PSY 6990

Project Report;
Adversarial attack on Google's Inception v3 Image Classifier

By

Nisarg Dave, Erin Richie, Sivaramakrishnan Sriram

PSY 6990: Explanations in AI Systems

Project 3

Overview

Abstract:

The adversarial attack on the Google's inception v3 is presented below. We performed the adversarial attack on the classifier using Fast Gradient Step Method with two different styles,

1. **Targeted attack**
2. **Non -Targeted attack**

After performing attack and we collected various non trivial, contrastive examples to demonstrate the systematicity of exploit. At last we are presenting the adversarial cases with samples of explanation. The training and explanations for the same is provided in the last part of the report.

Introduction:

The adversarial attack for the DNN based on Deep convolutional network. The system is pretty complex but robust. The system is robust enough for each cases. The main objective is to perform the adversarial attack on DNN of Google's inception v3.

The attack was targeted and random using the noise generator and perturbation is applied system wide for getting wrong predictions. The predictions should be tweaked to be wrong or expected. We declared the exploit and explained the systematic approach to the end users.

Explanation of the System:

Google's Inception v3:

The Image recognition is primarily the tough task for the computers. A human brain can interpret things easily and even it can distinguish it by seeing them. The vision is more advanced and natural when it comes to human perception. Google's inception v3 has good computer vision capabilities and can recognize the things with the exceptional power of trained deep convolutional neural network.

It's developed and validated considering the academic benchmark ImageNet. Google's system can identify the 1000 different classes. The system was tested on benchmarked computer vision system ImageNet. The classifier was used for the transfer learning too.

DNN Architecture and Structure:

DNN is Deep convolutional neural network. It has many convolutional and pooling layers coupled with each other. It's pretty complex in architecture with lots of nodes.

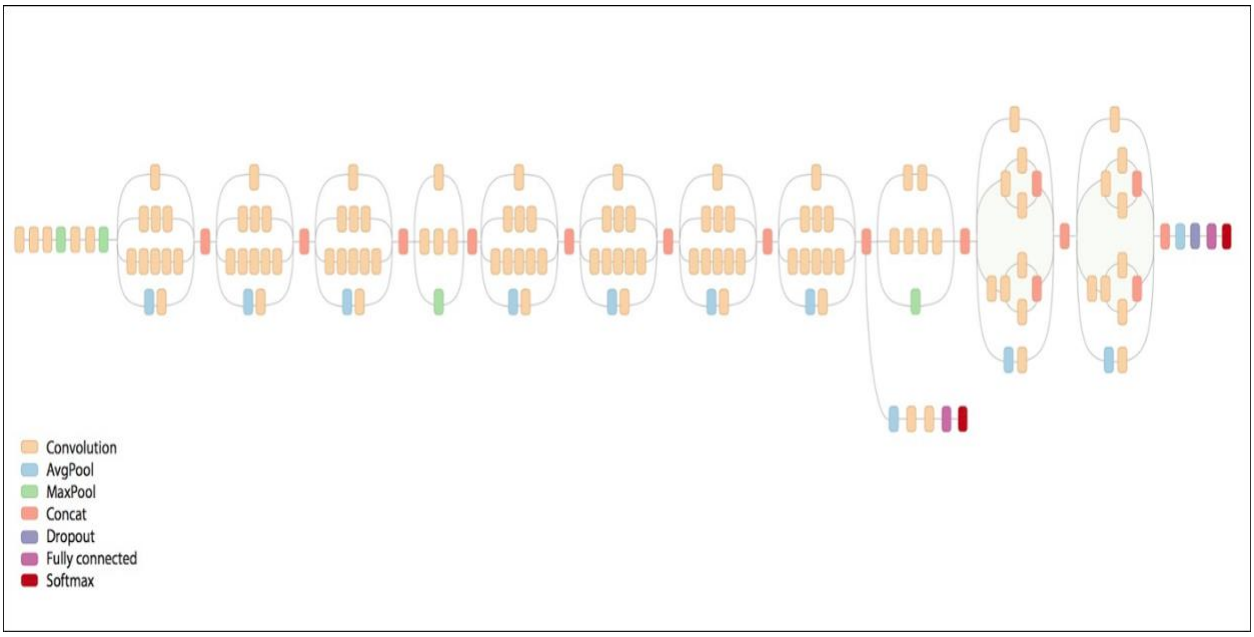

Image 1: Google Inception v3 DNN Architecture

Dropout method was used for avoiding the overfitting. The dropout based on node activations was the primary method for fully connected DNN. The softmax regularization was used to optimize the weight and activation functions.

Training:

After selecting the architecture, the complex architecture needs the qualitative training. The DNN was trained on very large training data and the data was broader in category, contains the various types of images for the same objects. The system was trained to recognize almost all type of defined image classes. For examples, the image of single flower, the grid of different flowers and the flower vase filled with flowers should be recognized correctly by the system.

For creating adversarial exploit, we need to understand all the expect related to the system. We need to get deeper knowledge regarding,

- System architecture
- Training features and data
- Training mechanism
- Regularization and principle components
- Testing and model performance parameters

Validation:

The DNN model is robust enough to reach nearly 97% accuracy for all classes.

Adversarial Attack:

Attacking the DNN using the adversarial sample is iterative process of exploiting the ML model mechanism. DNN can be fooled using the Adversarial samples, either create your own or tweak the existing training data.

We are examining the classifier of image. The task is classification so we need to change the Targeting class. For the classifier, let say if the system can identify the Rose then it should be able to identify the images of roses with different colors, background, type and orientation.

For fooling system with adversarial samples, we can create the different images samples for flowers such as,

- Rose with different colors
- Rose with different orientation
- Rose with different locations and environment

We can fool system by all of these kind of samples but It's easy to make system robust enough for those samples. We can re-train the model and make system identify every case correctly. So We thought the other way to make this exploit. This way is more robust and better enough to attack any image classifier.

So we used the fast gradient step method algorithm for creating the adversarial samples.

FGSM - Fast Gradient Step Method:

The FGSM is the Gradient based optimizer, The method is optimizer which optimizes the error function. The perturbation should be applied in each step. The FGSM working is discussed below,

- Start from random gradient direction
- Move in gradient direction and add gradient noise in each step
- The rate of learner is significantly fast, each step adds more proportional noise.
- Optimizes the error in each step and terminates when the error is maximum.

Non- Targeted attack using FGSM:

The simple attack in which the source image is required. The non targeted attack focuses on getting wrong predictions only. The goal is to get the wrong prediction for given samples.

- Normed vector of gradient is added in each step using the additive measure.

Ex. adv_img, noise = non_targeted_attack(img)

Targeted attack using FGSM:

The attack is complex and the goal is to get the object classified as some specific class. The desired class should be pass as the function call argument.

The classifier will find the feature weights for that particular class and then converges the input image features to the desired one.

It's like changing the feature set using the required class label

- Normed vector of gradient is added in each step using the additive measure with scaled transformation using the required label.

Ex. adv_img, noise = non_targeted_attack(img,859)

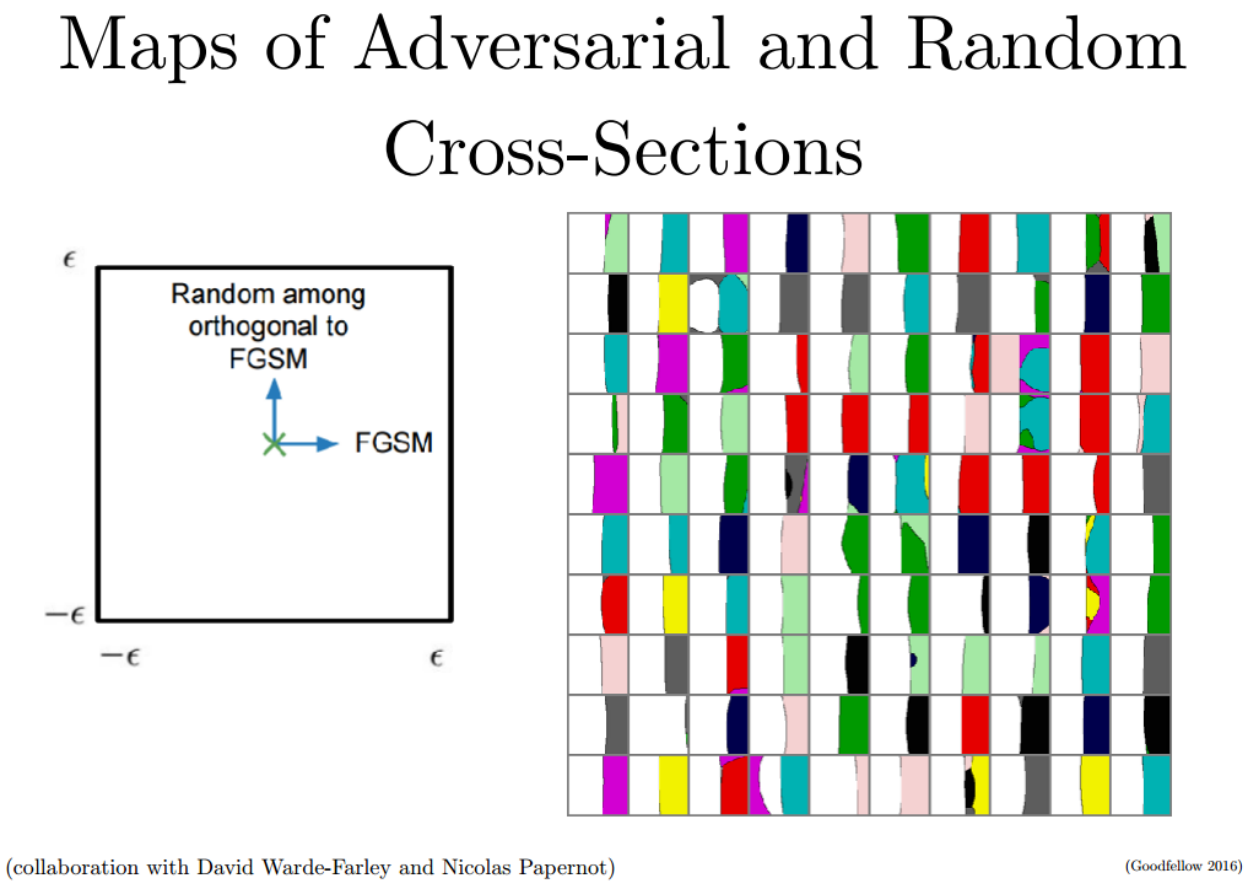

Image 2: Maps of Adversarial Cross Sections

Attacking the Classifier:

Non-Trivial Exploit: The cases in which the system fails to predict the correct class label. The cases on which classifier was trained and supposed to give the correct results but it fails!

1. The Car example using Non-targeted attack

Classifier was trained on car but it still can’t classify the adversarial sample correctly. It does classify it as an triumphal arch.

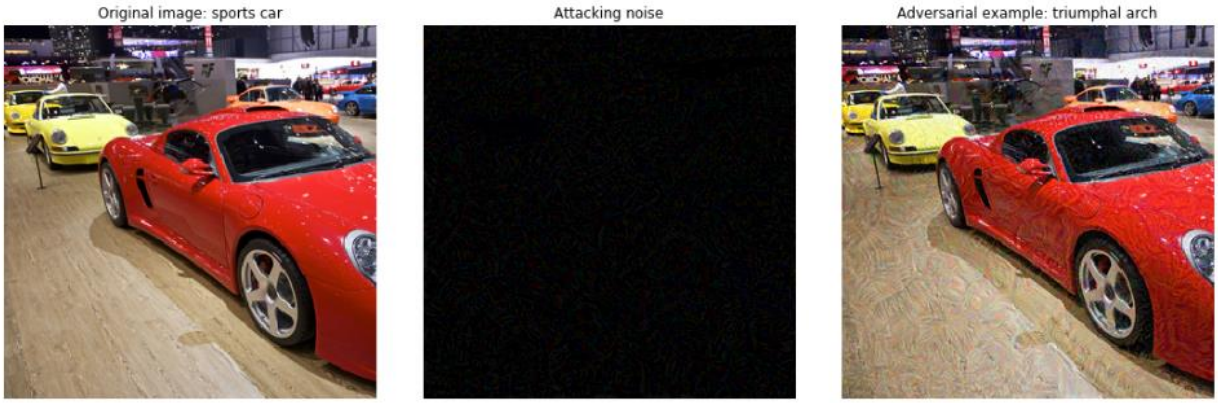

Image 3: Prediction of DNN on Original and Adversarial image of car

2. The Car example using targeted attack

Classifier was trained on car but it still can’t classify the adversarial sample correctly. It does classify it as a toaster.

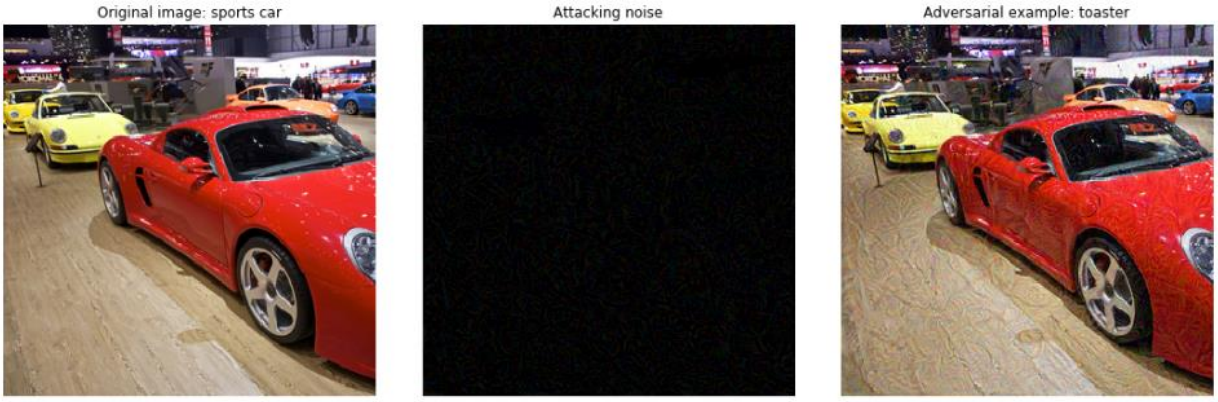

Image 4: Prediction of DNN on Original and Adversarial image of car

3. Banana example

Classifier was trained on banana but it still can’t classify the adversarial sample correctly. It does classify it as a mosquito net.

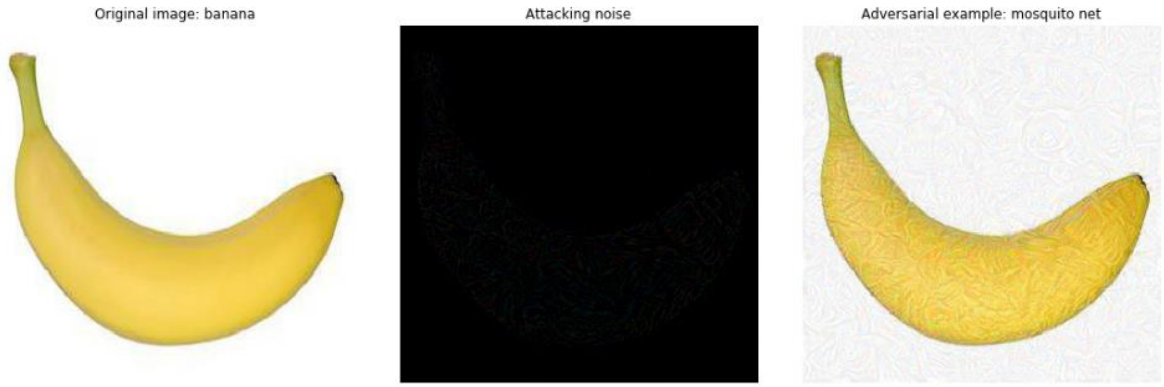

Image 5: Prediction of DNN on Original and Adversarial image of banana

Contrastive Cases: The cases in which the system is operating as expected, and then alternative cases that appear equivalent, but the system fails in some way.

1. The Indigo Bunting Bird

The classifier knows the Indigo bunting and can identify it but after adding noise it fails.

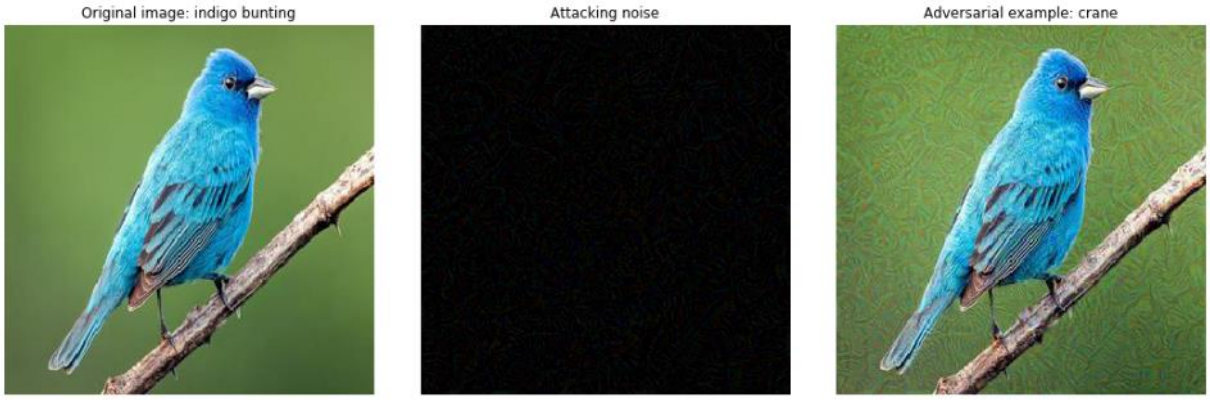

Image 6: Prediction of DNN on Original and Adversarial image of bird

Now let's try this for the group of indigo bunting birds.

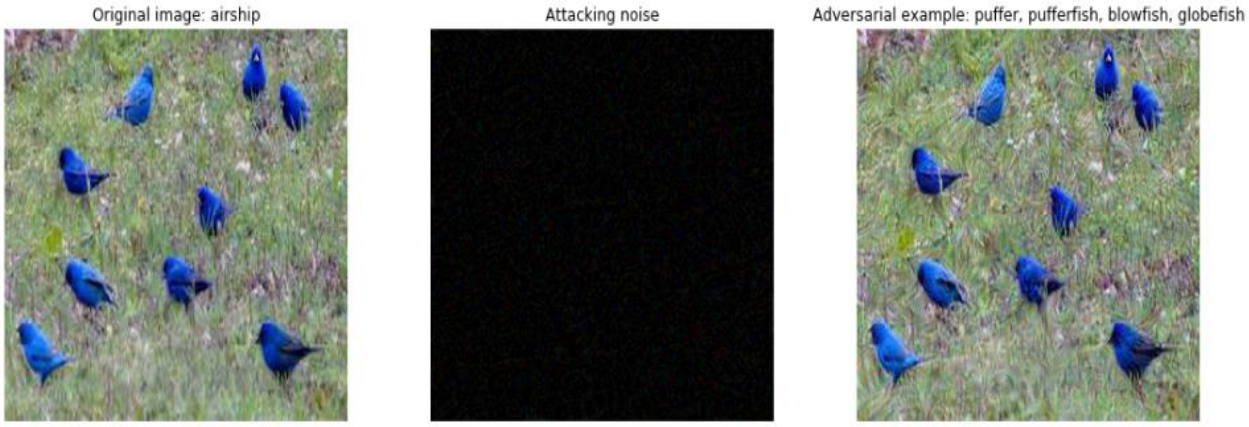

Image 7: Prediction of DNN on Original and Adversarial image of group of bird

We can see here that the classifier can classify single Indigo bunting bird but it fails when the group of birds are there. So this is contrastive case for the birds. Furthermore if we test it on another grid image,

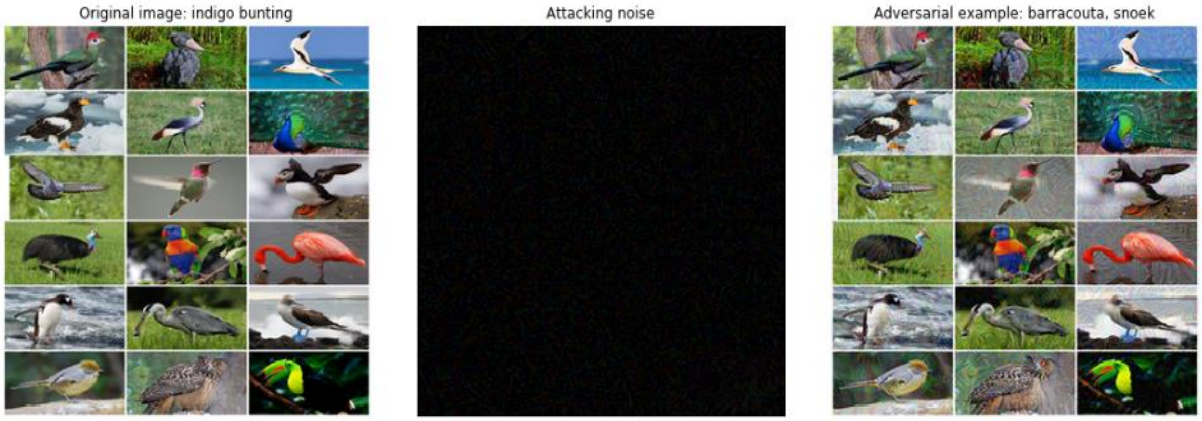

Image 8: Prediction of DNN on Original and Adversarial grid image of birds

Once again the system can classify only one bird out of many of them and after adding noise it even fails in that.

2. Flower example

We can clearly see this Contrastive case where grid of flowers is classified as vase, even before attack. After attack, it's classified as Wool.

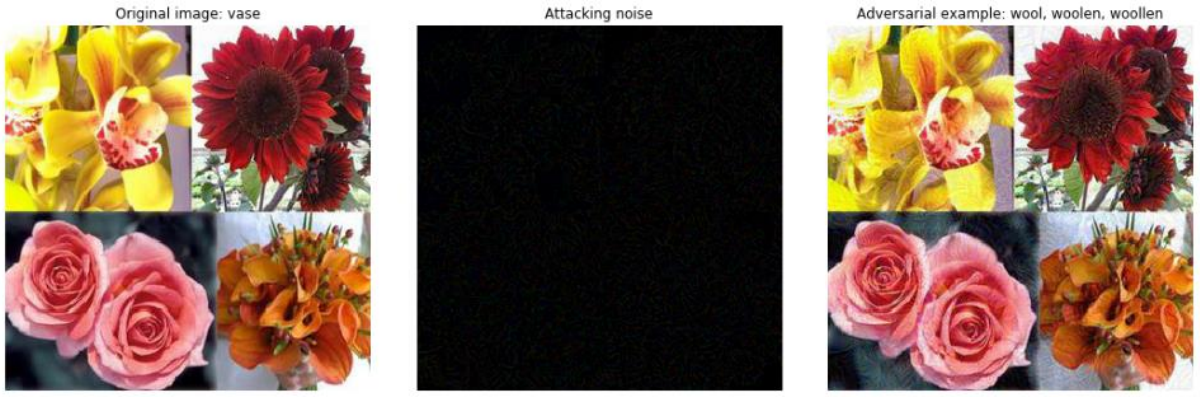

Image 9: Prediction of DNN on Original and Adversarial grid image of flowers

Before the attack, grid of flowers got still classified incorrectly (as Handkerchief). After the attack, flowers are classified as King snake.

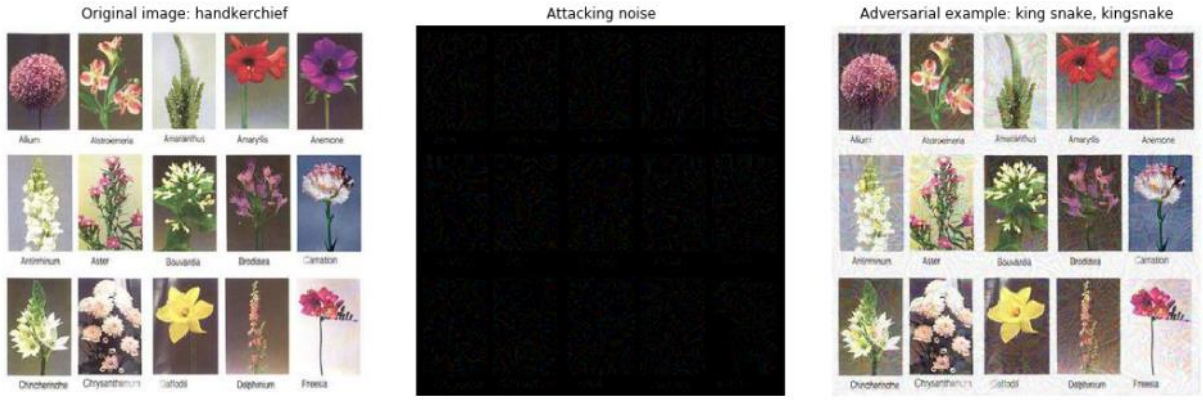

Image 10: Prediction of DNN on Original and Adversarial grid image of flowers

Furthermore, image of single parrot is classified correctly but the group of parrots is classified incorrectly before attack. After attack it is classified for sure incorrectly.

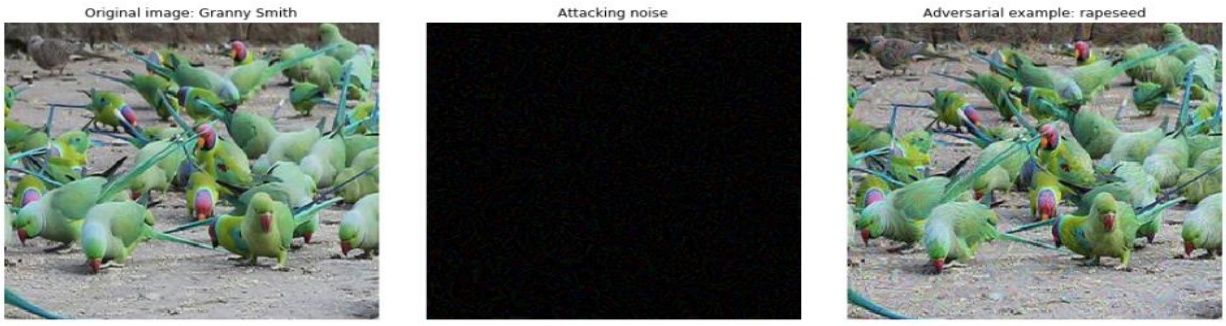

Image 11: Prediction of DNN on Original and Adversarial image of parrots

Systematicity of the approach: Example cases for the exploit can be applied to various different images.

Now let's apply the exploit to broader category of image samples. Let's test the approach on wide variety and different domain of images.

1. Cell Phone example

Before the attack, cell phone is classified correctly. After the attack, cell phone is classified as Lollipop.

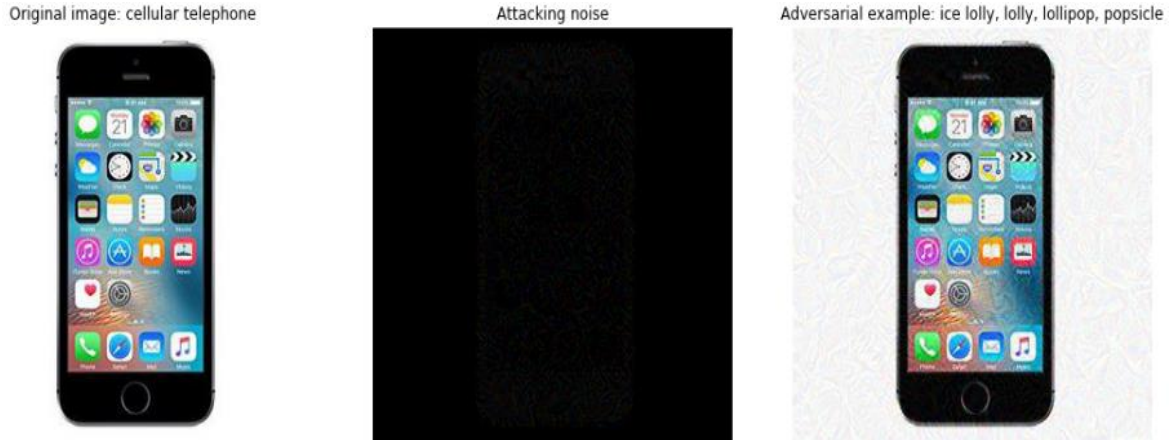

Image 12: Prediction of DNN on Original and Adversarial image of cell phone

2. Military uniform example

Before the attack, military uniform is classified correctly. After the attack, military uniform is classified as rapeseed.

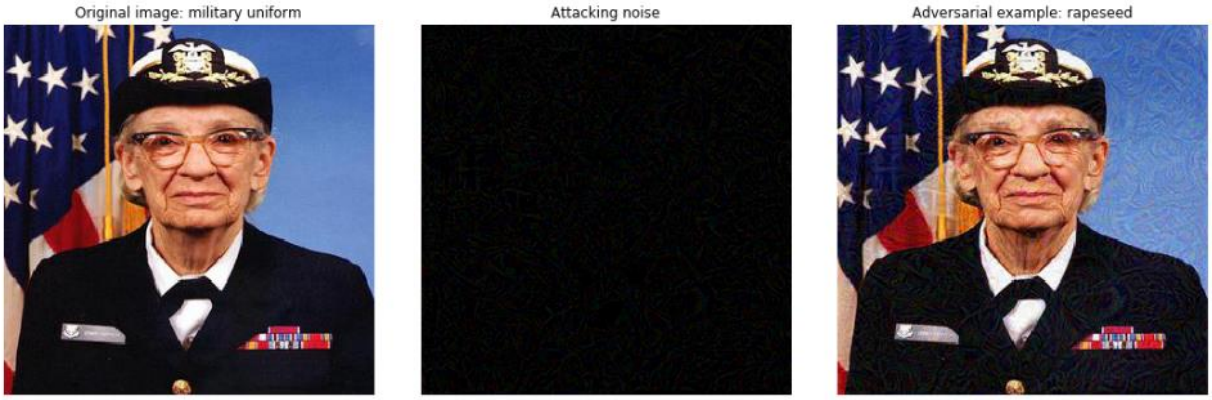

Image 13: Prediction of DNN on Original and Adversarial image of military uniform

3. Scooter example

Before the attack, Scooter is classified correctly. After the attack, Scooter is classified as daisy.

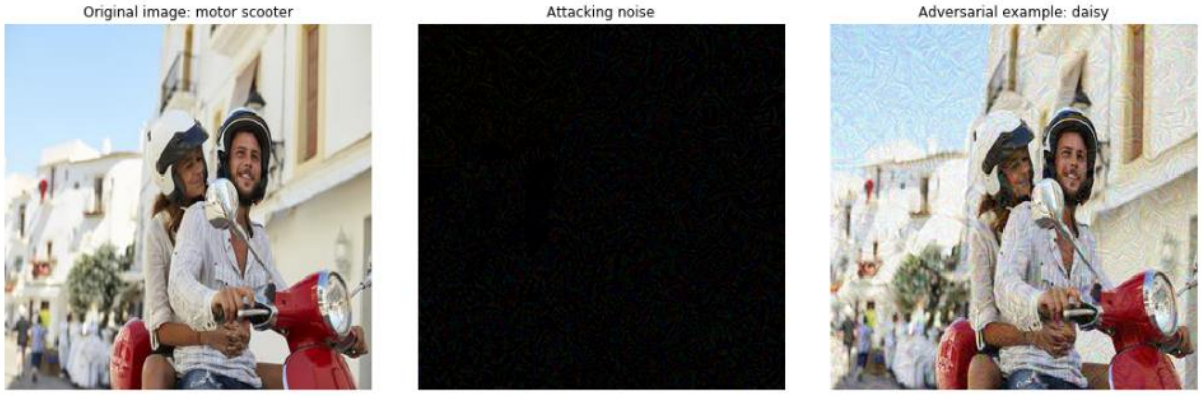

Image 14: Prediction of DNN on Original and Adversarial image of scooter

4. Moped example

Before the attack, Moped is classified correctly. After the attack, Moped is classified as racket.

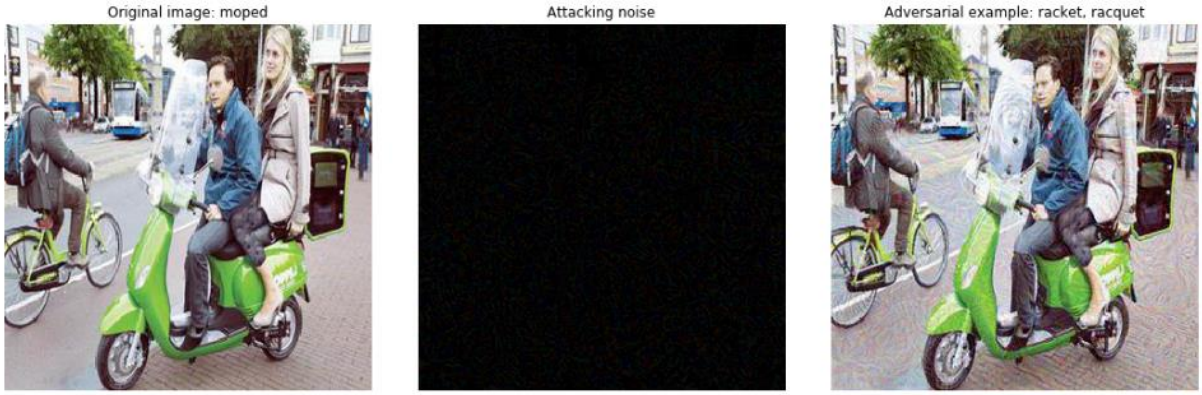

Image 15: Prediction of DNN on Original and Adversarial image of moped

Outlier Cases: The Outrageous cases which can be exemplified as Outliers to the system.

1. Game Character (Yennefer from Witcher 3)

Before the attack, classifier can’t classify the image correctly. After the attack, classifier surely can’t classify the image.

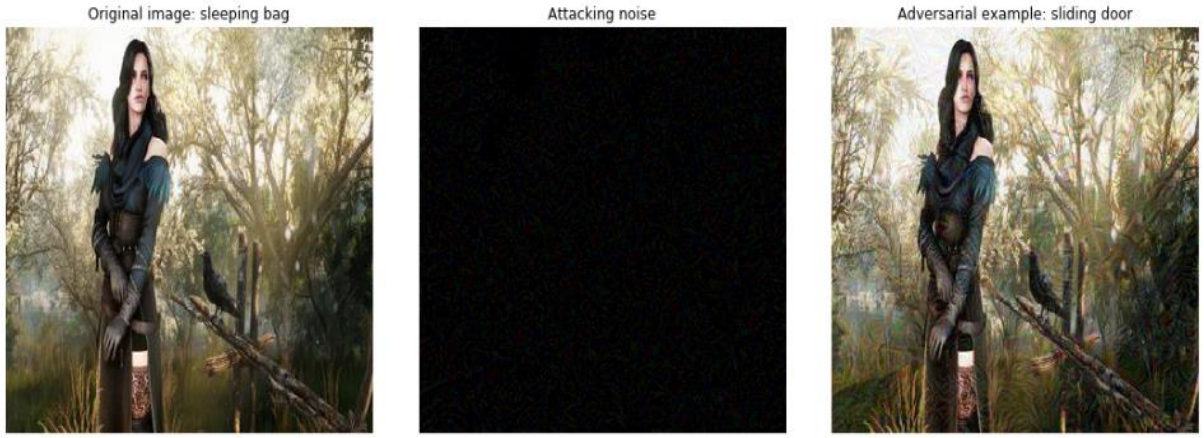

Image 16: Prediction of DNN on Original and Adversarial image of Yennefer

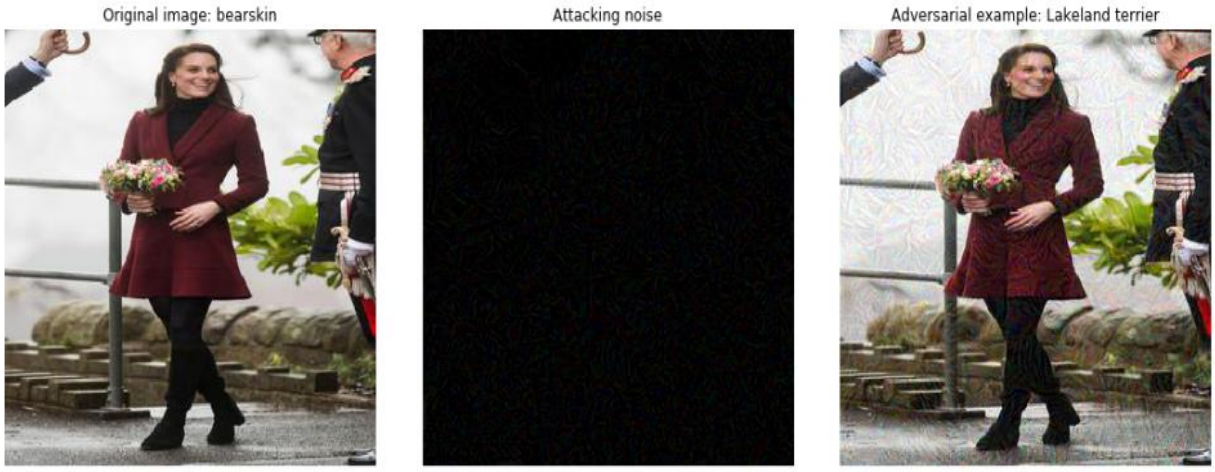

Image 17: Prediction of DNN on Original and Adversarial image of Kate Middleton

User Training and Explanation:

Please refer to the following guide to help you, as a user, understand the basics as to why your image search and classifying system, like a Google Image search, may be malfunctioning and how you can avoid and work around cyber attacks that may target such systems.

How and Why the Attack Works:

To better understand how an attack on Image classifier works, let's first consider how the system works as a whole. Search engines use keywords and phrases to gain understanding of how a person is interpreting an image or web page. While we, as humans, see pictures and read words- search engines are 'seeing' patterns and similarities in the way humans react to data. For example, when we see a picture of a car and know its a car because we see for wheels and it matches what we know to be the shape of a car, a search engine or image classifier may have a compilation of data that says when a certain number of blocks of color, or pixels, exists in a certain area of an image, that means it's a car. So while we actually see a car, the classifier may be calculating if the number of squares in the bottom half of the image match the 300 other images that humans have labeled as cars.

Hackers understand both the human level and the computer system level on which images are labeled. To confuse the system, hackers may write a program that rearranges all the blocks of color (pixels) so that they are clustered differently. What's more, is that hackers build these programs to rearrange the pixels in such a way that the image will still look like a car to the human eye, but to a classification like Google Images it looks completely different. In this way they cause confusion. You may type the word car into your search bar and see a dog instead. Or perhaps you are searching for a picture of an orange and a picture of a car comes up.

In the images below, for example, you may notice that something is 'off' about the car image- but it still looks like a car. The 'swirls' we see are a slight rearrangement of pixels that is enough to cause a system to 'see' a completely different image.

Image 18: Attack example, the left image is untouched by the attack and the right image has been attacked

This attack doesn't work when the image wasn't classified correctly by the system to begin with. For example, if a bug in the system already causes a group of birds to be mistakenly classified as a mango, the hacker's attack and rearrangement on the system may cause the image of birds to be labeled as a stack of pancakes rather than a mango- but the image is still incorrectly labeled.

In the images below, you can see that the 'swirls' from the rearranged pixels still exist in the second image, just as in the car example. However, this image was originally identified as an airship and after the attack was identified as a blowfish. Since both labels are incorrect, the attack is trivial and useless.

Image 19: The left image has not been attacked, the right image has been attacked

How to Avoid this Error:

To avoid this error while using Google Images, you can do several things.

1. If you are searching for images to use in a report or explanation, do not blindly copy and paste images into a report from such a search engine. Check out the source of the image and be sure it matches the message you are trying to send.
2. If you are making a website, consider adding captions to your pictures so that your patrons have more than one way to identify what they are looking at in an image.
3. If you are coding a system, consider new ways of image identification that acts more like the human eye and relies less on clumpings and arrangements of pixels.
4. For developers, use multiple training images such as the image with different color, orientation, blurred background or with varying environment.
5. For developers, use advanced computer vision for extracting more features for given input of image.
6. Use the Reinforcement learner for better adaptation of new test cases.

Conclusion:

In conclusion, we can say that today's machine learning systems are not robust enough in front of adversarial attacks. The attack can be applied with or without access to the training data. The demonstration of non-trivial, contrastive and outlier cases was outlined by the various explanations, In conclusion, systematicity was proved for broader variety of images.So the approach is robust and can be apply to any example. In last, we presented the explanation of how it works and how to avoid that.Our approach is capable of fooling pretty much any machine learning classifier in all the cases where there were no attempts to defend them.