Practical evading attacks on commercial AI image recognition services

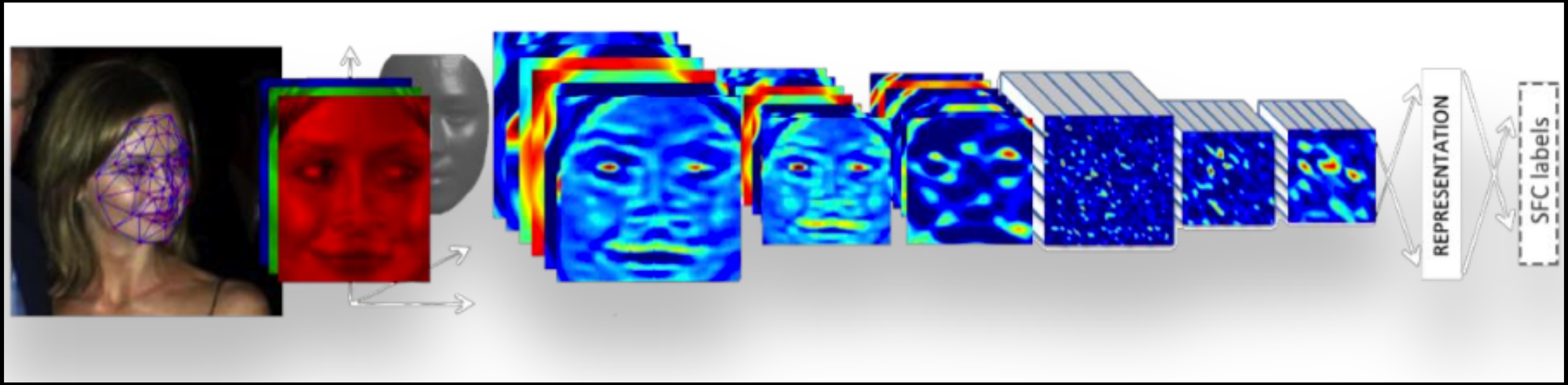
Kang Li
Department of Computer Science
University of Georgia

Collaborators: Yufei Chen, Qixue Xiao, Deyue Zhang

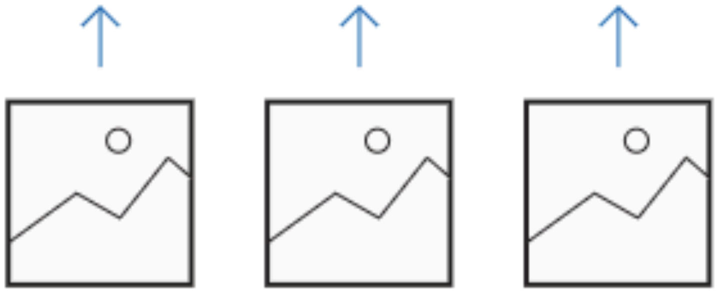
About Me

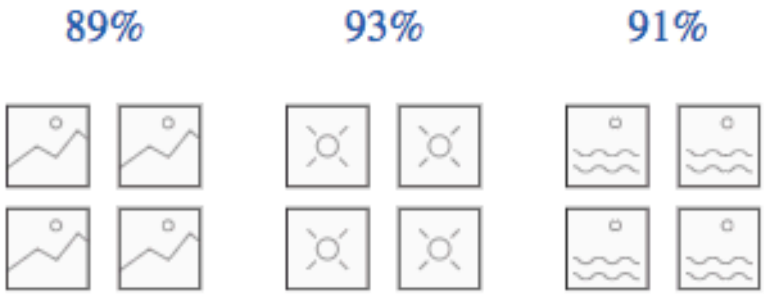
- Professor at the University of Georgia
- Director of UGA Institute for Cybersecurity and Privacy (ICSP)
- Founding mentor of *Blue-Lotus* CTF Team and *xCTF* League
- Founder of the *Disekt, SecDawgs* CTF Teams
- Finalist of 2016 DARPA Cyber Grand Challenge (CGC)

Deep Learning and Advances in AI Applications



https://www.cs.toronto.edu/~ranzato/publications/taigman_cvpr14.pdf https://www.customvision.ai





Upload Images

Bring your own labeled images, or use Custom Vision to quickly add tags to any unlabeled images.

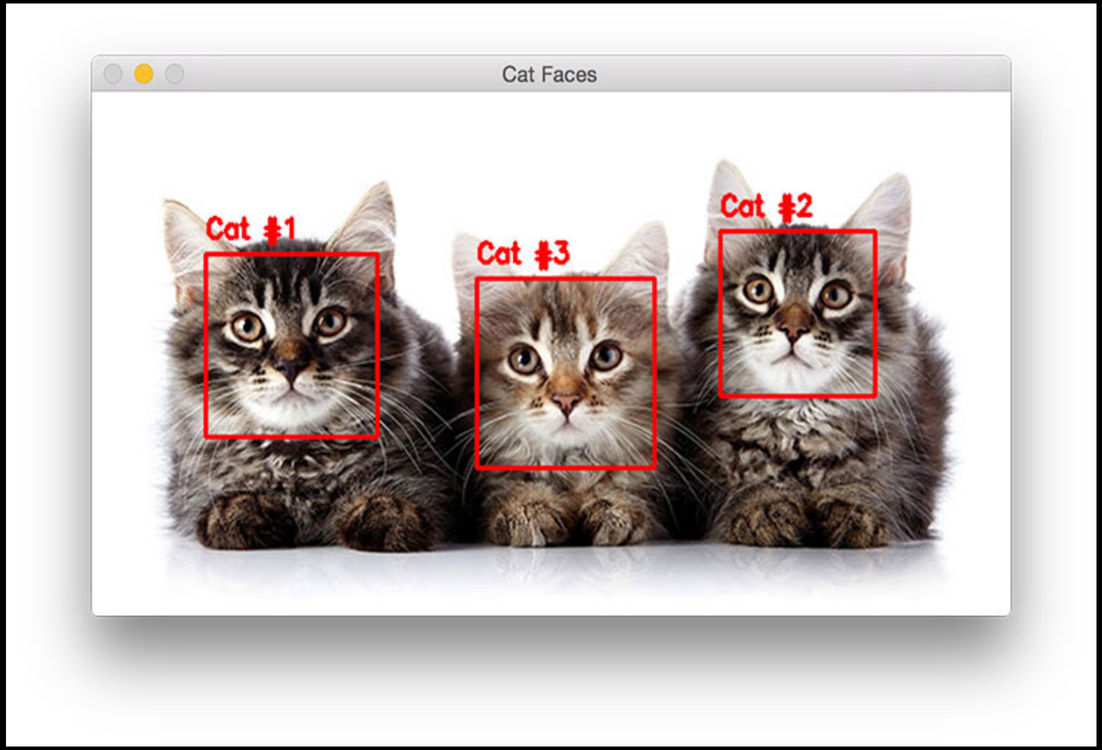
Train

Use your labeled images to teach Custom Vision the concepts you care about.

Evaluate

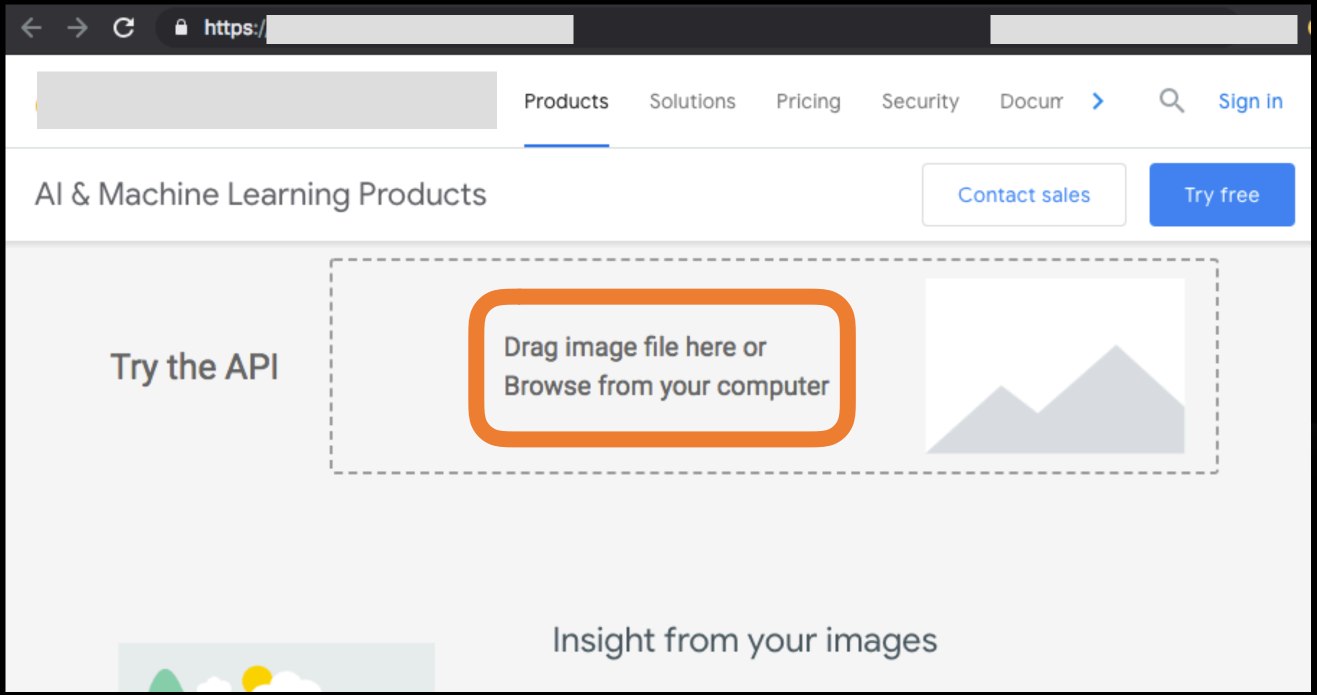
Use simple REST API calls to quickly tag images with your new custom computer vision model.

AI & Image Recognition System



https://www.pyimagesearch.com/2016/06/20/detecting-cats-in-images-with-opencv/

Image Recognition As a Service



Google Cloud Vision API

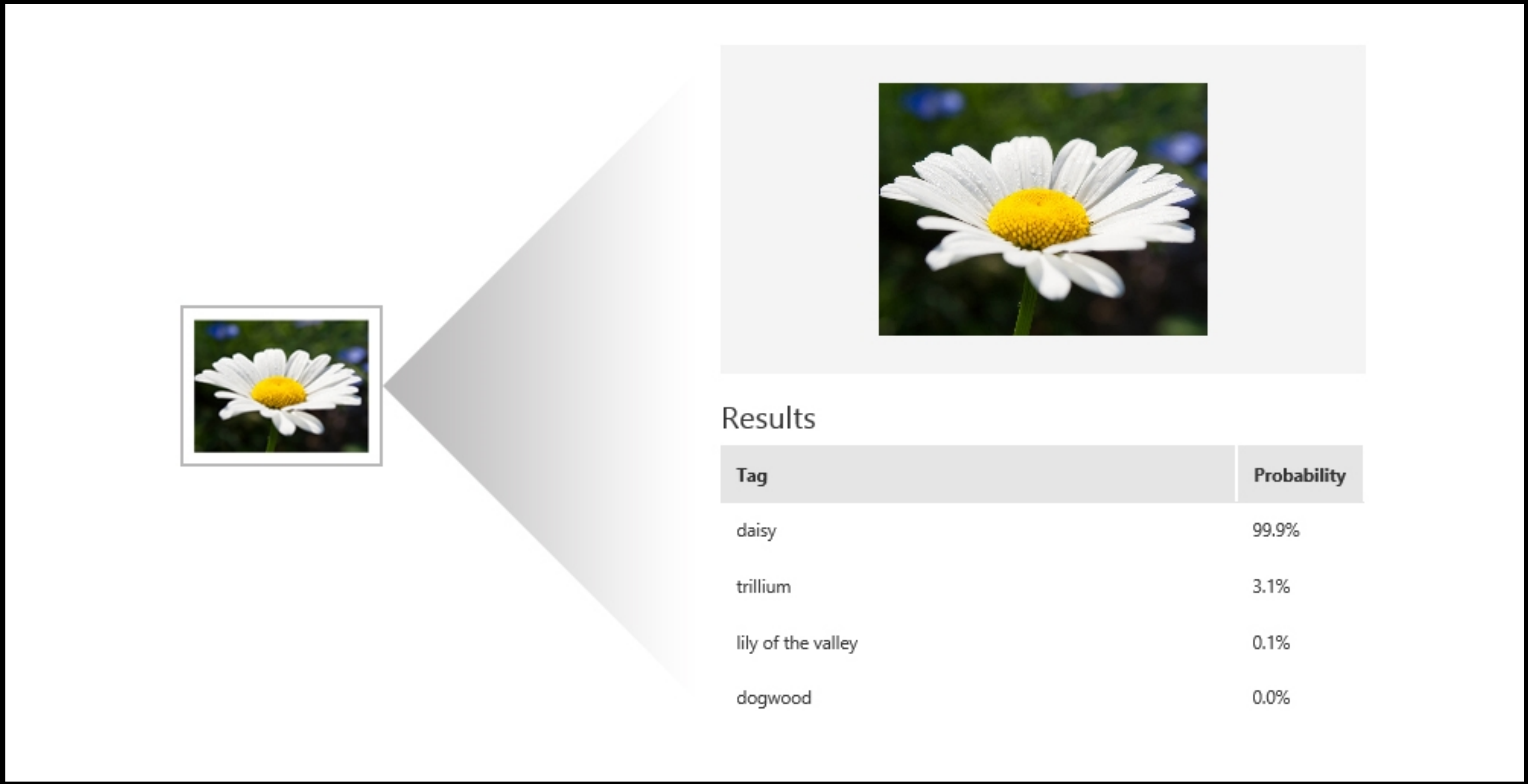
阿里云 aliyun.com

Microsoft Azure

Tencent AI OPEN PLATFORM

Baidu百度 | AI开放平台

Image Recognition As a Service



https://azure.microsoft.com/en-us/services/cognitive-services/custom-vision-service/

Image Recognition Service API

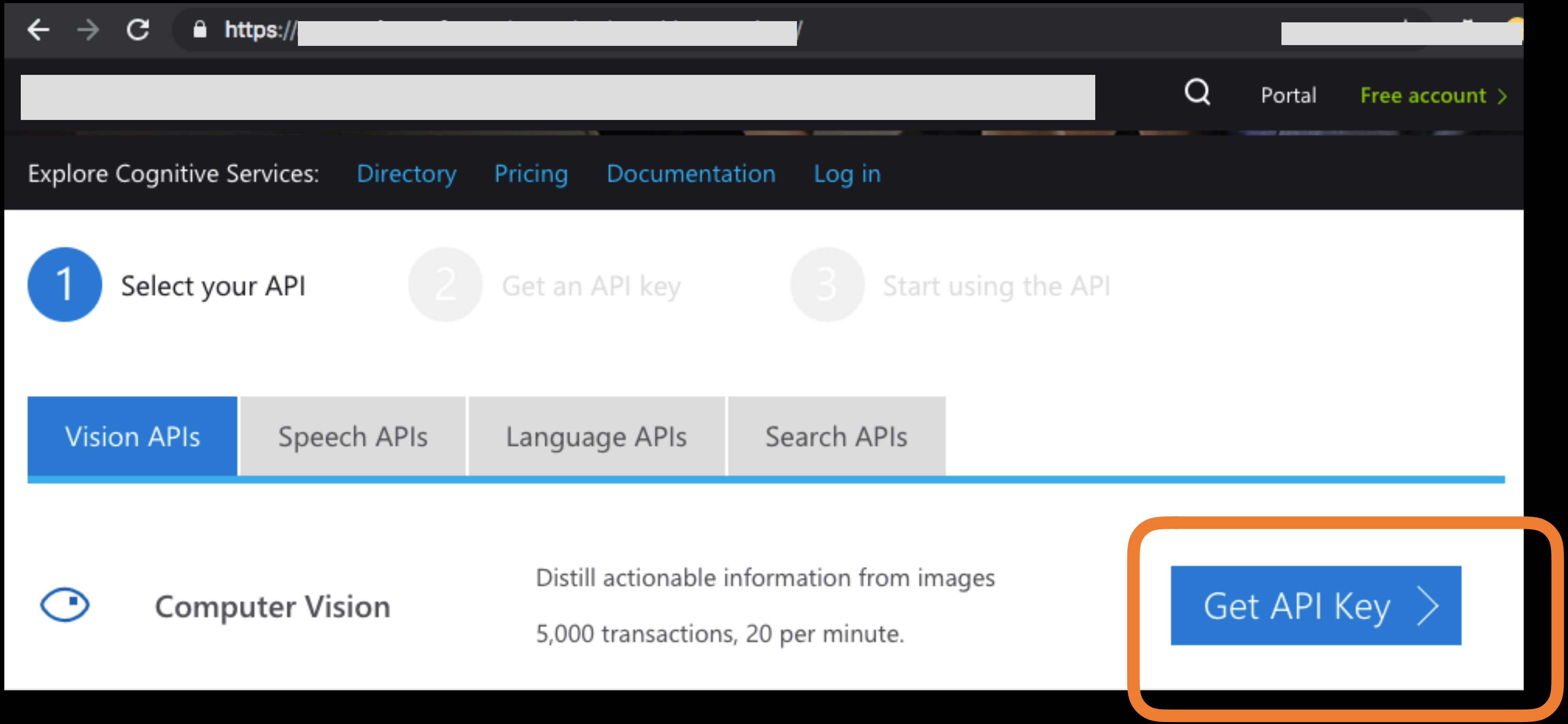


Image Recognition Service API (Example #1)

```python
'''
Animal Recognition [redacted]

** With the support of [redacted]SDK.
'''

from aip import AipImageClassify
import sys
import os
import time
import json


APP_ID = '117[redacted]'
API_KEY = 's3[redacted]'
SECRET_KEY = 'od[redacted]'
client = AipImageClassify(APP_ID, API_KEY, SECRET_KEY)


def get_file_content(image_path):
    with open(image_path, 'rb') as fp:
        return fp.read()


def api_test(image_path):
    image = get_file_content(image_path)
    #[redacted]nimal Recognition
    content = client.animalDetect(image)
    return content


def main():
    image_path = sys.argv[1]
    response = api_test(image_path)
    print('Result from the [redacted]on API:')
    print(json.dumps(response, ensure_ascii=False, indent=4, separators=(',', ':')))


if __name__ == '__main__':
    main()
```

```python
import AipImageClassify
```

```python
client = AipImageClassify(APP_ID, API_KEY, SECRET_KEY)
```

```python
content = client.animalDetect(image)
return content
```

Image Recognition Service API (Example #2)

```python
import requests
import sys
import os
import time
import json
from PIL import Image
from io import BytesIO


def api_test(image_path):
    subscription_key = "[redacted]"
    assert subscription_key

    vision_base_url = "https://[redacted]/"

    analyze_url = vision_base_url + "analyze"

    # Set image_path to the local path of an image that you want to analyze.

    # Read the image into a byte array
    image_data = open(image_path, "rb").read()
    headers    = {'Ocp-Apim-Subscription-Key': subscription_key,
                  'Content-Type': 'application/octet-stream'}
    params     = {'visualFeatures': 'Categories,Tags,Imagetype,Description,Color'}
    response = requests.post(
        analyze_url, headers=headers, params=params, data=image_data)
    response.raise_for_status()

    rsp = response.json()
    return rsp


def main():
    image_path = sys.argv[1]
    response = api_test(image_path)
    print('Resul[redacted] API:')
    print()
    print(json.dumps(response, ensure_ascii=False, indent=4, separators=(',', ':')))
```

set subscription_key to *"your valid subscription key"*

set recognition request target to regional *service URL*

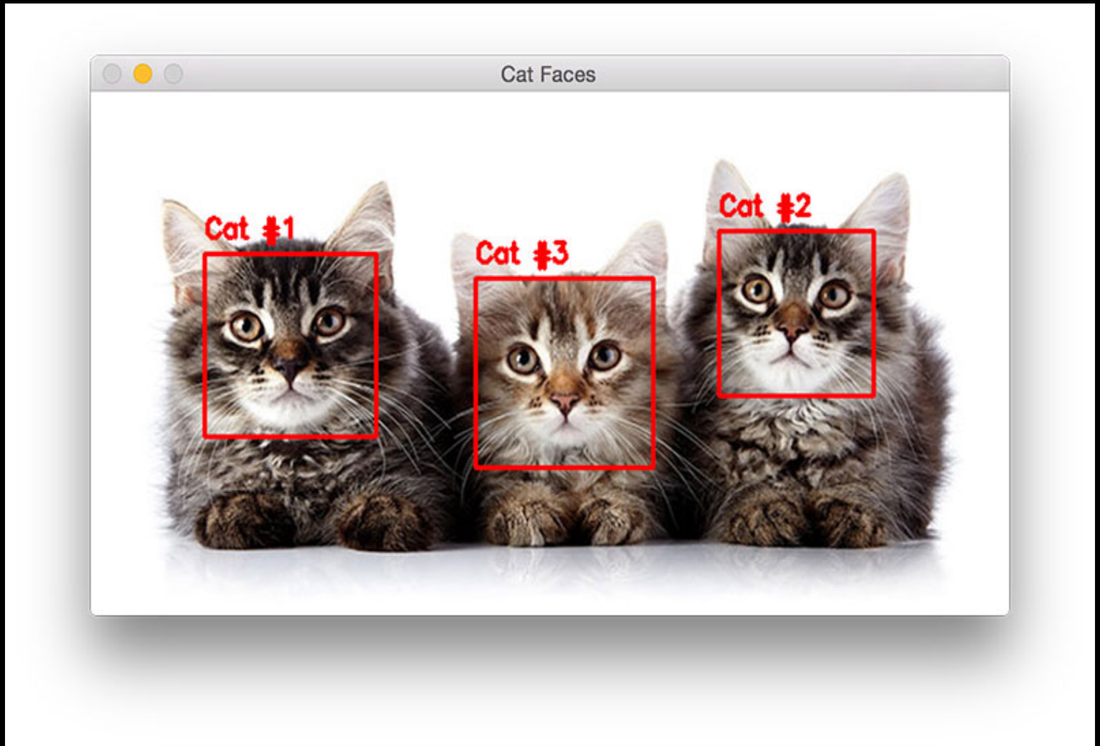
requests.post(analyze_url, headers=headers, params=params, json=data)

Applications based on Image Recognition Service

- Image Classifier
- Optical character recognition (OCR) in images
- Object, scene, and activity detection
- Person Identification and Emotion Recognition
- Explicit or offensive content moderation for images

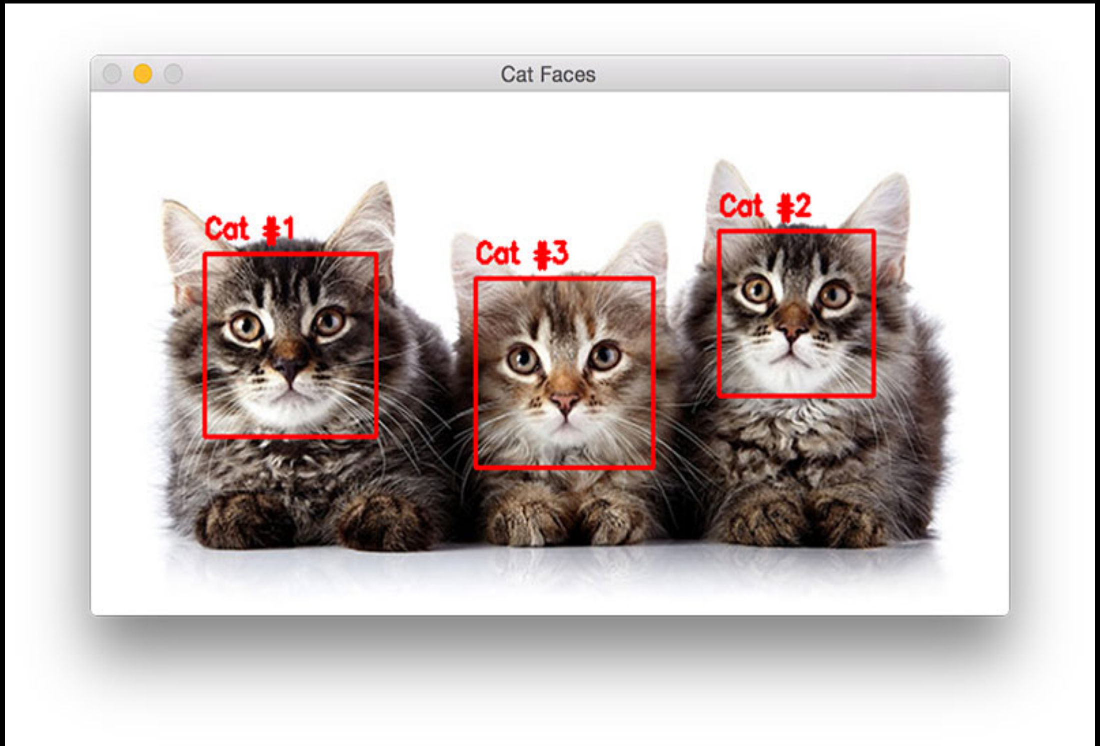
Attacks on Image Recognition Services

How to exploit an AI image recognition system?



https://www.pyimagesearch.com/2016/06/20/detecting-cats-in-images-with-opencv/

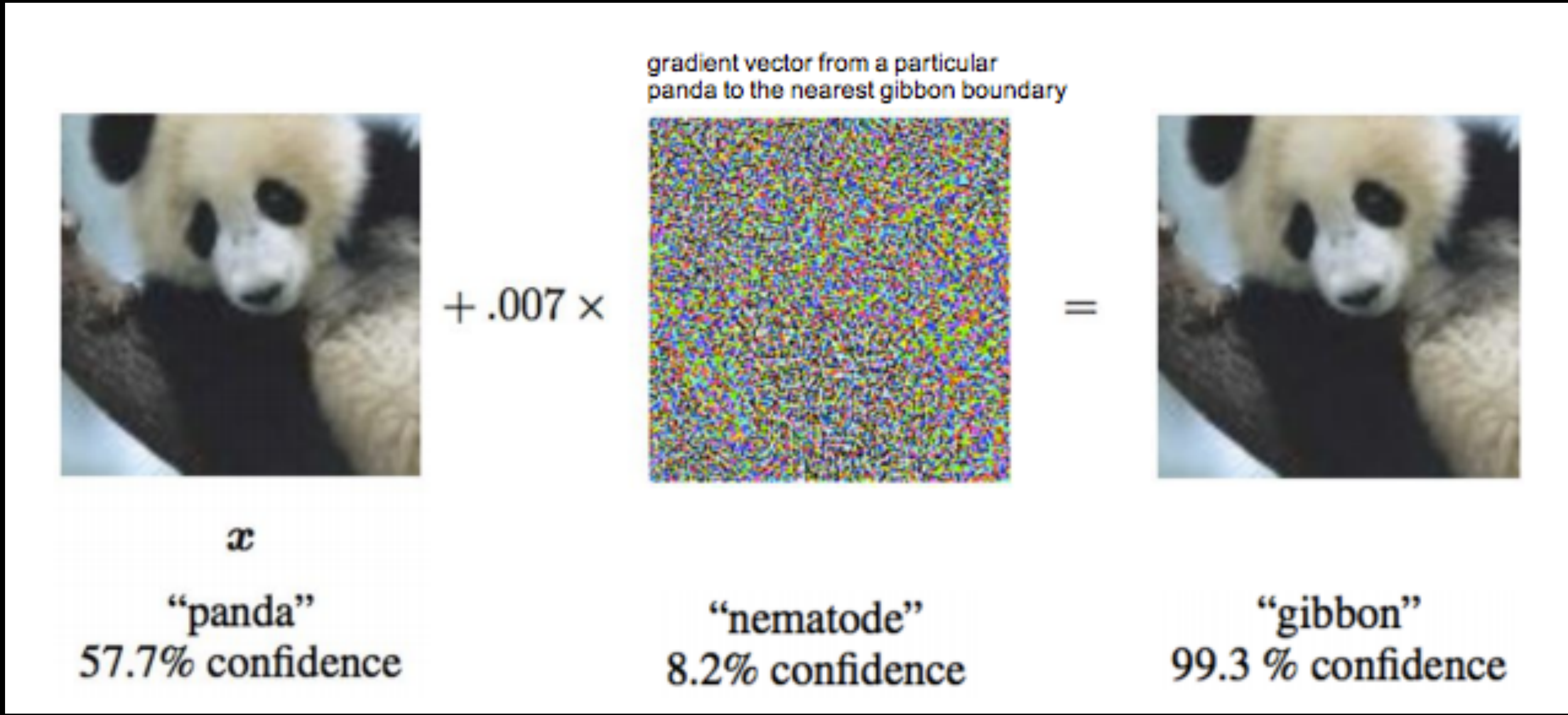
How to exploit an AI image recognition system?



https://www.pyimagesearch.com/2016/06/20/detecting-cats-in-images-with-opencv/

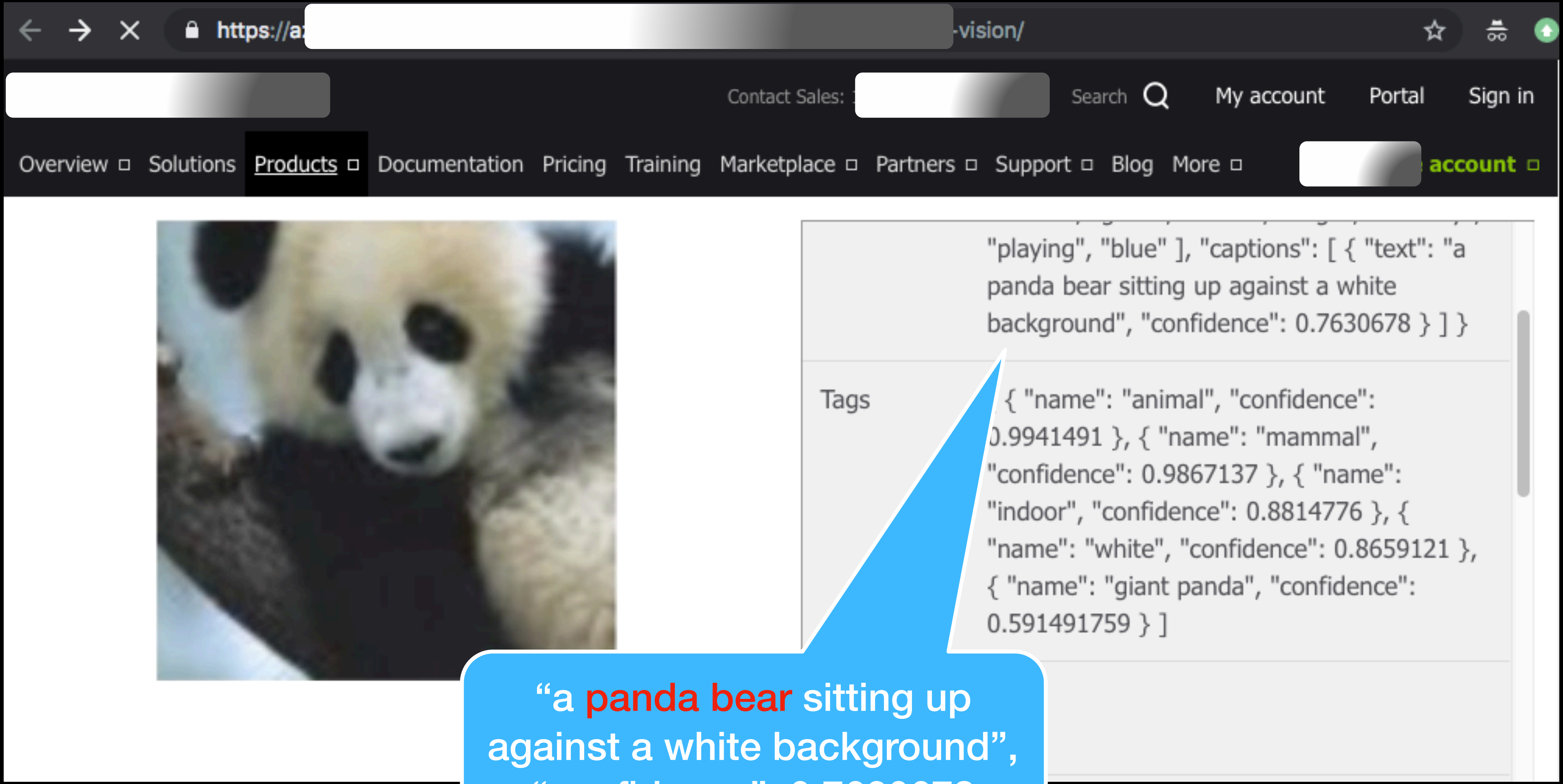
https://www.wired.com/story/tried-to-beat-face-id-and-failed-so-far/

The Famous Adversarial ML Example



Ian J. Goodfellow, Jonathon Shlens & Christian Szegedy, **EXPLAINING AND HARNESSING ADVERSARIAL EXAMPLES**

What if we feed this famous adversarial example to commercial image recognition systems?



"a panda bear sitting up against a white background", "confidence": 0.7630678

Other Examples from Adversarial ML Papers

Original Image (299x299)

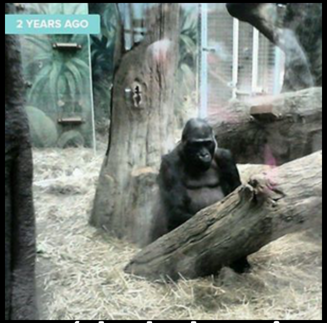
'gorilla': 0.96459390

'cheeseburger': 0.91612280

'balloon': 0.99278200

Theory

Adversarial Examples (299x299)



'sloth_bear': 0.14774416

'Dungeness_crab': 0.10425235

'parachute': 0.52863985

Model: Inception-V3
Data Source: NIPS 2017: Non-targeted Adversarial Attack Competition
https://www.kaggle.com/c/nips-2017-non-targeted-adversarial-attack#Dataset

Hidden Assumption

Adversarial Examples
(299x299)

'sloth_bear': 0.14774416

'Dungeness_crab': 0.10425235

'parachute': 0.52863985

Model: Inception-V3
Data Source: NIPS 2017: Non-targeted Adversarial Attack Competition
https://www.kaggle.com/c/nips-2017-non-targeted-adversarial-attack#Dataset

When Adversarial ML meet Reality

Adversarial Examples
(299x299)

'sloth_bear': 0.14774416

'Dungeness_crab': 0.10425235

'parachute': 0.52863985

Reality

Adversarial Examples
Central Cropped to 87.5%
(263x263)

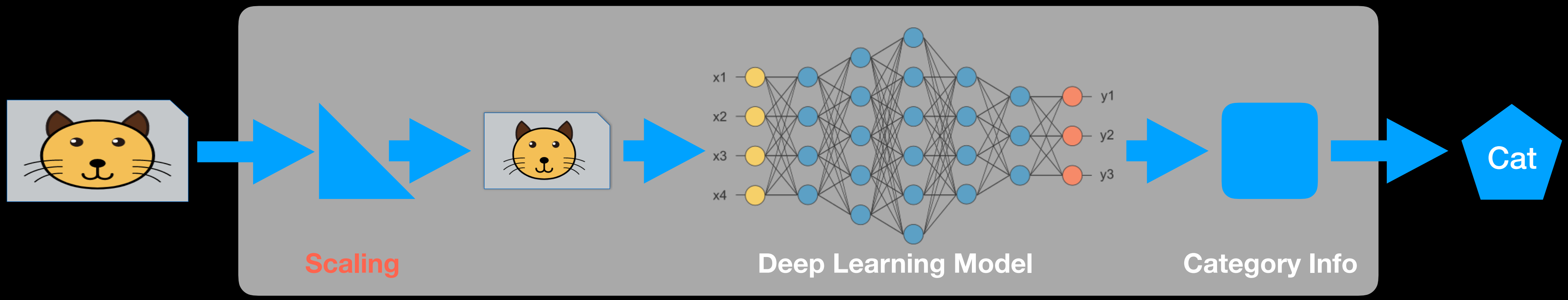
'gorilla': 0.91946507

'cheeseburger': 0.97768340

'balloon': 0.99310110

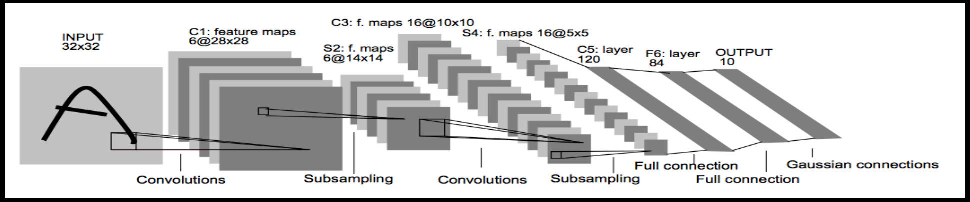
Adversarial samples failed with simple scaling!

Data Flow in Deep Learning Image Applications



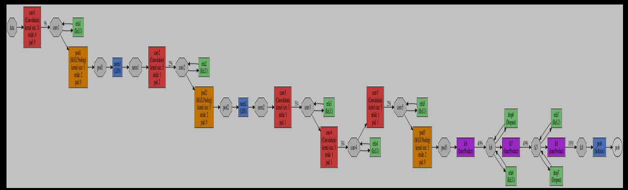
A Hidden Assumption of Deep Learning Applications

MNIST



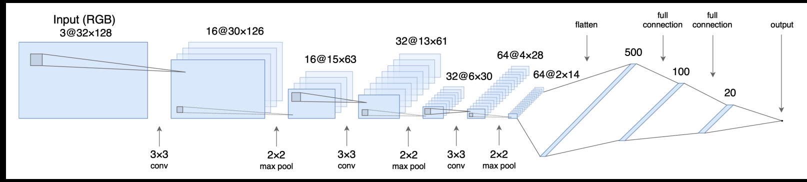
http://yann.lecun.com/exdb/publis/pdf/lecun-01a.pdf

ImageNet



https://github.com/BVLC/caffe/tree/master/examples/cpp_classification

NVIDIA PX DAVE-2



https://images.nvidia.com/content/tegra/automotive/images/2016/solutions/pdf/end-to-end-dl-using-px.pdf

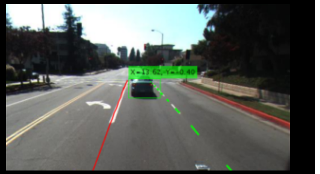
Deep Learning Model Input Requirement (pixel x pixel)

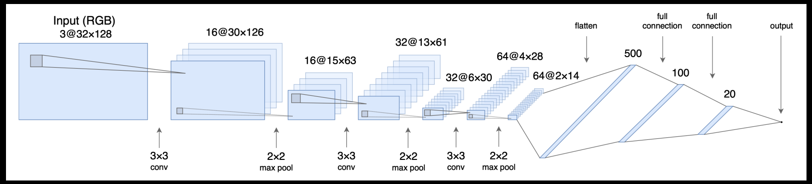
MNIST	28x28
ImageNet	
AlexNet	227x227
GoogleNet, VGG	224x224
ResNet	224x224
NVIDIA	
DAVE-2 Self-Driving	200x66

What if the input size does not match model scale?

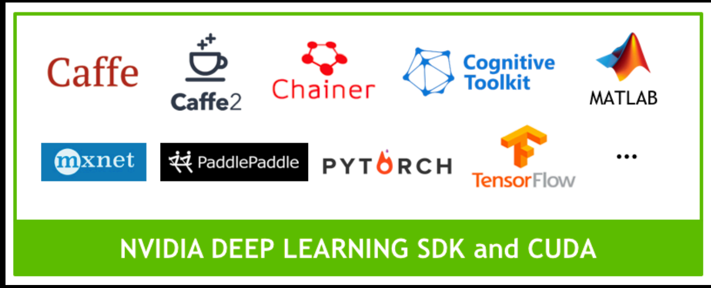
NVIDIA Self-Driving Models and Input Scales

NVIDIA Sample Self-Driving Models





NVIDIA PX2	
DAVE-2 Self-Driving	200x66



NVIDIA Recommended Ecosystem Camera Vendors

Filr	
A310	320x240
A615	640x480
Leopard	
LI-AR0231	1920x1208
SEKONIX	
SF3326-100	1920x1208

https://images.nvidia.com/content/tegra/automotive/images/2016/solutions/pdf/end-to-end-dl-using-px.pdf

https://developer.nvidia.com/drive/ecosystem

Scaling Function might be hidden from Developers

Scaling Functions Provided by Frameworks

TensorFlow Example

```
def read_tensor_from_image_file(file_name, input_height=299, input_width=299,
                input_mean=0, input_std=255):
  input_name = "file_reader"
  output_name = "normalized"
  file_reader = tf.read_file(file_name, input_name)
  if file_name.endswith(".png"):
    image_reader = tf.image.decode_png(file_reader, channels = 3,
                                       name='png_reader')
  elif file_name.endswith(".gif"):
    image_reader = tf.squeeze(tf.image.decode_gif(file_reader,
                                                  name='gif_reader'))
  elif file_name.endswith(".bmp"):
    image_reader = tf.image.decode_bmp(file_reader, name='bmp_reader')
  else:
    image_reader = tf.image.decode_jpeg(file_reader, channels = 3,
                                        name='jpeg_reader')
  float_caster = tf.cast(image_reader, tf.float32)
  dims_expander = tf.expand_dims(float_caster, 0);
  resized = tf.image.resize_bilinear(dims_expander, [input_height, input_width])
  normalized = tf.divide(tf.subtract(resized, [input_mean]), [input_std])
  sess = tf.Session()
  result = sess.run(normalized)

  return result
```

tf.image.**resize_bilinear**(dims_expander, [input_height, input_width])

Scaling Functions Provided by Frameworks

DeepDetect Example

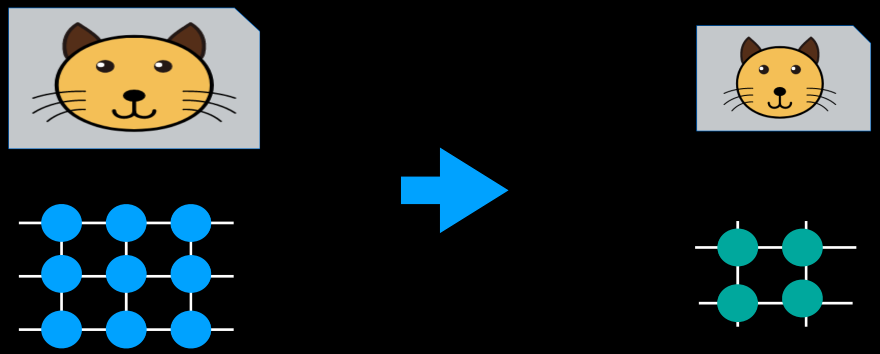
```
int read_file(const std::string &fname)
{
  cv::Mat img = cv::imread(fname,_bw ? CV_LOAD_IMAGE_GRAYSCALE :
                                        CV_LOAD_IMAGE_COLOR);
  if (img.empty())
{
  LOG(ERROR) << "empty image";
  return -1;
}
  _imgs_size.push_back(std::pair<int,int>(img.rows,img.cols));
  cv::Size size(_width,_height);
  cv::Mat rimg;
  cv::resize(img,rimg,size,0,0,CV_INTER_CUBIC);
  _imgs.push_back(rimg);
  return 0;
}
```

```
resize(img,rimg,size,0,0,CV_INTER_CUBIC);
```

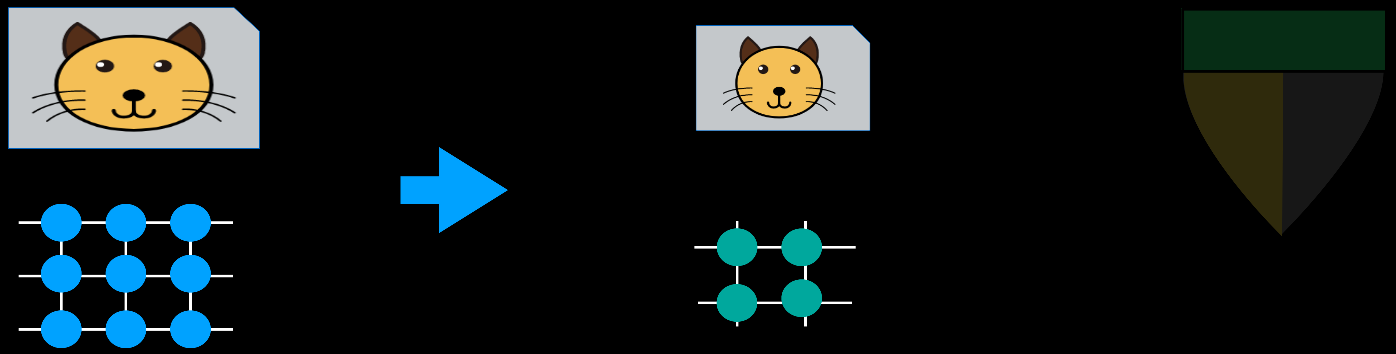
Common Scaling Algorithms and Scaling Attacks

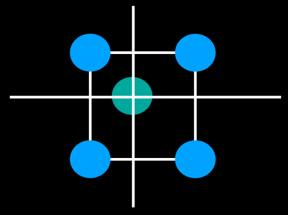
Scaling and Interpolation Algorithms

Scaling is supposed to preserve the visual features of an image and thus does not change its semantic meaning.



Scaling and Interpolation Algorithms

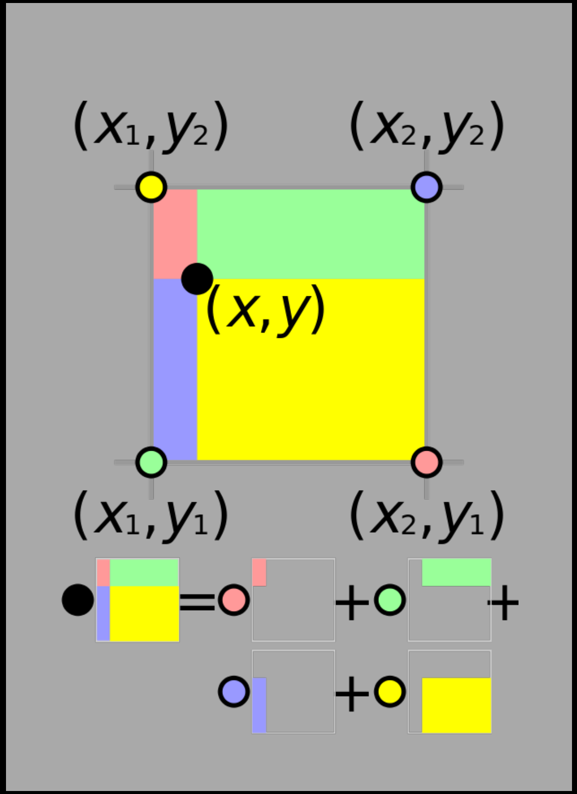




Interpolation: infer the pixel value at each missing point

Goal: to preserve visual features (and hopefully the semantic meanings)

Popular Scaling Algorithms



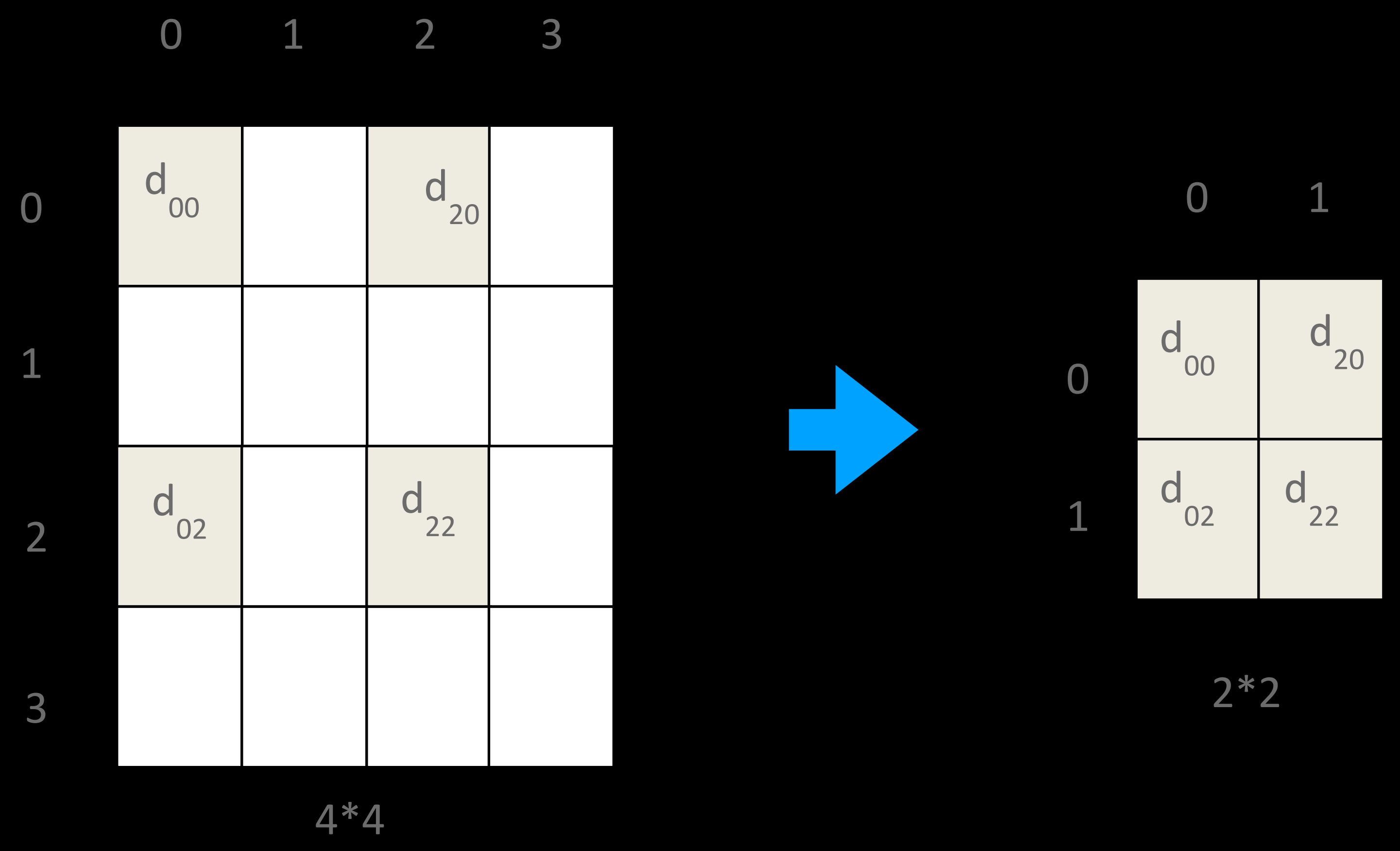
Bilinear Interpolation:

Value at (x,y) =
Sum of the value at each spot multiplied by the area of the rectangle divided by the total area of all four rectangles

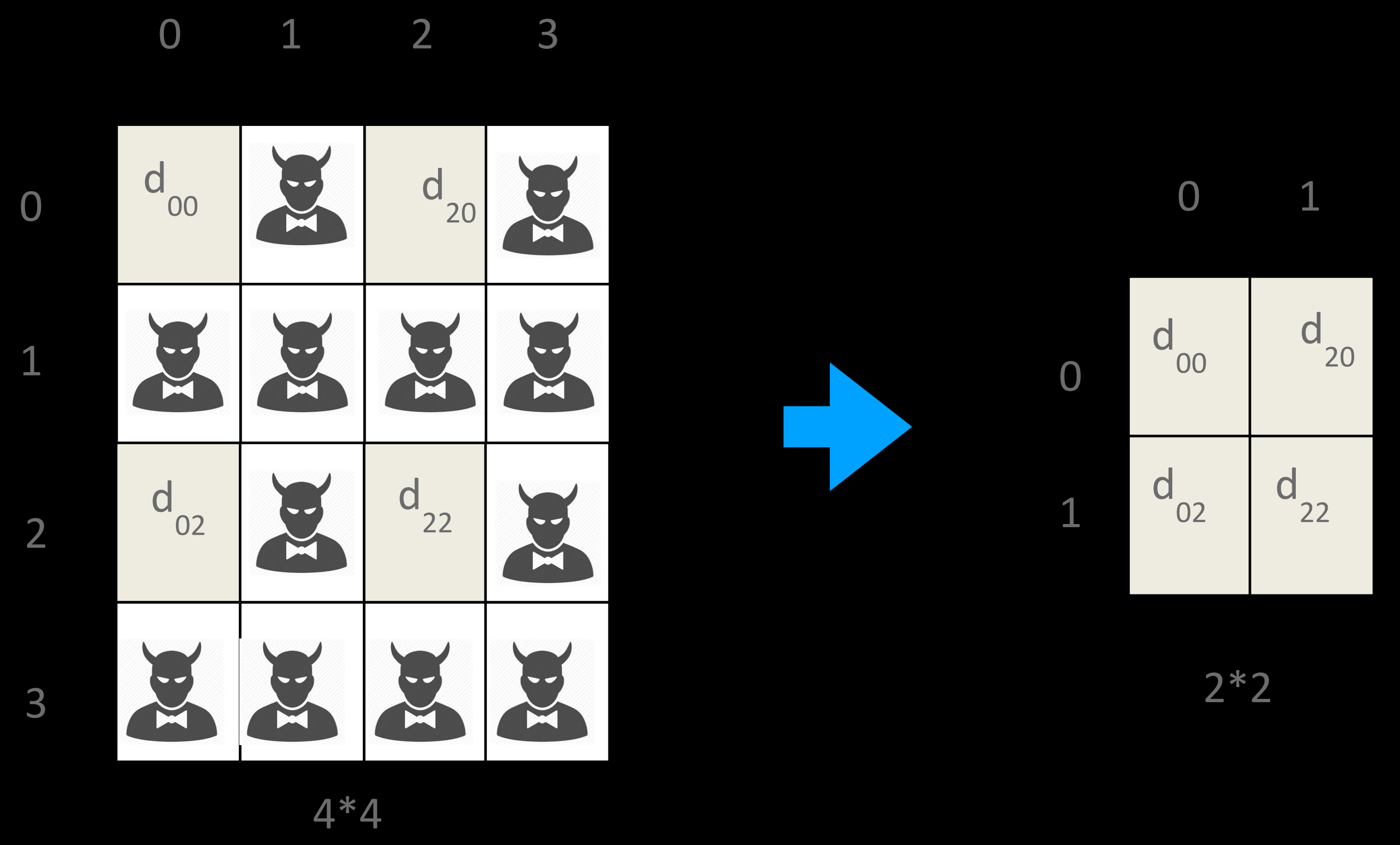
https://en.wikipedia.org/wiki/Bilinear_interpolation

$$f(i+u,j+v) = (1-u)(1-v)f(i,j) + (1-u)vf(i,j+1) + u(1-v)f(i+1,j) + uvf(i+1,j+1)$$

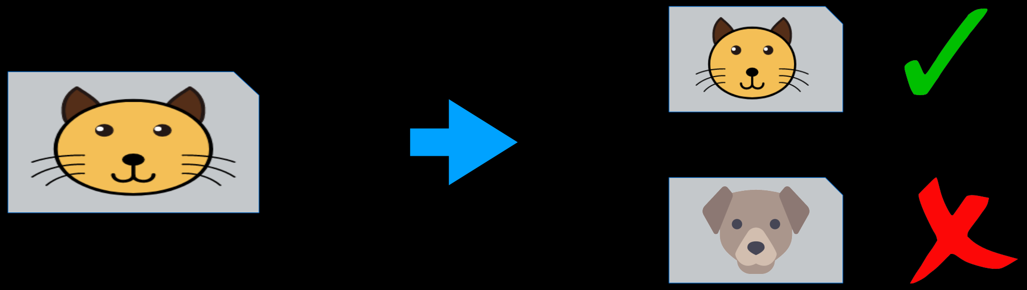
Nearest Neighbor Scaling Algorithm



Consequence of Scaling

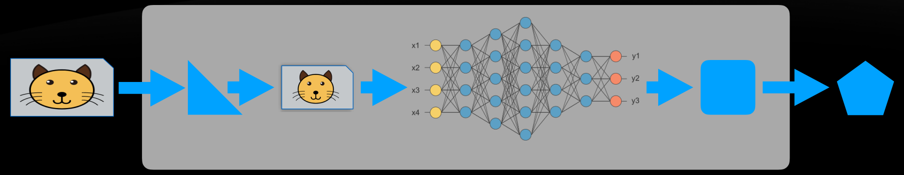


Examples of Scaling Effect



Scaling is not supposed to change the semantic meaning of the input image

If we know the scaling algorithms and sizes …



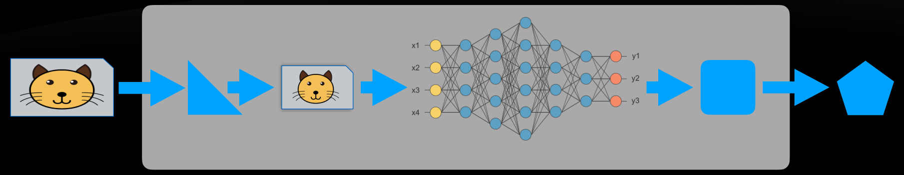
Attack Leveraging the Scaling Effect (prior work)

Data Scaling Attacks in Deep Learning Applications
https://www.defcon.org/html/defcon-china/dc-cn-speakers.html#LiKang

Attack Sample #2 (Traffic Sign)

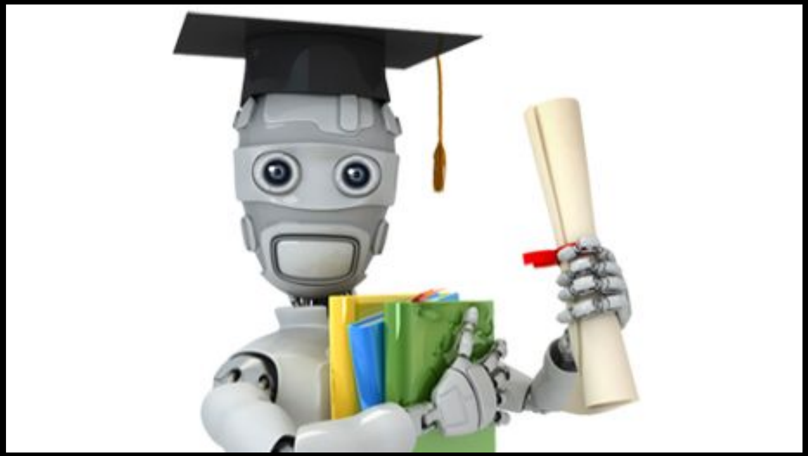
Attack Sample #2 (Traffic Sign)

How to infer the scaling factor in cloud services?



Inferring Preprocessing Parameters

inferring parameters by sending queries
and observe responses

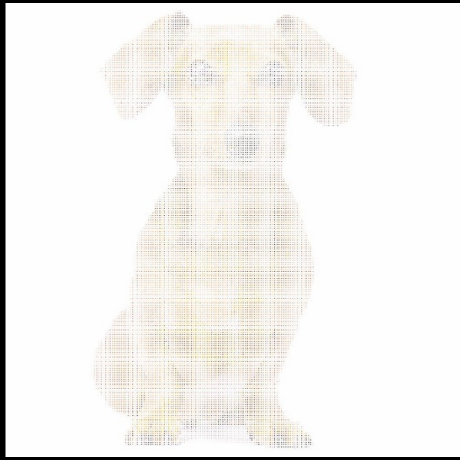


attacker

image recognition service

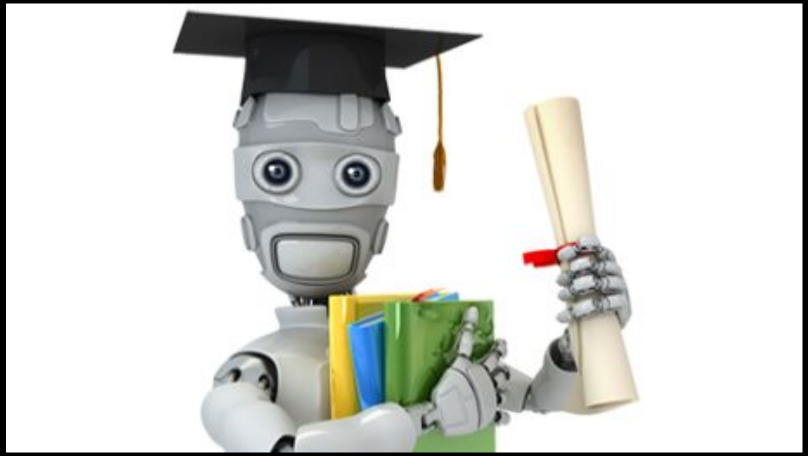
Inferring Preprocessing Parameters

“What can you see?”



{*probeImg*$_1$}

(1024->200,bilinear)



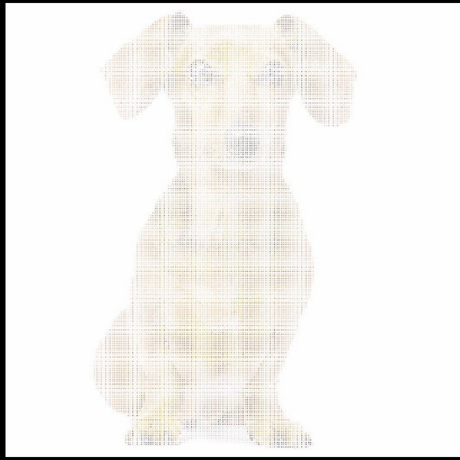
attacker

image recognition service

Using specially crafted images:
meaningful if scaling with the appropriate parameters

Inferring Preprocessing Parameters

"What can you see?"



{*probeImg$_1$*}

(1024->200,bilinear)

"A meaningless image."

attacker

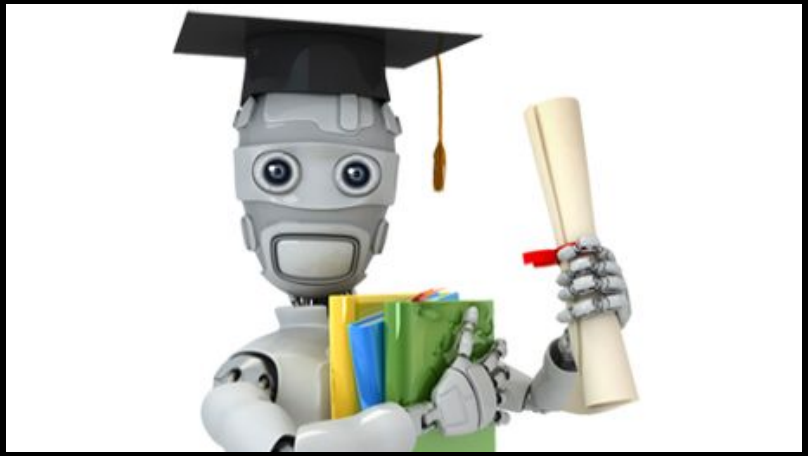


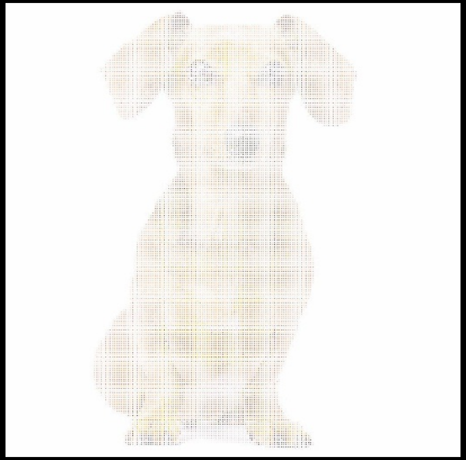
image recognition service

Using specially crafted images:
meaningful if scaling with the appropriate parameters

Inferring Preprocessing Parameters

“What can you see?”

attacker



{probeImg$_1$}

(1024->200,bilinear)

{probeImg$_k$}

(1024->201, bilinear)

{probeImg$_k$}

(1024->202, bilinear)

“A meaningless image.”

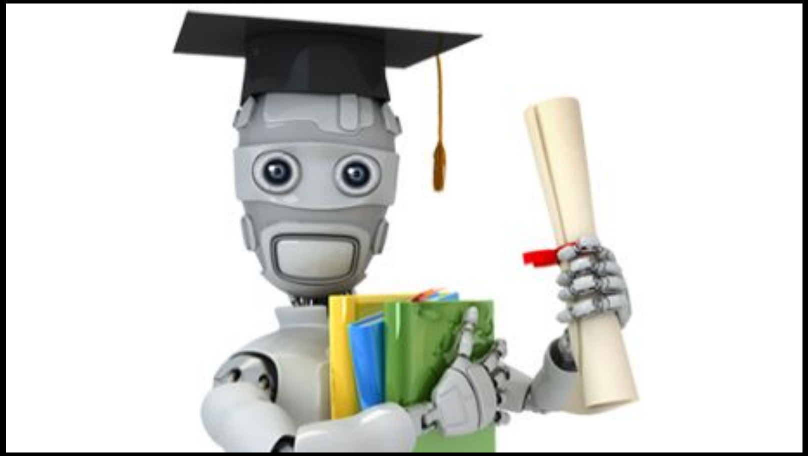
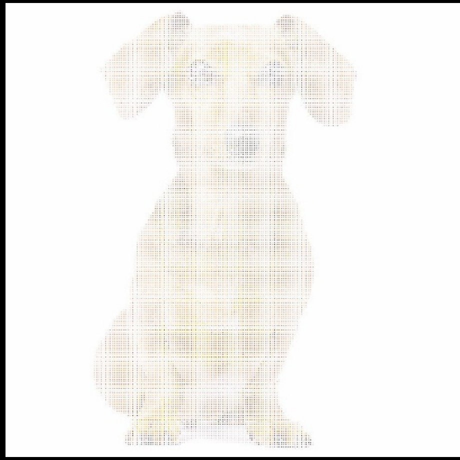


image recognition service

Inferring Preprocessing Parameters

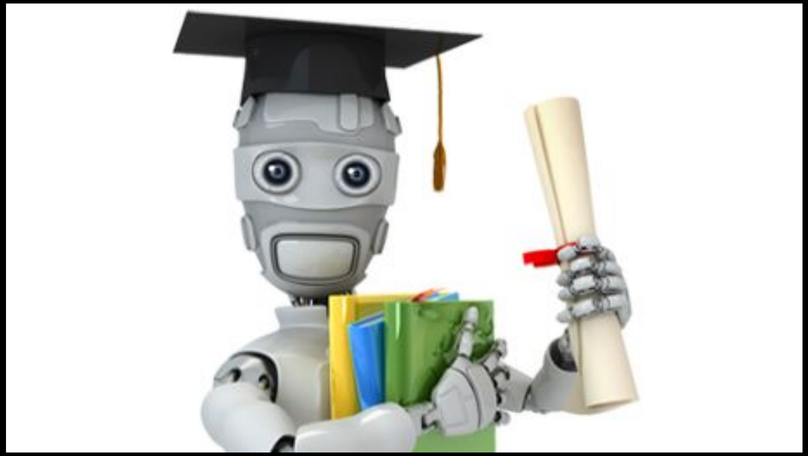
"What can you see?"



"A dog."

{*probeImg*$_k$}

(1024->202, bilinear)

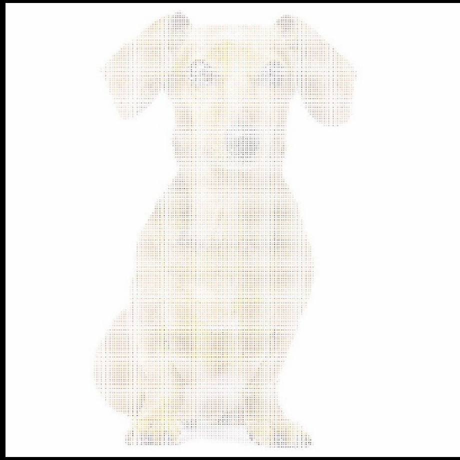


attacker

image recognition service

Inferring Preprocessing Parameters

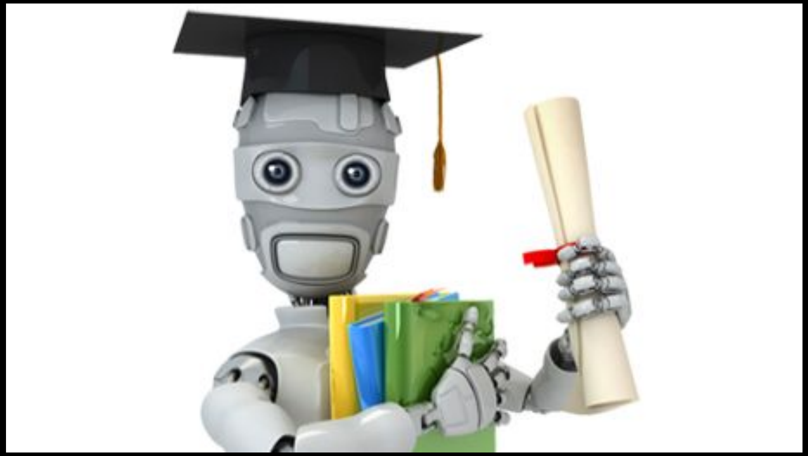
“What can you see?”



{$probeImg_k$}

(1024->202, bilinear)

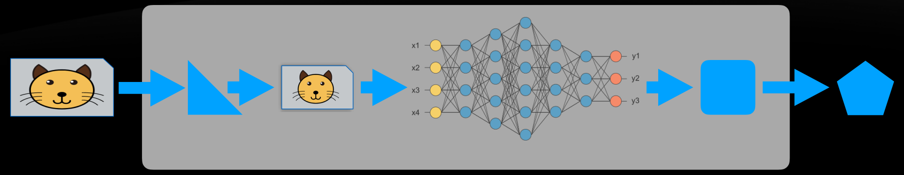
“A dog.”



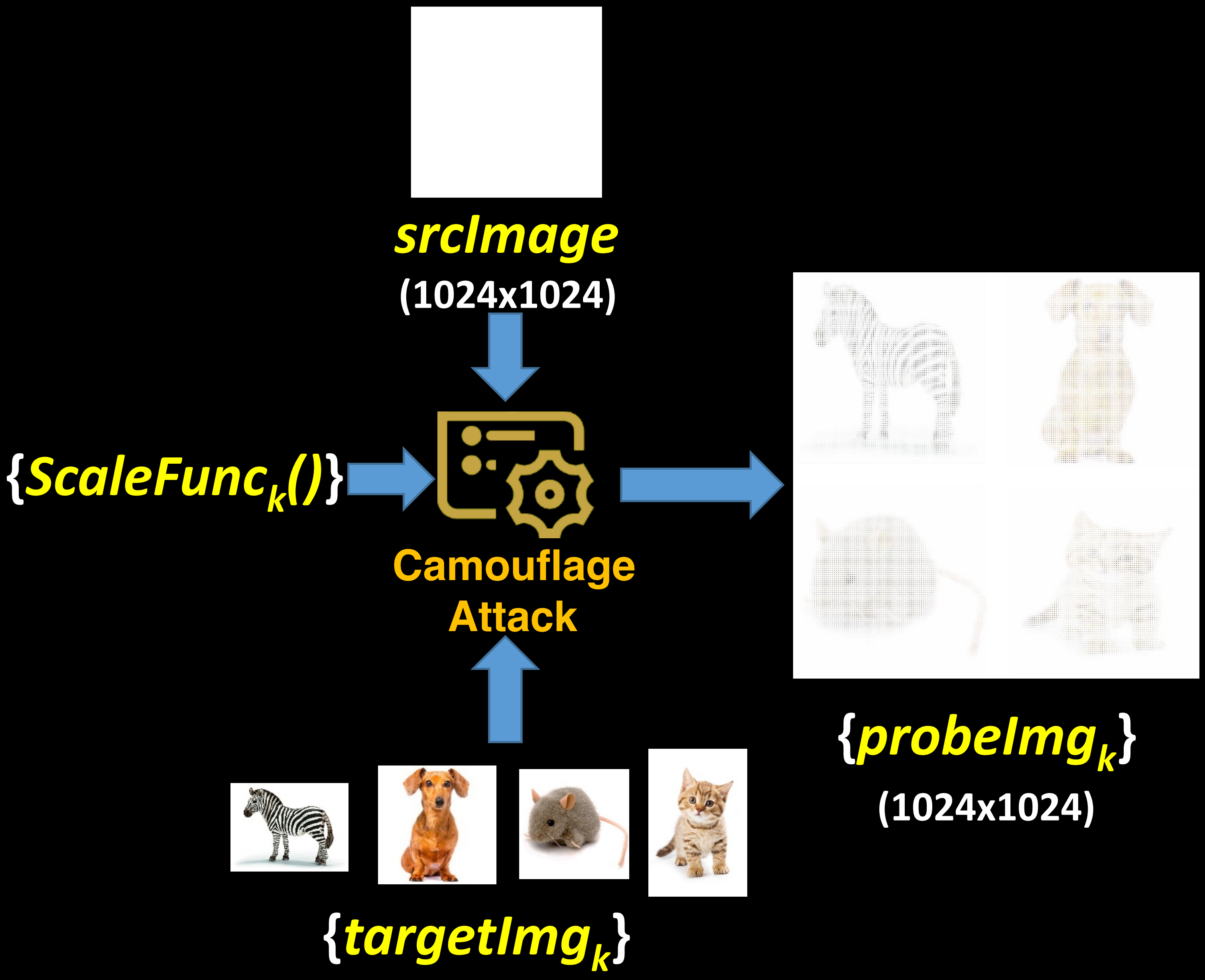
“Scaling method: *bilinear*
Model Input Size: 202x202”

Brute-force Scaling Attack

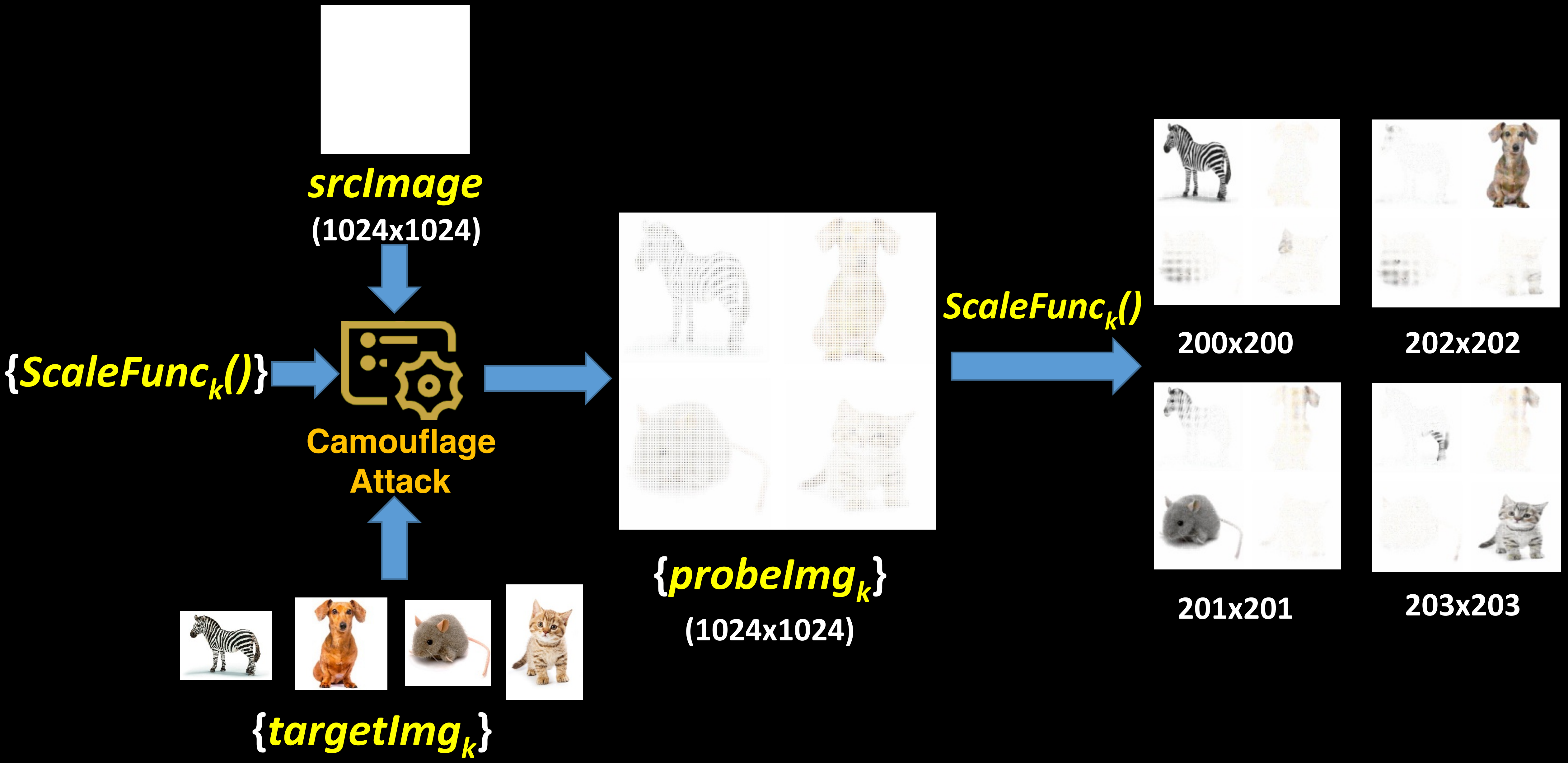
The efficiency of brute-force inference is low



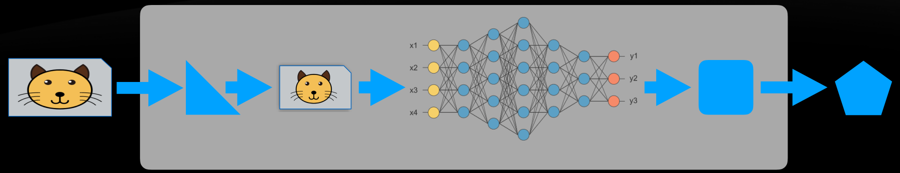
Improving Efficiency by Overlay Scaling Attacks



Improving Efficiency by Overlay Scaling Attacks



Once we know the preprocessing parameters



Attack Effect Examples

Result from AI Image Recognition Services ?

Attack Effect Examples

Attack Effect Example (Vendor A)

"grey wolf：88%

Eskimo dog：15%

white wolf：14%

…"

Attack Effect Example (Vendor B)

“grey wolf：0.939

mexican wolf：0.015

arctic wolf：0.011

…”

Attack Effect Example (Vendor M)

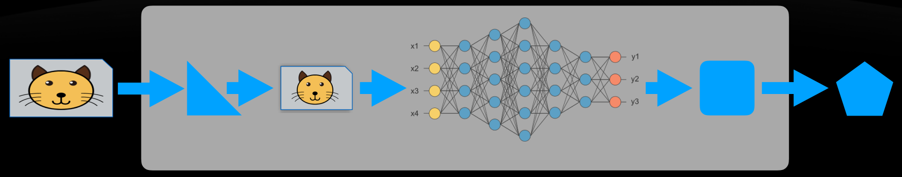
FEATURE NAME:	VALUE
Description	{ "tags": ["animal", "mammal", "wolf", "looking"], "captions": [{ "text": "a close up of a wolf", "confidence": 0.707954049 }] }
Tags	[{ "name": "animal", "confidence": 0.9989328 }, { "name": "mammal", "confidence": 0.9908992 }, { "name": "wolf", "confidence": 0.981169641 }]
Image format	"Jpeg"
Image dimensions	1024 x 1024
Clip art	0

- ✓ Description: { "tags": ["animal", "mammal", "wolf", "looking"], "captions": [{ "text": "a close up of a wolf", "confidence": 0.707954049 }] }
- ✓ Tags: [..., { "name": "wolf", "confidence": 0.981169641 }]

Attack Effect Example (Vendor T)

"white wolf： 98.52%
gray wolf： 0.50%
arctic fox： 0.40%
…"

Image Scaling Attack Toolkit



Summary

- AI-based image recognition services are getting increasingly popular
- Samples from adversarial ML fails to fool commercial image API
- Image pre-processing is widely and often implicitly used in AI-based image recognition services
- Attackers can infer image recognition service parameters and launch effective evading attacks