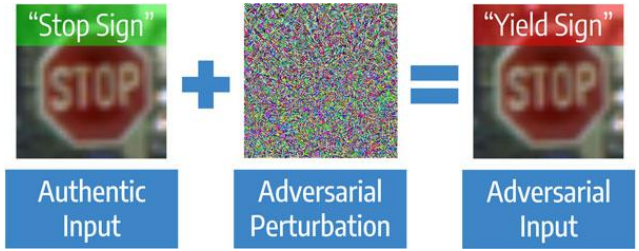
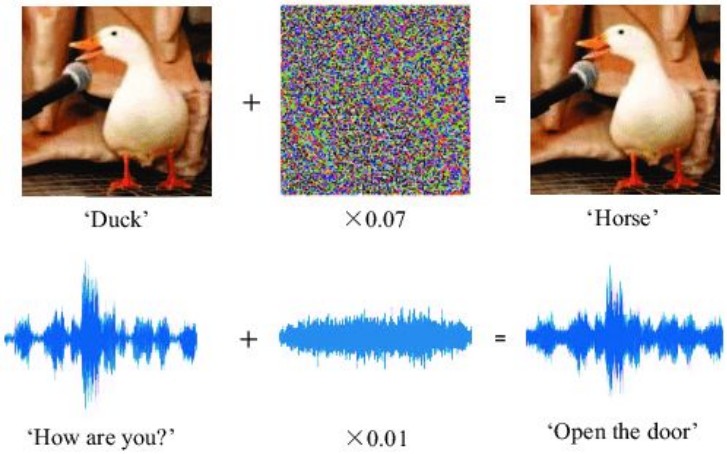
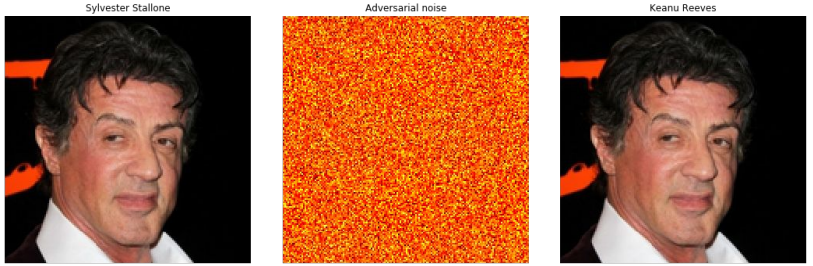
Adversarial Reprogramming of Neural Networks

Presented by Ramesh Kumar Sah





Adversarial Attacks



Adversarial Reprogramming

What is it?

Adversarial reprogramming is a way to repurpose an existing neural network for a new task chosen by the attacker, without the attacker needing to modify the network parameters.

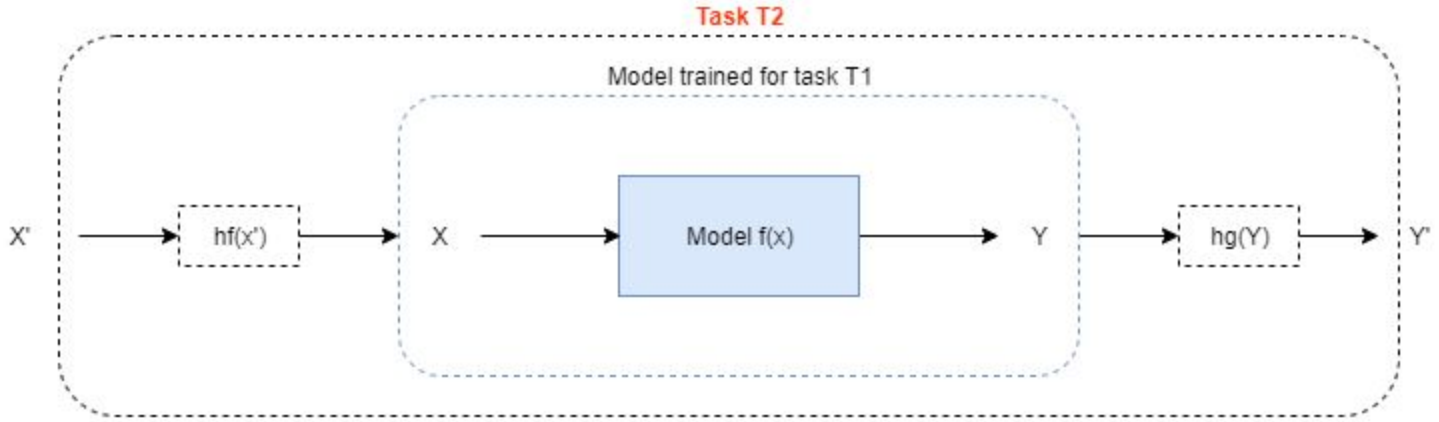
Finds a single adversarial perturbation, that can be added to inputs to a machine learning model in order to cause the model to perform a task chosen by the adversary.

How does it works?

The adversary first defines a hard-coded one-to-one label remapping function $Hg(.,\theta)$ that maps the output labels of the adversarial task to the label space of the classifier f;

And learning a corresponding adversarial reprogramming function $Hf(.;\theta)$ that transforms an input (X) from the input space of the new task to the input space of the classifier.

Threat Model



The adversary wishes to compute g(X')

hf converts inputs from the domain of X' into X

hg maps output of f(g(X')) back to outputs of g(X')

The parameters of the program are adjusted such that hg(f(hf(X'))) = g(X')

Is it Transfer Learning?

Very similar to the idea of transfer learning, but with a significant difference.

Transfer learning allows model parameters to be changed for the new task.

We don’t need to change the model parameters, but only access to the model such that we can feed inputs and observe outputs.

Nature of Adversarial Program

The adversarial program, P is formulated as an additive contribution to the input, and the same adversarial program is applied to all images.

$$P = \tanh(W \odot M)$$

where $W \in^{n \times n \times 3}$

n is the ImageNet width, and M is the masking matrix.

$$X_{adv} = h_f\left(\tilde{x}; W\right) = \tilde{X} + P$$

Here, W is the weights of a layer in a neural network, and is calculated using back-propagation.

$$\hat{W} = \underset{W}{\operatorname{argmin}} \left(-\log P(h_g(y_{adv}) | X_{adv}) + \lambda ||W||_F^2 \right)$$

Example of Adversarial Reprogramming

The authors repurposed an ImageNet model for the following adversarial tasks:

1. Square Counting
2. MNIST Classification
3. CIFAR 10 Classification

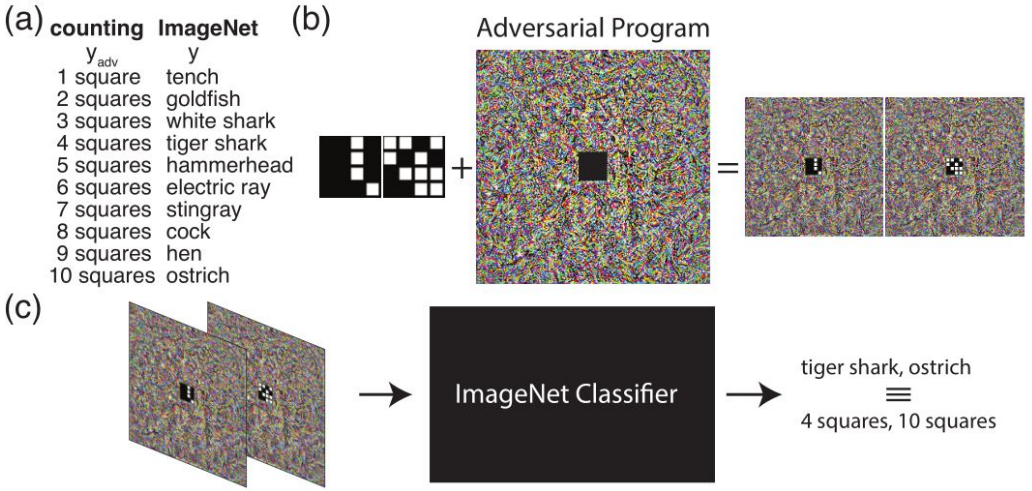


Figure 1: **Illustration of adversarial reprogramming.** (a) Mapping of ImageNet labels to adversarial task labels (squares count in an image). (b) Two examples of images from the adversarial task (left) are embedded at the center of an adversarial program (middle), yielding adversarial images (right). The adversarial program shown repurposes an Inception V3 network to count squares in images. (c) Illustration of inference with adversarial images. The network when presented with adversarial images will predict ImageNet labels that map to the adversarial task labels.

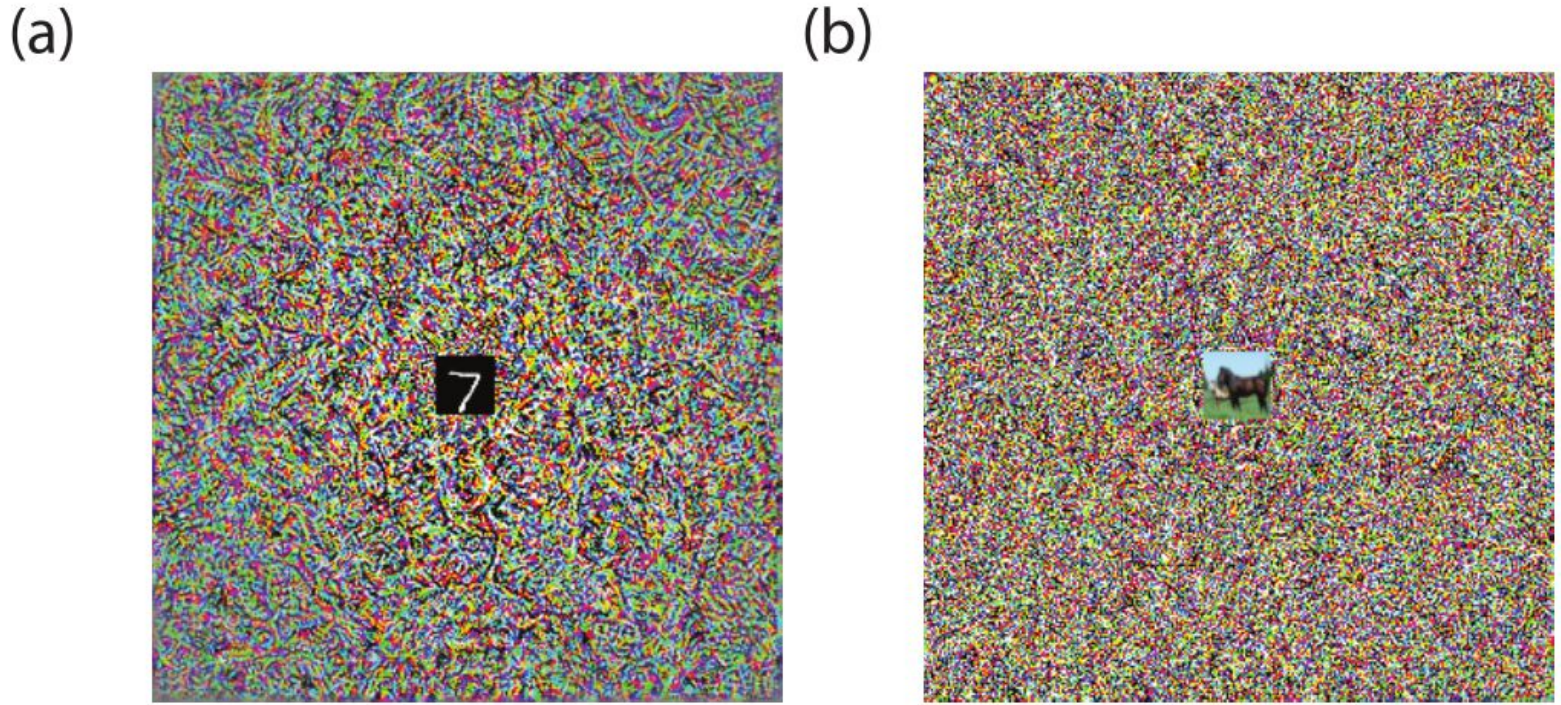


Figure 2: **Examples of adversarial programs.** Adversarial program which cause Inception V3 ImageNet model to function as (a) MNIST classifier. (b) CIFAR-10 classifier

After the optimization the adversarial program has minimal computation cost from the adversary's side as it only requires computing the Xadv.

Many other applications such as general purpose model, strengthening adversarial attacks, stealing of resources.