Face/Off

Alireza Torabian, MohammadMahdi Abdollahpour

Abstract. In this project, we tried to develop a plugin to obfuscate images to enhance face privacy on social media. The main approach is to perform an adversarial attack so that the resulting images are wrong classified or detected by systems while the face still looks like the original one.

Keywords: Adversarial attack, Computer vision, Privacy

1 Introduction

Facial image data is incredibly valuable and sensitive because it is key to your identity. It is used to identify you in many critical applications like smile-to-pay [1] and facial recognition shoplifter detection[2]. The fear is that detailed profiles of yourself may be created if your data gets in the wrong hands. So, your face data must be changed in a way that facial recognition systems can not be trained on it. However, the image must be still represent the content it is intended to have. Therefor, we choose to perform an adversarial attack. Moreover, our attack must be able to do well in black-box setting at the end.

2 Related Work

Many attacks have been proposed on attacking facial recognition models like adversarial attacks using gradient estimation [4] Evolutionary methods [3] generative networks [7] as well as on attacking facial detection models [5].

3 Enhancing Face Privacy

In order to fool the recognition systems, there are two approaches we can take: fooling face detection and fooling face recognition. Face detection is proposing areas in the image that contains a face and face recognition is classifying the face to the right person. At first, we try to attack face detection.

3.1 Attacking Face Detection

In order to create adversarial examples we use the Projected Gradient Descent [8]. We defined a loss function and computed the gradient of loss function w.r.t

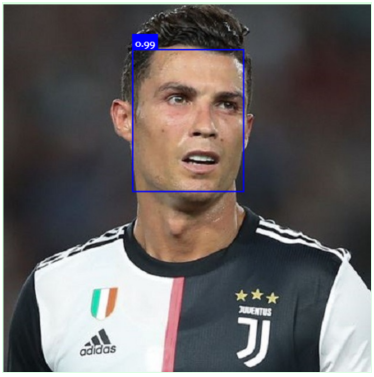
the input and update the example using the gradient direction. The initial loss function was:

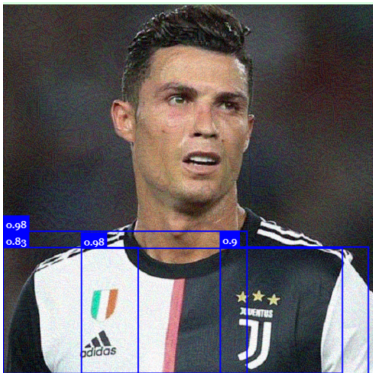
$$Loss = -\sum_{b \in boxes} score_b * (sign(d_b - t)) \tag{1}$$

where d is the distance from the real face position and t is a threshold. The real face was successfully undetected by the local model but we observed that fake face detection boxes had small areas which makes it unreasonable. So we changed the loss function to:

$$Loss = -\sum_{b \in boxes} score_b * area_b * (sign(d_b - t)) \tag{2}$$

where $area_b$ is the area of the detected face. Using this loss function we were able to have fake detection with big enough boxes. This attack is targeted since not only it lowers the probability of an area containing a face, it also generates some fake proposals with reasonable sizes. Unfortunately, this attack does not seem to be transferable.





3.2 Attacking Face Recognition

We use FaceNet[9] to embed faces in a vector, then dot product means similarity. So, we perform adversarial attack on FaceNet so that the victim will be recognized as similar to target. The percie algorithm is:

1. All faces in the image are detected and cropped and resized to 160*160 (Embedding model input size). (Using SSD Mobilenet from Face-API.js)
2. For each face distance (dot product of embeddings) to potential victims are computed and closest victim is chosen. (Embedding computed using FaceNet)
3. Projected Gradient Descent method with momentum and input diversity is applied to the objective function which is average of dot product of embeddings:

Fig. 1. Comparing unperturbed an perturbed images for the attack on face detection

(a) Input is resized to rnd*rnd which rnd is a random number uniformly sampled from [135,160] and then it is padded to 160*160 and rescaled image is not in center necessarily. The new image is chosen with probability of 0.9 otherwise original image is used [10]. (Input Diversity)
(b) The image values are normalized to [-1,1] and a random uniform noise is added. Then we compute embeddings and do L2 normalization.
(c) We compute objective function which is average of dot product of image embeddings with victim embeddings.
(d) We compute gradient of objective function w.r.t. input image then we add the gradient to previous gradient multiplied to 0.9 [6].
(e) We use Gradient Sign Method to get an adversary and clip it to L-infinity bound. We repeat this process to certain number of times [8]. (Projected Gradient Descent)

We have tested the recognition attack model on tools BetaFace and Clarifai.

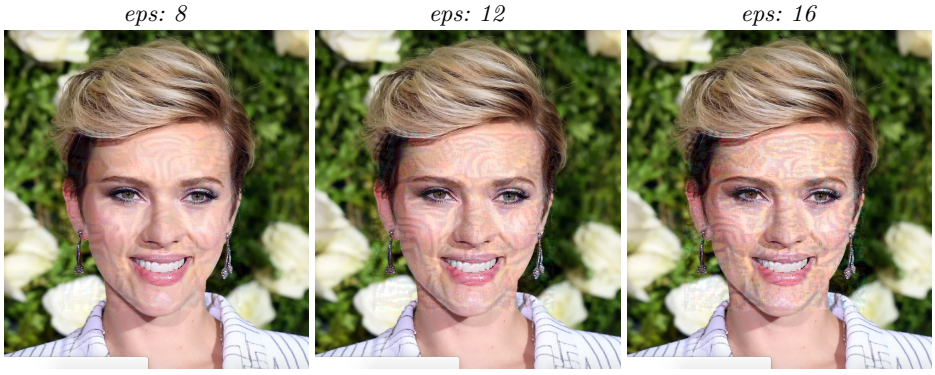
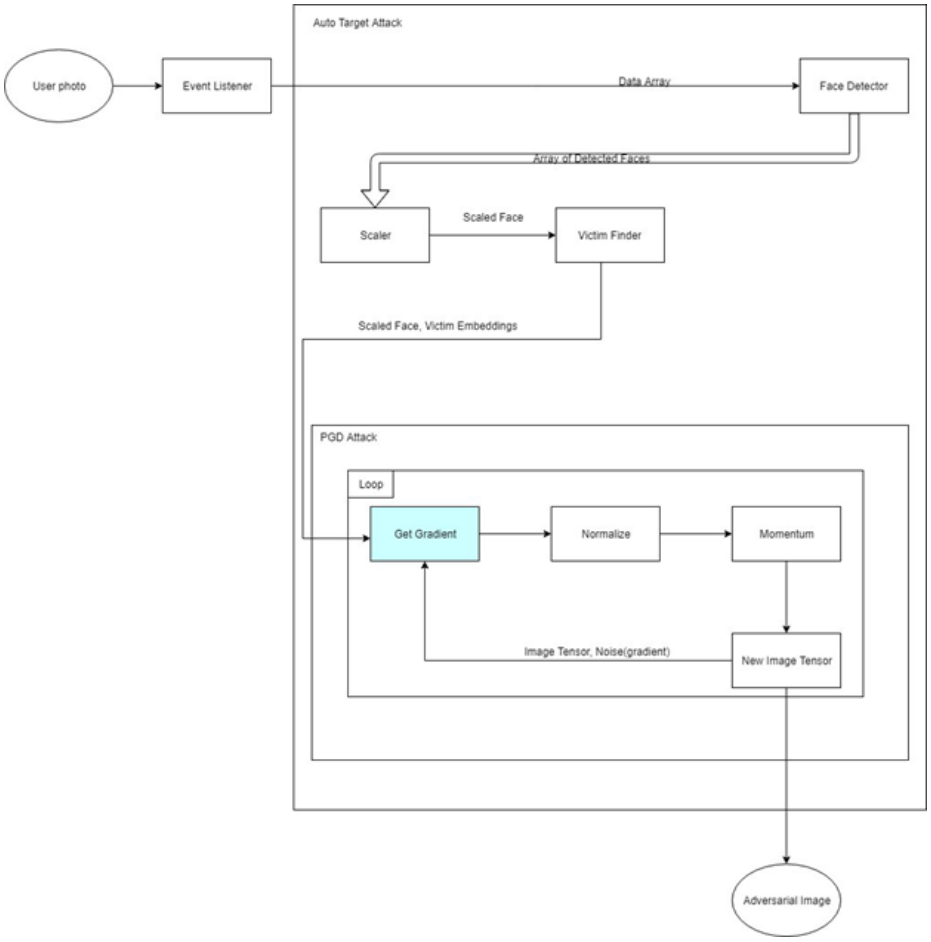


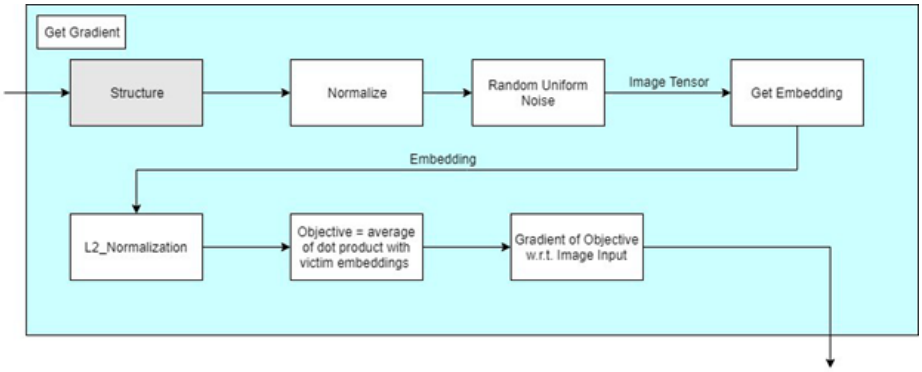
Fig. 2. Except BetaFace tool on the first perturbed image with epsilon 8, the recognition models have been fooled.

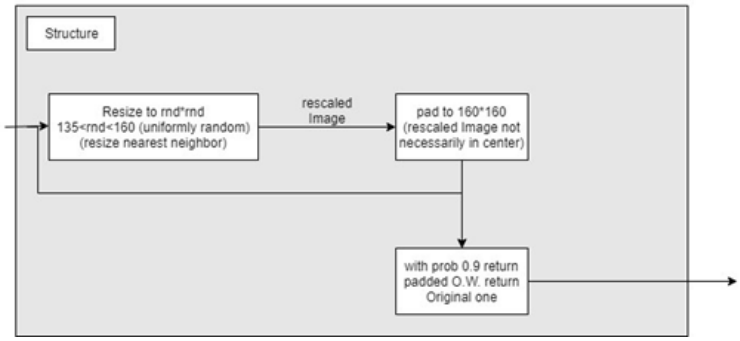
Pseudo code

```
ObjectiveFunction(imageInput) {
   With probability 0.1 resize image to random*random
   (random uniformly from [135,160]) and
   do a random padding
   Scale imageInput to [-1.1]
   Add randomUniform noise between [ -1e-2, 1e-2] imageInput
   Get imageInput Embedding
   L2_normalize the Embedding
  Return mean of dot product of victimEmbeddings and Embeddings
  // must be maximized
}

oneStepAttack(image, grad) {
    noise = gradient of ObjectiveFunction(image) w.r.t. image
    // div on L2 norm
    noise = noise/ mean(|noise|)
    // momentum
    noise = grad * 0.9 + noise
    // after this, we apply sign on noise, so it will be normalized
    let adv = image + sign(noise) * 1.0
    Clip image to lower bound and upper bound
    return adv, noise
}

let input = imageBatch.toFloat();

lowerBound = image - eps , clipped to 0-255
upperBound = image + eps , clipped to 0-255

grad = zeros
for ( i = 0; i < maxIter; i++) {
   Res = oneStepAttack(input, grad);
   input = res[0];
   grad = res[1];
}
adversarial = input.toInt();
```

Diagrams


Auto Target Attack
User photo
Event Listener
Data Array
Face Detector
Array of Detected Faces
Scaler
Scaled Face
Victim Finder
Scaled Face, Victim Embeddings
PGD Attack
Loop
Get Gradient
Normalize
Momentum
Image Tensor, Noise(gradient)
New Image Tensor
Adversarial Image





4 Summary

We realized that the face detection attack model is not transferable, so we cannot use both attack models together. Our face recognition attack model has achieved 35% success rate in fooling face recognition tools BetaFace and Clarifai. However, our model couldn't impersonate the person in the picture to the pre-specified target person in most cases. Our evaluations on various samples are reported here.

References

1. https://www.theguardian.com/world/2019/sep/04/smile-to-pay-chinese-shoppers-turn-to-facial-payment-technology(CSRF)
2. https://www.cnet.com/news/with-facial-recognition-shoplifting-may-get-you-banned-in-places-youve-(CSRF)

3. Alzantot, M., Sharma, Y., Chakraborty, S., Zhang, H., Hsieh, C.J., Srivastava, M.: Genattack: Practical black-box attacks with gradient-free optimization (2018)
4. Bhagoji, A.N., He, W., Li, B., Song, D.: Practical black-box attacks on deep neural networks using efficient query mechanisms. In: European Conference on Computer Vision. pp. 158–174. Springer (2018)
5. Bose, A.J., Aarabi, P.: Adversarial attacks on face detectors using neural net based constrained optimization (2018)
6. Dong, Y., Liao, F., Pang, T., Su, H., Zhu, J., Hu, X., Li, J.: Boosting adversarial attacks with momentum (2017)
7. Dong, Y., Su, H., Wu, B., Li, Z., Liu, W., Zhang, T., Zhu, J.: Efficient decision-based black-box adversarial attacks on face recognition (2019)
8. Madry, A., Makelov, A., Schmidt, L., Tsipras, D., Vladu, A.: Towards deep learning models resistant to adversarial attacks. In: International Conference on Learning Representations (2018), https://openreview.net/forum?id=rJzIBfZAb
9. Schroff, F., Kalenichenko, D., Philbin, J.: Facenet: A unified embedding for face recognition and clustering. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (Jun 2015). https://doi.org/10.1109/cvpr.2015.7298682, http://dx.doi.org/10.1109/CVPR.2015.7298682
10. Xie, C., Zhang, Z., Zhou, Y., Bai, S., Wang, J., Ren, Z., Yuille, A.: Improving transferability of adversarial examples with input diversity (2018)