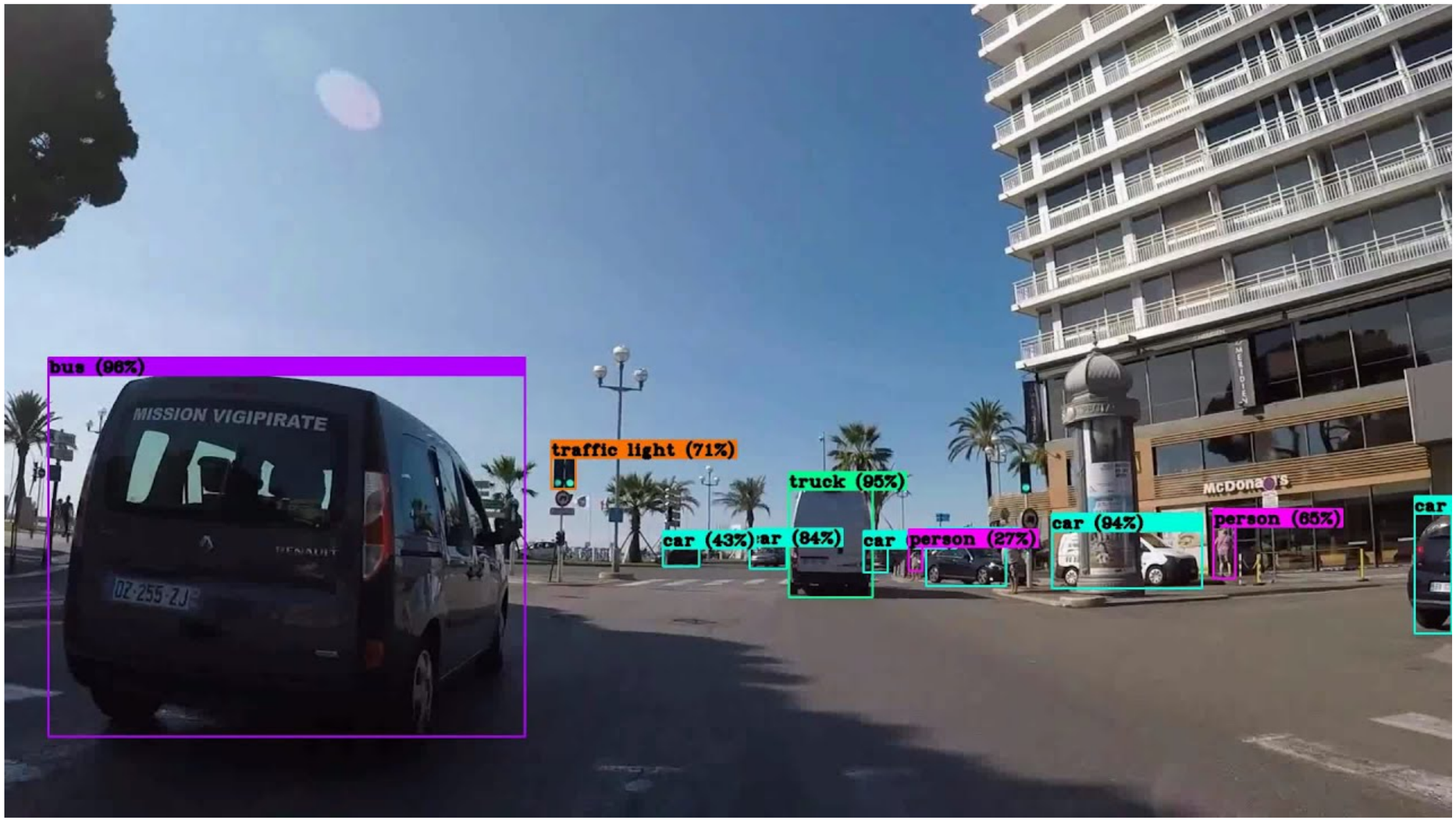
Machine Learning - Part 6

Apr. 24, 2025

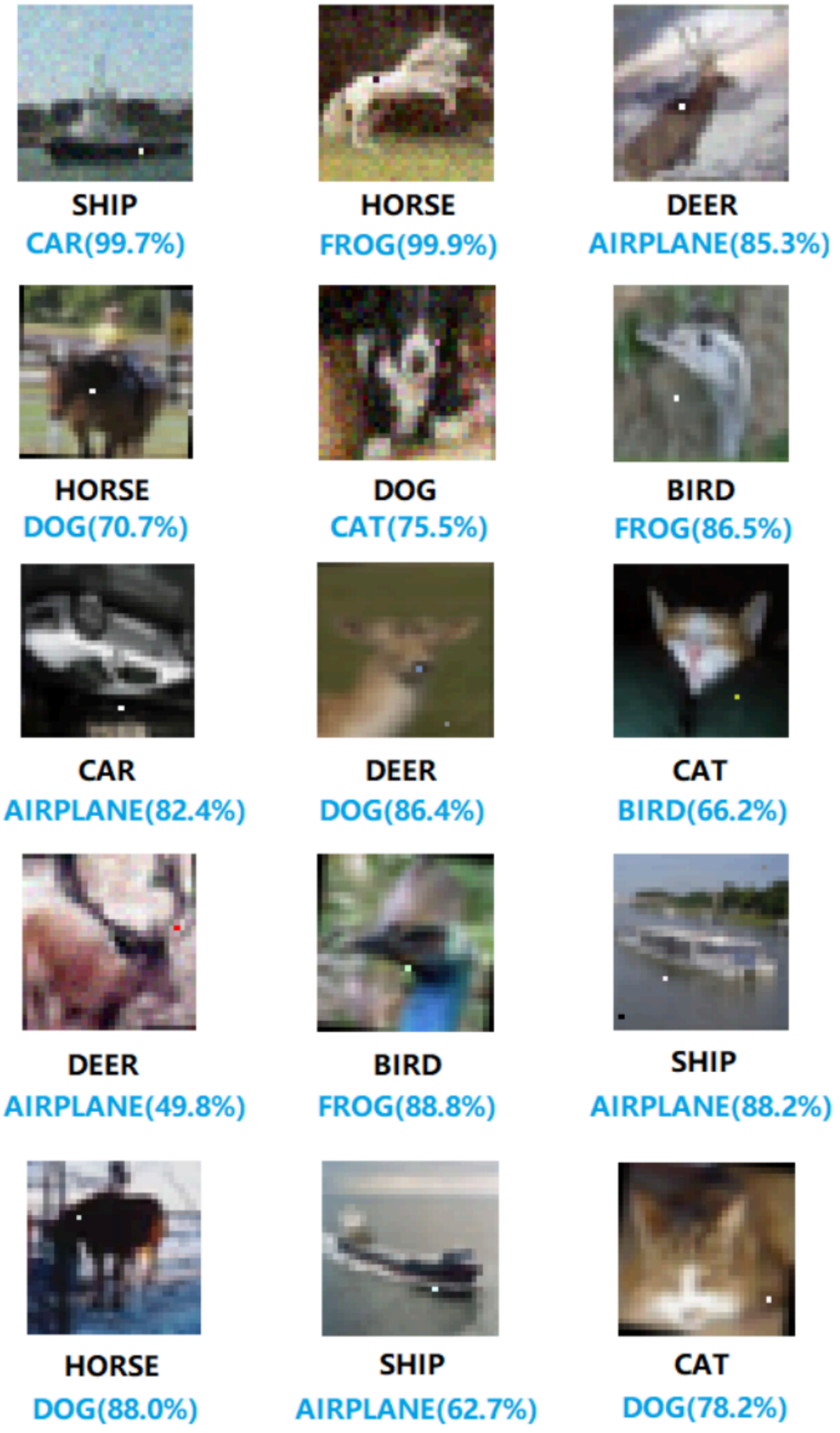
Unintentionally tricking neural networks

Example: Neural networks are used for classification in autonomous vehicles. If classification goes wrong, the car will have the wrong information about its surrounding, with dangerous consequences.



This can happen unintentionally, by e.g., glare from the sun, or any situation that the car has not encountered often enough in its training.

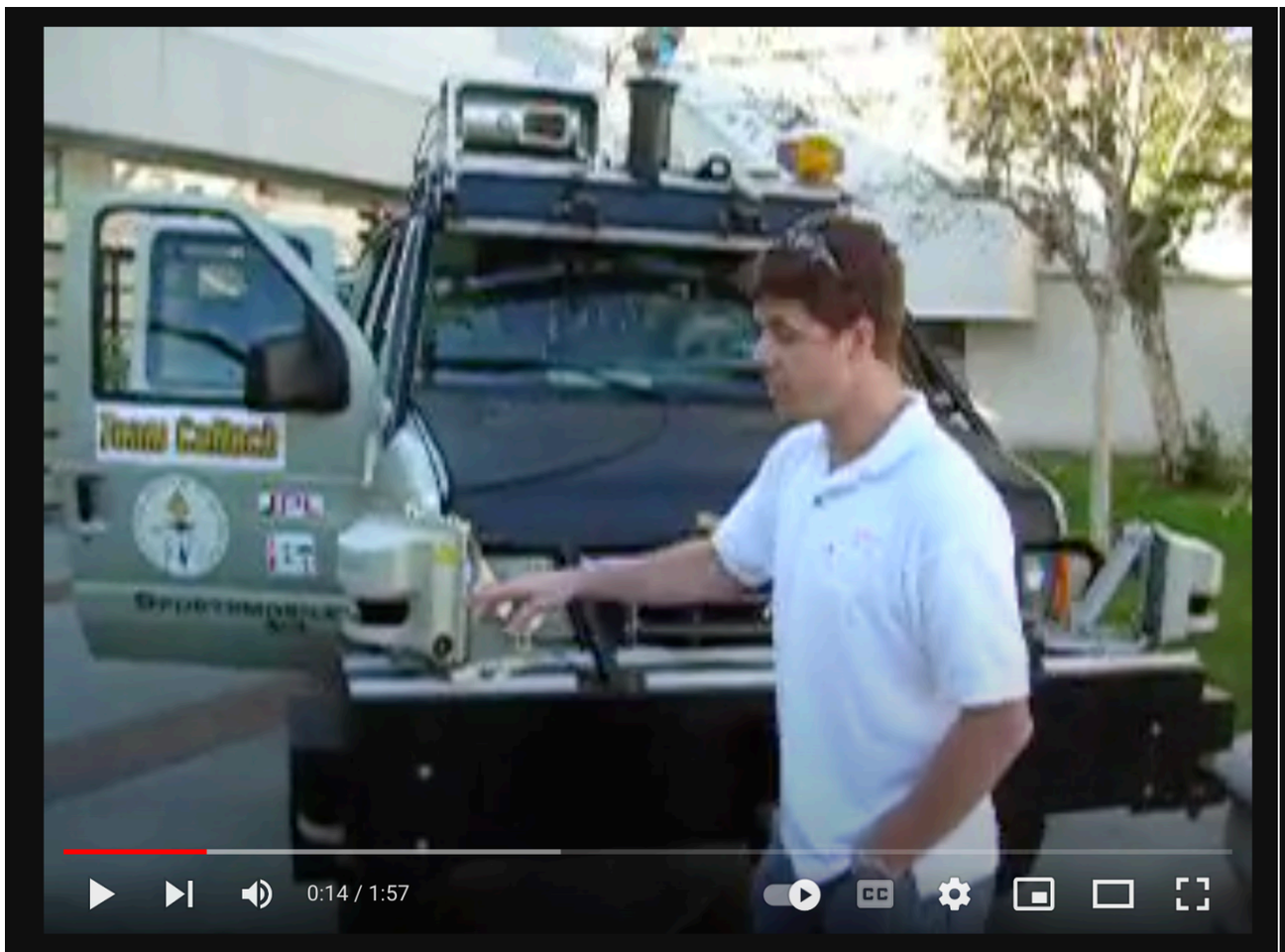
It can be enough to change a single pixel in an image, which could also happen by spilling coffee on an object, having some dust on your camera lens etc.



DARPA Grand Challenge 2005

Defense Advanced Research Projects Agency

132 mile off-road race for autonomous cars



CalTech "Alice" Pt 1 - DARPA Grand Challenge Autonomous SUV

CalTech's Autonomous SUV from DARPA Grand Challenge

DARPA Grand Challenge 2005

132 mile off-road race for autonomous cars

2019


65
MPH
55 %
65
50
5:41 PM 72° F
Scott Kubo
2.0 mi
B Ave
NAVIGATE ON AUTOPILOT
CANCEL
3.3 mi
7 min
5:47 PM
B Ave
CA-75 S San Diego, CA
73

Changing the building blocks of the neural networks, training with more data (i.e., more miles driven), or having more sensors (like cameras, radar etc) could help the algorithms, but will this get rid of every problem?

How safe does a self-driving car have to be before people want to rely on it?

Intentionally tricking neural networks

Neural nets as security guards

Imagine that we run an auction website like eBay. On our website, we want to prevent people from selling prohibited items — things like live animals. We can use deep learning to automatically check auction photos for prohibited items and flag the ones that violate the rules.

Neural nets as security guards

Imagine that we run an auction website like eBay. On our website, we want to prevent people from selling prohibited items — things like live animals. We can use deep learning to automatically check auction photos for prohibited items and flag the ones that violate the rules.

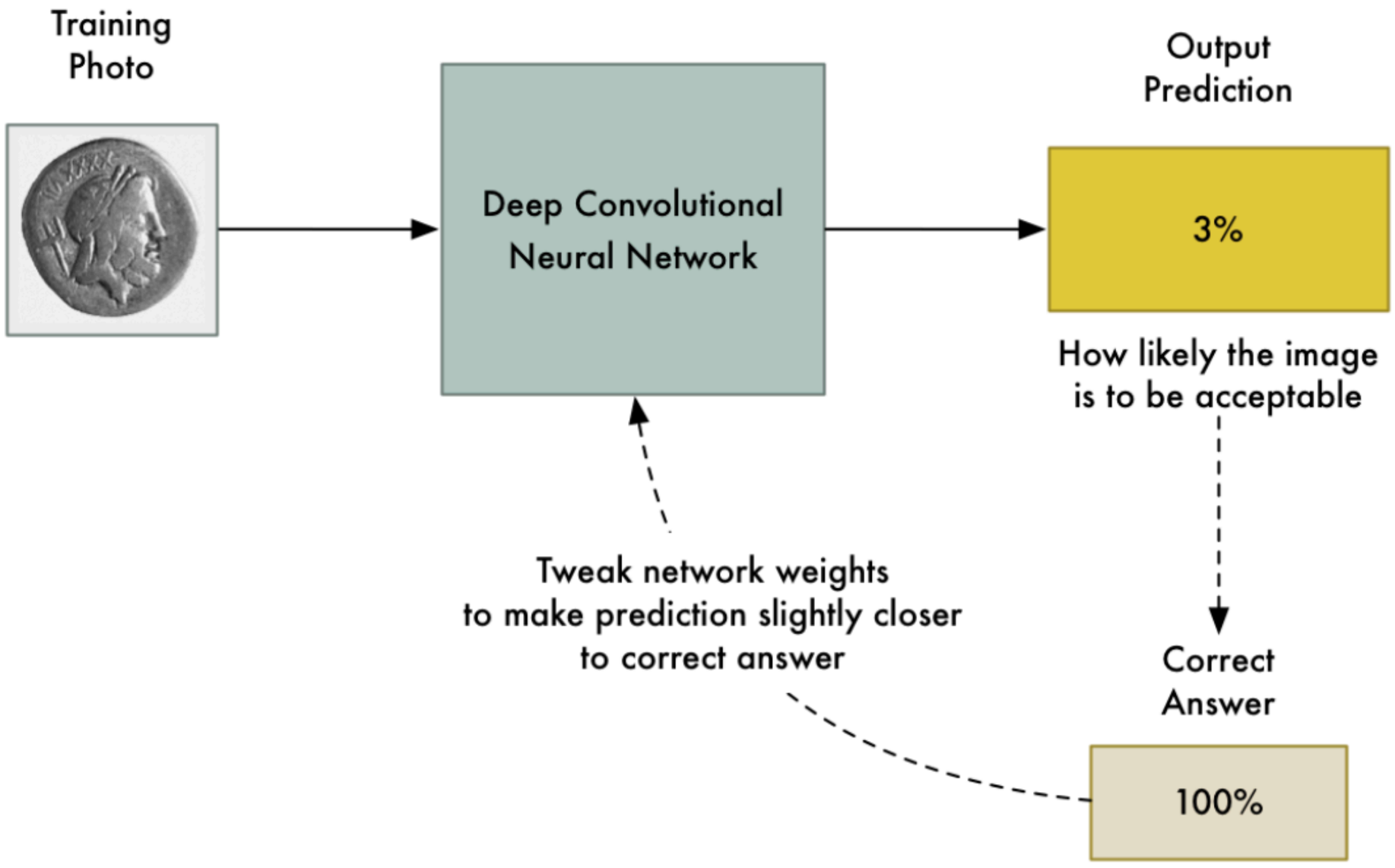
This is a typical image classification problem. To build this, we'll train a deep convolutional neural network to tell prohibited items apart from allowed items and then we'll run all the photos on our site through it.

First, we need a dataset of thousands of images from past auction listings. We need images of both allowed and prohibited items so that we can train the neural network to tell them apart.

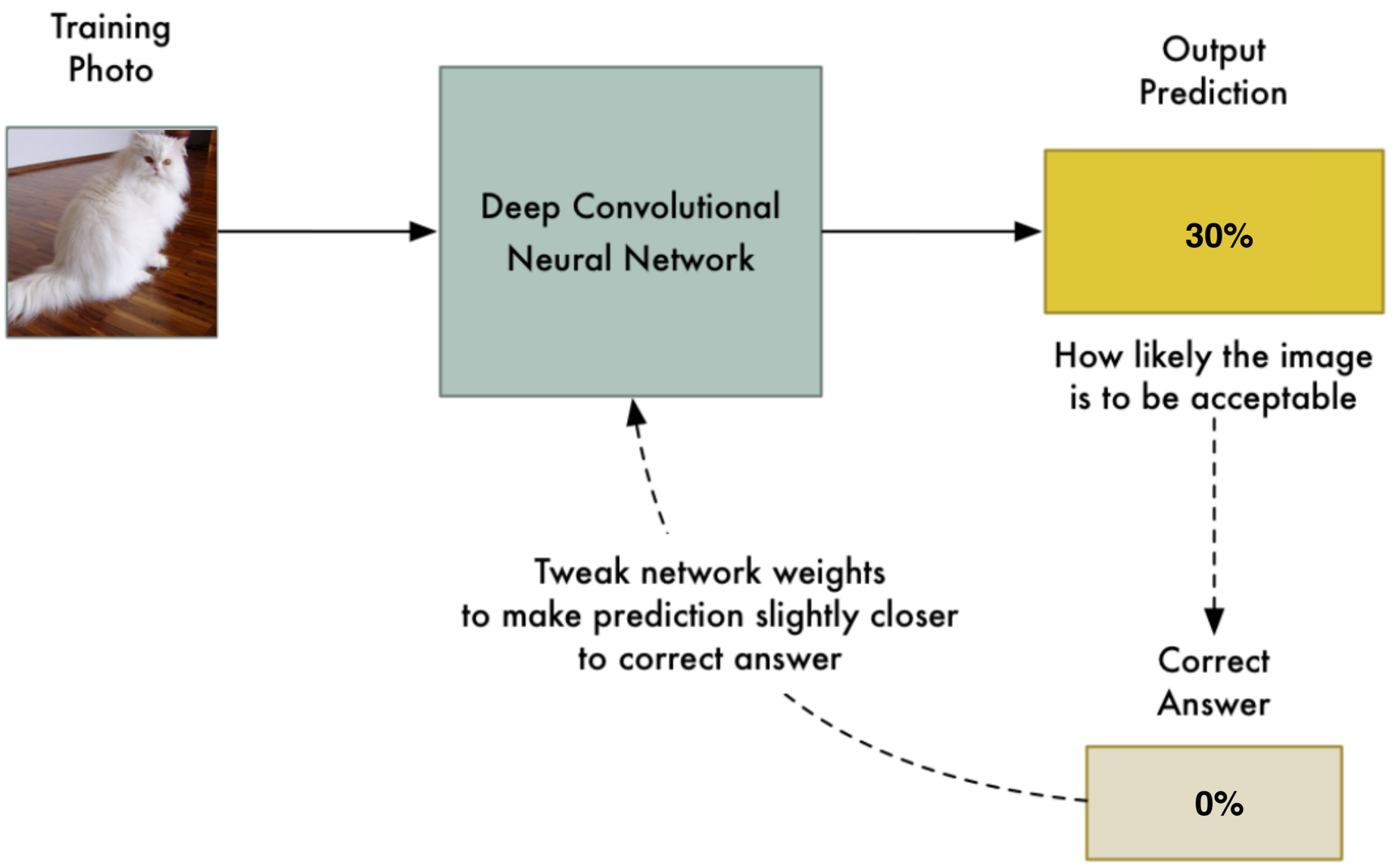
Photos of
Allowed Items

Photos of
Prohibited Items

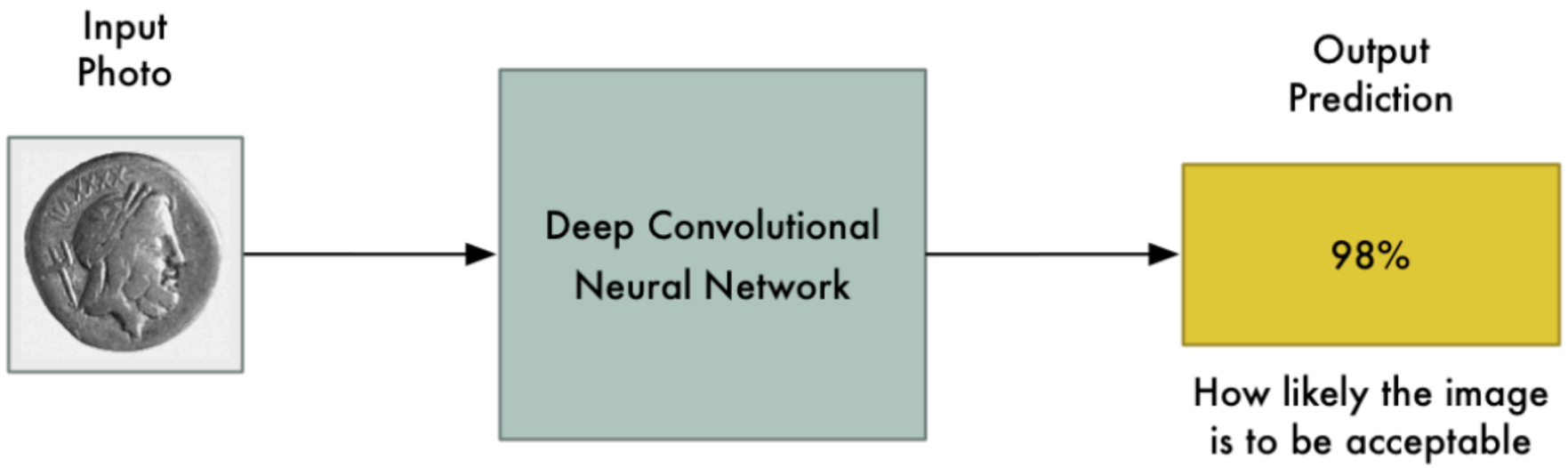
Training the Neural Network



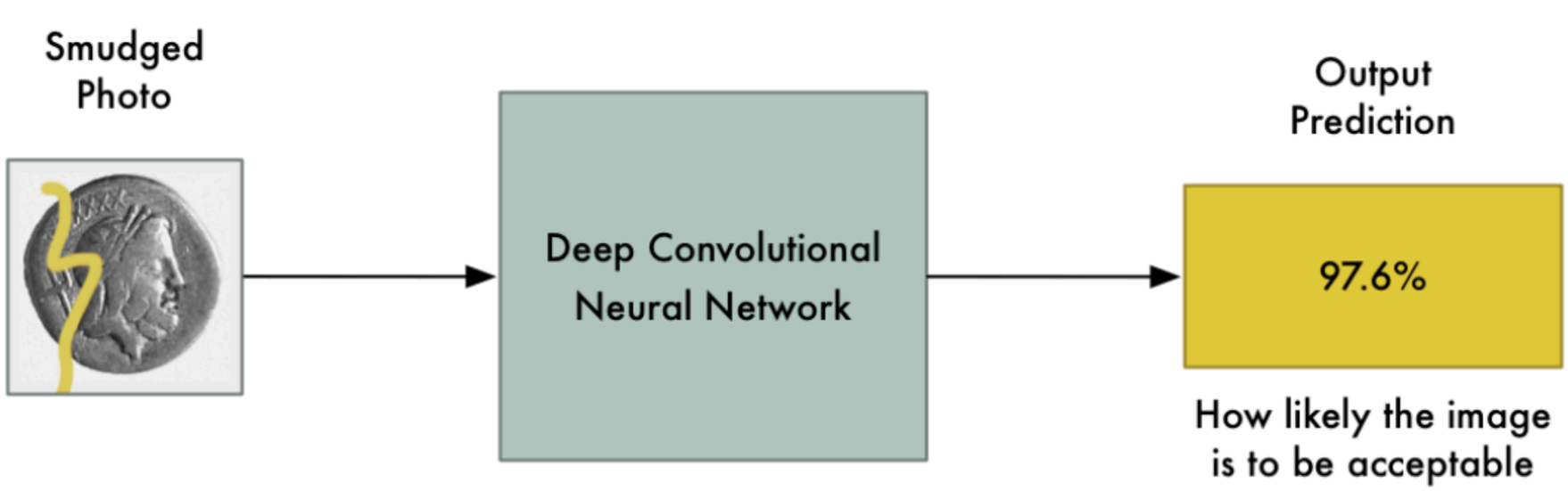
Training the Neural Network



We repeat this thousands of times with thousands of photos until the model reliably produces the correct results with an acceptable accuracy. The end result is a neural network that can reliably classify images.

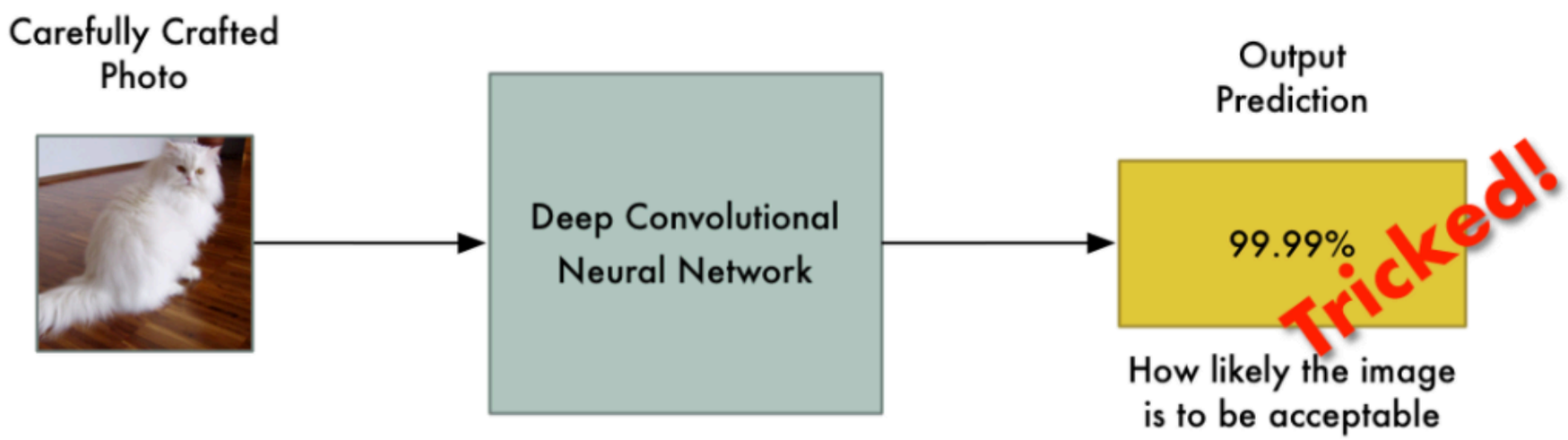


With a well-trained neural network, we expect that small changes to the input photo should only cause small changes to the final prediction.



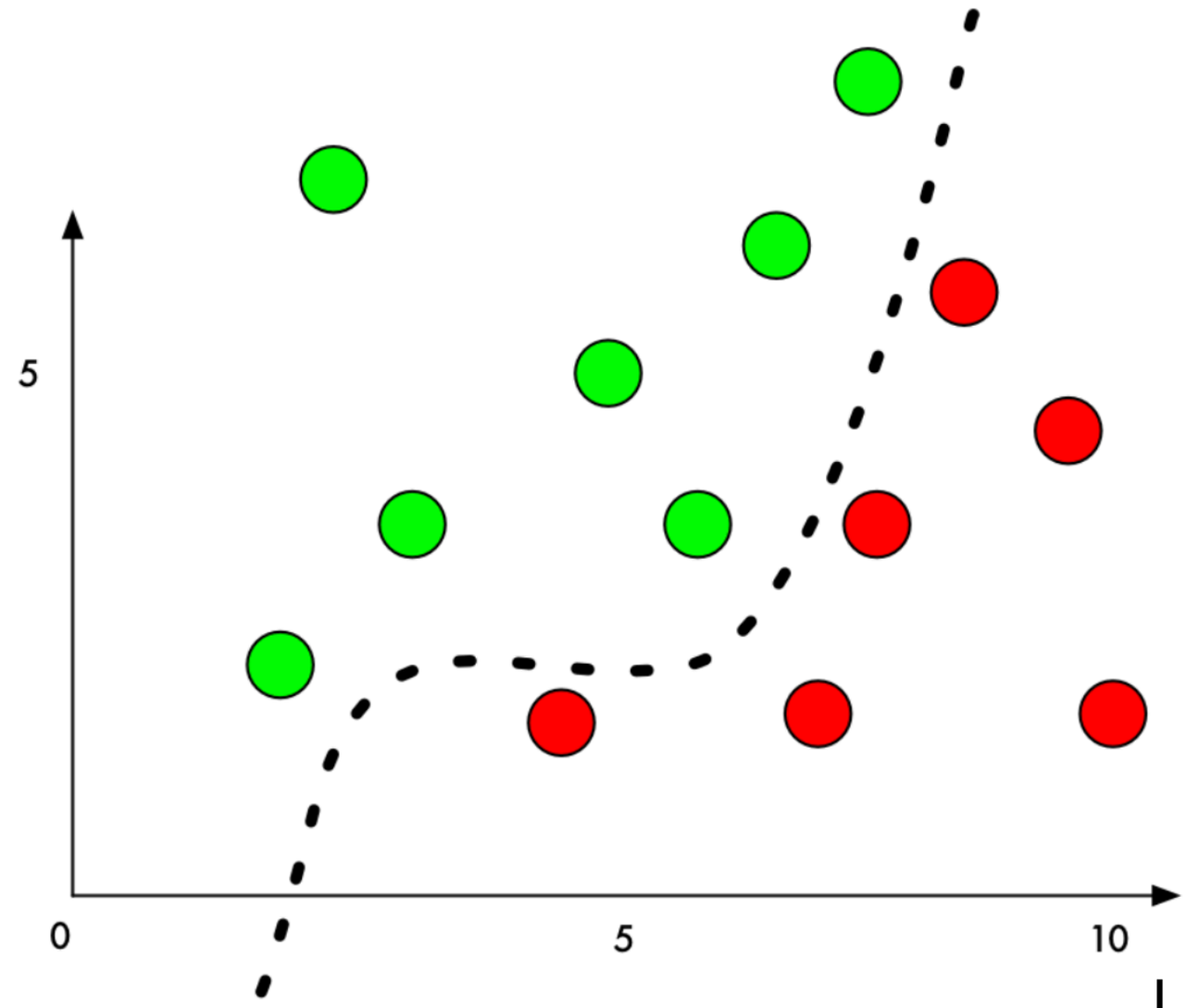
However, this is not always true. If you know *exactly which pixels to change* and *exactly how much to change them*, you can intentionally force the neural network to predict the wrong output for a given picture without changing the appearance of the picture very much.

That means we can intentionally craft a picture that is clearly a prohibited item but which completely fools our neural network.



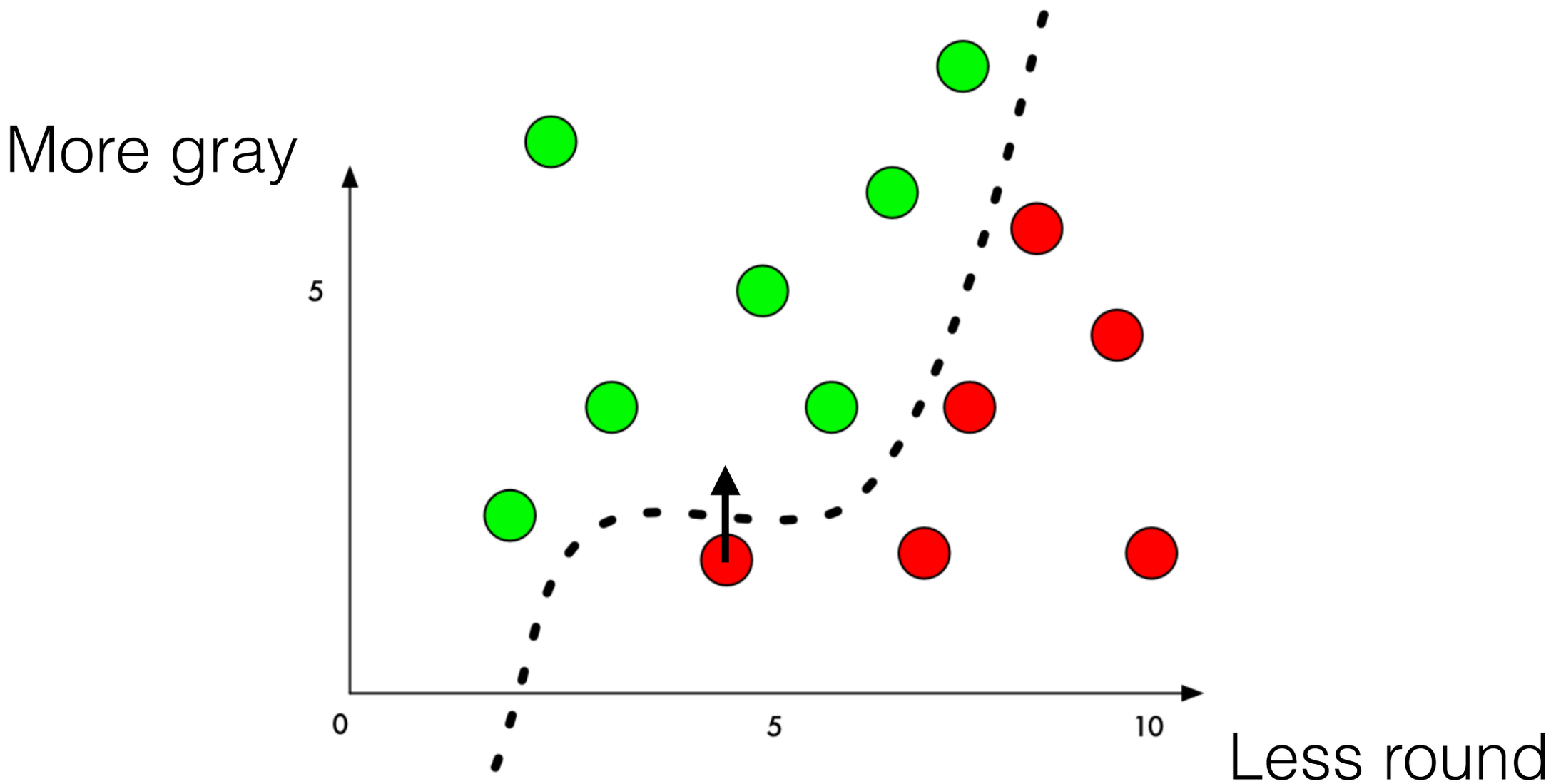
A machine learning classifier works by finding a dividing line between the things it's trying to tell apart.

More gray

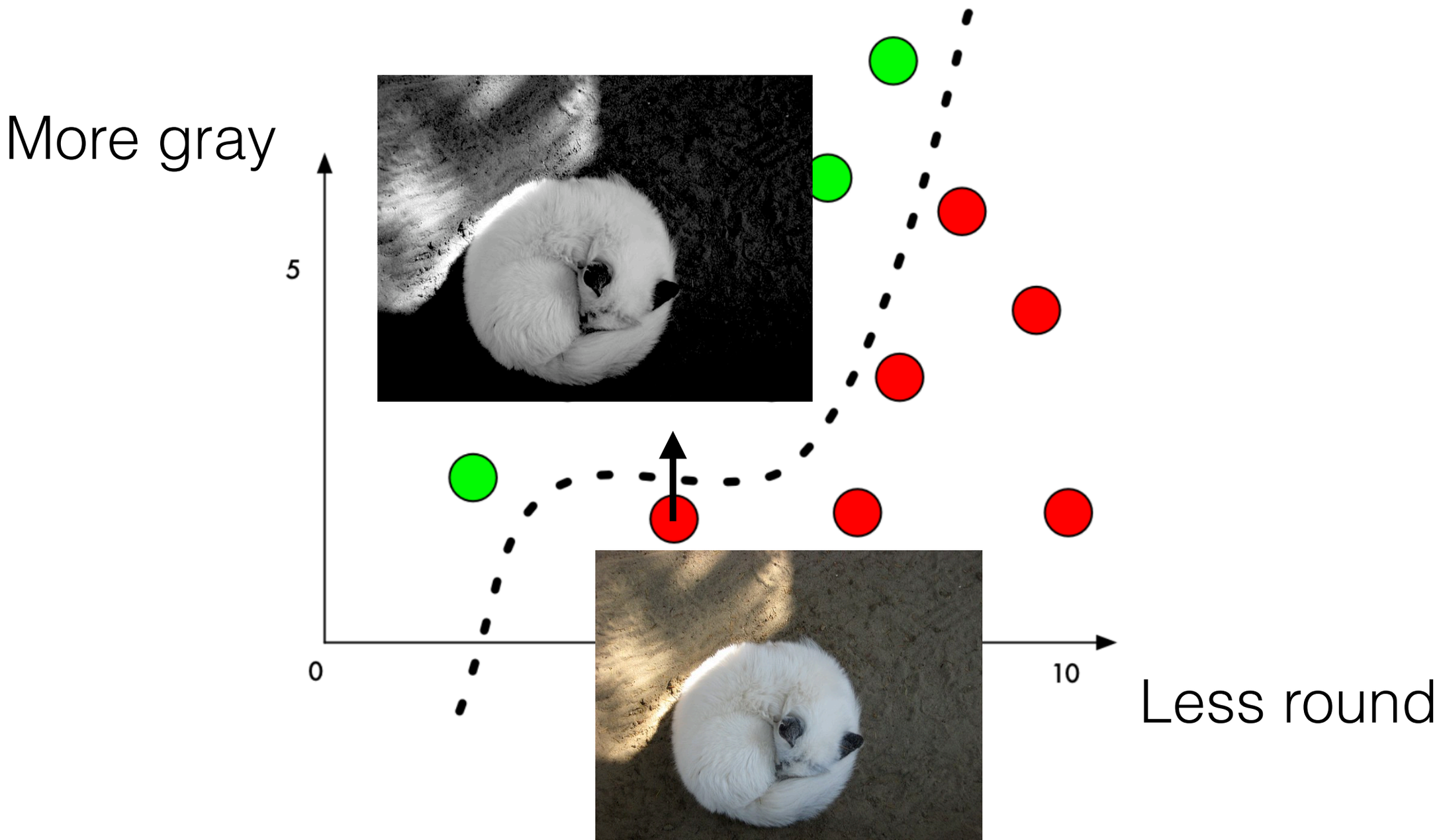


Less round

If we add a small amount to the Y value of a red point right beside the boundary, we can just barely push it over into green territory.



If we add a small amount to the Y value of a red point right beside the boundary, we can just barely push it over into green territory.



In image classification with deep neural networks, each “point” we are classifying is an entire image made up of thousands of pixels.

In image classification with deep neural networks, each “point” we are classifying is an entire image made up of thousands of pixels. That gives us *thousands* of possible values that we can tweak to push the point over the decision line.

In image classification with deep neural networks, each “point” we are classifying is an entire image made up of thousands of pixels. That gives us *thousands* of possible values that we can tweak to push the point over the decision line. And if we make sure that we tweak the pixels in the image in a way that isn’t too obvious to a human, we can fool the classifier without making the image look manipulated.

Original Image

Persian cat 87%
lynx 0%
Angora 0%
dishwasher 0%
Pomeranian 0%

Hacked Image

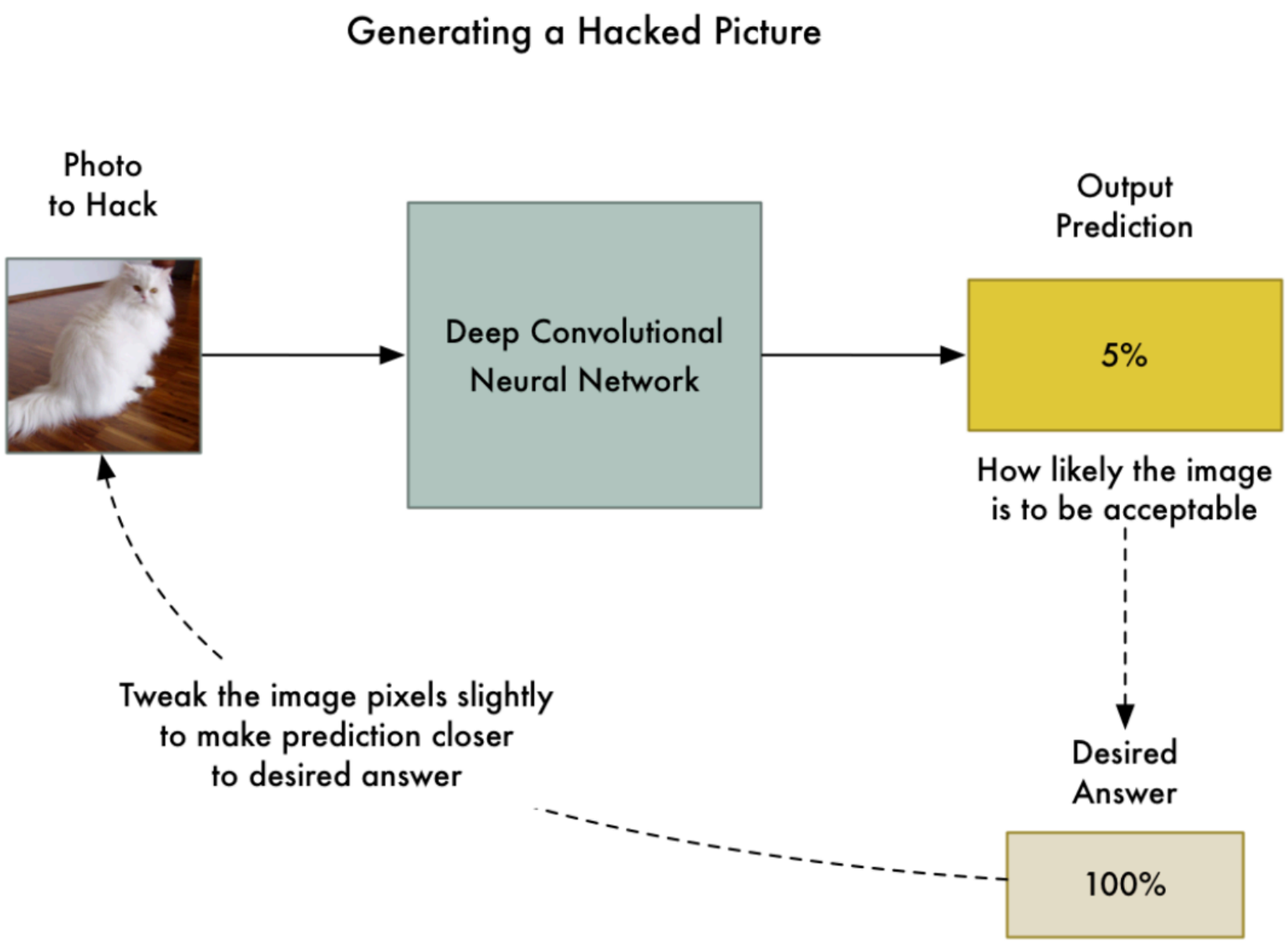
Original Image

Persian cat 87%
lynx 0%
Angora 0%
dishwasher 0%
Pomeranian 0%

Hacked Image

toaster 98%
Crock Pot 1%
Siamese cat 0%
wallaby 0%
carton 0%

How to trick a neural network



The only problem is that by allowing any single pixel to be adjusted without any limitations, the changes to the image can be drastic enough that you'll see them.

The only problem is that by allowing any single pixel to be adjusted without any limitations, the changes to the image can be drastic enough that you'll see them. To prevent these obvious distortions, we can add a simple constraint to our algorithm. We'll say that no single pixel in the hacked image can ever be changed by more than a tiny amount from the original image — let's say something like 0.01%.

The only problem is that by allowing any single pixel to be adjusted without any limitations, the changes to the image can be drastic enough that you'll see them. To prevent these obvious distortions, we can add a simple constraint to our algorithm. We'll say that no single pixel in the hacked image can ever be changed by more than a tiny amount from the original image — let's say something like 0.01%. That forces our algorithm to tweak the image in a way that still fools the neural network without it looking too different from the original image.

There is still a big limitation with how we create these images — our attack requires direct access to the neural network itself.

There is still a big limitation with how we create these images — our attack requires direct access to the neural network itself. Because we are actually "training" against the neural network to fool it, we need a copy of it. In the real world, no company is going to let you download their trained neural network's code, so that means we can't attack them.

There is still a big limitation with how we create these images — our attack requires direct access to the neural network itself. Because we are actually "training" against the neural network to fool it, we need a copy of it. In the real world, no company is going to let you download their trained neural network's code, so that means we can't attack them… Right?

It turns out that you can train your own substitute neural network to mirror another neural network by probing it to see how it behaves. Then you can use your substitute neural network to generate hacked images that still often fool the original network! This is called a *black-box attack*.

It turns out that you can train your own substitute neural network to mirror another neural network by probing it to see how it behaves. Then you can use your substitute neural network to generate hacked images that still often fool the original network! This is called a *black-box attack*.

It turns out that knowing the details of exactly how a network is constructed is not crucial. An attack that works on one network is in practice likely to work on other networks.

The applications of black-box attacks are limitless. Here are some plausible examples:

1. Trick self-driving cars into seeing a stop sign as a green light — this could cause car crashes!

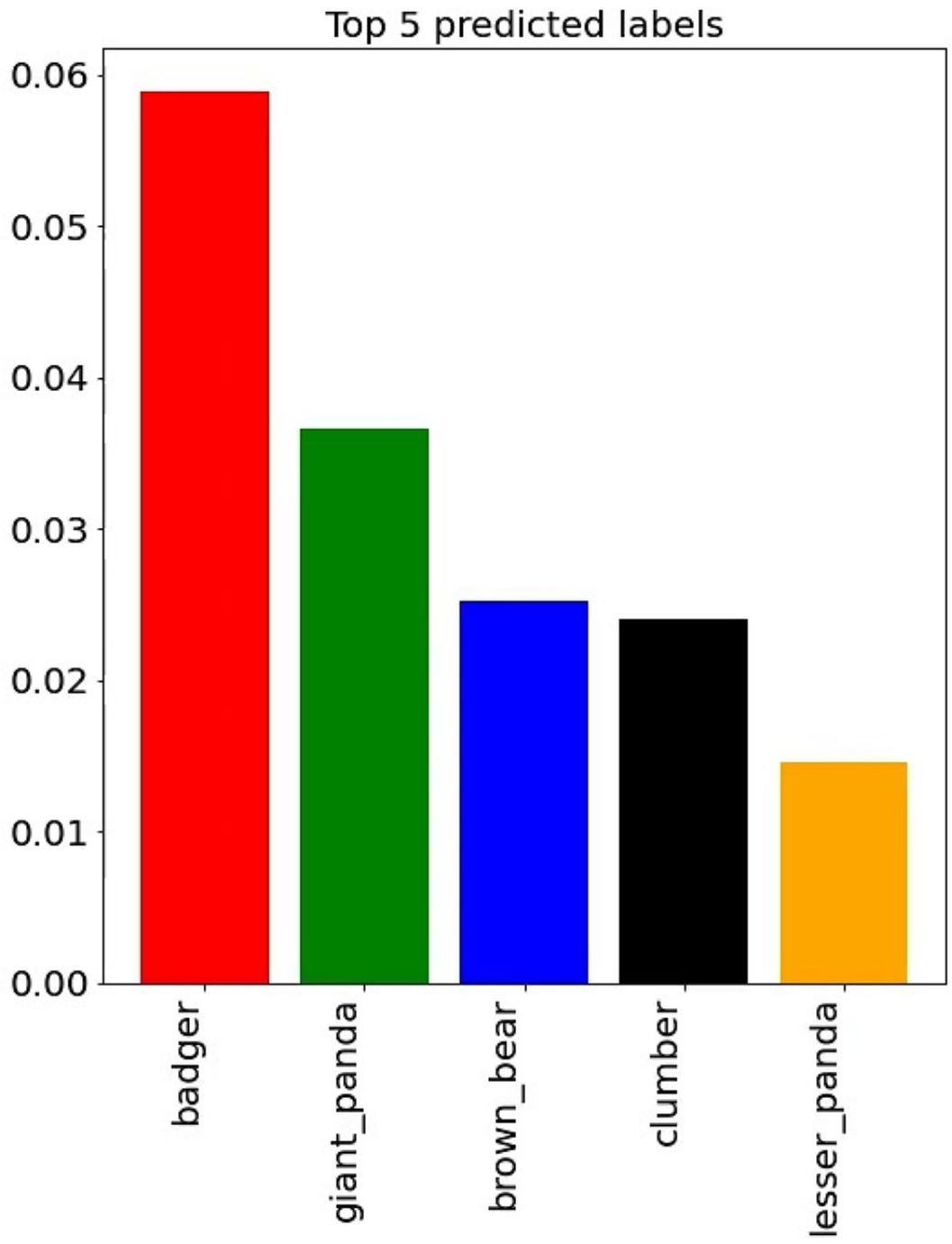
The applications of black-box attacks are limitless. Here are some plausible examples:

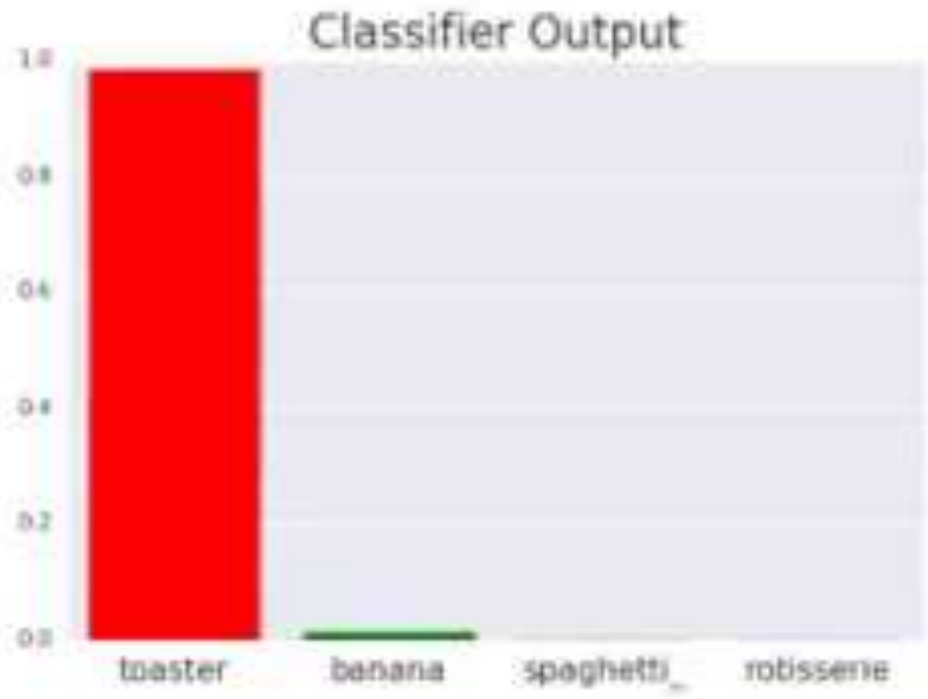
1. Trick self-driving cars into seeing a stop sign as a green light — this could cause car crashes!
2. Trick content filtering systems into letting offensive/illegal content through.

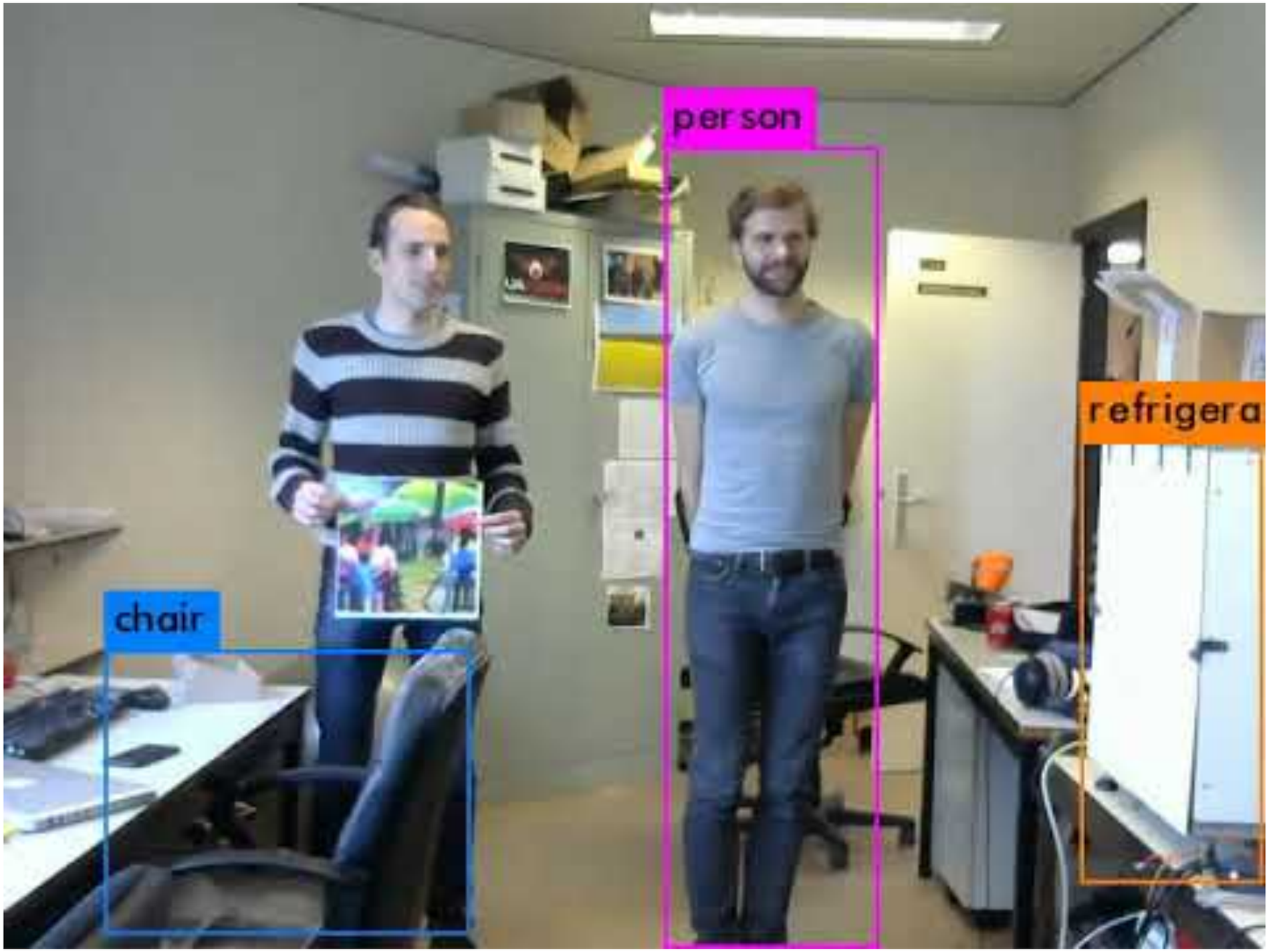
The applications of black-box attacks are limitless. Here are some plausible examples:

1. Trick self-driving cars into seeing a stop sign as a green light — this could cause car crashes!
2. Trick content filtering systems into letting offensive/illegal content through.
3. Trick ATM/smartphone check scanners into thinking the writing on a check says the check is for a greater amount than it actually is.

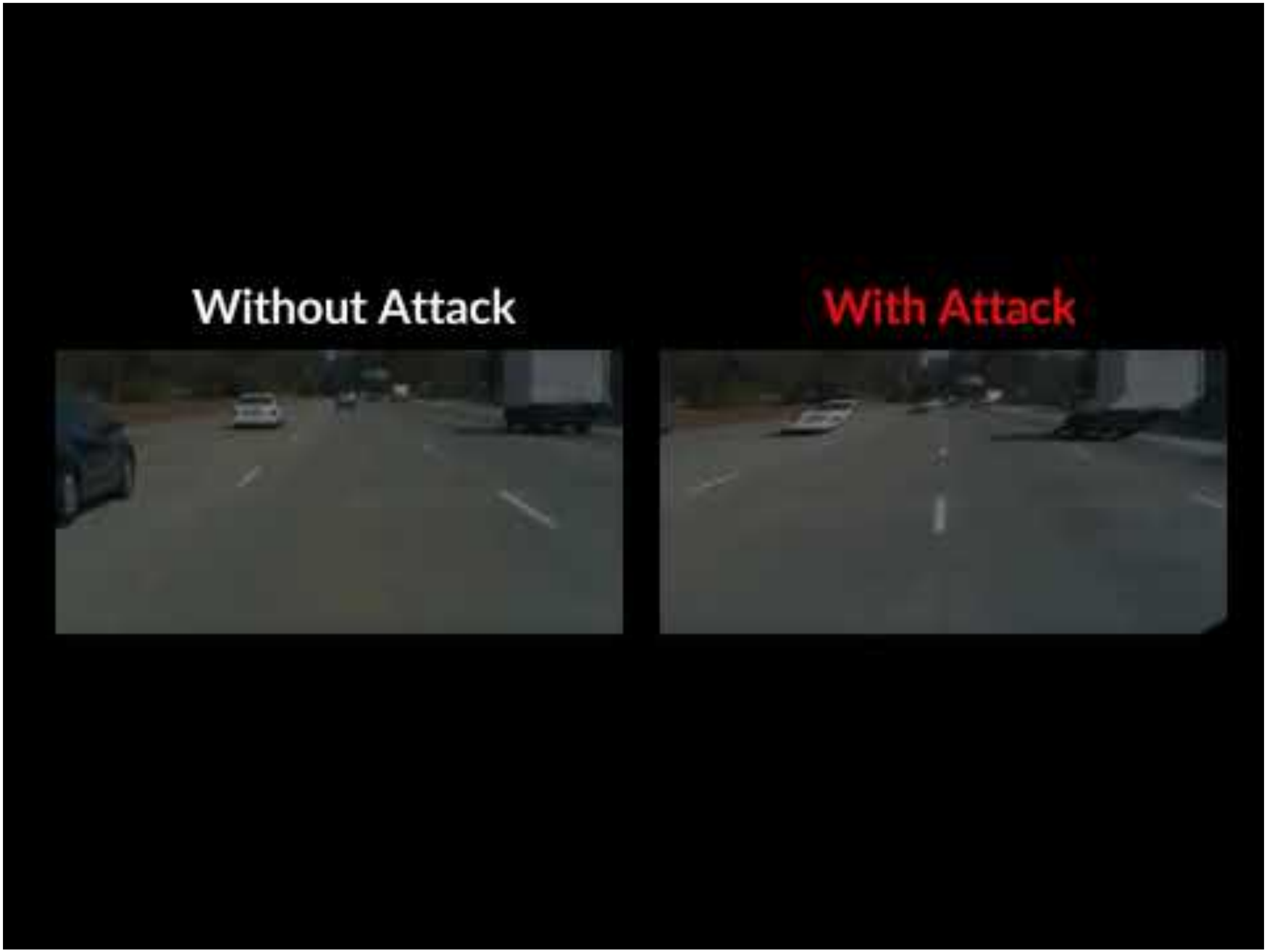

Advarsarial Attack with FGSM (Untargetted)
(ε = 0.020): (badger, 5.88% Confidence)


Top 5 predicted labels
0.06
0.05
0.04
0.03
0.02
0.01
0.00
badger
giant_panda
brown_bear
clumber
lesser_panda


Classifier Output
1.0
0.8
0.6
0.4
0.2
0.0
toaster
banana
spaghetti_
robisserie


person
refrigera
chair

Lane-Keeping Assistance System




Projector : ON
Attack with OPAD
STOP

But how difficult is it to carry out these attacks? Do you need a specialized projector, set up at a certain distance away from the sign, and does it only work when the sun is at a certain angle in the sky?

But how difficult is it to carry out these attacks? Do you need a specialized projector, set up at a certain distance away from the sign, and does it only work when the sun is at a certain angle in the sky?

No! Placing stickers in the way shown here results in neural networks classifying the stop sign as a 45 mph speed limit sign!

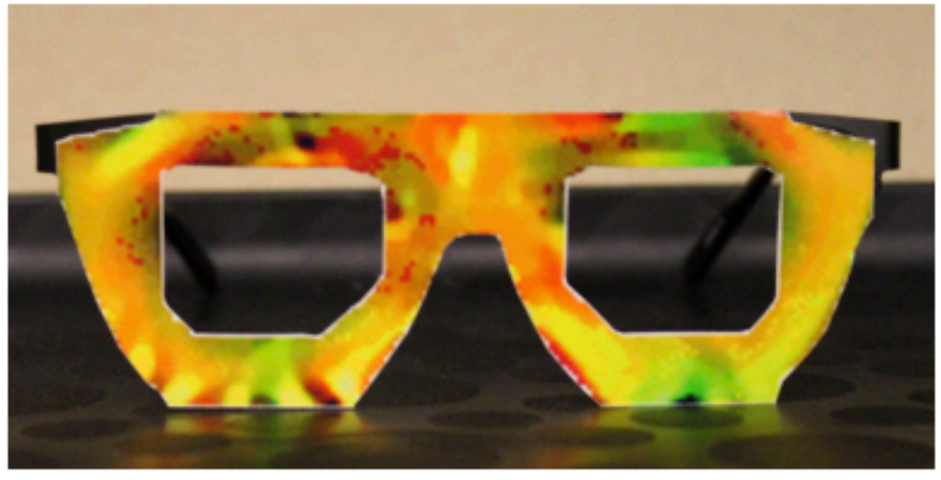
But how difficult is it to carry out these attacks? Do you need a specialized projector, set up at a certain distance away from the sign, and does it only work when the sun is at a certain angle in the sky?

No! Placing stickers in the way shown here results in neural networks classifying the stop sign as a 45 mph speed limit sign!

So how confident are we that it works correctly if there is graffiti, dirt or ice on the sign?

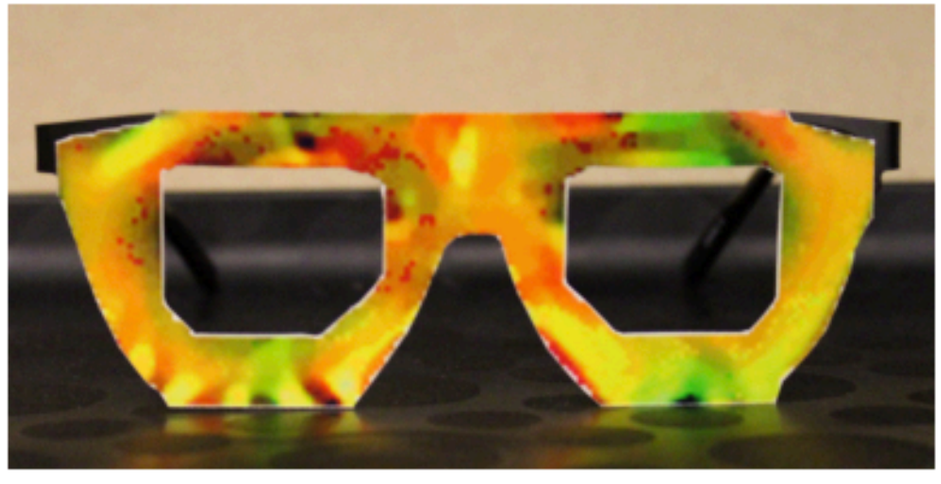
How about facial recognition?

How about facial recognition?



It turns out that one way of protecting against being recognized is to wear weirdly colored glasses.

How about facial recognition?



It turns out that one way of protecting against being recognized is to wear weirdly colored glasses.

The people in the top row were recognized by the neural network as the people in the bottom row.

How about facial recognition?

Wearing different accessories can make it so that the neural network does not even recognize that there is a person in the picture at all, i.e., you become invisible.

Figure 6: An example of an invisibility attack. Left: original image of actor Kiefer Sutherland. Middle: Invisibility by perturbing pixels that overlay the face. Right: Invisibility with the use of accessories.

Possible protections

This is a subject under current research!

1. Create lots of hacked images and include them in the training dataset. It seems to make the neural network more resistant to these attacks. This is called Adversarial Training and is probably the most reasonable defense to consider adopting right now.
2. Build in fail-safes into your application for when a user intentionally sets out to fool your neural networks and think of ways to mitigate those scenarios.

Discussion

How do you envision the impact of AI on different industries, such as healthcare, finance, transportation, etc., in the coming years?

Healthcare: improve diagnostics, treatment planning, patient care, …

Finance: fraud detection, algorithmic trading, risk management, …

Transportation: autonomous vehicles, intelligent traffic management, …

Discussion

What ethical considerations should be taken into account as AI technologies continue to advance? How can we ensure AI is developed and used responsibly?

Fairness

Transparency and explainability

Privacy and data security

Safety and reliability

Discussion

How might advances in AI impact our understanding of intelligence, consciousness, and what it means to be human?

Challenge anthropocentrism?

Moral responsibility?

Rights and dignity of AI systems?

Thank you for the term!