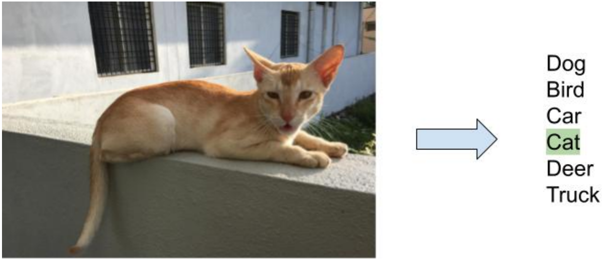
Deep Learning for Computer Vision

Dr. Konda Reddy Mopuri
Mehta Family School of Data Science and Artificial Intelligence
IIT Guwahati

Aug-Dec 2022

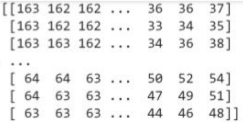
Image Classification

- Familiar CV task: i/p is an image and o/p is a category label



Dog
Bird
Car
Cat
Deer
Truck

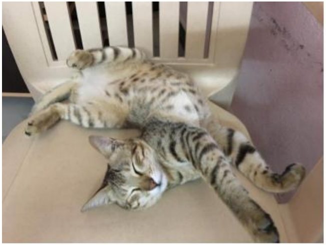
Challenge: Semantic gap

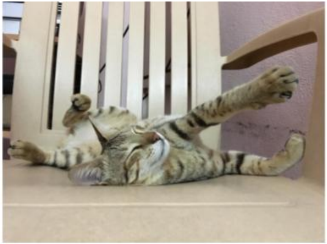


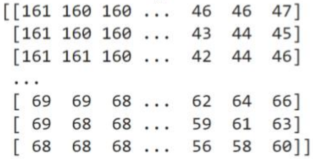
```
[[163 162 162 ...  36  36  37]
 [163 162 162 ...  33  34  35]
 [163 163 162 ...  34  36  38]
 ...
 [ 64  64  63 ...  50  52  54]
 [ 64  63  63 ...  47  49  51]
 [ 63  63  63 ...  44  46  48]]
```

For computers

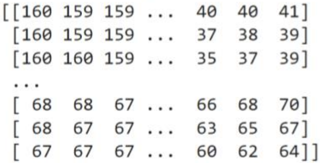
Challenge: View point variation





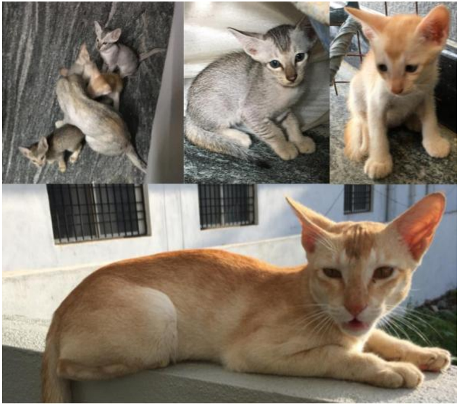


```
[[161 160 160 ...  46  46  47]
 [161 160 160 ...  43  44  45]
 [161 161 160 ...  42  44  46]
 ...
 [ 69  69  68 ...  62  64  66]
 [ 69  68  68 ...  59  61  63]
 [ 68  68  68 ...  56  58  60]]
```

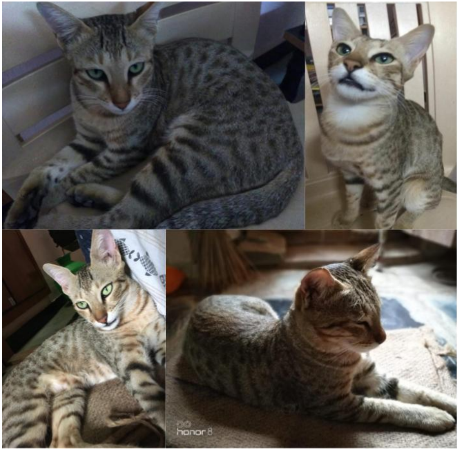


```
[[160 159 159 ...  40  40  41]
 [160 159 159 ...  37  38  39]
 [160 160 159 ...  35  37  39]
 ...
 [ 68  68  67 ...  66  68  70]
 [ 68  67  67 ...  63  65  67]
 [ 67  67  67 ...  60  62  64]]
```

Challenge: intra-class-variation



Challenge: lighting-variation



Other Challenges

- Occlusion
- Deformation
- Clutter
- . . .

Image classification: elementary task for other CV tasks

- Object detection

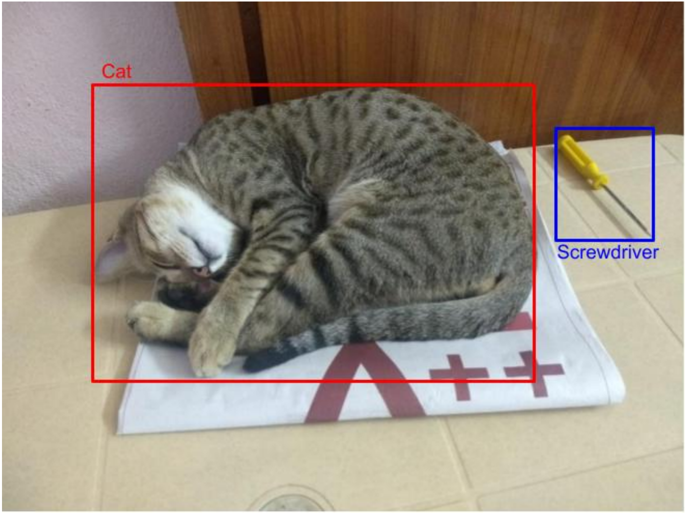
Image classification: elementary task for other CV tasks

- Object detection
- Caption Generation

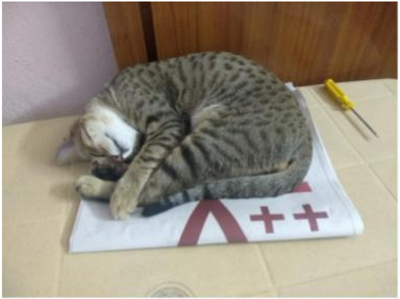
Image classification: elementary task for other CV tasks

- Object detection
- Caption Generation
- Playing Chess/Go
- . . .

Object Detection



Caption Generation

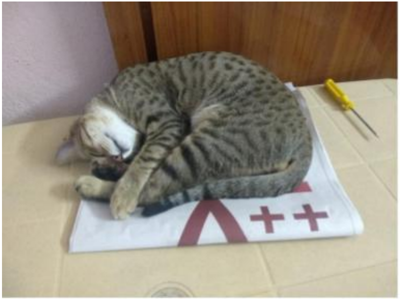


→

Dog
Paper
Cat
Table
Screwdriver
Truck
Pen
Sleeping
.....
<EoS>

Predict the next word → **?**

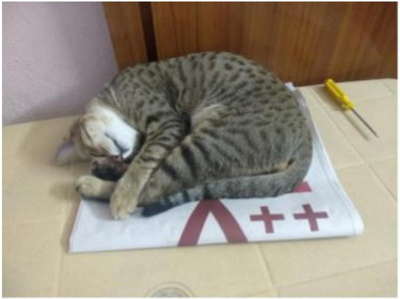
Caption Generation



Dog
Paper
Cat
Table
Screwdriver
Truck
Pen
Sleeping
.....
<EoS>

Predict the next word **Cat**

Caption Generation

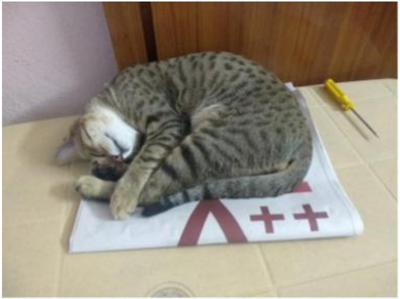


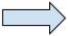
Dog
Paper
Cat
Table
Screwdriver
Truck
Pen
Sleeping
.....
<EoS>

Predict the next word

Cat Sleeping

Caption Generation



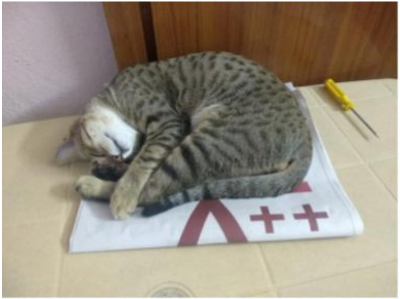


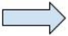
Dog
Paper
Cat
Table
Screwdriver
Truck
Pen
Sleeping
.....
<EoS>

Predict the next word

Cat Sleeping on

Caption Generation



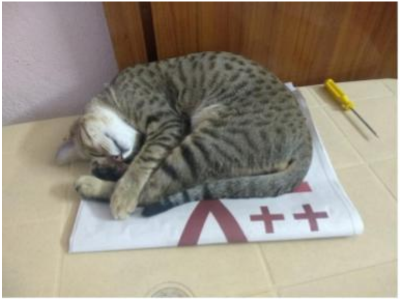


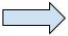
Dog
Paper
Cat
Table
Screwdriver
Truck
Pen
Sleeping
.....
<EoS>

Predict the next word

Cat Sleeping on the

Caption Generation



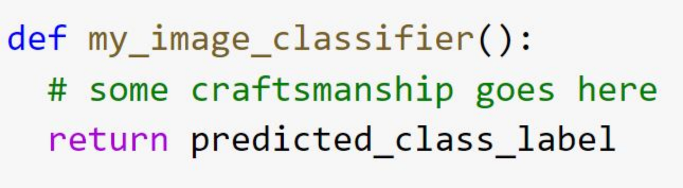


Dog
Paper
Cat
Table
Screwdriver
Truck
Pen
Sleeping
.....
<EoS>

Predict the next word

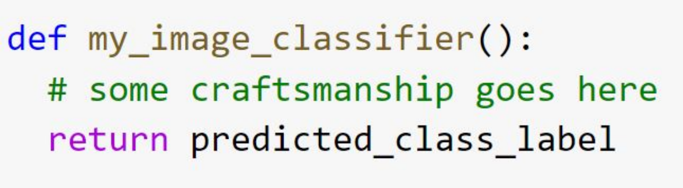
Cat Sleeping on the table

How to build an image classifier?



```
def my_image_classifier():
  # some craftsmanship goes here
  return predicted_class_label
```

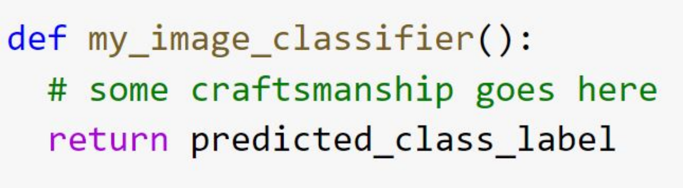
How to build an image classifier?



```
def my_image_classifier():
  # some craftsmanship goes here
  return predicted_class_label
```

- Are there any rules that can we can hard-code? (unlike writing a program for addition of two numbers)

How to build an image classifier?



```
def my_image_classifier():
  # some craftsmanship goes here
  return predicted_class_label
```

- Are there any rules that can we can hard-code? (unlike writing a program for addition of two numbers)
- One can see that such an algorithm is not (i) gonna be robust, and (ii) transferable across categories

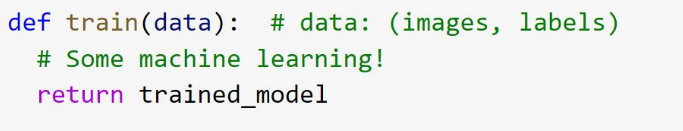
Here comes Machine Learning!

- Instead of trying to encode our knowledge of the objects, we take a data-driven approach

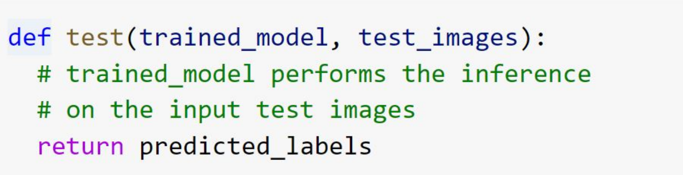
Here comes Machine Learning!

- Instead of trying to encode our knowledge of the objects, we take a data-driven approach
- Build algorithms that can learn from the data

Here comes Machine Learning!



```python
def train(data):  # data: (images, labels)
  # Some machine learning!
  return trained_model
```



```python
def test(trained_model, test_images):
  # trained_model performs the inference
  # on the input test images
  return predicted_labels
```

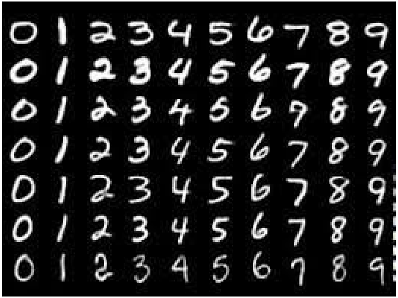
Common datasets for image classification: MNIST

- 10-class problem: $\{0, 1, 2, \ldots, 9\}$

Source

Common datasets for image classification: MNIST

- 10-class problem: $\{0, 1, 2, \ldots, 9\}$
- 28×28 gray-scale images



Source

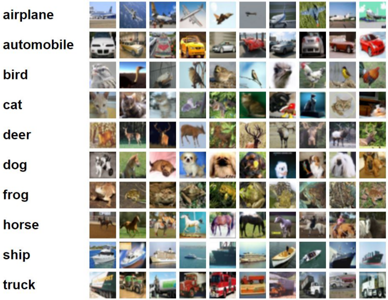
Common datasets for image classification: MNIST

- 10-class problem: $\{0, 1, 2, \ldots, 9\}$
- 28×28 gray-scale images
- $50K$ for training, and $10K$ for testing

Source

Common datasets for image classification: CIFAR-10

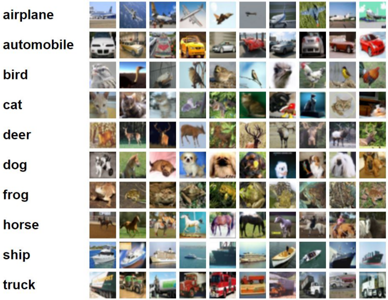
- 10-class problem: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck



Source

Common datasets for image classification: CIFAR-10

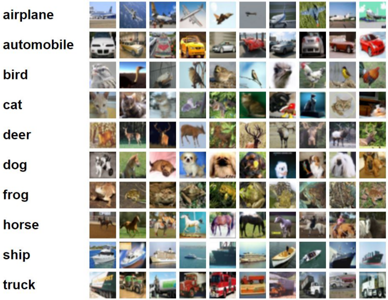
- 10-class problem: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck
- 32×32 RGB images



Source

Common datasets for image classification: CIFAR-10

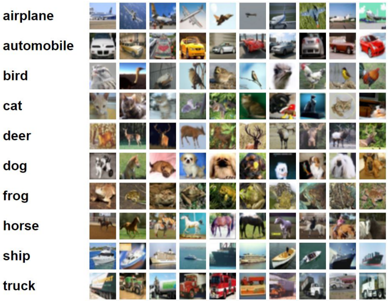
- 10-class problem: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck
- 32×32 RGB images
- $50K$ for training, and $10K$ for testing



Source

Common datasets for image classification: CIFAR-10

- 10-class problem: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck
- $32 \times 32 \times 3$ (RGB) images
- $50K$ for training, and $10K$ for testing



Source

We work with CIFAR-10

- We use CIFAR-10 for most of our assignments and experiments
- CIFAR-100 is a related dataset

Common datasets for image classification: ImageNet

- 1000 object categories

Source

Common datasets for image classification: ImageNet

- 1000 object categories
- $1.3M, 50K, 100K$ training, validation and testing images

Source

Common datasets for image classification: ImageNet

- 1000 object categories
- $1.3M, 50K, 100K$ training, validation and testing images
- Considered gold standard (as of 2020s)

Source

Common datasets for image classification

- MIT places

Common datasets for image classification

- MIT places
- Omniglot

Common datasets for image classification

- MIT places
- Omniglot
- iNaturalist
- . . .

Simple Classifier: Nearest neighbor

Simple Classifier: Nearest neighbor

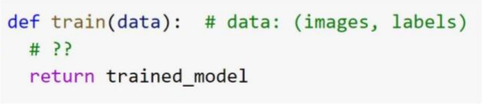
- Training: Remember the labels of all the training data samples

Simple Classifier: Nearest neighbor

- Training: Remember the labels of all the training data samples
- Testing: Predict the label of the nearest training sample

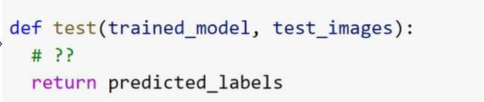
Nearest neighbor classifier

Remember the labels of training samples



```
def train(data):   # data: (images, labels)
  # ??
  return trained_model
```

Pick the label of the closest training sample



```
def test(trained_model, test_images):
  # ??
  return predicted_labels
```