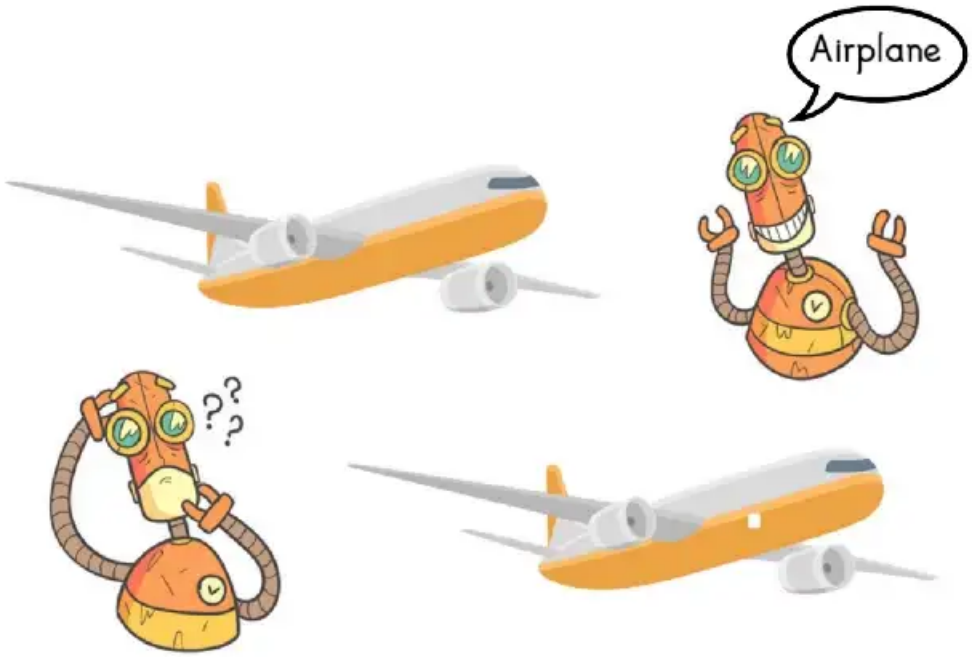
# One pixel attack

Pavan Sudeesh Peruru.
IMT2018517.

Neural networks have been used to perform various machine learning tasks, among which image classification is one such task where convolution neural networks(CNN), a class of neural networks, have found great success. With CNNs such as AlexNet and VGG16 achieving human-competitive results, deep neural networks are the current de-facto standard for image classification. But do you know by just changing one pixel we can entirely change the classification of the image.




Modifying a pixel in the image changes the classification prediction of a Neural Network

where introducing a small amount of noise that is supposedly imperceptible to the human eye causes neural networks to misclassify images, the one-pixel attack aims to achieve the same while being limited by an extreme constraint, modify only a single pixel of the image.

- one-pixel attack is more threatening than other types of adversarial attack because it is a type of black box attack it does not require any additional information of the DNN models.

- Its effectiveness was 31.40 % on the CIFAR 10 dataset and 16.40% on the ImageNet dataset.

Technical challenges in one pixel attack:

(1) Extremely Small Modification. The one-pixel attack modifies only one pixel in an image, which is significantly less than other types of adversarial attack.
(2) Randomness of Pixel Modification. For an image, there may be more than one feasible pixel that can cause the change of classification.

Coming to the detection of one pixel attack there are two methods to detect a one pixel attack.

1. Trigger detection.
2. Candidate detection.

## Adversarial Attack:

Adversarial attack means attackers intend to mislead to classifiers by construction adversarial samples. It is impossible to learn the Internal characteristics of DNN models, so they invented Black box attack here only pixel is allowed to be modified. As we are modifying only one pixel it becomes extremely hard to find the one pixel. The one-pixel attack requires only black box feedback that is the probability label without any inner information of the target network, like gradients or structure.
Detection of Adversarial attack:

One method that was designed was to defend image adversarial attack using image compression. But it also have its own issues, compression may also lead to a large loss of classification accuracy of the attacked images. In particular, two novel detection mechanisms are proposed with one using a gradient calculation-based method and the other using a differential evolution-based method.

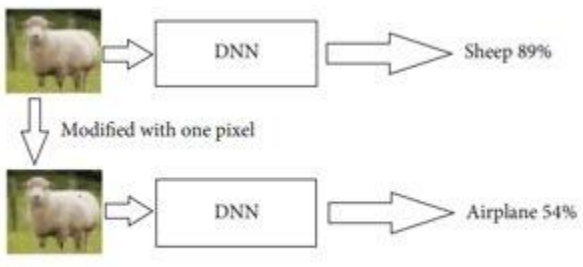



FIGURE 2: Illustration of one-pixel attack.

**Detection Model:** Suppose that a set of adversarial images, which are modified by the aforementioned one-pixel attack, are given to the system. With these given images, our objective is to distinguish which pixel has been modified by a one-pixel attack.

The **trigger detection model** is designed for white box detention that requires all the network information including inner gradients and network structure. Basically, here we detect a trigger for the image data.

**Candidate detection** is a type of black box detection where only the output probabilities of labels are needed for detection. In the candidate detection model, we aim to find a set of pixels as the candidate victim pixels. If the selected victim pixels include the pixel modified by the one-pixel attack, our detection is successful.

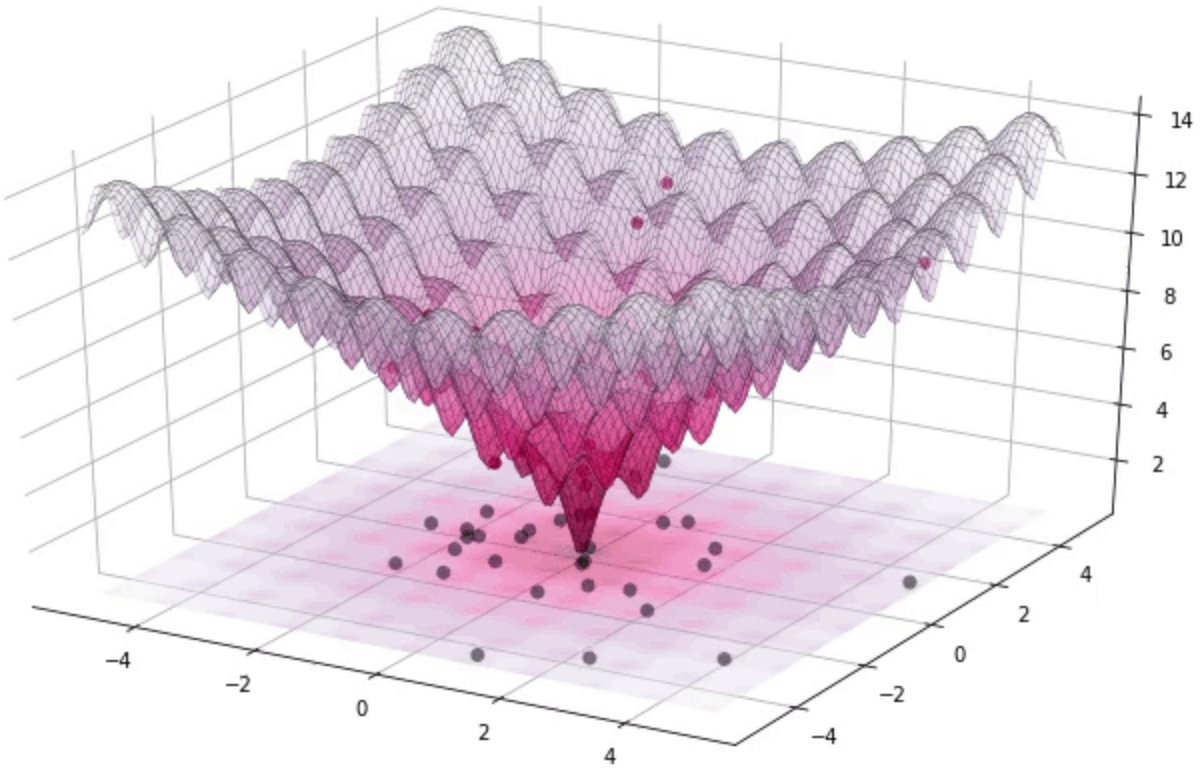
**Targeted vs. Untargeted Attacks:**
The objective of an untargeted attack is to cause a model to misclassify an image. This means we want to perturb an image as to minimize the confidence probability of the correct classification category and maximize the sum of the probabilities of all other categories.

The objective of a targeted attack is to cause a model to classify an image as a given target class. We want to perturb an image as to maximize the probability of a class of our own choosing.

1_one-pixel-attack-cifar10 code: Link

For this attack, we will use the Cifar10 dataset. The task of the dataset is to correctly classify a 32x32 pixel image in 1 of 10 categories (e.g., bird, deer, truck). The black-box attack requires only the probability labels (the probability value for each category) that get outputted by the neural network. We generate adversarial images by selecting a pixel and modifying it to a certain color.

By using an Evolutionary Algorithm called Differential Evolution (DE), we can iteratively generate adversarial images to try to minimize the confidence (probability) of the neural network's classification.




First, generate several adversarial samples that modify a random pixel and run the images through the neural network. Next, combine the previous pixels' positions and colors together, generate several more adversarial samples from them, and run the new images through the neural network. If there were pixels that lowered the confidence of the network from the last step, replace them as the current best known solutions. Repeat these steps for a few iterations; then on the last step return the adversarial image that reduced the network's confidence the most. If successful, the confidence would be reduced so much that a new (incorrect) category now has the highest classification confidence.

## Image Perturbation

To begin, we need a function to modify one or more pixels in an image.

We can define the perturbation of a pixel as a 5-tuple

$$\mathbf{x} = (x, y, r, g, b)$$

where $x, y$ are the coordinates of the pixel from 0 to 31, and $r, g, b$ are the red, green, and blue values from 0 to 255. Then multiple perturbations can simply be a concatenation of these tuples:

$$X = (x_1, y_1, r_1, g_1, b_1, x_2, y_2, r_2, g_2, b_2, \dots)$$

We could instead use an array of tuples, but the optimization algorithm we will use requires it to be a flat 1-d vector.

Then the function to perturb an image can take as an input the image and $X$, and output a copy of the image with each pixel at $x_i, y_i$ modified to have the color $r_i, g_i, b_i$. To speed up computation, we will batch together an array of $X$ perturbations, denoted $X_S$.

In the context of the one pixel attack, our input will be a flat vector of pixel values:

$$X = (x_1, y_1, r_1, g_1, b_1, x_2, y_2, r_2, g_2, b_2, \dots)$$

These will be encoded as floating-point values, but will be floored back into integers to calculate image perturbations. First we generate a random population of $n$ perturbations

$$\mathbf{P} = (X_1, X_2, \dots, X_n)$$

Then, on each iteration we calculate $n$ new mutant children using the formula

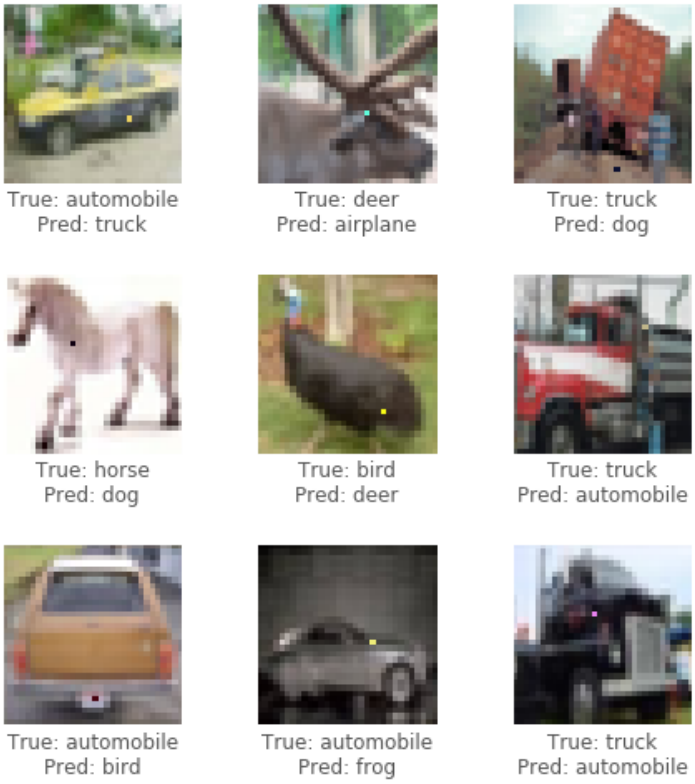
$$X_i = X_{r1} + F(X_{r2} - X_{r3})$$

such that

$$r1 \neq r2 \neq r3$$

where $r1, r2, r3$ are random indices into our population $\mathbf{P}$, and $F = 0.5$ is a mutation parameter. Basically, we pick 3 random individuals from the previous generation and recombine them to make a new candidate solution. If this candidate $X_i$ gives a lower minimum at position $i$ (i.e., the attack is closer to success), replace the old $X_i$ with this new one. This process repeats for several iterations until our stopping criterion, `attack_success`, which is when we find an image that successfully completes the attack.

See below for some examples of successful attacks:




Final results all are shown in the code. Link

## Conclusions

It appears that the accuracy of a model is not strongly correlated with the chance of performing a successful attack on an image. Perhaps surprisingly, the purely convolutional model is the most resistant CNN to these types of attacks. In addition, the capsule network CapsNet has the lowest attack success rate out of all the models, although it is still vulnerable to attack.

One pixel attack for fooling DNN code: Link

Imported all the required libraries then used the VGG model pre-trained weights and used some basic sequential layers and tested them in the CIFAR dataset. And results are shown in the code.