0.364%

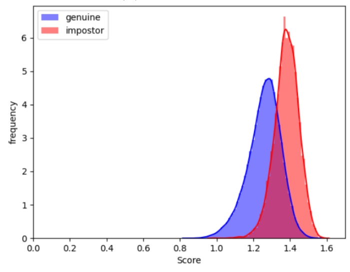
Fig. 6. Performances of ArcFace model (left column) and re-trained model (right column) on obfuscated FRGC validation dataset using Fawkes. Top row: obfuscated vs. obfuscated. Bottom row: normal vs. obfuscated.



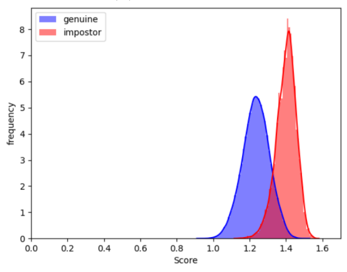
(a) EER = 16.247%



(b) EER = 6.311%



(c) EER = 20.994%



(d) EER = 10.065%

Fig. 7. Performances of ArcFace model (left column) and re-trained model (right column) on obfuscated FRGC validation dataset using method from Kelly et al. Top row: obfuscated vs. obfuscated. Bottom row: normal vs. obfuscated.

model. Additionally, the method from Una et al. obtained an ASR of 99.0% on the ArcFace model and 77.4% on the finetuned model. Once again, this shows that the additional training was effective. However, Fig. 7 and the Attack Success Rate of 77.4% show that this obfuscation method was still able to fool the finetuned model at a high rate.

5 CONCLUSION

In this research we showed that a Face Recognition model trained only on regular images is vulnerable to adversarial attacks. This highlights the effectiveness of state-of-the-art obfuscation methods, as the model could neither reliably verify obfuscated vs. obfuscated images, nor normal vs. obfuscated images.

We then re-trained the same Face Recognition model on regular images, as well as obfuscated images. After comparing the performances of this re-trained model to the original, it showed the new training approach was successful. The re-trained model was able to perform on a similar level on regular images. However, when comparing obfuscated images, especially when comparing normal images against obfuscated images, the re-trained model outperforms the regular model.

In order to fully test the finetuned model's robustness, we experimented on a slightly different dataset containing some more difficult angles and facial expressions. Additionally, we experimented with two obfuscation methods the finetuned model was not trained on, in order to see whether the re-training would lead to an increase in performance against these obfuscation methods as well. The results of these experiments showed that in each case, the finetuned model performed significantly better at verifying obfuscated images compared to the regular model. However, in some cases the Equal Error Rate was not low enough to call the model completely robust and the corresponding Attack Success Rates confirmed this.

This research has lead to some potentially useful insights, worth exploring further. For example, re-training the FR model with a more extensive dataset could lead to an increase in robustness of the model. Furthermore, using a different loss function that supports the volatility of training on alternating regular images and obfuscating images more might lead to promising results.

REFERENCES

[1] Naveed Akhtar, Jian Liu, and Ajmal S. Mian. 2017. Defense against Universal Adversarial Perturbations. *CoRR* abs/1711.05929 (2017). arXiv:1711.05929 http://arxiv.org/abs/1711.05929

[2] Moustapha Cisse, Piotr Bojanowski, Edouard Grave, Yann Dauphin, and Nicolas Usunier. 2017. Parseval Networks: Improving Robustness to Adversarial Examples. https://doi.org/10.48550/ARXIV.1704.08847

[3] Debayan Deb, Jianbang Zhang, and Anil K. Jain. 2019. AdvFaces: Adversarial Face Synthesis. *CoRR* abs/1908.05008 (2019). arXiv:1908.05008 http://arxiv.org/abs/1908.05008

[4] Jiankang Deng, Jia Guo, Jing Yang, Niannan Xue, Irene Kotsia, and Stefanos Zafeiriou. 2022. ArcFace: Additive Angular Margin Loss for Deep Face Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44, 10 (2022), 5962–5979. https://doi.org/10.1109/TPAMI.2021.3087709

[5] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. 2014. Explaining and Harnessing Adversarial Examples. https://doi.org/10.48550/ARXIV.1412.6572

[6] Ian J. Goodfellow and David Warde-Farley. 2016. 11 adversarial perturbations of deep neural networks. Perturbations, Optimization, and Statistics. , 311-339 pages. http://youngwei.com/page/CS4301-0U3/Slides/Perturbations%20Optimization%20and%20Statistics.pdf

[7] Ayush Gupta. 2022. *A Comprehensive Guide on Deep Learning Optimizers*. Retrieved January 20, 2023 from https://www.analyticsvidhya.com/blog/2021/10/a-comprehensive-guide-on-deep-learning-optimizers/#:~:text=An%20optimizer%20is%20a%20function,loss%20and%20improve%20the%20accuracy

[8] Kashmir Hill. 2020. *Twitter Tells Facial Recognition Trailblazer to Stop Using Sites Photos*. Retrieved November 22, 2022 from https://www.nytimes.com/2020/01/22/technology/clearview-ai-twitter-letter.html?searchResultPosition=1

[9] Una M. Kelly, Luuk Spreeuwers, and Raymond Veldhuis. 2022. Exploring Face De-Identification using Latent Spaces. In *2022 IEEE International Joint Conference on Biometrics (IJCB)*. 1–7. https://doi.org/10.1109/IJCB54206.2022.10007990

[10] Diederik P. Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. https://doi.org/10.48550/ARXIV.1412.6980

[11] Pouya Samangouei, Maya Kabkab, and Rama Chellappa. 2018. Defense-GAN: Protecting Classifiers Against Adversarial Attacks Using Generative Models. *CoRR* abs/1805.06605 (2018). arXiv:1805.06605 http://arxiv.org/abs/1805.06605

[12] Shawn Shan, Emily Wenger, Jiayun Zhang, Huiying Li, Haitao Zheng, and Ben Y. Zhao. 2020. Fawkes: Protecting Personal Privacy against Unauthorized Deep Learning Models. *CoRR* abs/2002.08327 (2020). arXiv:2002.08327 https://arxiv.org/abs/2002.08327

[13] Yang Song, Taesup Kim, Sebastian Nowozin, Stefano Ermon, and Nate Kushman. 2017. PixelDefend: Leveraging Generative Models to Understand and Defend against Adversarial Examples. *CoRR* abs/1710.10766 (2017). arXiv:1710.10766 http://arxiv.org/abs/1710.10766

[14] Yang Song, Rui Shu, Nate Kushman, and Stefano Ermon. 2018. Constructing Unrestricted Adversarial Examples with Generative Models. *CoRR* abs/1805.07894 (2018). arXiv:1805.07894 http://arxiv.org/abs/1805.07894

[15] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. 2019. One Pixel Attack for Fooling Deep Neural Networks. *IEEE Transactions on Evolutionary Computation* 23, 5 (oct 2019), 828–841. https://doi.org/10.1109/tevc.2019.2890858

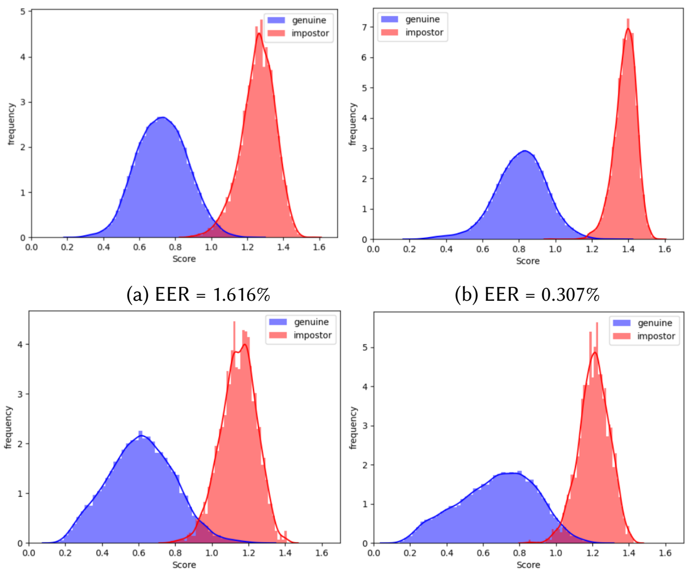
[16] Jiaming Zhang, Jitao Sang, Xian Zhao, Xiaowen Huang, Yanfeng Sun, and Yongli Hu. 2020. Adversarial Privacy-preserving Filter. *CoRR* abs/2007.12861 (2020). arXiv:2007.12861 https://arxiv.org/abs/2007.12861

[17] Yaoyao Zhong and Weihong Deng. 2022. OPOM: Customized Invisible Cloak Towards Face Privacy Protection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2022), 1–1. https://doi.org/10.1109/TPAMI.2022.3175602

A APPENDIX

	ArcFace	Finetuned
FRGC normal vs. obfuscated (AdvFaces)	91.0	0.55
PUT normal vs. obfuscated (AdvFaces)	76.7	7.90
FRGC normal vs. obfuscated (Fawkes)	90.7	0.65
FRGC normal vs. obfuscated (Una et al.)	99.0	77.4

Table 1. Table giving an overview of all the Attack Success Rates obtained in this research (in %).



(a) EER = 1.616% (b) EER = 0.307%

(c) EER = 3.279% (d) EER = 2.615%

Fig. 8. Additional scores of obfuscated vs. obfuscated comparisons. Performances of ArcFace model (left column) and re-trained model (right column) on... Top row: FRGC validation dataset. Bottom row: PUT validation dataset.