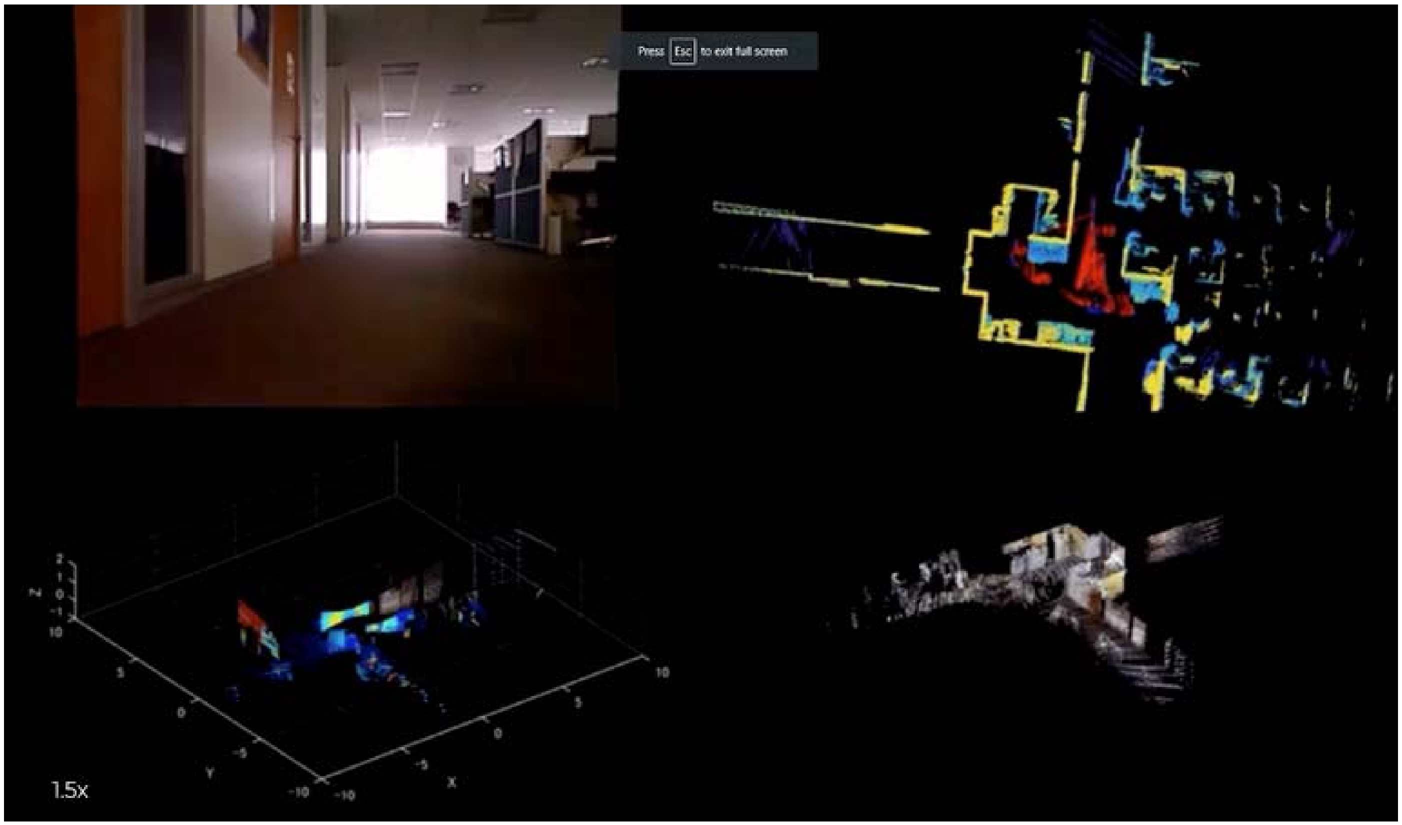
CMPE 185 Autonomous Mobile Robots

Perception: Learning Based Object Classification and Detection

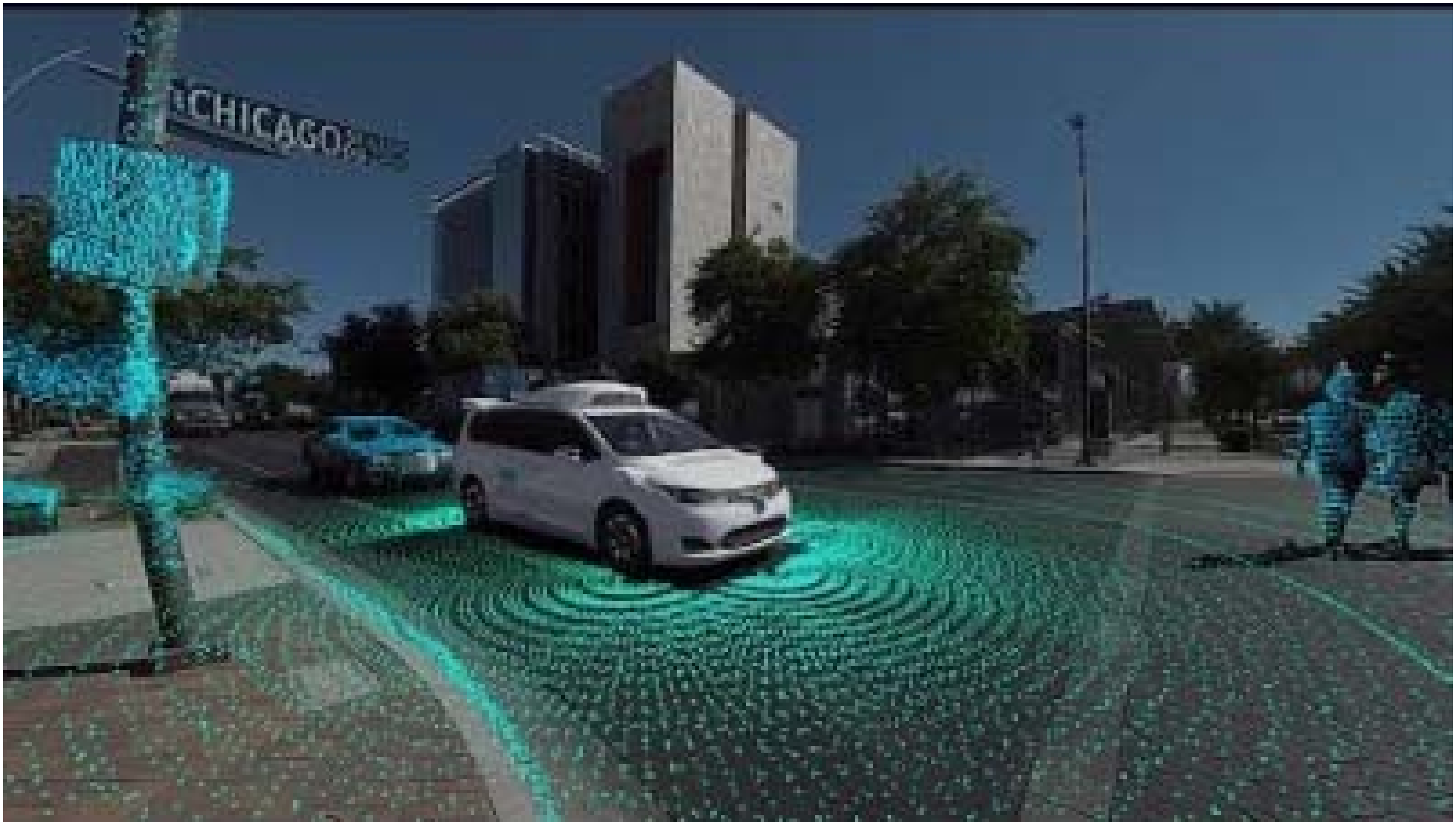
Dr. Wencen Wu
Computer Engineering Department
San Jose State University

Computer Vision

- **Enable robot vision to build environment maps and localize your mobile robot**



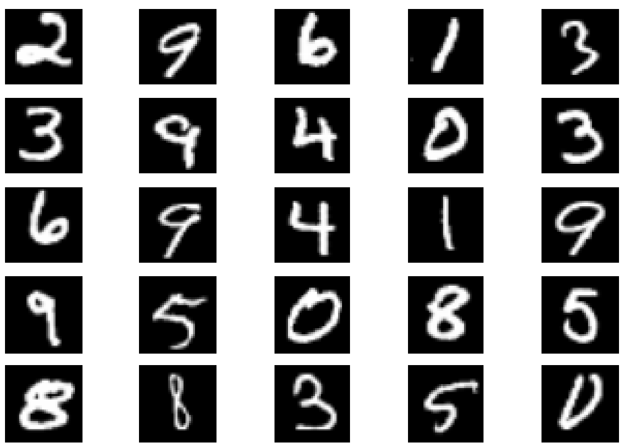
Example: Waymo Experience – Sensor Fusion



Example: Autonomous Parking

Image Classification

- K classes
- Task: assign correct class label to the whole image



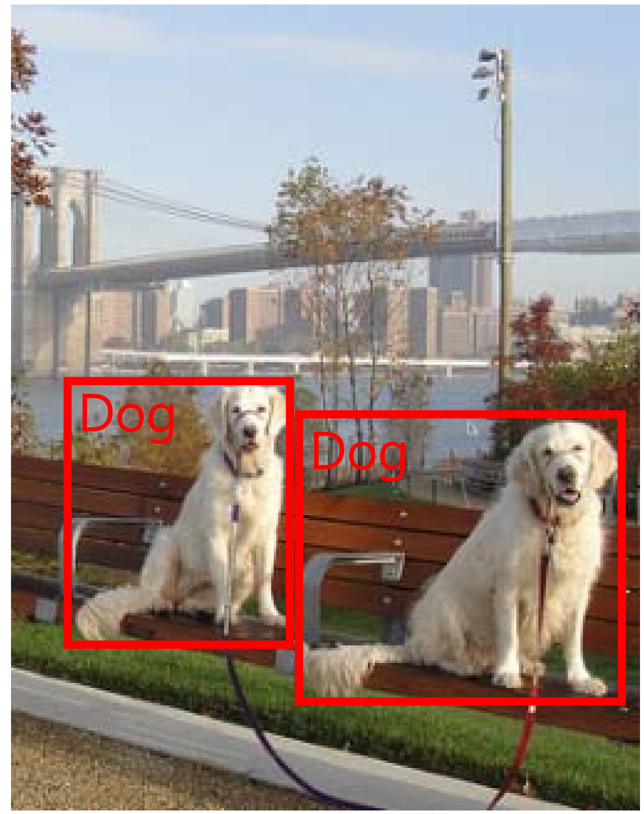
Digit classification (MNIST)

Object recognition (Caltech-101)

Classification v.s. Detection

Classification

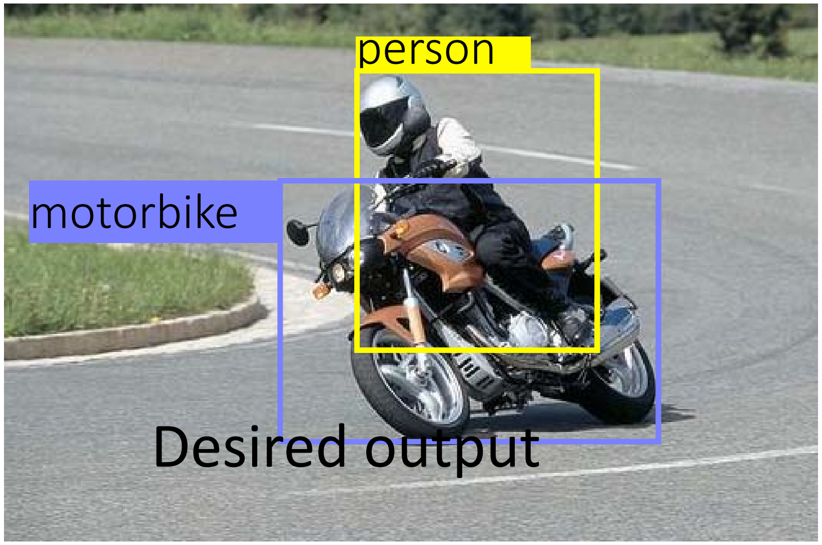
Detection



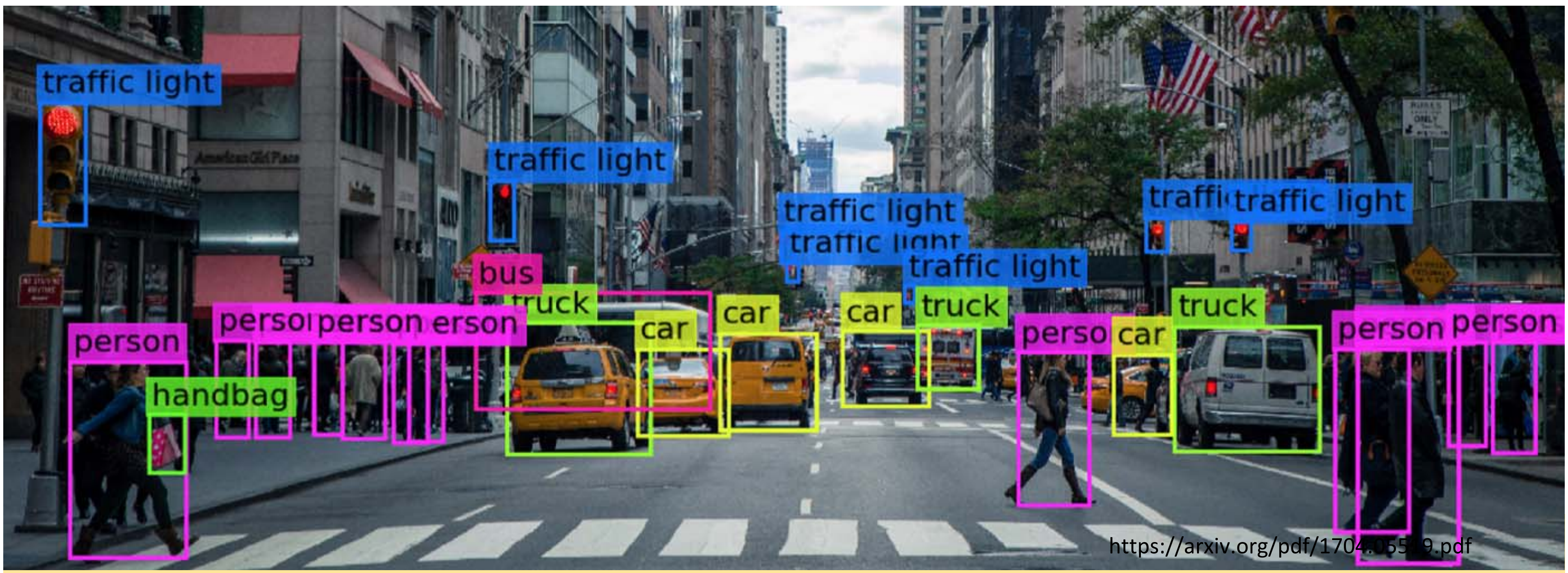
Problem Formulation

- **When performing object detection, we wish to obtain:**
  - A **list of bounding boxes**, or the *(x, y)*-coordinates for each object in an image
  - The **class label** associated with each bounding box
  - The **probability/confidence score** associated with each bounding box and class

{ airplane, bird, motorbike, person, sofa }



Object Detection in Autonomous Driving



https://arxiv.org/pdf/1704.05519.pdf

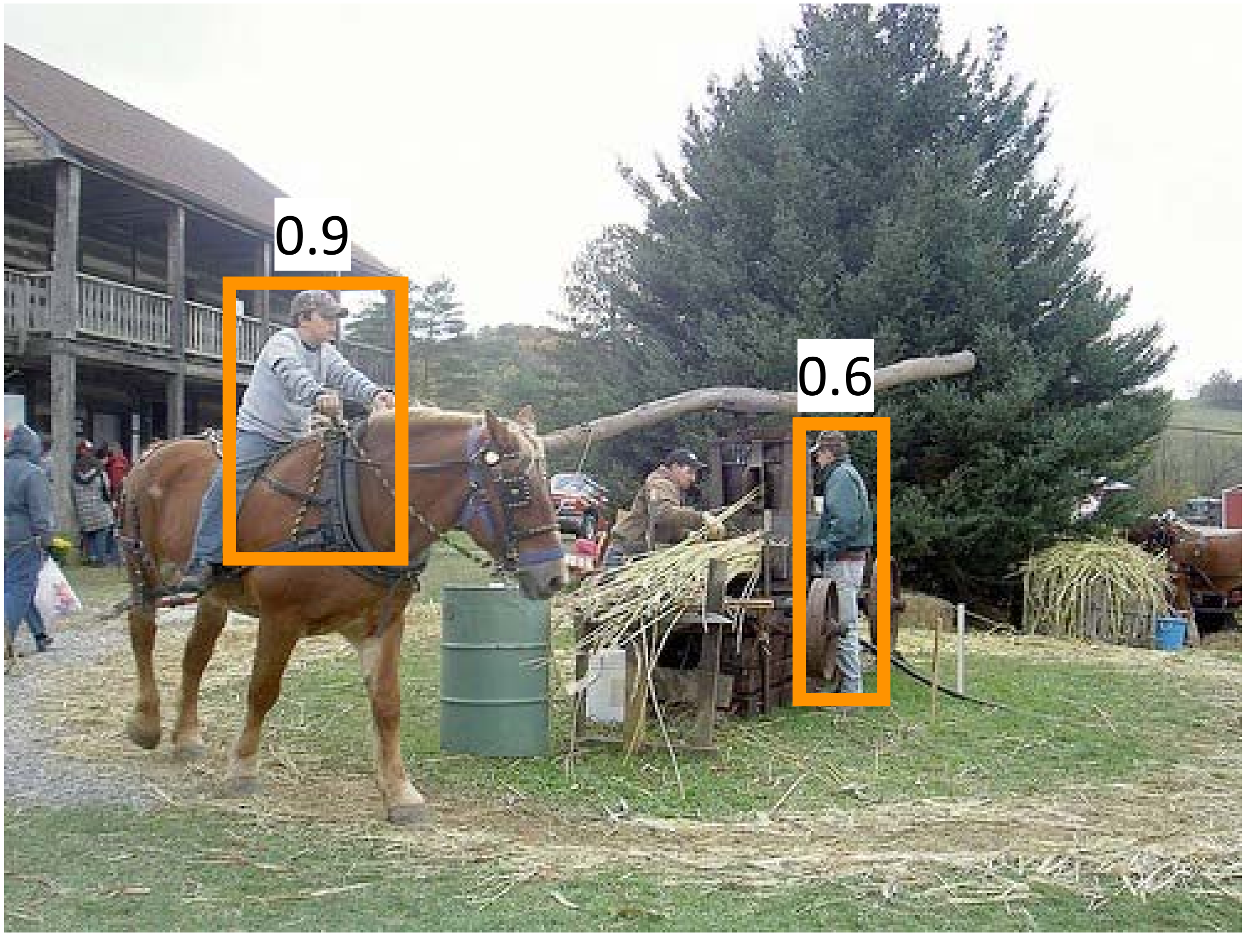
Evaluating a Detector

Test image (previously unseen)

First Detection

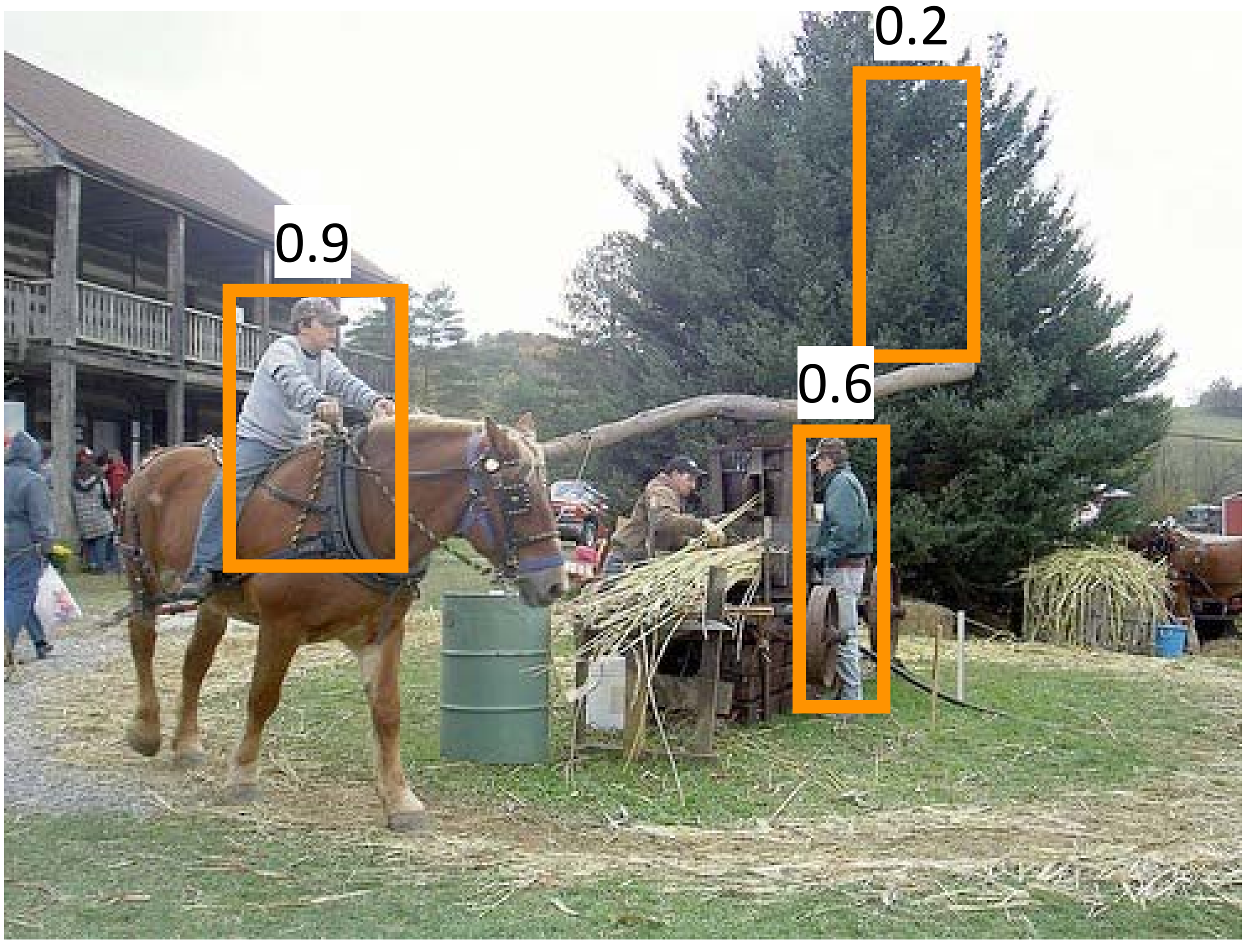
 'person' detector predictions

Second Detection



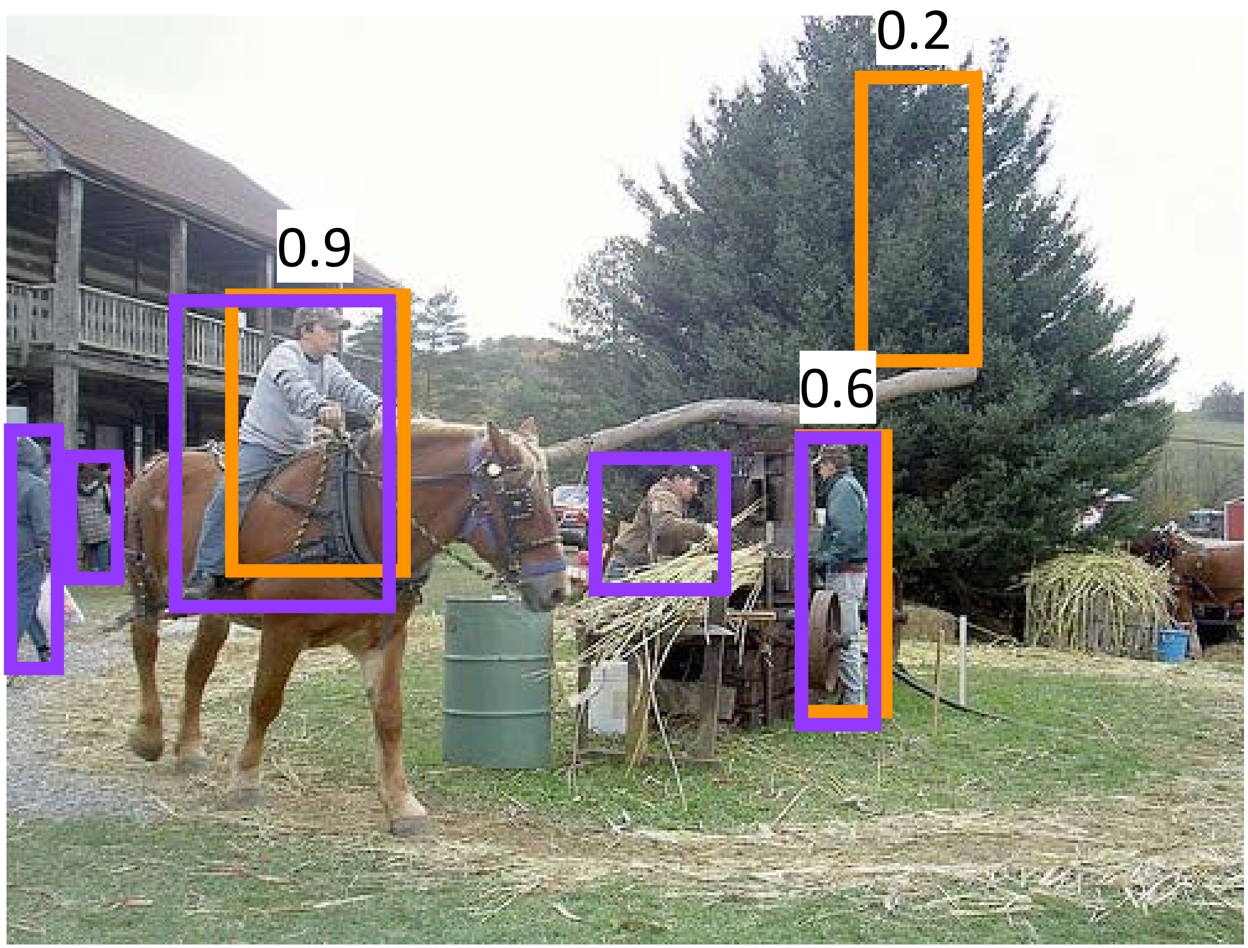
'person' detector predictions

Third Detection



'person' detector predictions

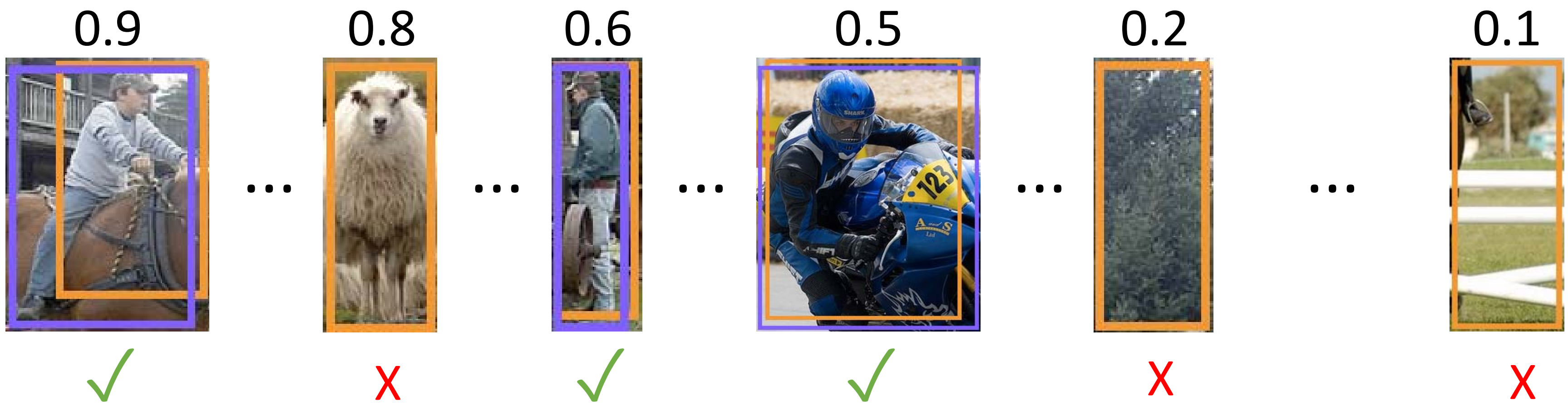
Compare to Ground Truth



'person' detector predictions

ground truth 'person' boxes

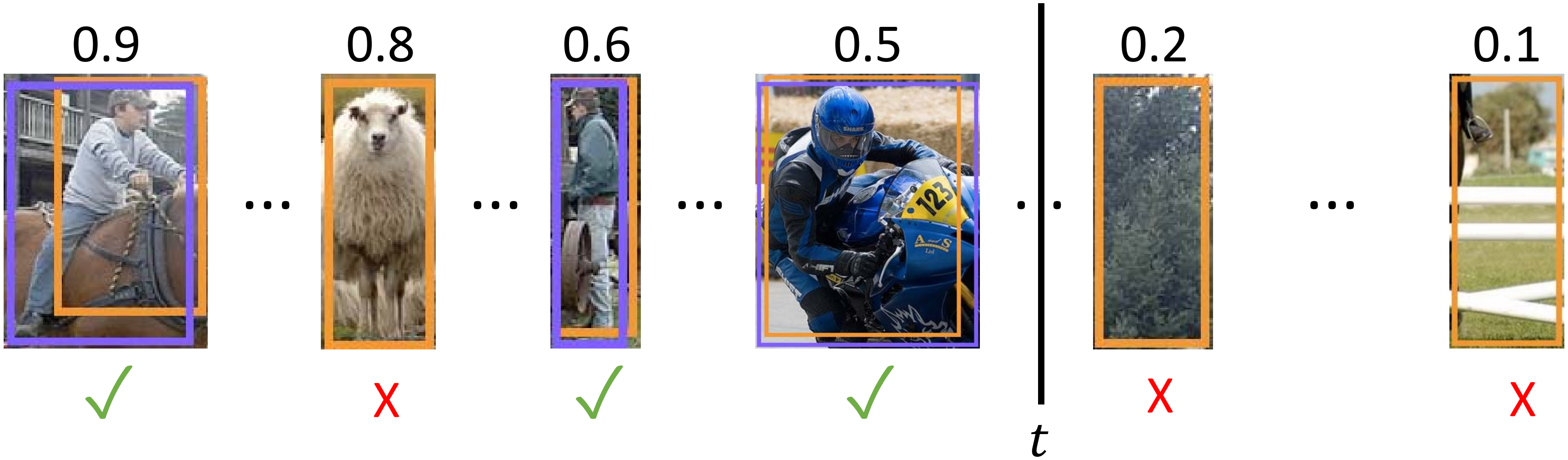
Sort by Confidence



true positive
(high overlap)

false positive
(no overlap, low overlap, or duplicate)

Evaluation Metric



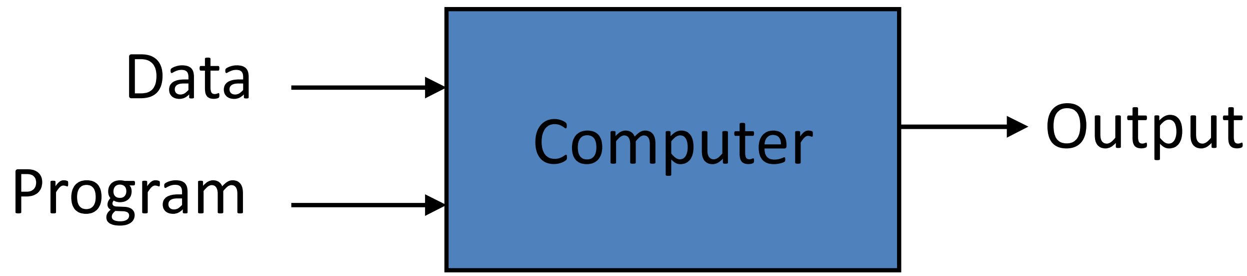
$$precision@t = \frac{\#true\ positives@t}{\#true\ positives@t + \#false\ positives@t} \qquad \frac{\checkmark}{\checkmark + X}$$

$$recall@t = \frac{\#true\ positives@t}{\#ground\ truth\ objects}$$

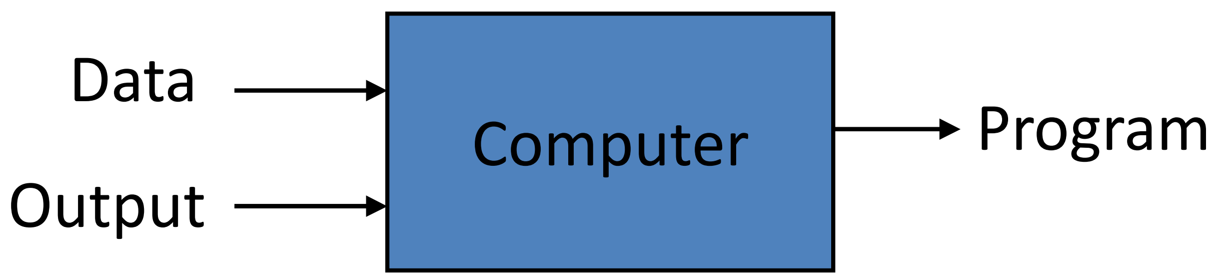
Machine Learning Based Object Detection

Traditional Programming v.s. Machine Learning

- Traditional Programming



- Machine Learning



Why Machine Learning is Hard?

Is this a tree?

A brown trunk moving upwards
and branching with leaves…?

Types of Learning

- **Supervised (inductive) learning**
  - Given: training data + desired outputs (labels)
- **Unsupervised learning**
  - Given: training data (without desired outputs)
- **Semi-supervised learning**
  - Given: training data + a few desired outputs
- **Reinforcement learning**
  - Rewards from sequence of actions

Supervised Learning: Classification

- Given $(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)$
- Learn a function *f(x)* to predict *y* given *x*
  - *y* is categorical == classification

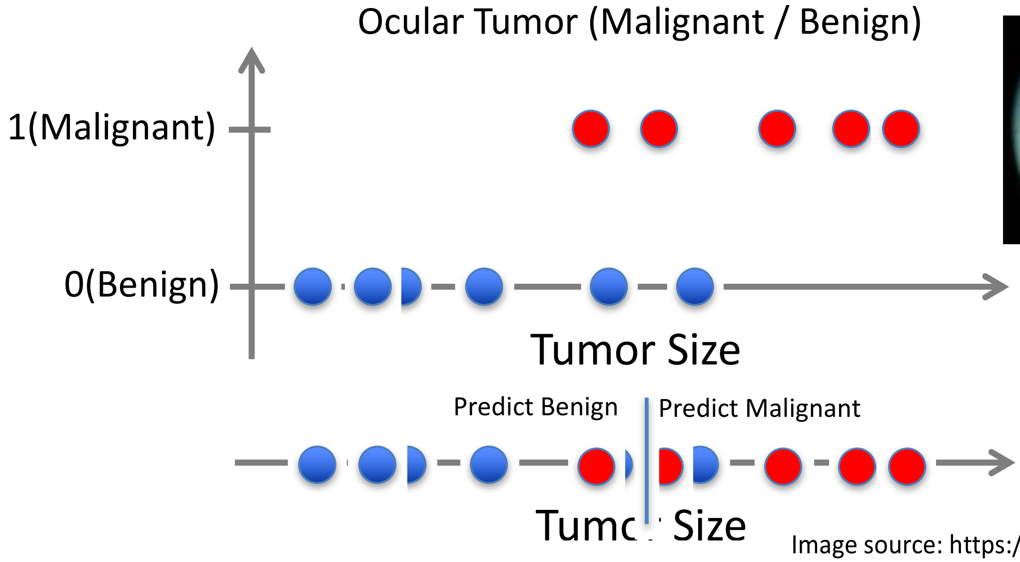
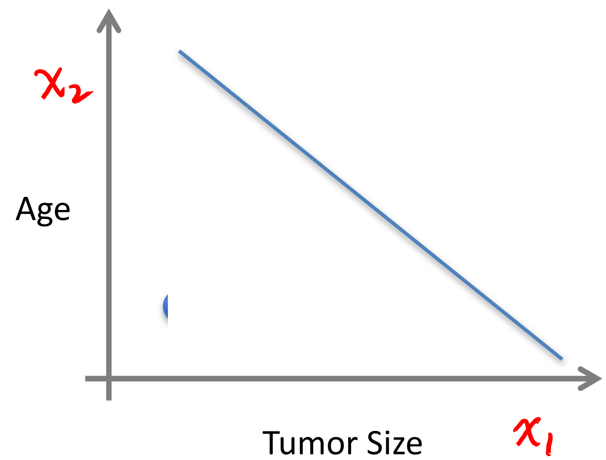


Image source: https://eyecancer.com/uncategorized/choroidal-metastasis-test/

Supervised Learning: Classification

- *x* can be multi-dimensional
  - each dimension corresponds to an attribute:
    - clump thickness
    - color
    - distance from optic nerve
    - …



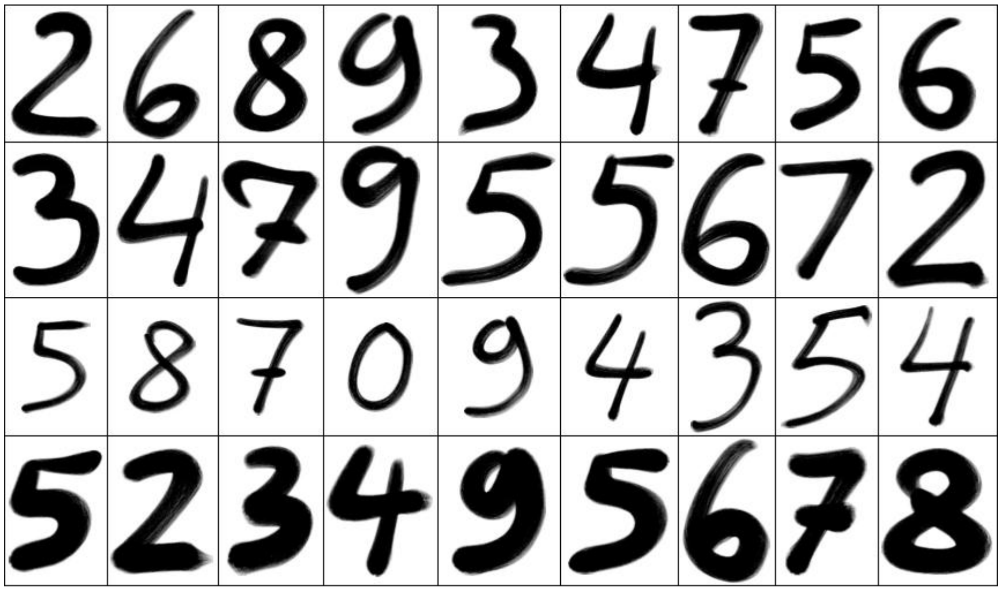
$\vec{x}_1 = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$

$y_1 = 0$ or 1

- Cell type is the most telling feature, but it's risky to do a biopsy of the eye
  - ML can help determine *when* a feature is needed

Supervised Learning: Multiclass Classification

- Multiclass classification problems
  - Written digits → 0, 1, …, 9
  - Pictures → apple, orange, strawberry
  - Emails → spam, primary, social, promotion

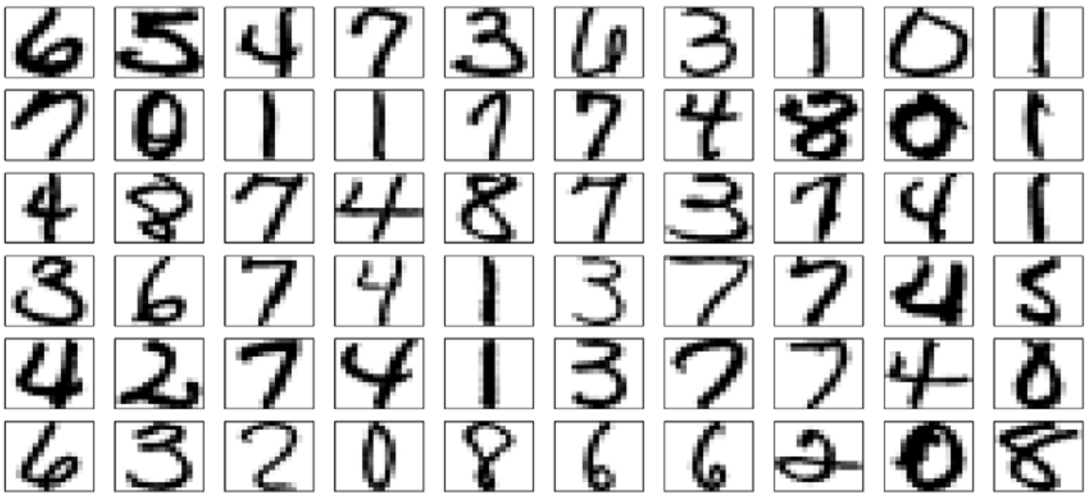


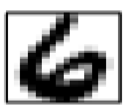
Example Data from ImageNet

Machine Learning Example

- Digit recognition

Each digit is a 16×16 image.





[-1 -1 -1 -1 -1 -1 -1 -0.63 0.86 -0.17 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -0.99 0.3 1 0.31 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -0.41 1 0.99 -0.57 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -0.68 0.83 1 0.56 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -0.94 0.54 1 0.78 -0.72 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 0.1 1 0.92 -0.44 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -0.26 0.95 1 -0.16 -1 -1 -1 -0.99 -0.71 -0.83 -1 -1 -1 -1 -1 -0.8 0.91 1 0.3 -0.96 -1 -1 -0.55 0.49 1 0.88 0.09 -1 -1 -1 -1 0.28 1 0.88 -0.8 -1 -0.9 0.14 0.97 1 1 1 0.99 -0.74 -1 -1 -0.95 0.84 1 0.32 -1 -1 0.35 1 0.65 -0.10 -0.18 1 0.98 -0.72 -1 -1 -0.63 1 1 0.07 -0.92 0.11 0.96 0.30 -0.88 -1 -0.07 1 0.64 -0.99 -1 -1 -0.67 1 1 0.75 0.34 1 0.70 -0.94 -1 -1 0.54 1 0.02 -1 -1 -1 -0.90 0.79 1 1 1 1 0.53 0.18 0.81 0.83 0.97 0.86 -0.63 -1 -1 -1 -1 -0.45 0.82 1 1 1 1 1 1 1 1 0.13 -1 -1 -1 -1 -1 -1 -0.48 0.81 1 1 1 1 1 1 0.21 -0.94 -1 -1 -1 -1 -1 -1 -1 -0.97 -0.42 0.30 0.82 1 0.48 -0.47 -0.99 -1 -1 -1 -1]

$$\mathbf{x} = (1, x_1, \cdots, x_{256}) \longleftarrow \text{input}$$

$y = 6$

The Key Players

- Pictures
  - input $\mathbf{x} \in R^d = \boldsymbol{X}$
- Classes: cat, dog, desk, etc.…
  - output y $\in \{1, 2, \ldots\} = \boldsymbol{Y}$
- True relationship between x and y
  - target function f: $\boldsymbol{X} \rightarrow \boldsymbol{Y}$
- Data
  - data set $\boldsymbol{D} = (\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$
  - $y_n = f(\mathbf{x}_n)$

We learn the function f from the data $\boldsymbol{D}$

$f(x) \rightarrow y$ ✓

learn

- $\boldsymbol{X}$, $\boldsymbol{Y}$, and $\boldsymbol{D}$ are given by the learning problem; the target f is fixed but unknown

Learning

- Start with a set of candidate hypotheses $\boldsymbol{H}$ which you think are likely to represent f

$$\boldsymbol{H} = \{h_1, h_2, \ldots,\}$$

  $\boldsymbol{H}$ is called the hypothesis set or model

- Select a hypothesis h from $\boldsymbol{H}$. The way we do this is called a *learning algorithm*
- Use h for new input. We hope $h \approx f$
- Again, $\boldsymbol{X}$, $\boldsymbol{Y}$, and $\boldsymbol{D}$ are given by the learning problem; the target f is fixed but unknown

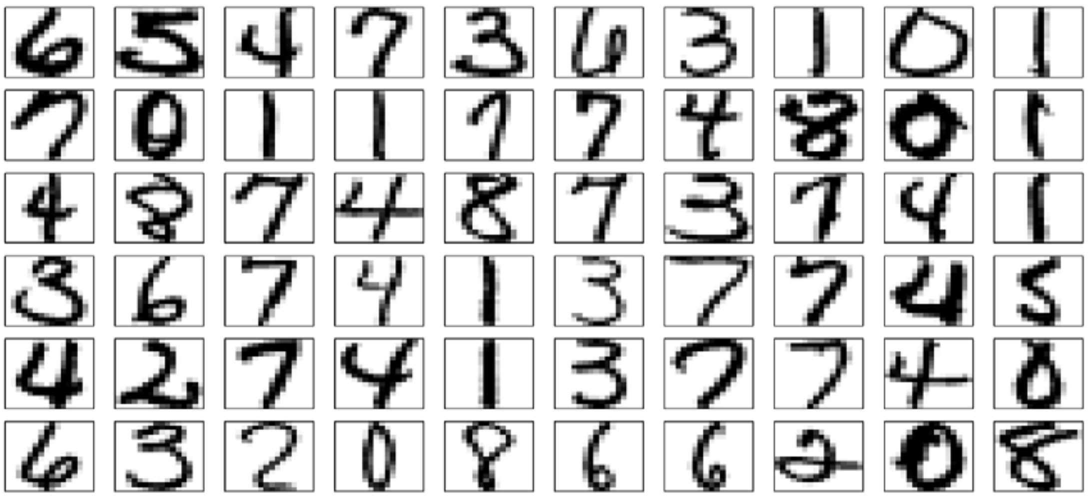
We choose $\boldsymbol{H}$ and the learning algorithm

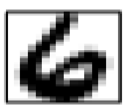
This is a very general setup (e.g. choose H to be all possible hypotheses)

Revisit: Digit Recognition Problem

- Digit recognition

Each digit is a 16 × 16 image.





[-1 -1 -1 -1 -1 -1 -1 -0.63 0.86 -0.17 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -0.99 0.3 1 0.31 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -0.41 1 0.99 -0.57 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -0.68 0.83 1 0.56 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -0.94 0.54 1 0.78 -0.72 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 0.1 1 0.92 -0.44 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -0.26 0.95 1 -0.16 -1 -1 -1 -0.99 -0.71 -0.83 -1 -1 -1 -1 -1 -0.8 0.91 1 0.3 -0.96 -1 -1 -0.55 0.49 1 0.88 0.09 -1 -1 -1 -1 0.28 1 0.88 -0.8 -1 -0.9 0.14 0.97 1 1 1 0.99 -0.74 -1 -1 -0.95 0.84 1 0.32 -1 -1 0.35 1 0.65 -0.10 -0.18 1 0.98 -0.72 -1 -1 -0.63 1 1 0.07 -0.92 0.11 0.96 0.30 -0.88 -1 -0.07 1 0.64 -0.99 -1 -1 -0.67 1 1 0.75 0.34 1 0.70 -0.94 -1 -1 0.54 1 0.02 -1 -1 -1 -0.90 0.79 1 1 1 1 0.53 0.18 0.81 0.83 0.97 0.86 -0.63 -1 -1 -1 -1 -0.45 0.82 1 1 1 1 1 1 1 1 0.13 -1 -1 -1 -1 -1 -1 -0.48 0.81 1 1 1 1 1 1 0.21 -0.94 -1 -1 -1 -1 -1 -1 -1 -0.97 -0.42 0.30 0.82 1 0.48 -0.47 -0.99 -1 -1 -1 -1]

$y = 6$

$\mathbf{x} = (1, x_1, \cdots, x_{256}) \longleftarrow$ input

$\mathbf{w} = (w_0, w_1, \cdots, w_{256}) \longleftarrow$ linear model

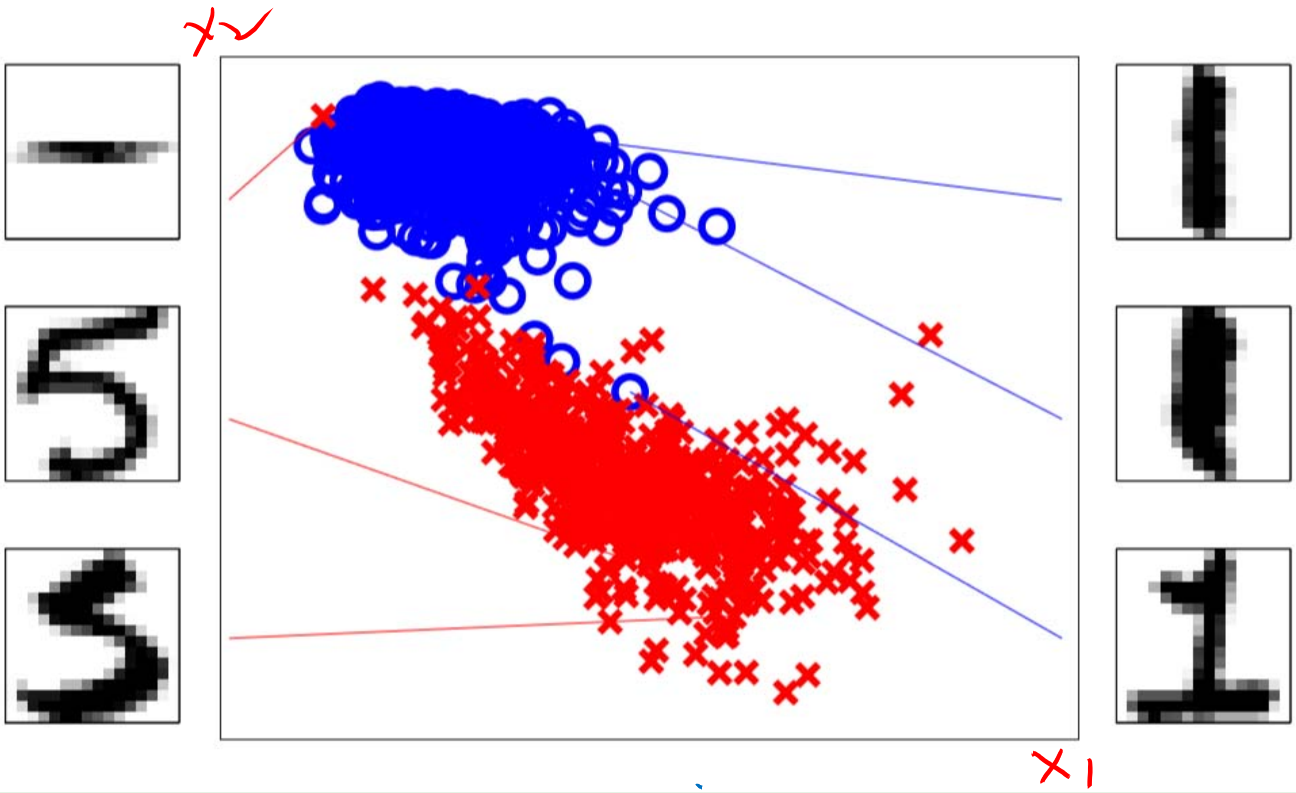
$(\vec{x} \cdot \vec{w})$

Hypothesis: $h = g(\boldsymbol{w}^T \boldsymbol{x})$

e.g.,: $h = \text{sign}(\boldsymbol{w}^T \boldsymbol{x})$

Machine Learning Example

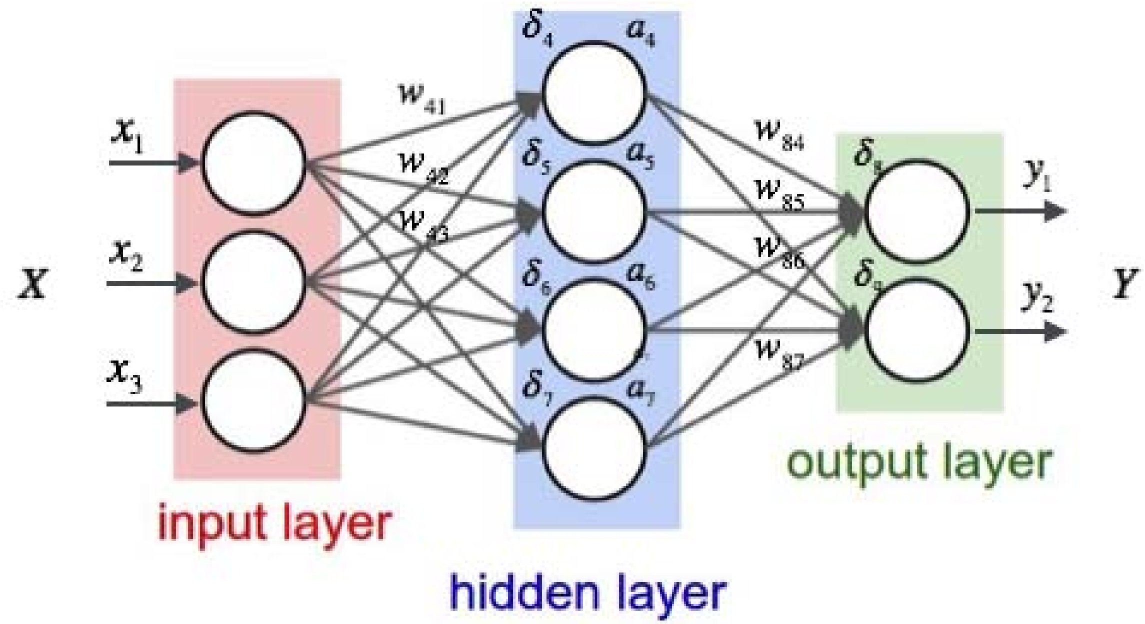
- Feature: an important property of the input that you think is useful for classification, e.g.,

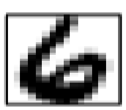


$\mathbf{x} = (x_0, x_1, x_2)$ x_1: intensity x_2: symmetry

Neural Network

- How does a neural network work?

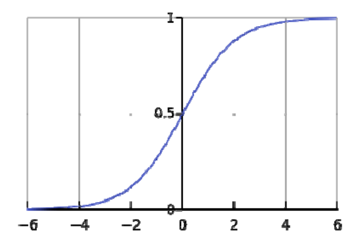




Input
$\boldsymbol{x} = (x_1, x_2 \dots,)^T$

Activation function
$a_j = g(\boldsymbol{w}_j^T \boldsymbol{x})$

Example:
$$g(t) = \frac{1}{1 + e^{-t}}$$



Goal: learn $\boldsymbol{w}$!

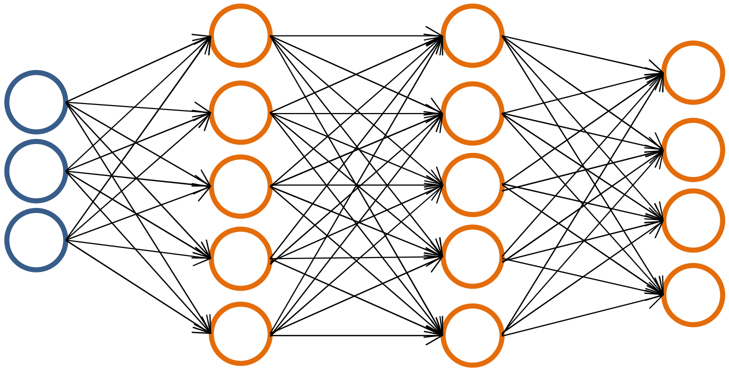
Neural Network for Object Classification

Pedestrian

Car

Motorcycle

Truck



$$h_\Theta(\mathbf{x}) \in \mathbb{R}^K$$

- We want

$$h_\Theta(\mathbf{x}) \approx \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

when pedestrian

$$h_\Theta(\mathbf{x}) \approx \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

when car

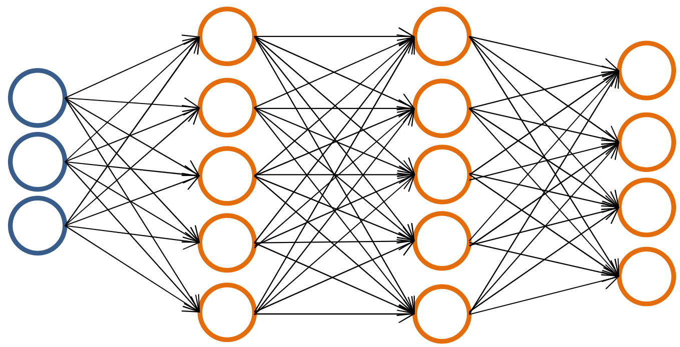
$$h_\Theta(\mathbf{x}) \approx \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

when motorcycle

$$h_\Theta(\mathbf{x}) \approx \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

when truck

Neural Network for Object Classification



$$h_\Theta(\mathbf{x}) \in \mathbb{R}^K$$

- We want

$h_\Theta(\mathbf{x}) \approx \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$ when pedestrian

$h_\Theta(\mathbf{x}) \approx \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$ when car

$h_\Theta(\mathbf{x}) \approx \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$ when motorcycle

$h_\Theta(\mathbf{x}) \approx \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$ when truck

- Given $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\}$
- Must convert labels to 1-of-K representation
  - e.g., $y_i = [0 \quad 0 \quad 1 \quad 0]^T$ when motorcycle, $y_i = [0 \quad 1 \quad 0 \quad 0]^T$ when car

- Thank You!