DEFENCE RESEARCH & DEVELOPMENT ORGANIZATION

SCIENTIFIC ANALYSIS GROUP LAB

Internship Project Report

on

Adversarial Evasion Attacks Analysis

Submitted By

Maan Jayesh Patel

&

Vishw Prakash Joshi

Guided By

Scientist Pooja Yadav

Tabel of Content

1. Problem Definition: -

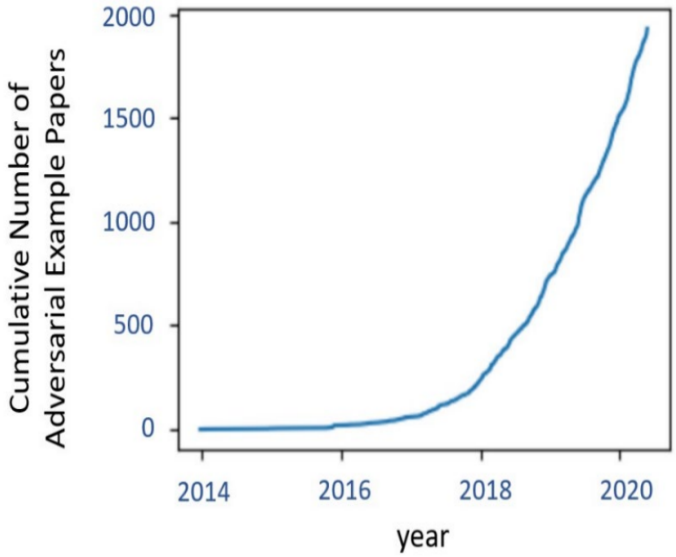
An adversarial evasion attack in the context of image data is a type of attack on machine learning models, particularly those used in computer vision, where an attacker modifies the input images in subtle ways to deceive the model into making incorrect predictions or classifications. These modifications are often imperceptible to the human eye but can cause significant changes in the model's output

In advanced facial recognition (FR) models, enhancing recognition efficiency is not enough; the system must also resist various potential attacks aimed at undermining its proficiency. Recent research has shown that deep FR systems are susceptible to different types of attacks that introduce data variations to mislead classifiers. These attacks can be executed either through (a) physical attacks, which alter the physical appearance of a face before the image is captured, or (b) digital attacks, which modify the image after it has been captured (Singh et al., 2020). Physical attacks, often referred to as spoofing or presentation attacks (Marcel et al., 2014), involve manipulating the face directly. On the other hand, digital attacks include adversarial attacks (Yuan et al., 2019) and morphing attacks (Scherhag et al., 2020), which alter the digital image itself. While adversarial attacks primarily fall under digital attacks, some techniques are designed to work physically.

Adversarial attacks are particularly intriguing because they generally target deep neural networks (DNNs), especially convolutional neural networks (CNNs), which are the foundation of state-of-the-art FR models. The significant increase in the number of research papers on adversarial example generation each year highlights the growing interest in this type of attack (see Fig. 1).

There are various adversarial evasion attacks, but there is no specific mention of which attack performs well with respect to a particular class of model.

In this project, we are analyzing a selected list of these attacks to determine which one is the most efficient in term of human perceptibility and success rate



2. Scope of Work: -

Consider a facial recognition system used for secure access to buildings. An adversary might slightly alter the pixels of their photo using a technique like the Simple Black Box Attack (simBA). These alterations are so minor that a human observer wouldn't notice any difference. However, the modified image could trick the facial recognition system into misclassifying the adversary as an authorized individual, thereby granting them access.

Adversarial Attacks are crucial for several reasons:

- **Security:** Adversarial attacks can compromise the security of systems relying on facial recognition, such as access control and surveillance.
- **Reliability:** These attacks highlight the need for robust and reliable machine learning models that can resist manipulation.
- **Ethical Implications:** Ensuring that facial recognition systems are fair and unbiased involves protecting them from adversarial influences that could exacerbate existing biases or create new ones.

3. Approach: -

3.1 Types of Attacks: -

I. Black Box Attack - In black-box attacks, the attacker does not have access to the internal parameters, architecture, or training data of the model. Instead, they can only observe the model's input-output behaviour, i.e., they can query the model with inputs and obtain the corresponding outputs. This limited knowledge makes black-box attacks more challenging but also more realistic in many real-world scenarios where attackers do not have full access to the system.

II. White Box Attack - In white-box attacks, the attacker has complete knowledge of the model, including its architecture, parameters, and training data. This comprehensive access allows the attacker to create highly effective adversarial examples by directly exploiting the model's vulnerabilities.

From these two above attack techniques we decided to perform Black Box Attack.

3.2 Black Box Techniques: -

I. Transferability Attacks: -

The transferability of adversarial examples refers to that adversarial examples crafted on an accessible service (or model) are also able to fool other inaccessible models with unknown architectures or parameters. A number of transferable adversarial attacks have been proposed to enhance the transferability of adversarial examples so as to improve the attack success rates in the totally black-box setting, as follows:

a) Transferable Adversarial Attacks can be classified by target of the attacks, namely the following:

a.1) Decision Layer Attack: The probability is predicted by the direct attack substitution model through one-step attack or multi-step iterative attack, optimizing adversarial samples.
a.2) Feature Layer Attack: Different from the decision layer attack, feature layer attack mainly attacks the middle layer features of the alternative model so as to achieve the purpose of attacking the target service.

b) Transferable Adversarial Attacks can also be classified by attack generation method, as follows:

b.1) Gradient Optimization Attack: In this attack, a substitute model is obtained that has a similar decision boundary with the target model, the according loss function of the substitute model is set, and the gradient of the loss function is calculated against the input space for optimized adversarial perturbations. In practice, to approximate to the gradient of the target model, multiple gradient calculation methods are used, such as fast gradient sign method or momentum-based. gradient calculation method.
b.2) Generative Attack: Different from the previous gradient based anti-migration attacks, the alternative model is attacked by optimizing the generation model, and then the generation model is used to generate anti disturbance to attack the target service.

c) There are strategies to improve the transferability of the attacks in Items a) and b) as follows:
c.1) Input transformation: This strategy adopts various input transformations to further improve the transferability of adversarial examples. Specifically, such attacks create a batch of various patterns based on an input image and use the average gradient to optimize the adversarial examples.
c.2) Model ensemble: This strategy uses multiple models simultaneously to improve the attack transferability. Often the predictions, logits, or losses of multiple models are fused. If an

adversarial example can mislead multiple models simultaneously, it is likely to mislead another one as well.

II. Query Based attacks:

Query Attack is one kind of black-box adversarial attack, which allows probing target models with queries. Compared with other kinds of black-box adversarial attacks, which can be grouped as zero-knowledge attacks, Query Attack can access the target model to obtain important information such as local gradient. Query Attack can be divided into two groups, including Score-Based Attack and Decision-Based Attack. Score-Based Attack can obtain not only the predictive labels but also the probability of the target labels. Decision-Based Attack can only obtain the predictive labels, such as boundary attack.

Query attacks can be classified by the information provided by the target service for input images, namely the following:

a) Score-based attacks: The target service not only returns the predicted category but also feeds back
the predicted probability and the output logits that could be utilized, which is convenient for the adversaries to generate adversarial examples. The researchers usually leverage the zero-order, first-order, or second-order information of the victim model to estimate the gradient with respect to the input sample. Then the white-box attacks could be naturally performed for efficient black-box attacks.

b) Decision-based attacks: These are much more relevant in real-world machine learning applications where confidence scores or logits are rarely accessible, compared to score-based attacks. At the same time, decision-based attacks have the potential to be much more robust against standard defences like gradient masking, intrinsic stochasticity, or robust training than attacks from other categories. Finally, compared to transfer-based attacks, they need much less information about the model (neither architecture nor

training data) and are much simpler to apply. However, decision-based attacks are much more challenging due to the minimum information requirement for the attacks.

3.3 Query Based Attacks: -

❖ Types: -

I. Score-Based Attacks - Score-based attacks assume that the model provides confidence scores for its predictions. The attacker uses these scores to iteratively refine the adversarial example.

- Method: -
  - Initial Perturbation: Start with an initial image x and a small perturbation δ.
  - Gradient Estimation: Estimate the gradient of the loss function with respect to the input by querying the model multiple times.
  - Update Rule: Use the estimated gradient to update δ in a direction that increases the loss, making the model more likely to misclassify the perturbed image $x+\delta$.

- Mathematical Formulation:
  - Let $f(x)$ be the confidence score vector for input x and $L(f(x),y)$ be the loss function for true label .
  - The goal is to maximize $L(f(x+\delta),y$ while keeping δ small

$$\delta^* = \arg\max_{\delta} L(f(x+\delta), y) \quad \text{s.t.} \quad \|\delta\| \leq \epsilon$$

- Use finite differences to approximate the gradient, where h is a small constant and u is a random vector.

$$\nabla_\delta L \approx \frac{L(f(x+\delta+hu), y) - L(f(x+\delta), y)}{h}$$

II. Decision Based Attacks: Decision-based attacks only require the final classification decision (label) from the model. These attacks are more challenging as they lack gradient information.

- Method: -
  - Initial Misclassified Example: Start with an adversarial example x' that is already misclassified.
  - Boundary Search: Iteratively adjust x' to find the closest point on the decision boundary.
  - Perturbation Refinement: Minimize the perturbation while maintaining misclassification.

- Mathematical Formulation:
  - Let f(x) be the model's prediction (hard label) for input x.
  - The goal is to find the smallest perturbation δ such that f(x+δ) ≠ y:

$$\delta^* = \arg\min_\delta \|\delta\| \quad \text{s.t.} \quad f(x+\delta) \neq y$$

  - Use binary search to find the decision boundary and iterative refinement to reduce ‖δ‖.

4. Selecting Facial Recognition Model: -

4.1 Model and Dataset Selection: -

The availability of sufficiently large training datasets has driven the development of increasingly complex datasets to support facial recognition (FR) research. Early deep FR models like DeepFace, FaceNet, and DeepID (Sun et al., 2014) were trained on private, controlled, or small-scale datasets, limiting the ability to compare new models effectively. To address this, CASIA-Webface (Yi et al., 2014), comprising 0.5 million images of 10,000 celebrities, was introduced as the first widely used public training dataset. Subsequently, larger public datasets such as MS-Celeb-1M (Guo et al., 2016), VGGface2 (Cao et al., 2018), and Megaface (Kemelmacher-Shlizerman et al., 2016), each containing over 1 million images, were introduced to facilitate the development and evaluation of advanced deep learning methods. We selected Inception Resnet V1 and VGG-16 pre-trained model. LFW, VGG Face 2 and Casia-Webface Datasets for feature extraction and fine-tuning.

4.2 Transfer Learning: -

4.2.1 Feature Extraction:

For evaluating the evasion attacks we require a facial recognition model and dataset on which it is trained. So, we decide to perform transfer learning on state of model on a facial image set.

We choose VGG-16 model state of art model and VGG Face 2 Dataset. Dataset contain 30,87,656 images of 8631 different people. But we failed to perform feature extraction successfully due to insufficient computational power available.

Next, we performed feature extraction on Inception Resnet V1 on Labelled Faces in the Wild (LFW) Dataset. We remove head of Inception Resnet v1 model and added our own dense layers. We made all trainable parameters of convolution layer False and then train the model. In results we found model getting overfitting with training accuracy 0.9567 and validation accuracy 0.3040, shown in Fig. 2. To overcome overfitting, we perform data augmentation and added dropout layer

in the head part of model architecture. But still results were same, we came conclusion that overfitting is due to uneven image distribution in each class.

We decided to change the dataset to Casia Webface with 4,90,623 images of 10,572 peoples. We divided dataset into 80/20 ratio of training set and test set. Image distribution between classes was better than LFW dataset. As mention in the Keras's documentation first feature extraction should be applied and then fine-tuning, so we first applied feature extraction on Inception Resnet V1 model using Casia Webface dataset. All trainable parameters convolutional layers were made false and new dense layers (model's head) were added. For training we used Adam optimizer with learning rate 1e-5, Categorical Cross Entropy as Loss function (because labels were categorical on-hot encoded) and epochs value was 30. Training accuracy was 0.77 and validation accuracy was 0.78.

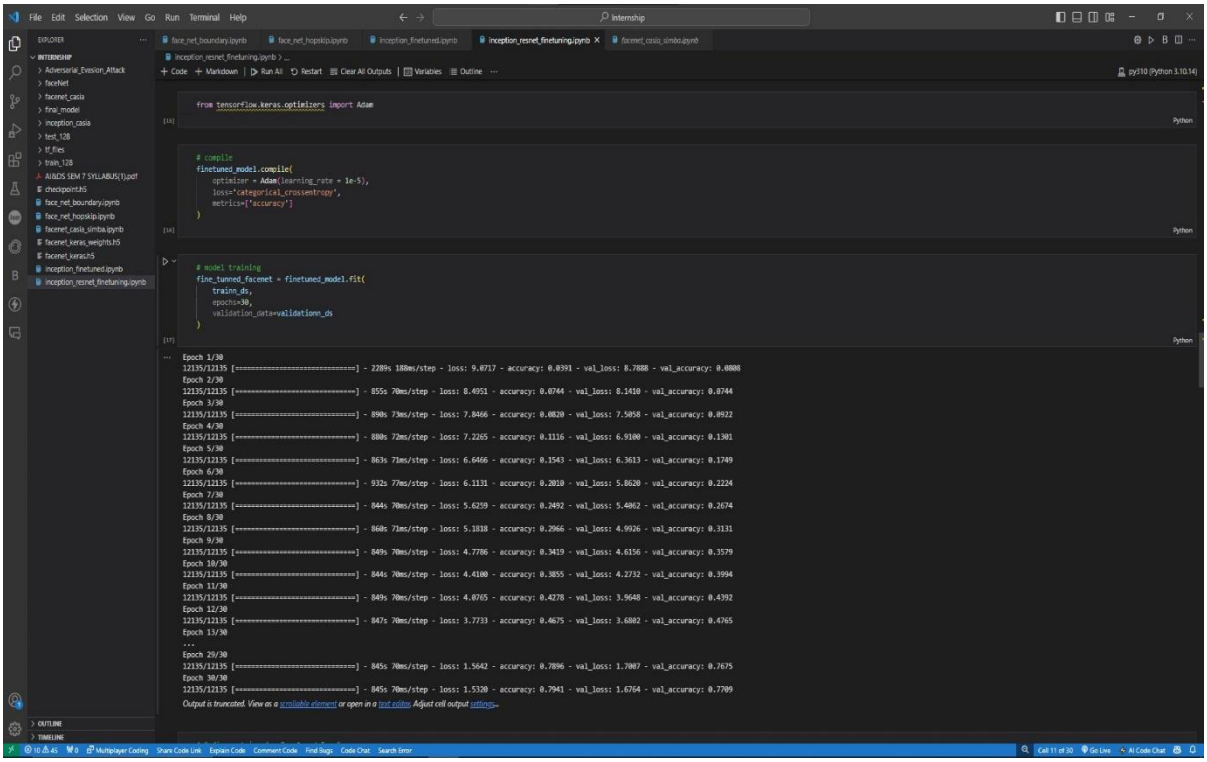


Fig. 2

4.2.2 Fine Tuning:

Fine-tuning means that along with training the dense layer parameters, the trainable parameters of the convolutional layers are also trained during the training process.

After feature extraction, to increase the accuracy of facial recognition model we perform fine-tuning by making convolutional layer's parameters trainable. We made parameters trainable after 327th layer in convolutional part of the model. For training parameter of fit function were kept same as of feature extraction. Training Accuracy was 0.8970 and Validation Accuracy was 0.8101. We will perform multiple Black Box Attacks on this fine-tunned CNN based Model (inception resnet v1). In Top-k accuracy, Top-1 accuracy was 0.81 and Top-5 accuracy was 0.87.

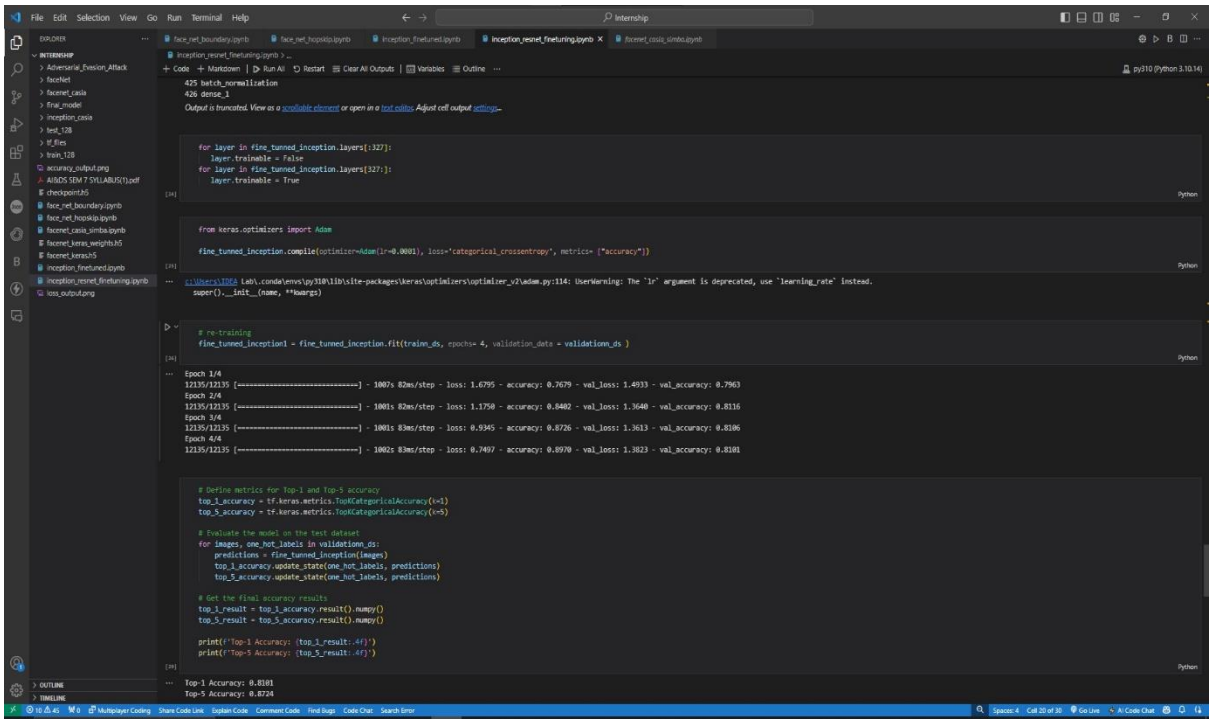


Fig.3

5. Adversarial Evasion Attacks Performed: -

5.1 Simple Black Box Attack (simBA): - Simple Black-Box Attack (simBA) is an adversarial attack that generates perturbations by randomly modifying individual pixels or groups of pixels in the input image and evaluating the impact on the model's output. It is a query-efficient black-box attack that does not require gradient information from the model, relying instead on observing changes in the model's predictions to guide the generation of adversarial examples.

Internal Working: -

- It is prediction probability-based attack. Firstly, NumPy zeros array (Identity matrix) is formed with same shape of original image.
- Calculate the initial probability of the true label and also get true label of the image. Randomly selected direction vectors (noise) are added to original image.
- Both negative and positive perturbation are added. After adding changes to the image prediction probability is calculated and label is predicted again if it does not match with given targeted label then again perturbation is added to the image until targeted label is achieved.
- For example, consider image size is (160,160,3). Flattening this image gives vector size of 76800. Choosing vector size equal to the flattening size result in an exhaustive search but this is computationally infeasible. Suppose we want to generate 500 orthonormal vectors for perturbation. Then in each iteration q, one of the 500 vectors will add ensuring direction is unique and effective.

Parameters: -

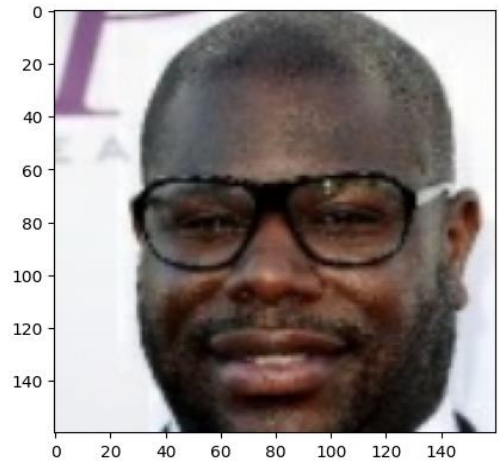
We used functional call for simBA attack provided by Adversarial Robustness Toolbox. Following parameter were used:

- Classifier: TensorflowV2Classifier(loss=categoricalcrossentroy, Optimizer = Adam)
- attack: ‘dct’
- max_iter: 3000
- epsilon: 0.1
- order: ‘random’
- freq_dim: 4
- stride: 2
- targeted_attack: True
- batch_size: 1
- verbose: True

Results: -

- We performed simBA attack on randomly five selected classes from the dataset.
- In first class, out of 19 images, 17 of them were successfully attacked with target label resulting in 89.4% success rate. Structure Similarity Index Measure (SSIM) score between original image and adversarial image was 0.964.
- In second class, out of 34 images, 33 of them were successfully attacked, but three of them were predicted with non-targeted label. So, 30 images were wrongly predicted with target label resulting in 88.2% success rate. Structure Similarity Index Measure (SSIM) score between original image and adversarial image was 0.968.
- In Third class out 67 images 63 of them were successfully attacked but two of them were predicted with non-targeted label. So, 61 images were wrongly predicted with target label resulting in 91.0% success rate. Structure Similarity Index Measure (SSIM) score between original image and adversarial image was 0.97.
- In Fourth class out 82 images 81 of them were successfully attacked but two of them were predicted with non-targeted label. So, 79 images were wrongly predicted with target label resulting in 96.3% success rate. Structure Similarity Index Measure (SSIM) score between original image and adversarial image was 0.97.

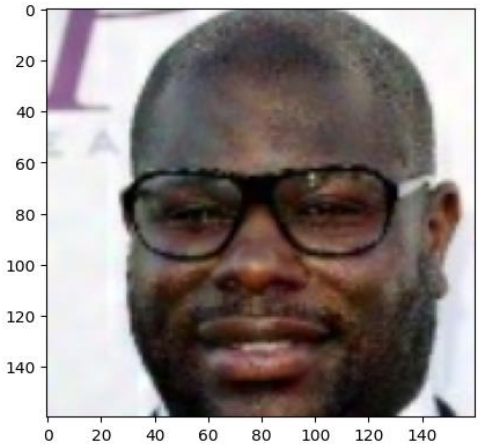
Samples of Attack: -



Original Image

Original label: 8715

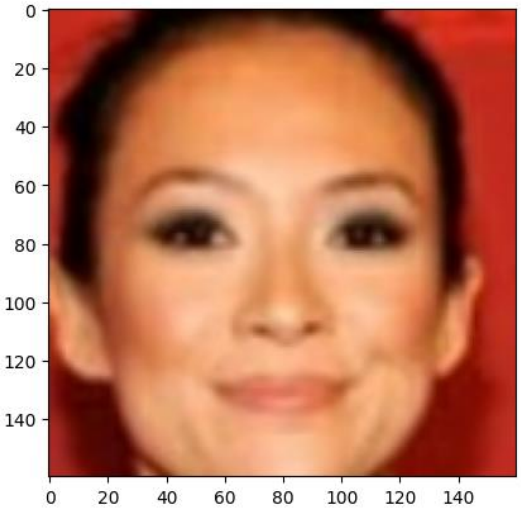
Confidence before attack: 0.9996



Adversarial Image

Adversarial label: 38

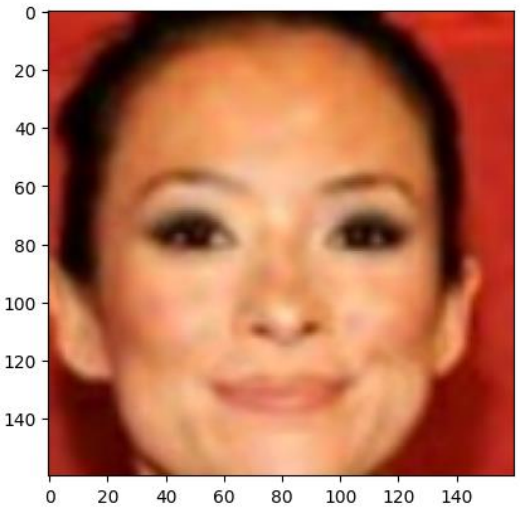
Confidence after attack: 0.2260



Original Image

Original label: 5177

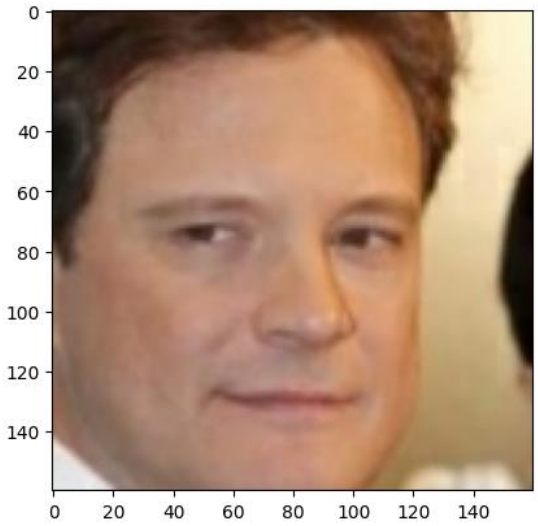
Confidence before attack: 0.9970



Original Image

Original label: 38

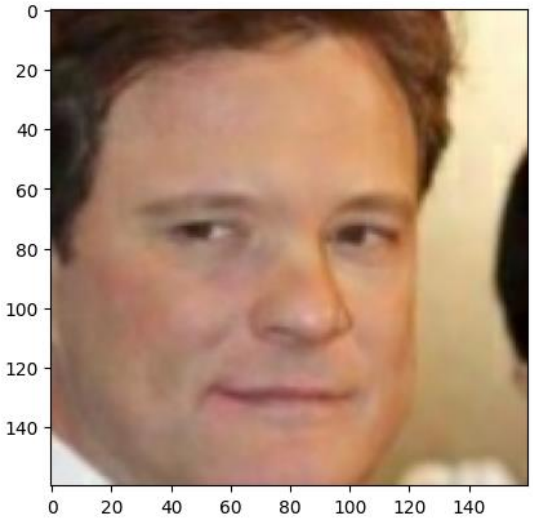
Confidence before attack: 0.1768



Original Image

Original label: 18

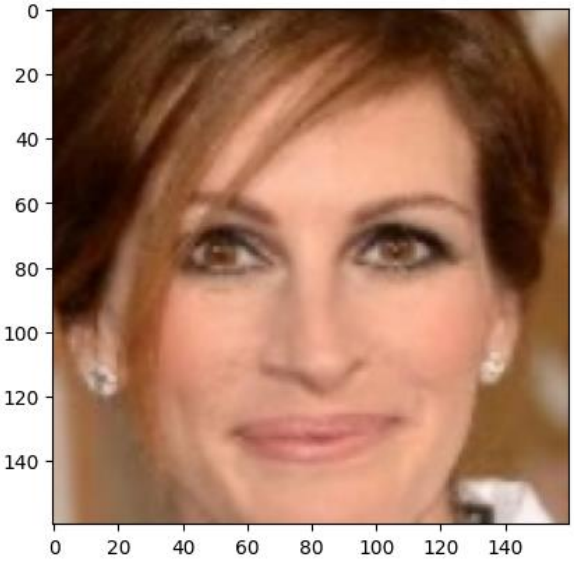
Confidence before attack: 0.9996



Adversarial Image

Adversarial label: 38

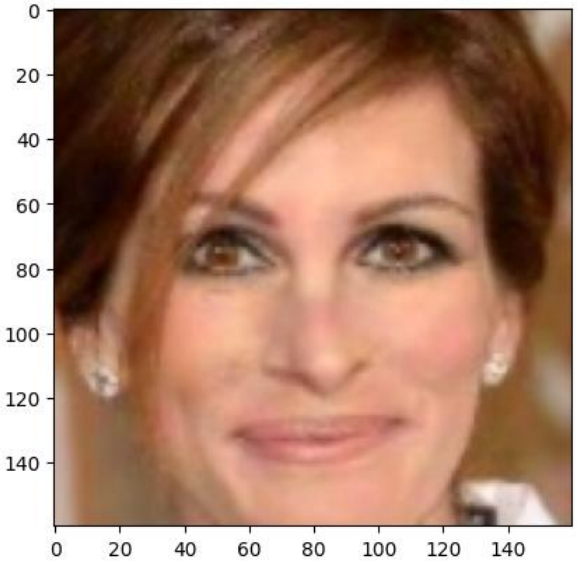
Confidence after attack: 0.1918



Original Image

Original label: 41

Confidence before attack: 0.9816



Adversarial Image

Adversarial label: 38

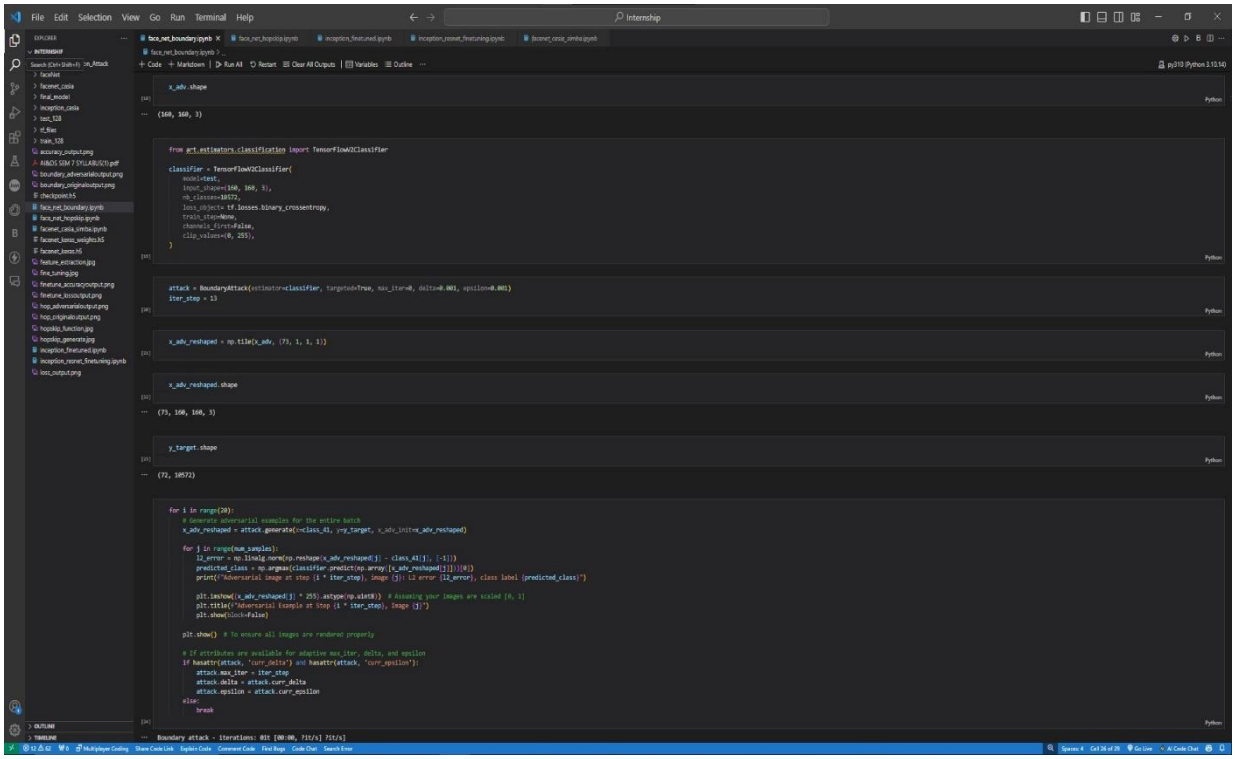
Confidence after attack: 0.1246

5.2 Boundary/Decision Based Attack: - Boundary Attack is a black-box adversarial attack that starts with a large, easily detectable perturbation and iteratively reduces its size while maintaining its adversarial properties. The attack performs a random walk along the decision boundary of the model's classification regions, gradually refining the adversarial example to find the smallest possible perturbation that still causes misclassification.

Internal Working:

- The attack starts with an initial adversarial example, x_adv_init, which is a perturbed version of the original input x that is already classified incorrectly by the model.
- The attack iteratively refines the adversarial example by combining two types of steps, **Orthogonal Step & Target Step.** Perturbation in a direction orthogonal to the decision boundary to ensure the adversarial example remains adversarial. Perturbation towards the target class in the Target step.
- Unlike gradient-based attacks, the Boundary attack does not rely on gradient information. It uses random perturbations and acceptance criteria to gradually find adversarial examples.
- Calculate the L2 norm of the perturbation to measure the magnitude of the change between the original and adversarial images.
- Adjust the '**delta**' and '**epsilon**' parameters based on the current state of the attack to fine-tune the perturbations.
- The process is repeated for a maximum of '**max_iter**' iterations. Each iteration may involve up to '**num_trial**' attempts to find a successful perturbation.
- The final adversarial examples are returned in an array.

Boundary Attack Function & Parameters: -



Function:

BoundaryAttack (estimator=classifier,

targeted=True,

max_iter=0,

delta=0.001,

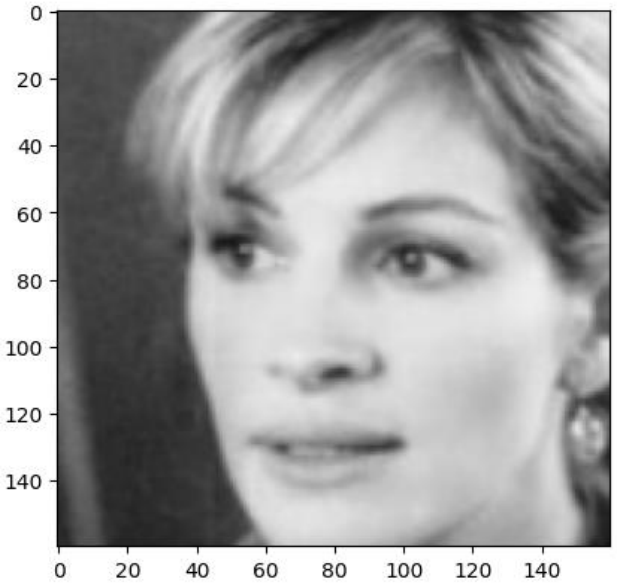
epsilon=0.001)

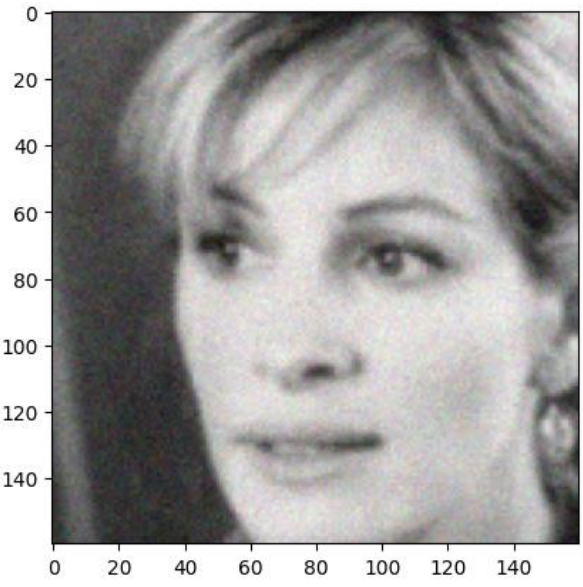
Parameters:

- classifier: The model to be attacked.
- targeted=True: The attack aims to modify the input to be classified as a specific target class.
- max_iter=0: Initially set to 0, this will be updated in the loop.

- max_eval=1000: Maximum number of evaluations (queries to the classifier) allowed for each iteration.
- init_eval=10: Initial number of evaluations for the gradient estimation.

Results: In this particular class we took all the images whose original confidence score before attack was > 0.99. We found 72 total images satisfying this criterion and out of these 72, we were able to achieve a successful attack on 48 images leading to a success rate around 66.66%. And overall, the Structural Similarity Index Measure (SSMI) score is found to be 0.8408 leading to a very similar image as original with no visually perceivable perturbations in the adversarial image. Below is representation of one of the image from the class 41 in the dataset.





Original Image
Original label: 41
Confidence before attack: 0.999044

Adversarial Image
Original label: 38
Confidence before attack: 0.15139474

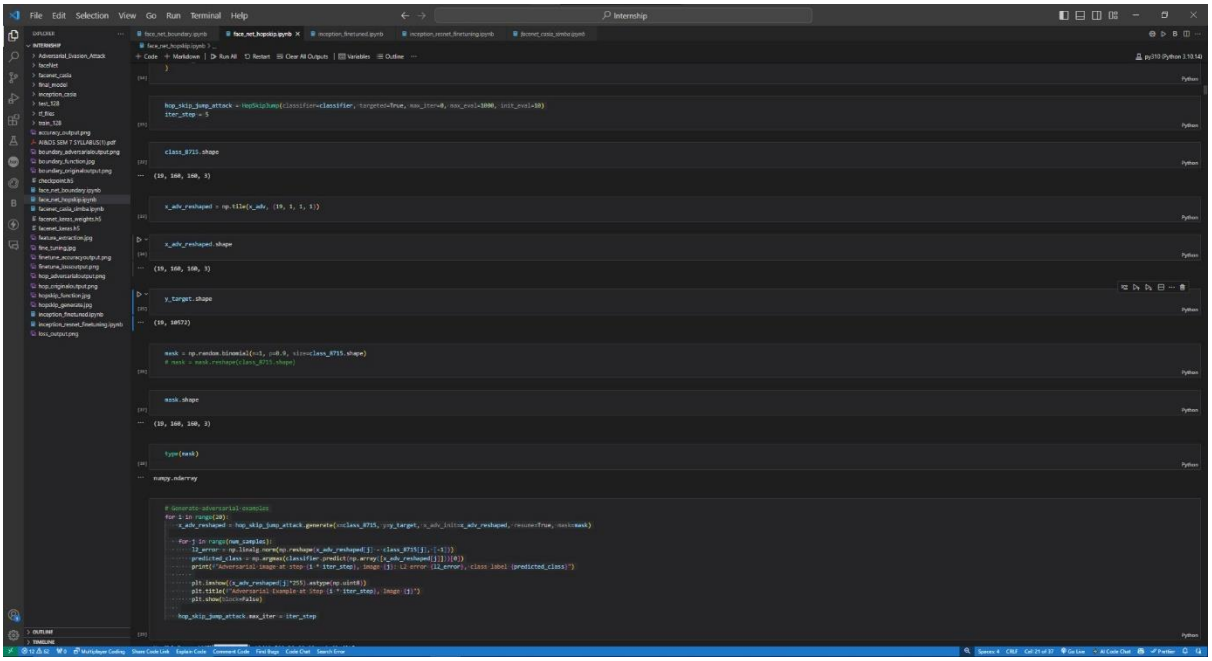
5.3 HopSkipJump Attack:- It is a black-box adversarial attack that iteratively searches for the decision boundary of a model using output labels. It combines binary search and gradient estimation techniques to generate adversarial examples efficiently. The attack minimizes the perturbation size while ensuring the perturbed image remains misclassified, making it highly effective in black-box settings where only the model's output can be queried.

Internal Working:

- The attack starts with an initial adversarial example (x_adv_init). This example is typically a random perturbation of the original input or an input from a different class.
- Estimate the gradient of the loss function by querying the classifier around the decision boundary. This is done using finite differences by evaluating the classifier's decisions on slightly perturbed versions of the current image.
- Use the classifier's output to determine the distance and direction to the decision boundary.
- Restrict the number of queries (max_eval) to the classifier for efficiency, making the attack practical even with limited access to the model. Perform small perturbations iteratively to gradually misclassify the image, updating the maximum number of iterations as needed.
- Use a binary mask to specify which pixels in the image can be altered, allowing for targeted perturbation only in certain areas.
- Calculate the L2 norm of the perturbation to measure the magnitude of the change between the original and adversarial images. Continuously check the classifier's

prediction on the adversarial example to ensure the targeted misclassification is achieved.

HopSkipJump Function and Parameters: -



Function:

hop_skip_jump_attack = HopSkipJump (classifier=classifier, targeted=True,

max_iter=0,

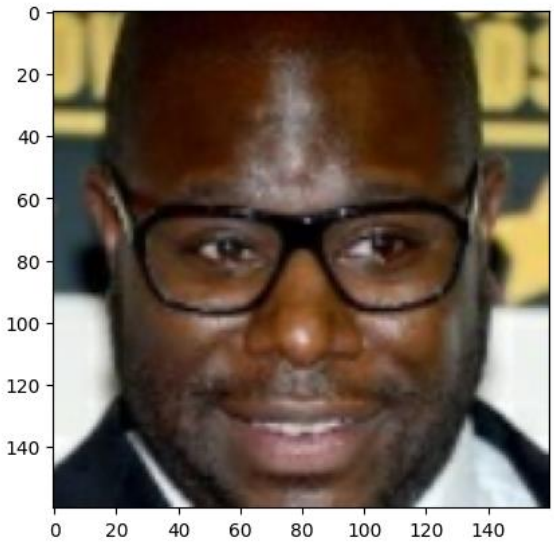
max_eval=1000,

init_eval=10)

Parameters:

- Estimator (classifier): This is the target model to be attacked.
- Targeted (targeted=True): This specifies that the attack is targeted.
- Max Iterations (max_iter=0): The maximum number of iterations the attack will perform. This parameter controls how many steps the attack takes to refine the adversarial example.
- Delta (delta=0.001): The step size for the perturbation in each iteration. It defines how large each individual step in the adversarial perturbation process is.

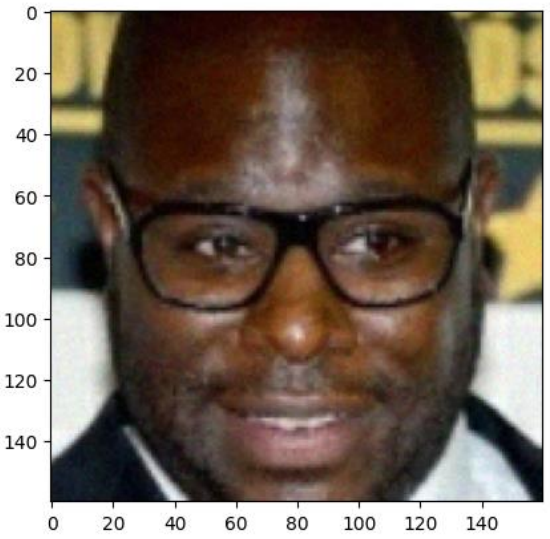
- Epsilon (epsilon=0.001): This controls the maximum allowed perturbation. It ensures that the perturbation does not exceed a certain threshold, maintaining the adversarial example's similarity to the original image.
  Results: In this particular class we took all the images whose original confidence score before attack was > 0.99. We found 19 total images satisfying this criterion and out of these 19, we were able to achieve a successful attack on 16 images leading to a success rate around 84.21%. And overall, the Structural Similarity Index Measure (SSMI) score is found to be 0.8140 leading to a very similar image as original with no visually perceivable perturbations in the adversarial image. One of the example is below for reference:



Original Image

Original label: 8715

Confidence before attack: 0.9911



Adversarial Image

Adversarial label: 38

Confidence before attack: 0.1011

6. References: -

- SimBA Attacks: are efficient black-box attacks that exploit the confidence scores to create the adversarial perturbations (Guo, Gardner, You, Wilson, and Weinberger)

- Boundary Attack: starts from a large adversarial perturbation and performs random walks on the decision boundary while keeping adversarial (Brendel, Rauber and Bethge).

- HopSkipJumpAttack (HSJA): boosts BoundaryAttack by estimating the gradient direction via binary information at the decision boundary, which is based on a Monte Carlo estimate (Chen, Jordan, and Wainright)

- Adversarial Attacks against Face Recognition: A Comprehensive Study