Introduction to Machine Learning & Deep Learning Concepts

July 2021

Zafri Baharuddin

Based on work by
Dickson Neoh
Justin Johnson

intel®

ITXOTIC

Deep Learning for Computer Vision

Building artificial systems that

process,

perceive, and

reason about visual data

Computer Vision is everywhere




Left to right:
Image by Intel is copyrighted
Image from Unsplash
Image from Unsplash
Image by Nasa is public




Left to right:
Image by Nasa is public
Image from Unsplash
Image from Unsplash
Image from Unsplash




Left to right:
Image from Unsplash
Image from Unsplash
Image from Unsplash
Image copyrighted by ExtremeTech

Deep Learning for Computer Vision

Building artificial systems that
learn from data and experience

Deep Learning for Computer Vision

Hierarchical learning algorithms
with many “layers”,
(very) loosely inspired by the brain

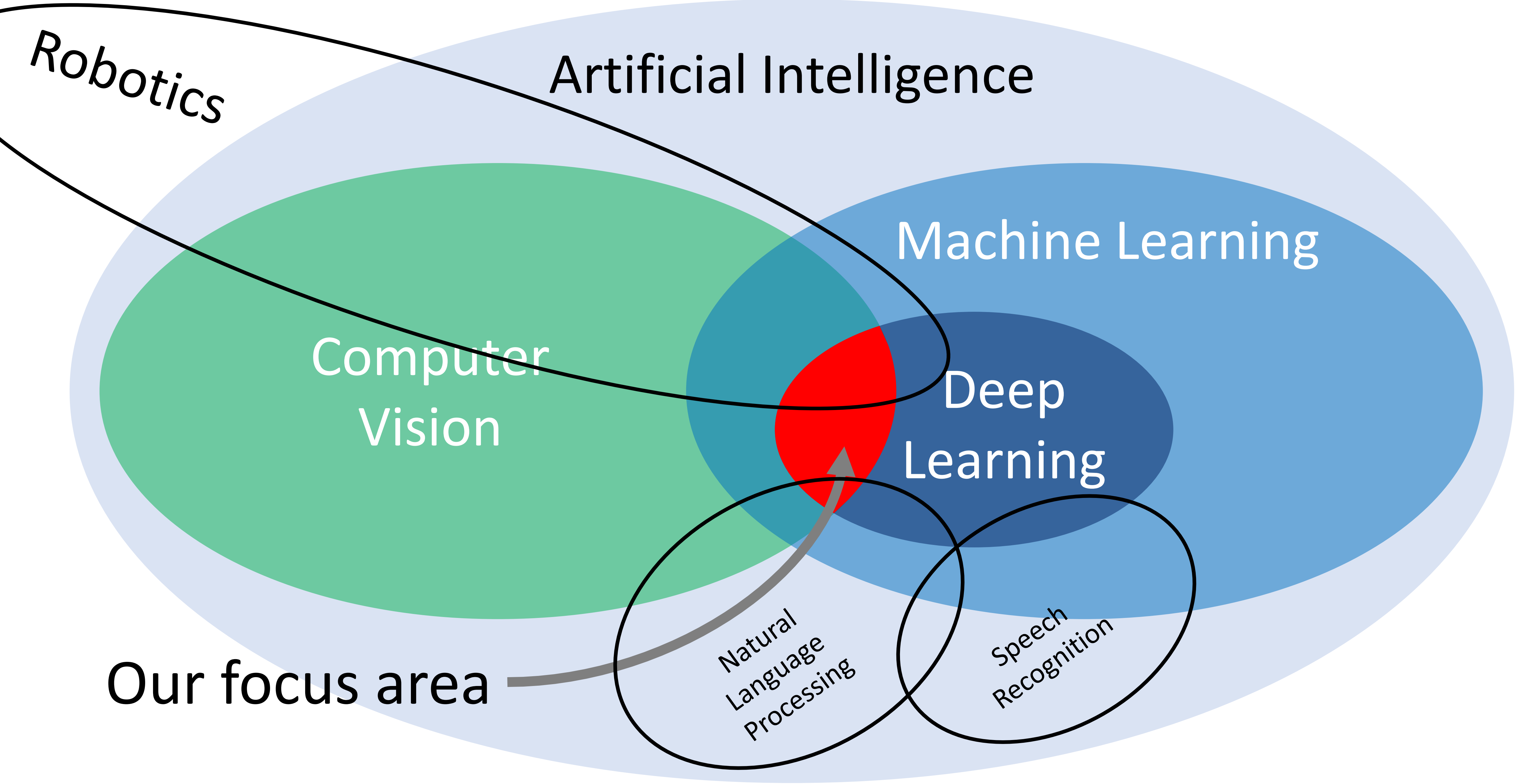

Robotics
Artificial Intelligence
Machine Learning
Computer Vision
Deep Learning
Natural Language Processing
Speech Recognition
Our focus area

Image Classification: Challenges

- Change view angle

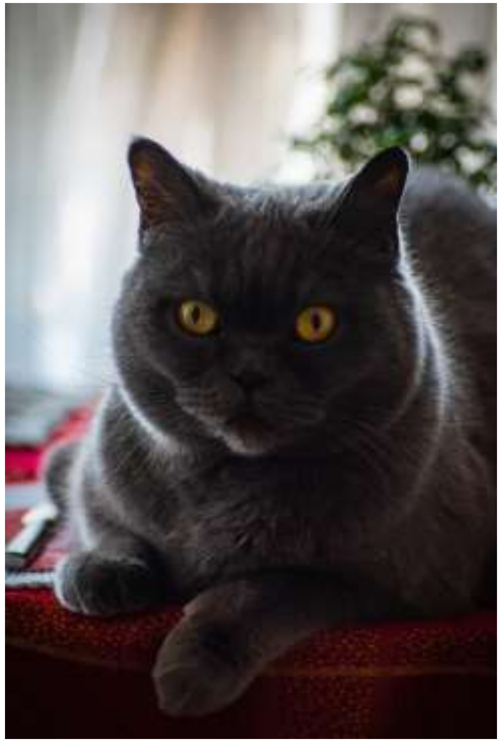
Image Classification: Challenges

- Variation within class

Image Classification: Challenges

- Sub-classes (different cat breeds)

Maine Coon



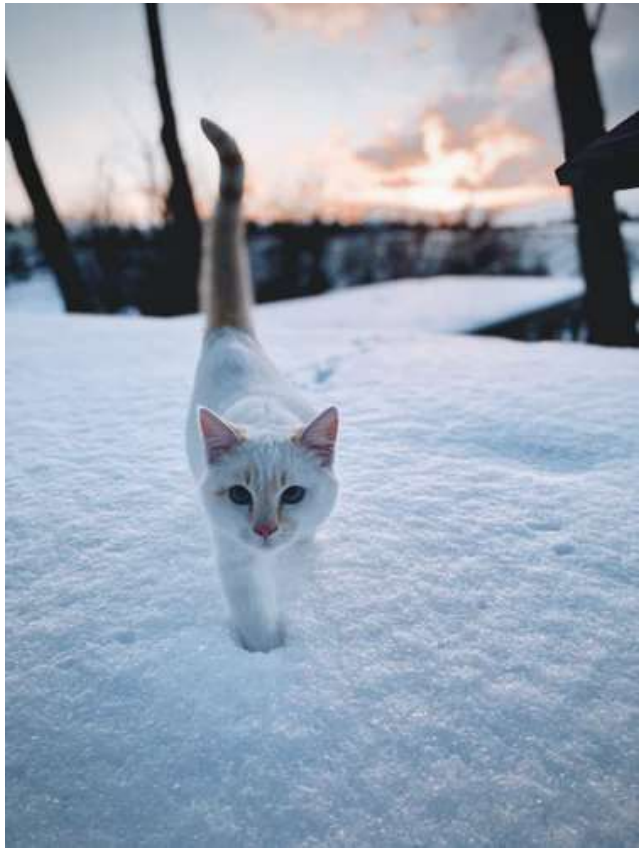
Bristish Shorthair

Grumpy Cat
(Mixed breed)
2012-2019

ITXOTIC intel

Image Classification: Challenges

- Background blending / camouflage



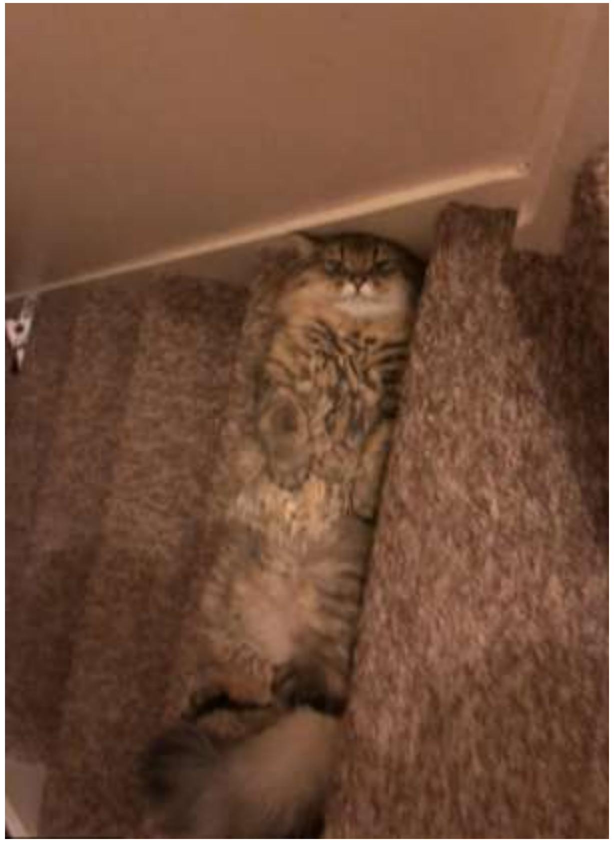


Image Classification: Challenges

- Lighting change



Image Classification: Challenges

- Deformation

Image Classification: Challenges

- Occlusion

Image Classification: Building Blocks for other tasks.

- Object detection

Output: bounding boxes and class

Image Classification: Building Blocks for other tasks.

- **Object segmentation**

Output: segments and class

Image Classification: Building Blocks for other tasks.

- Image captioning

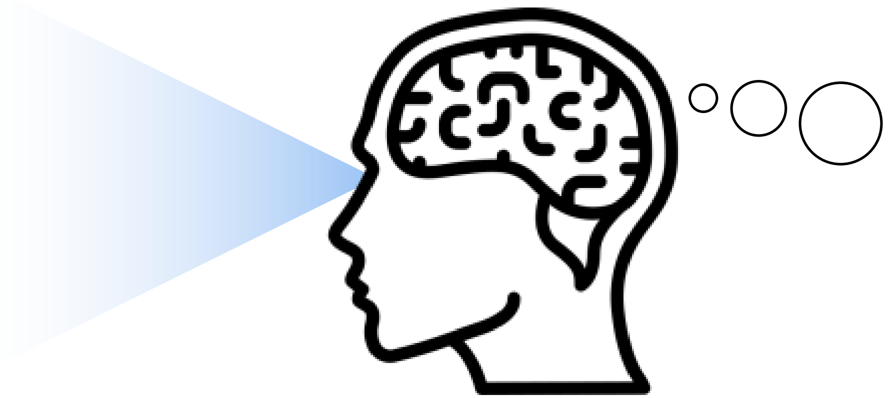
Output: description: There are two cars and a bike on the road

ITXOTIC intel

Motivation: Classification – Basic Human task

Input: image

Resolution:
800 x 600 x 3 (RGB)



Output:

cat
bird
deer
dog
truck

All images from https://unsplash.com/s/photos/cat

Motivation: Classification – Basic AI task

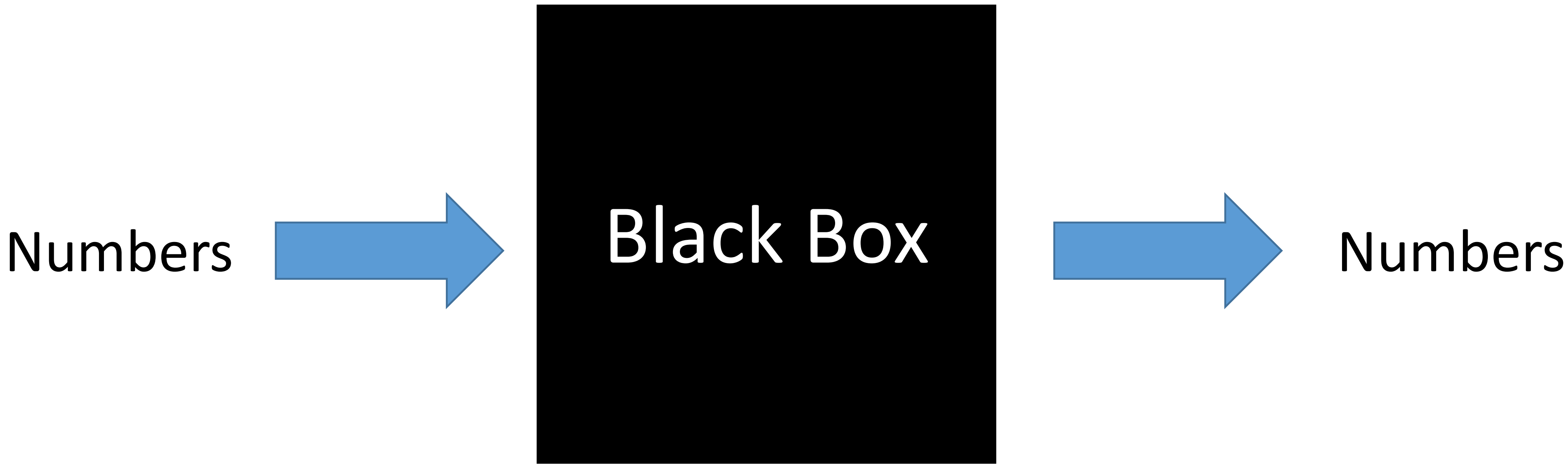
Input: image

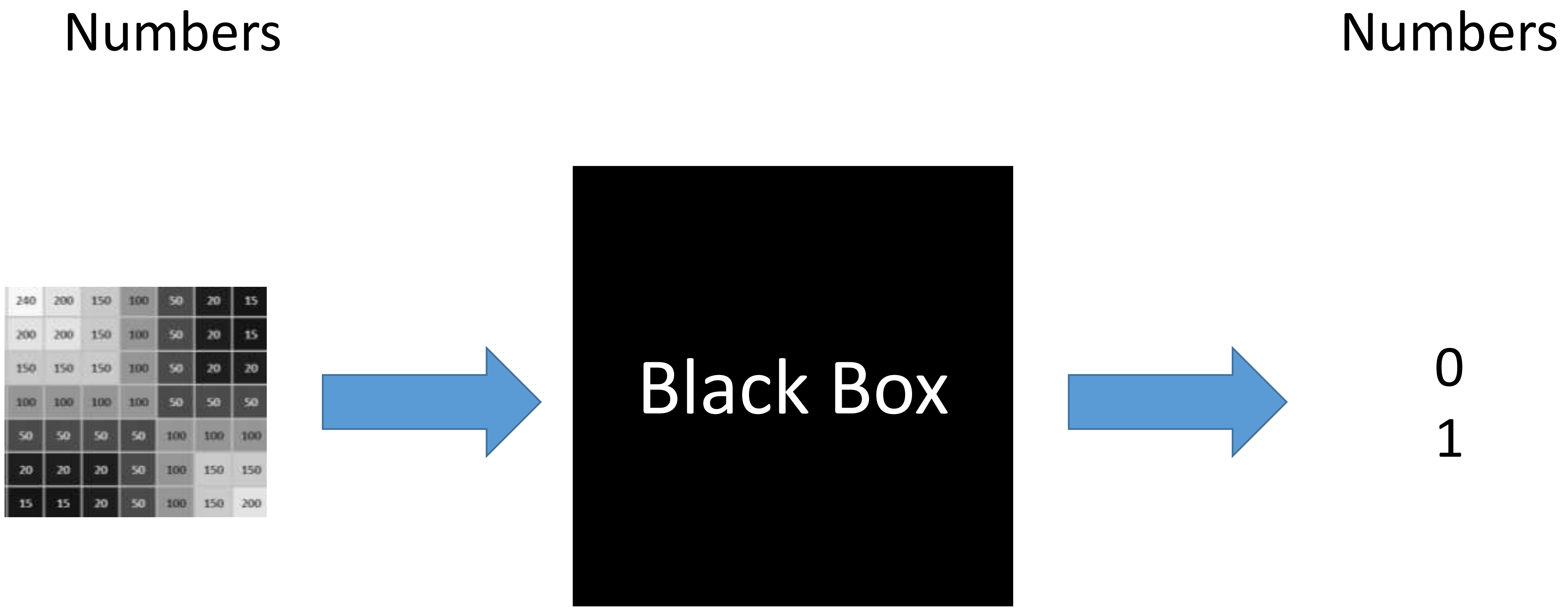
Resolution:
800 x 600 x 3 (RGB)

AI Magic Box

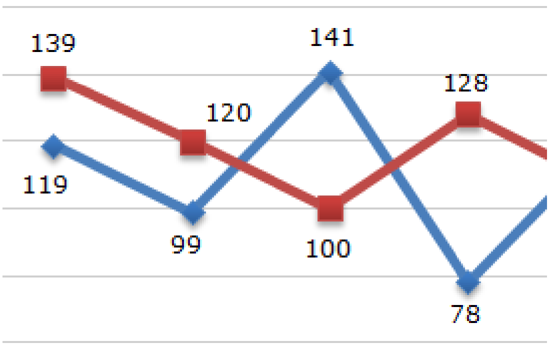
Output:	Probability:
cat	0.82
bird	0.02
deer	0.04
dog	0.10
truck	0.02

All images from https://unsplash.com/s/photos/cat


Numbers
Black Box
Numbers

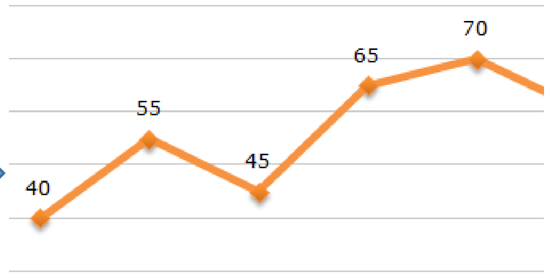

Numbers
Numbers
240 200 150 100 50 20 15
200 200 150 100 50 20 15
150 150 150 100 50 20 20
100 100 100 100 50 50 50
50 50 50 50 100 100 100
20 20 20 50 100 150 150
15 15 20 50 100 150 200
Black Box
0
1

Numbers



Black Box

Numbers



Numbers

Numbers

Deep
learning
for
humans

Black Box

인간을위
한 딥 러닝

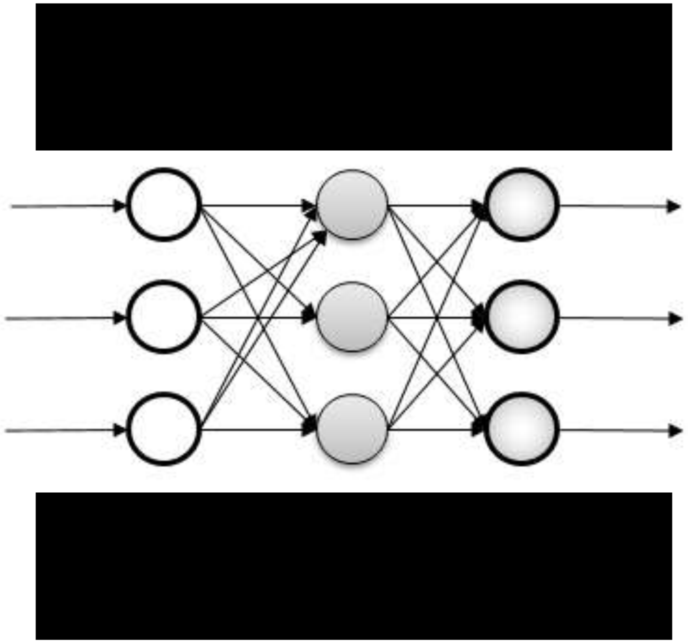
Numbers

Numbers

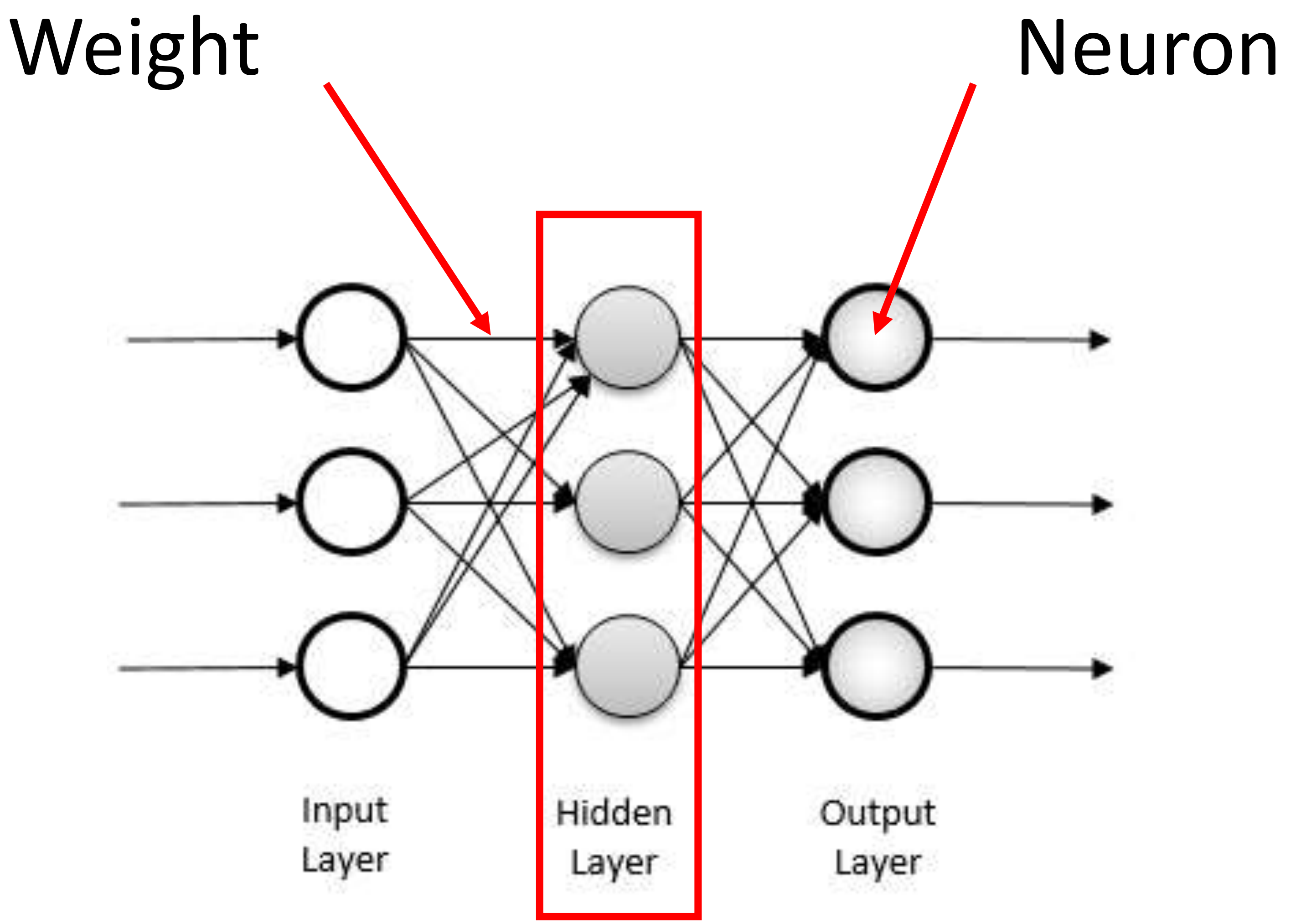
Black Box

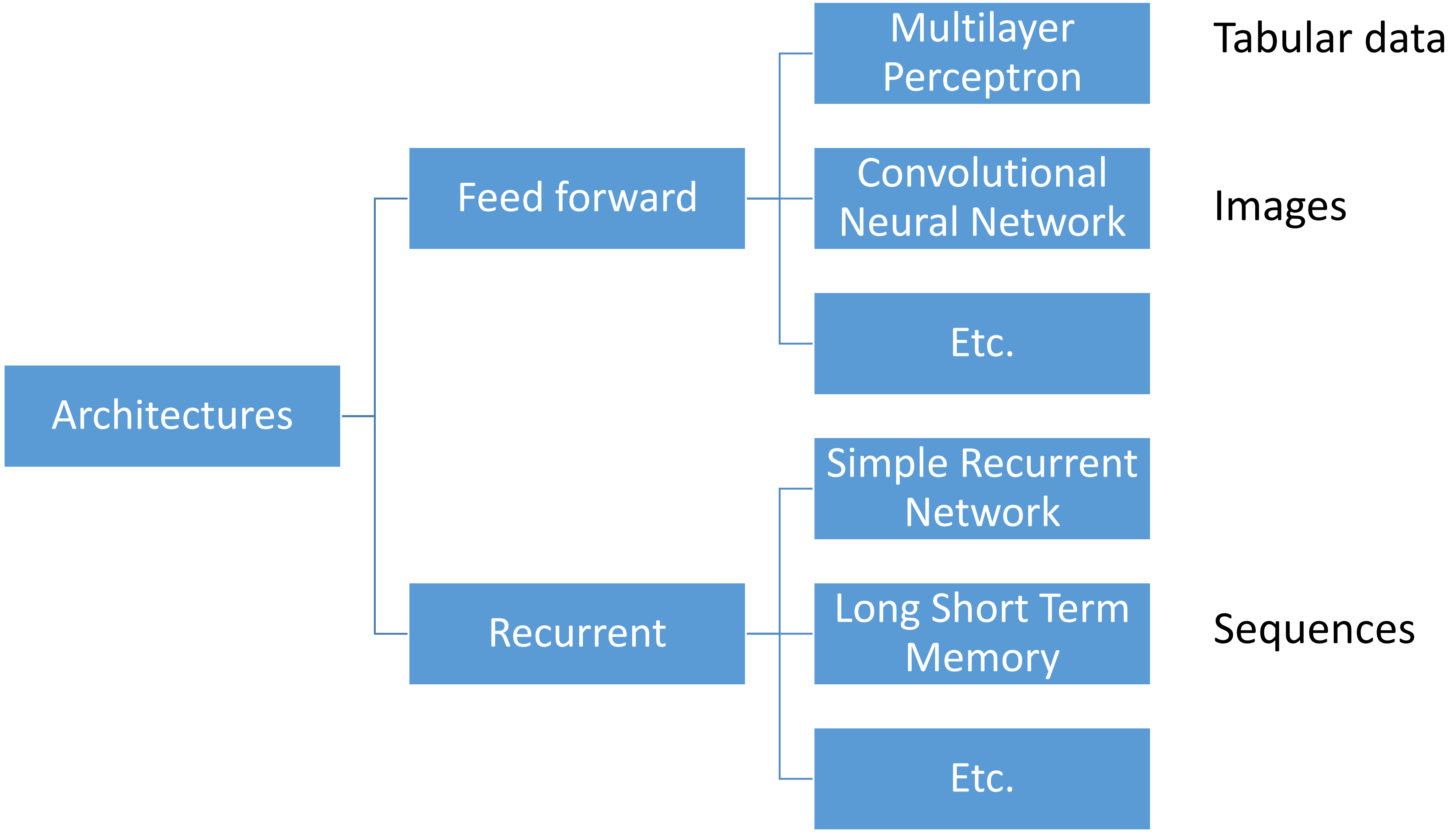
인간을위
한 딥 러닝

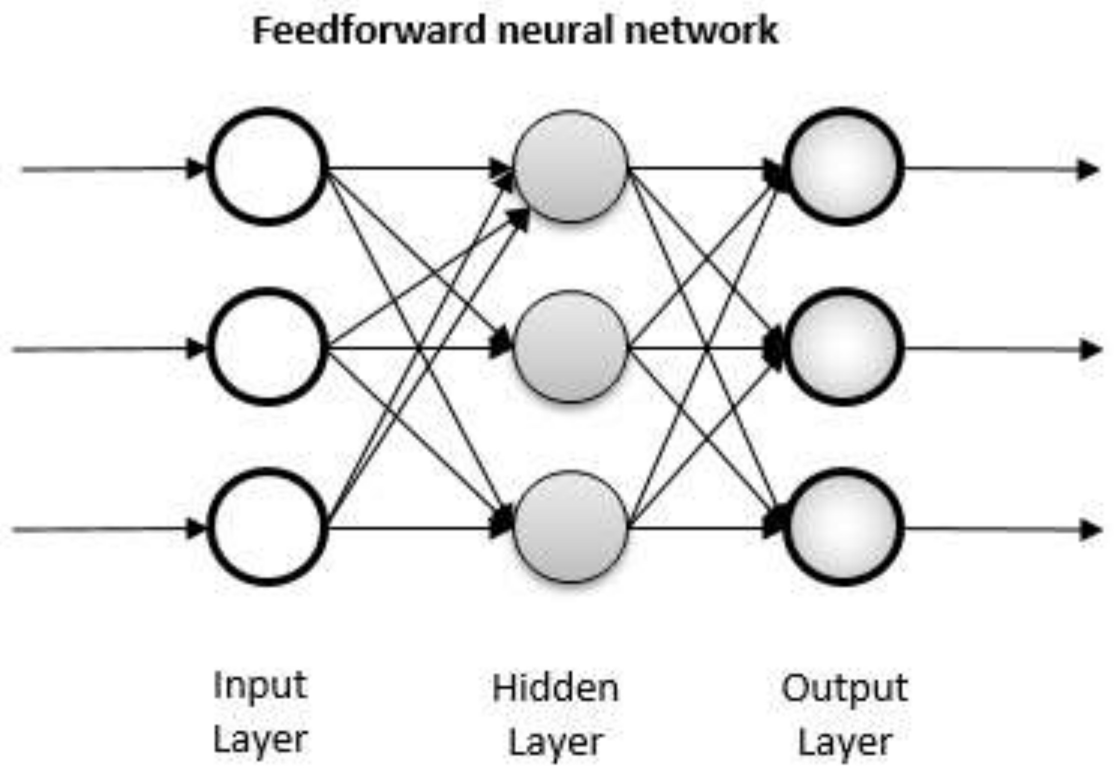
Numbers

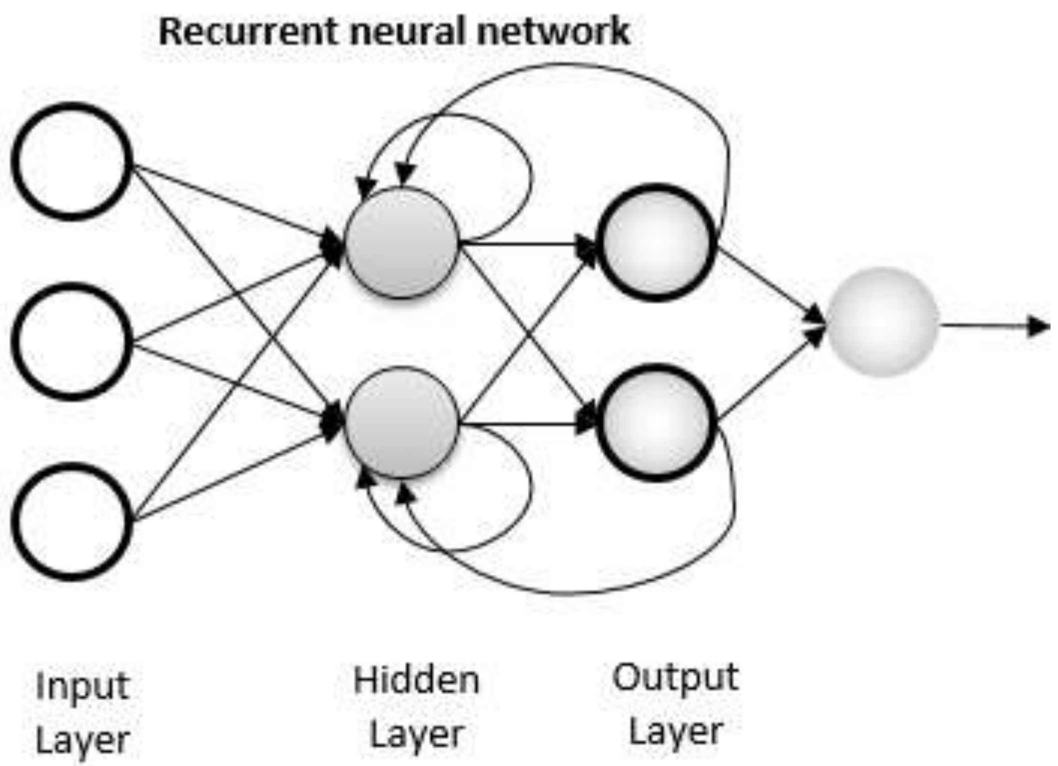


Numbers

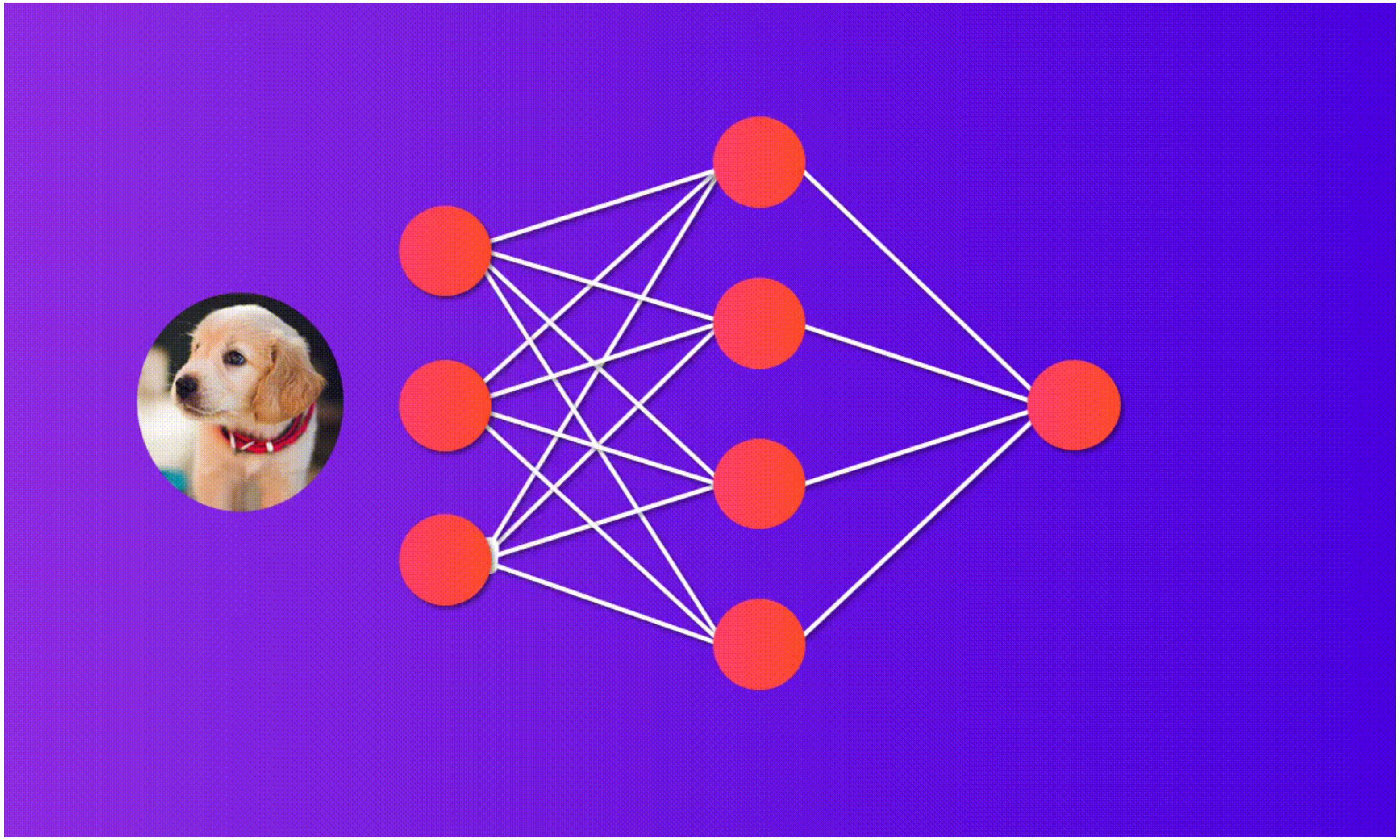

Weight
Neuron
Input
Layer
Hidden
Layer
Output
Layer


Architectures
Feed forward
Recurrent
Multilayer Perceptron
Convolutional Neural Network
Etc.
Simple Recurrent Network
Long Short Term Memory
Etc.
Tabular data
Images
Sequences

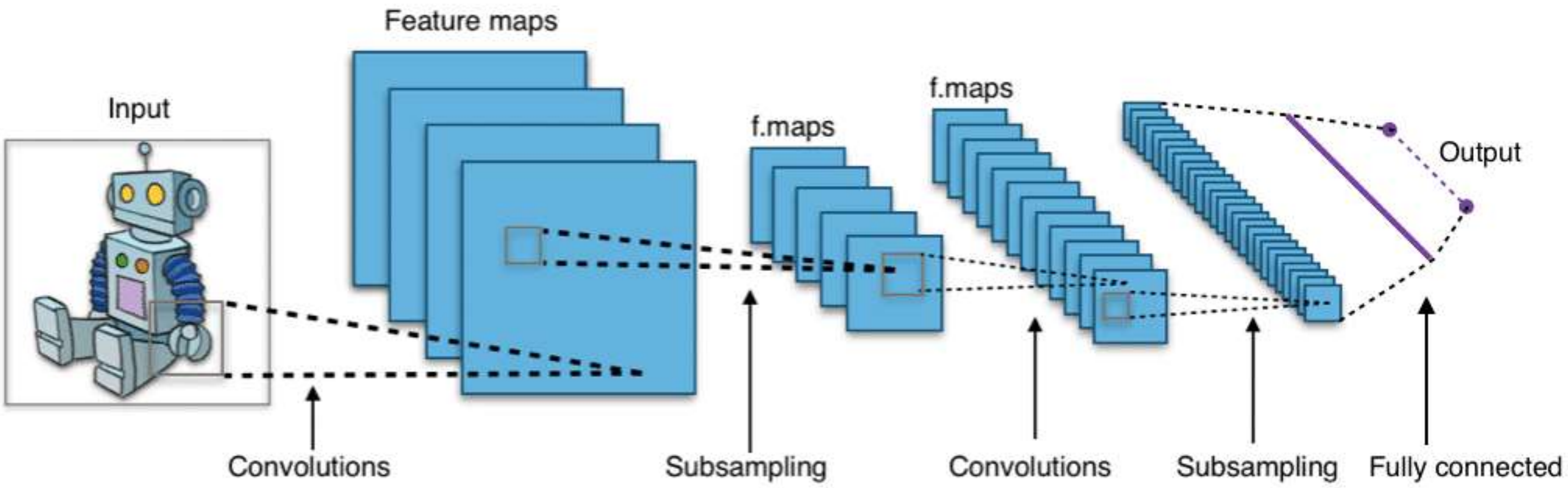

Feedforward neural network
Input Layer
Hidden Layer
Output Layer


Recurrent neural network
Input Layer
Hidden Layer
Output Layer

MLP

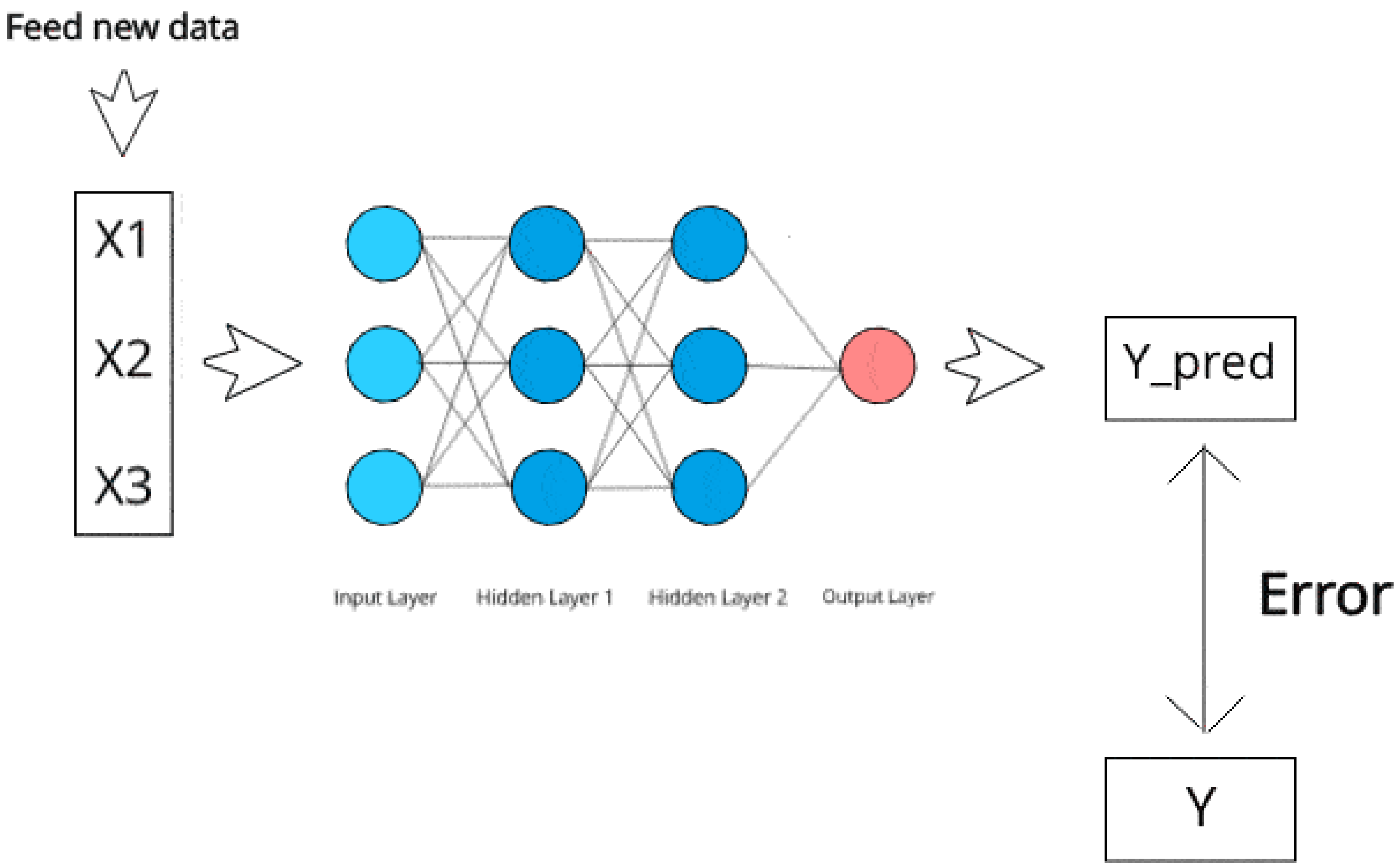


CNN

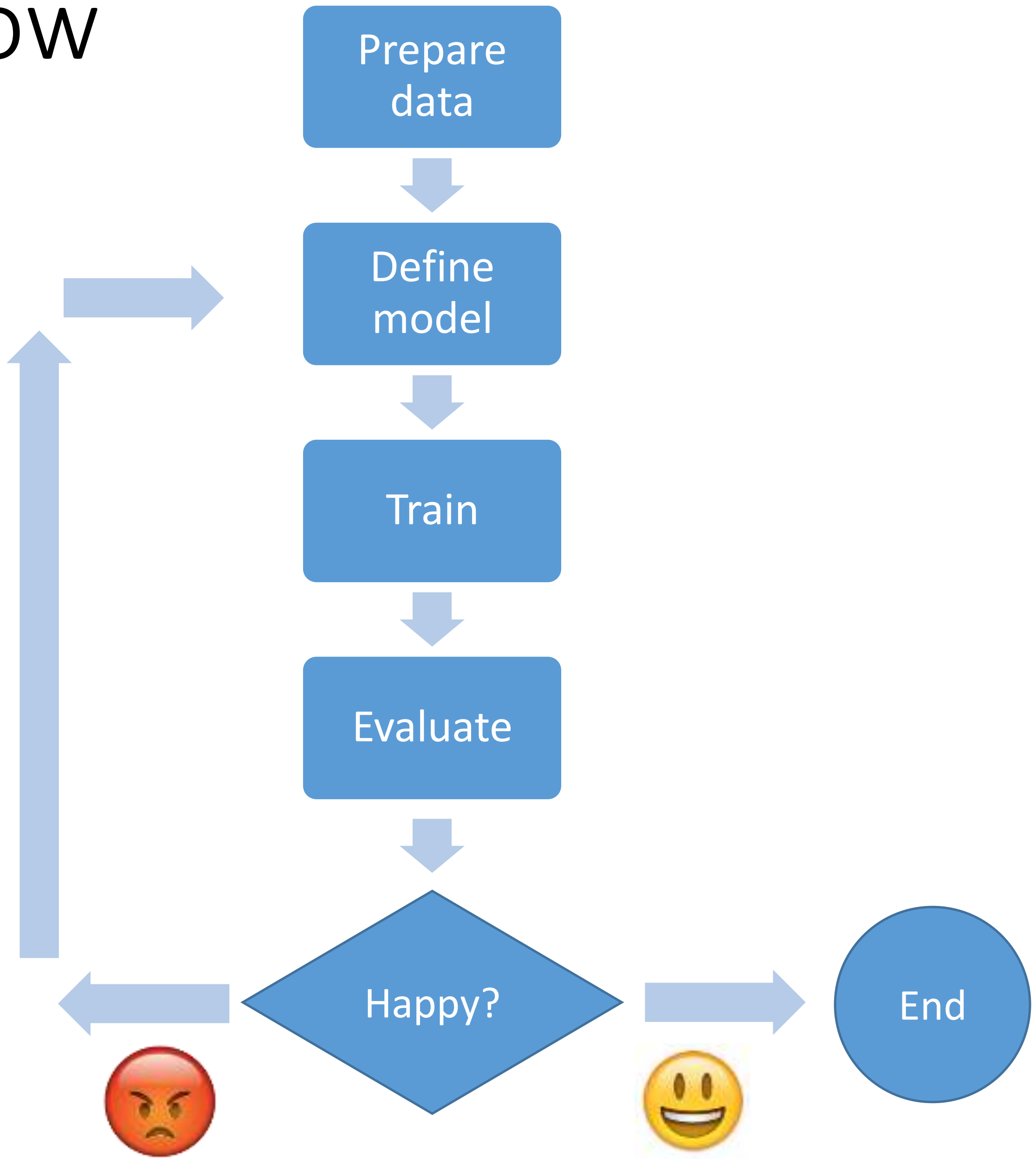


Feature maps
Input
f.maps
f.maps
Output
Convolutions
Subsampling
Convolutions
Subsampling
Fully connected

Learning with Backpropagation



Typical DL workflow



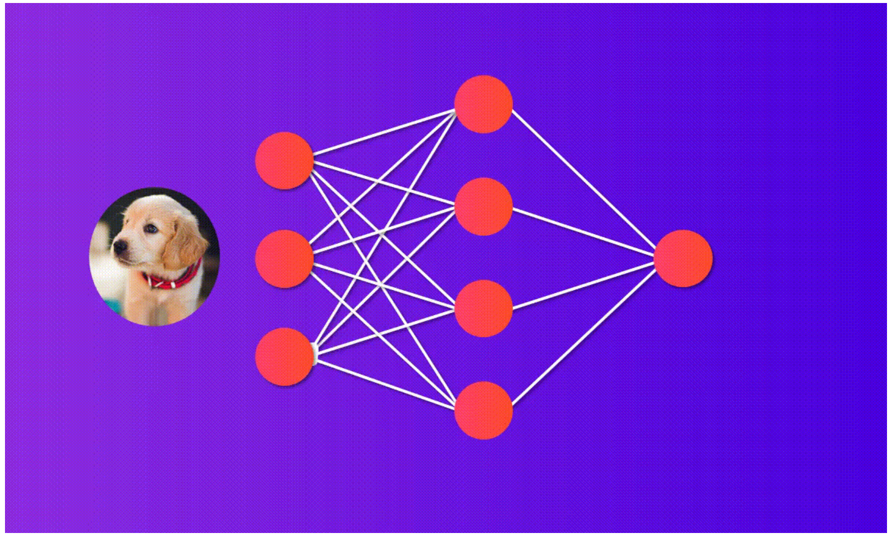
How

- Learn by doing.
- Minimal theory/math to get started with Tensorflow

What

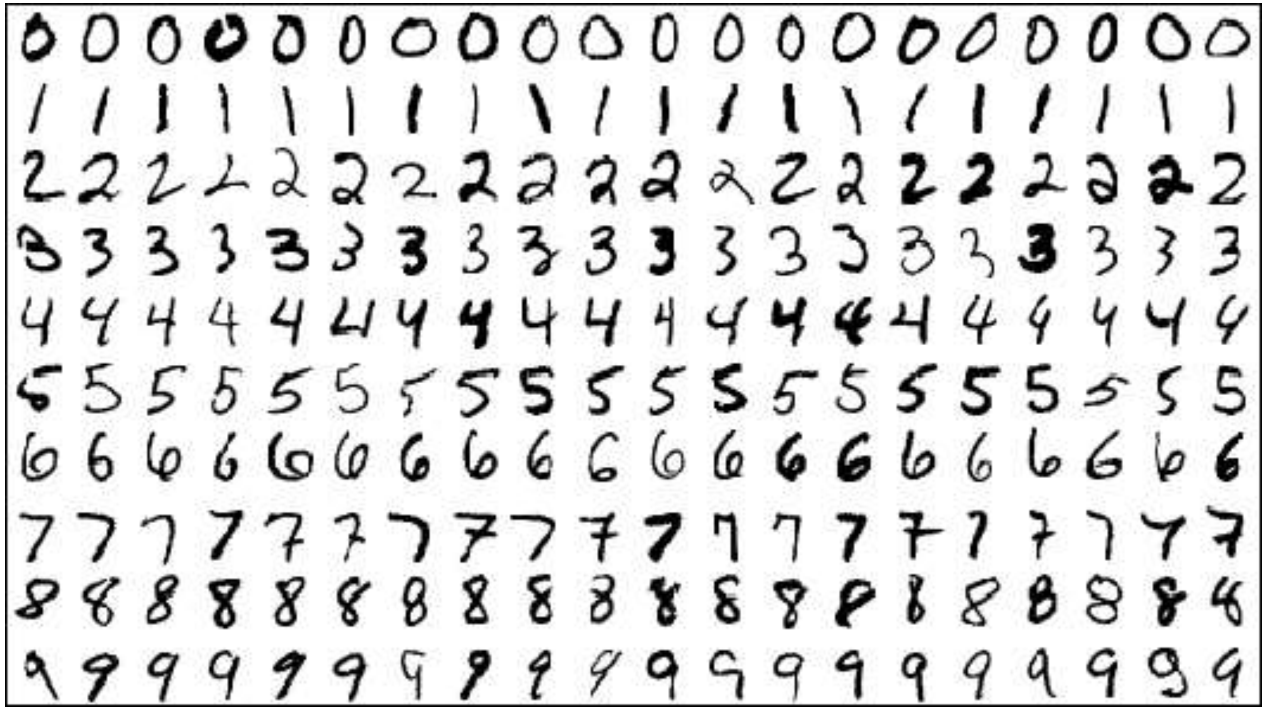
- Jupyter Notebook
- Google Colab (or go to colab.research.google.com)
- Interactive demo

Lab 1: MNIST with MLP

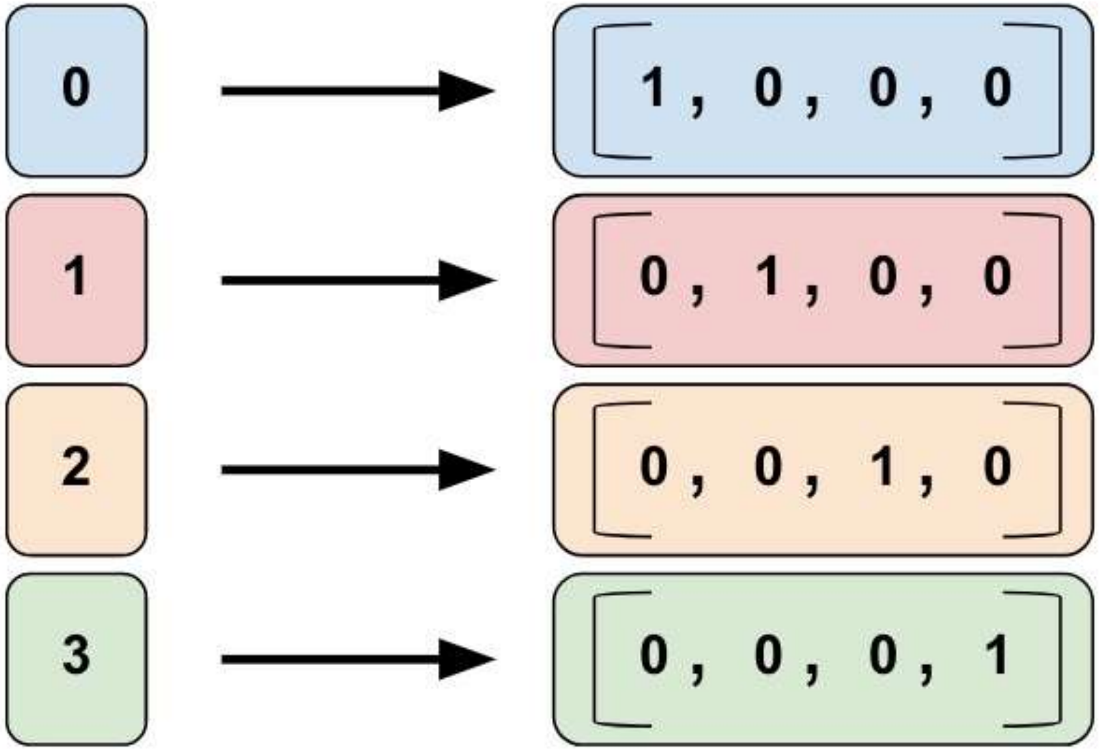


MNIST

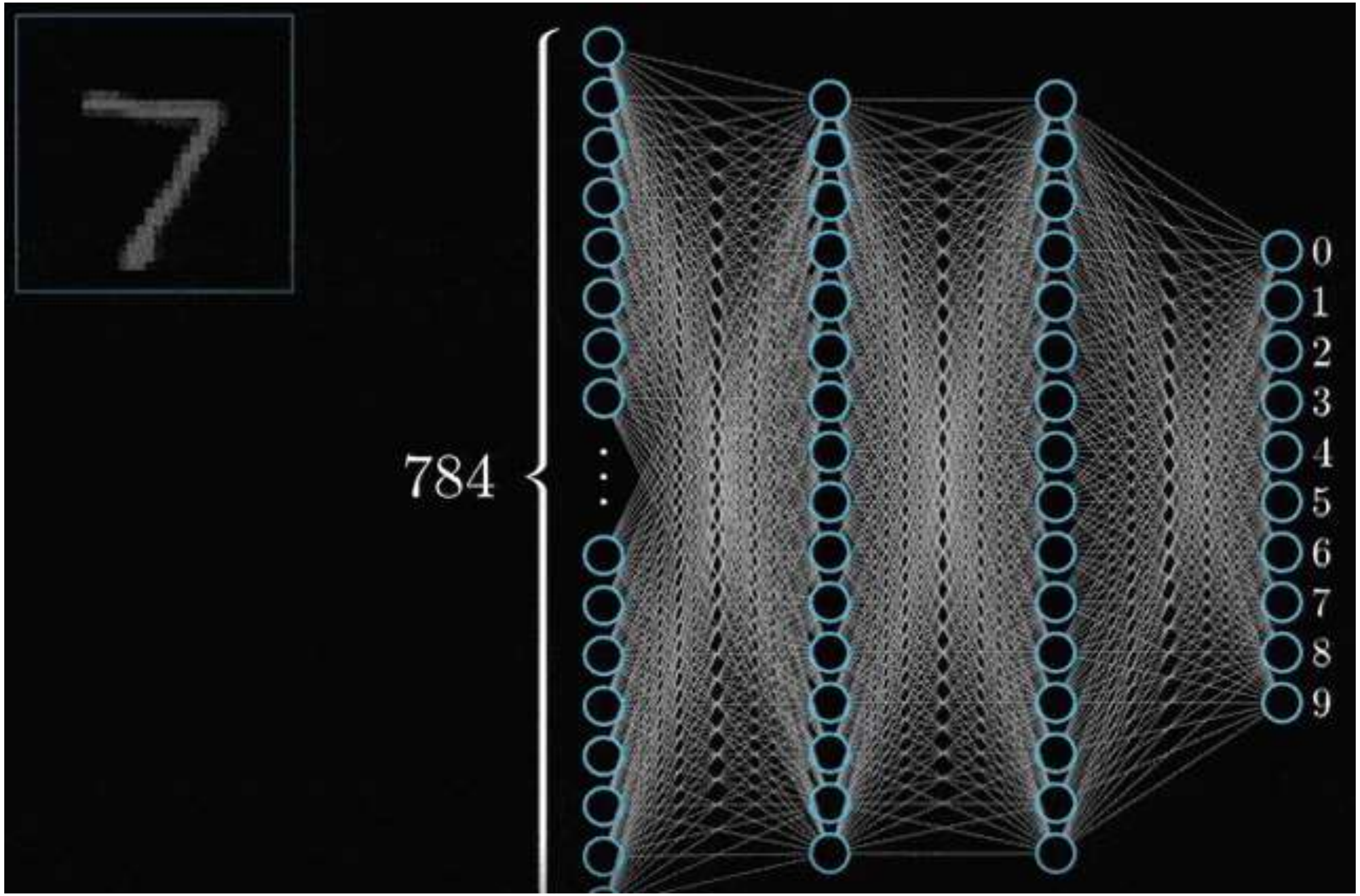
- Modified National Institute of Standards and Technology
- The Hello World of Deep Learning
- 10 classes (0-9)
- Image size: 28 x 28 pixels
- SOTA: 99.84%
- Human: 99.80%



One hot encoding



MLP



ITXOTIC intel

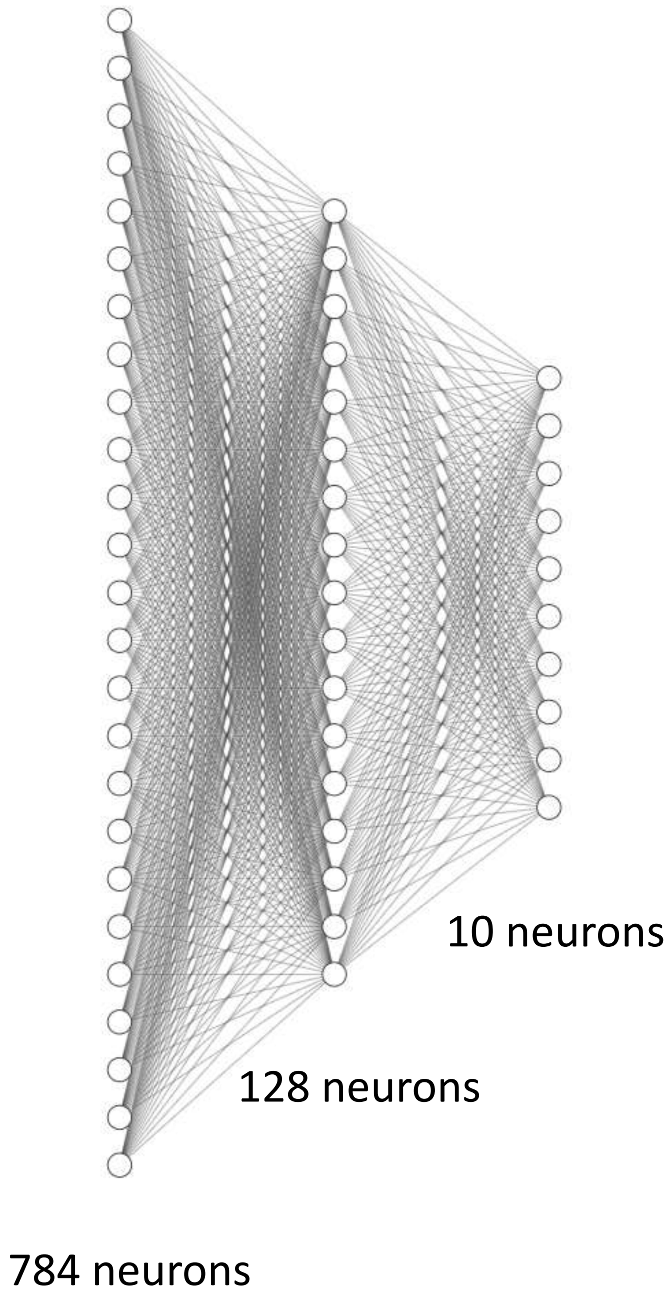
MLP: 97.0 ++ %

- 97+% accuracy with < 20 lines of codes
- Open MNIST with MLP

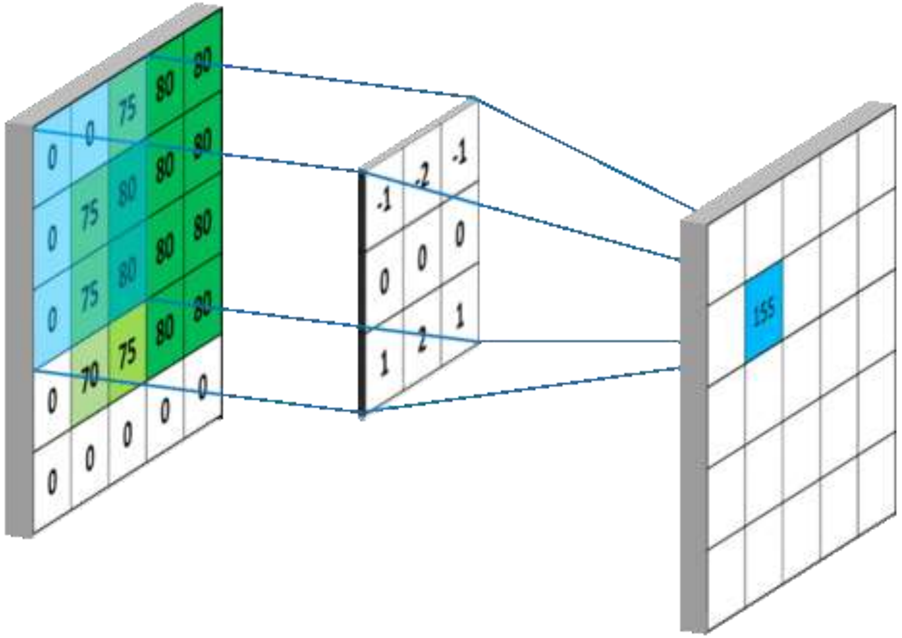
```
import numpy as np
from tensorflow import keras
num_classes = 10
input_shape = (28, 28, 1)
(x_train, y_train), (x_test, y_test) = keras.datasets.mnist.load_data()
x_train = x_train.astype("float32") / 255
x_test = x_test.astype("float32") / 255
x_train = np.expand_dims(x_train, -1)
x_test = np.expand_dims(x_test, -1)
y_train = keras.utils.to_categorical(y_train, num_classes)
y_test = keras.utils.to_categorical(y_test, num_classes)
model = keras.Sequential([
        keras.Input(shape=input_shape),
        keras.layers.Flatten(),
        keras.layers.Dense(128, activation='relu'),
        keras.layers.Dense(num_classes, activation="softmax"),
])
model.compile(loss="categorical_crossentropy", optimizer="adam", metrics=["accuracy"])
model.fit(x_train, y_train, batch_size=128, epochs=5, validation_split=0.1)
```

Hyperparameters

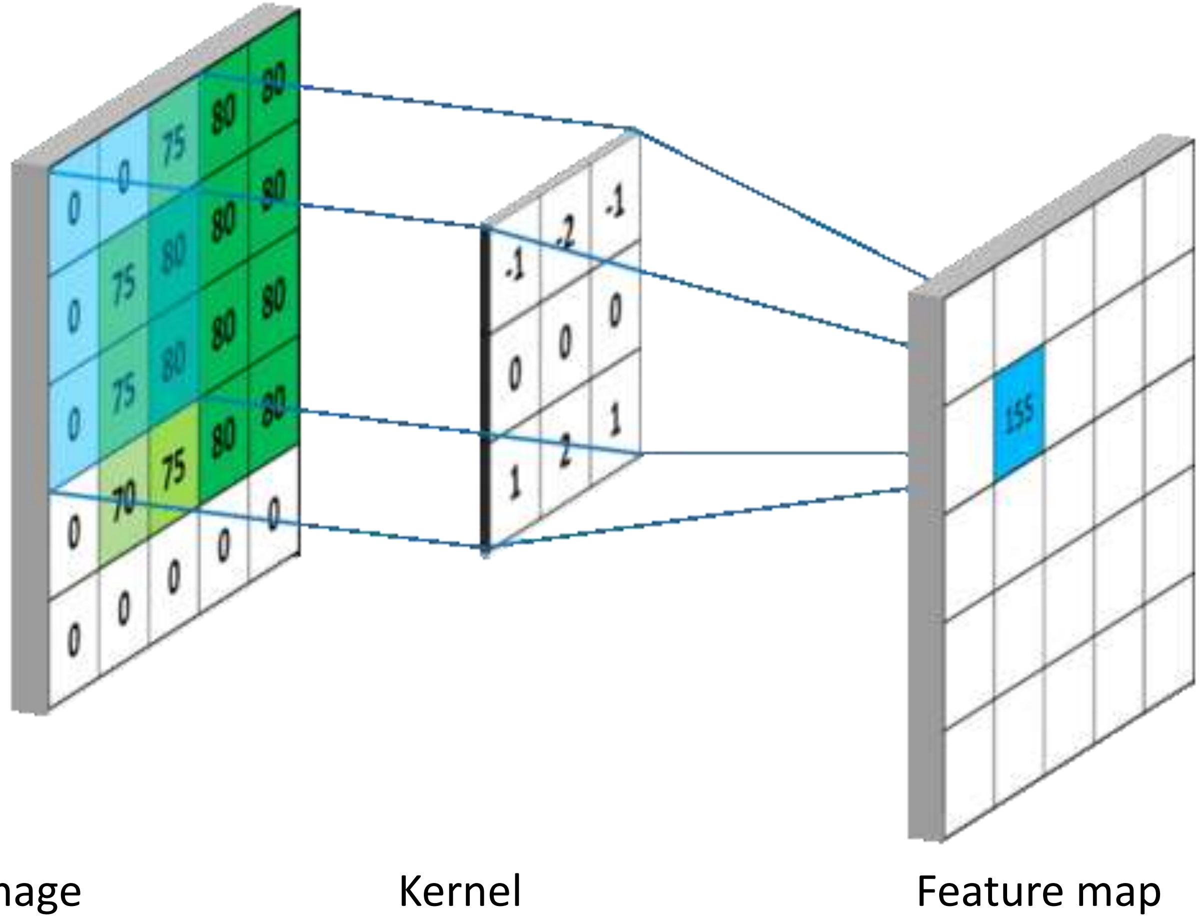
```
model = keras.Sequential(
    [
        keras.Input(shape=input_shape),
        layers.Flatten(),
        layers.Dense(128, activation='relu'),
        layers.Dense(num_classes, activation="softmax"),
    ]
)

model.summary()
```



Lab 2: MNIST with CNN



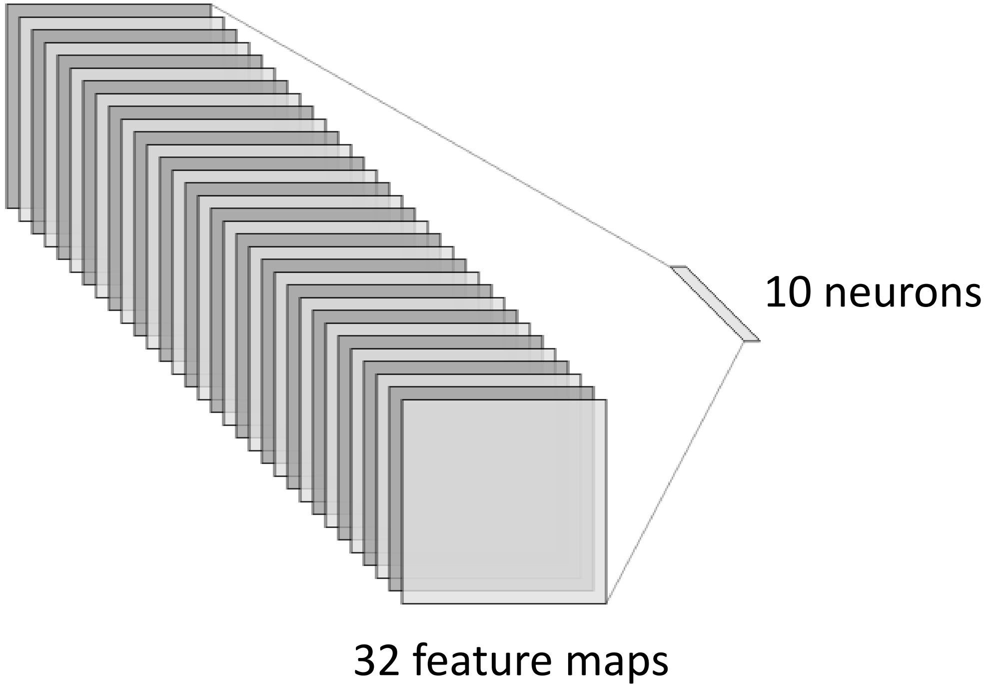
Convolution Operation



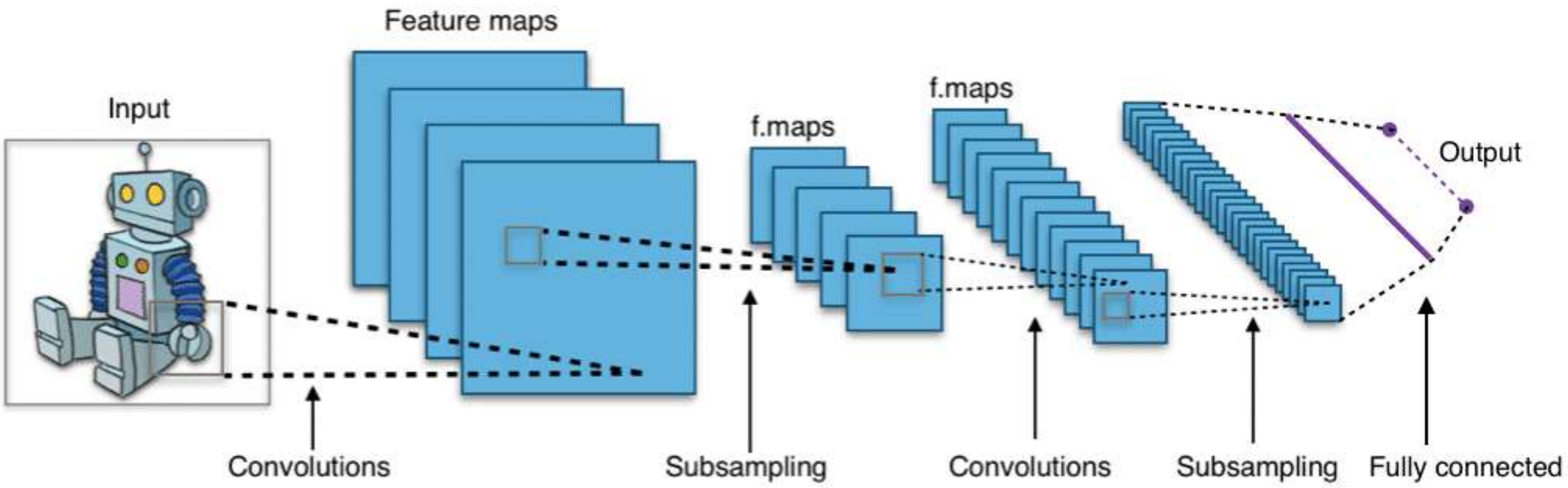
ITXOTIC intel

CNN: 98.0 ++ %

```
model = keras.Sequential(
    [
        keras.Input(shape=input_shape),
        layers.Conv2D(32, kernel_size=(3, 3), activation="relu"),
        layers.Flatten(),
        layers.Dense(num_classes, activation="softmax"),
    ]
)
```



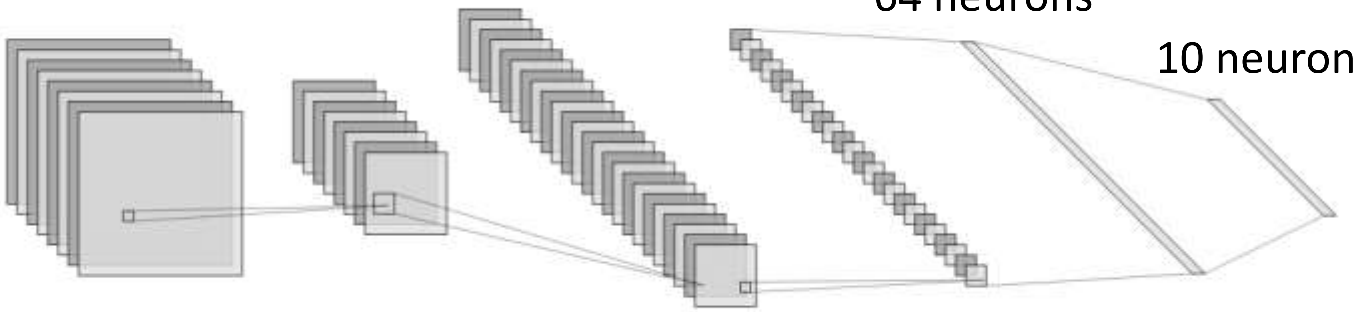
CNN


Feature maps
Input
f.maps
f.maps
Output
Convolutions
Subsampling
Convolutions
Subsampling
Fully connected

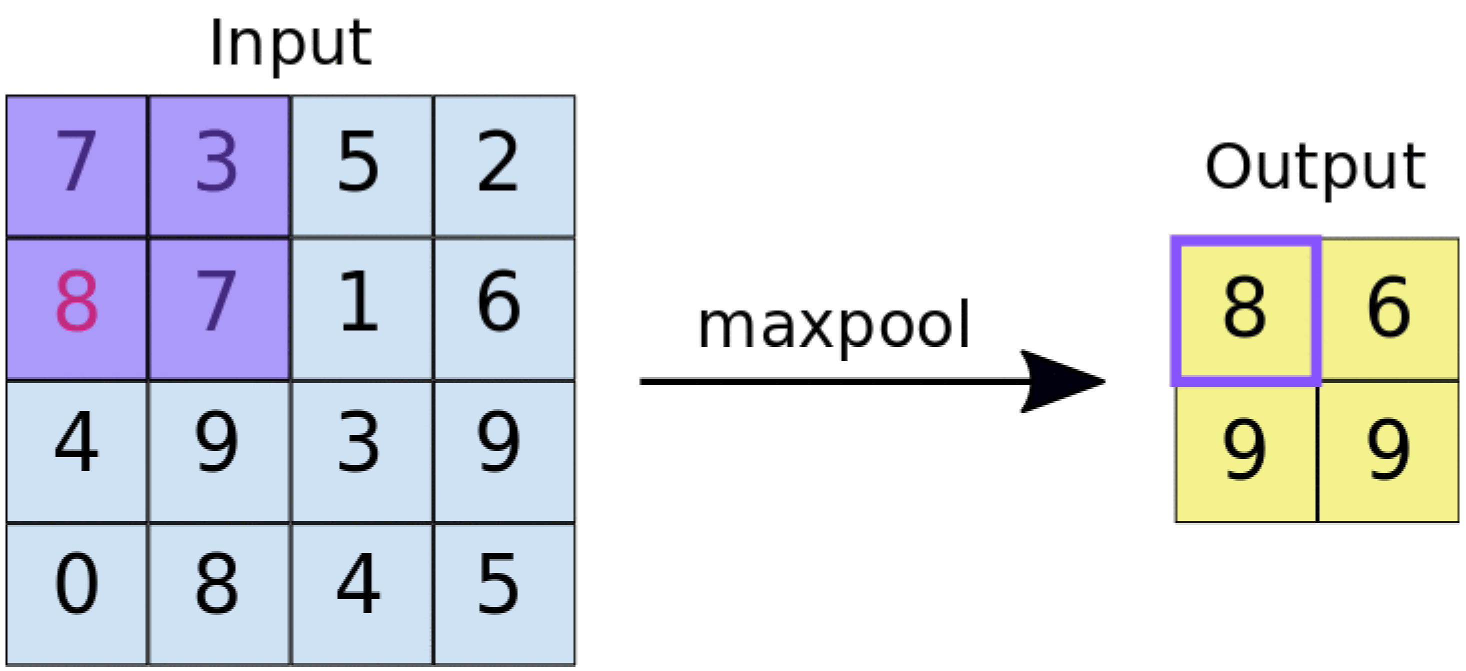
CNN Lenet: 99.0 ++ %

Open [MNIST with Lenet]

```
model = keras.Sequential(
    [
        keras.Input(shape=input_shape),
        layers.Conv2D(32, kernel_size=(3, 3), activation="relu"),
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Conv2D(64, kernel_size=(3, 3), activation="relu"),
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Flatten(),
        layers.Dense(64, activation='relu'),
        layers.Dense(num_classes, activation="softmax"),
    ]
)
```



Max Pooling Operation



Feature maps learned from data

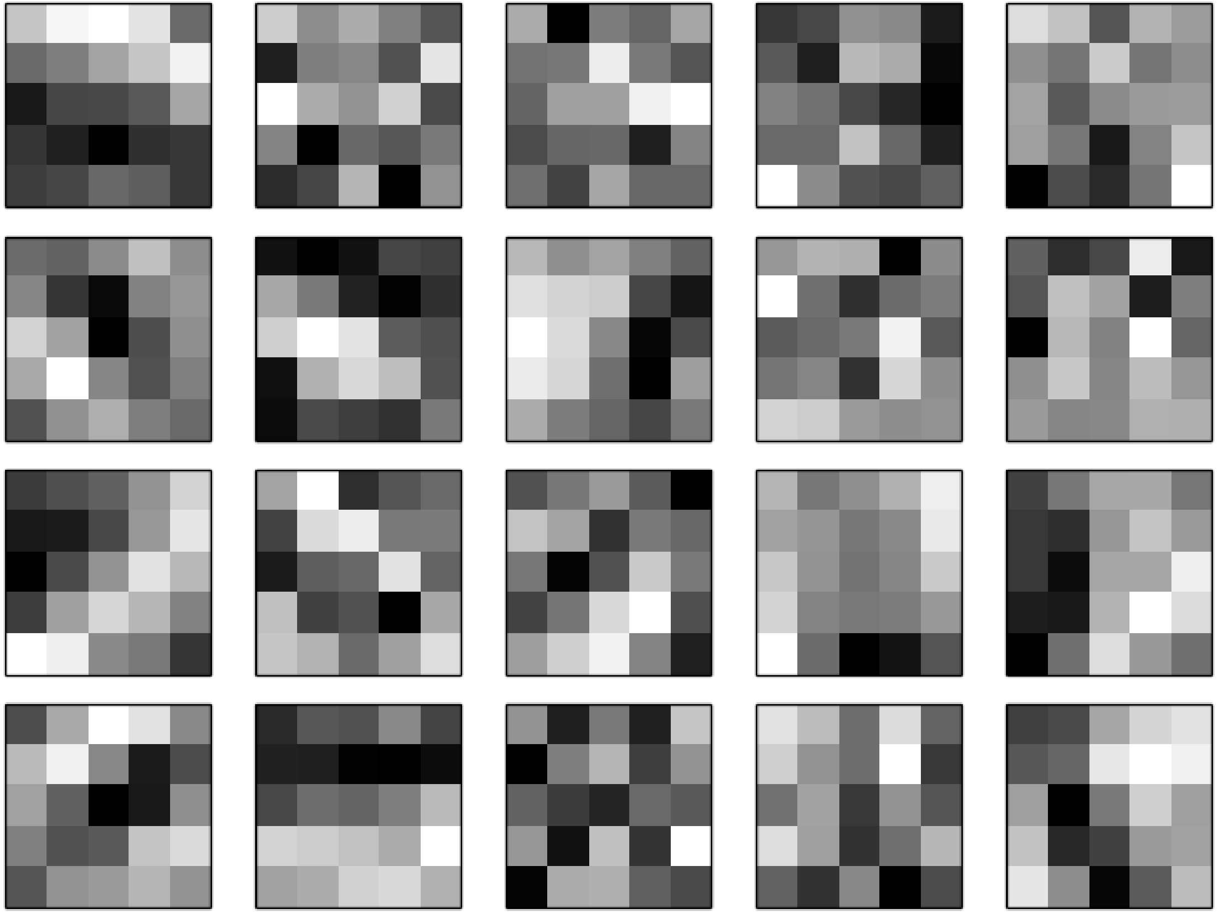
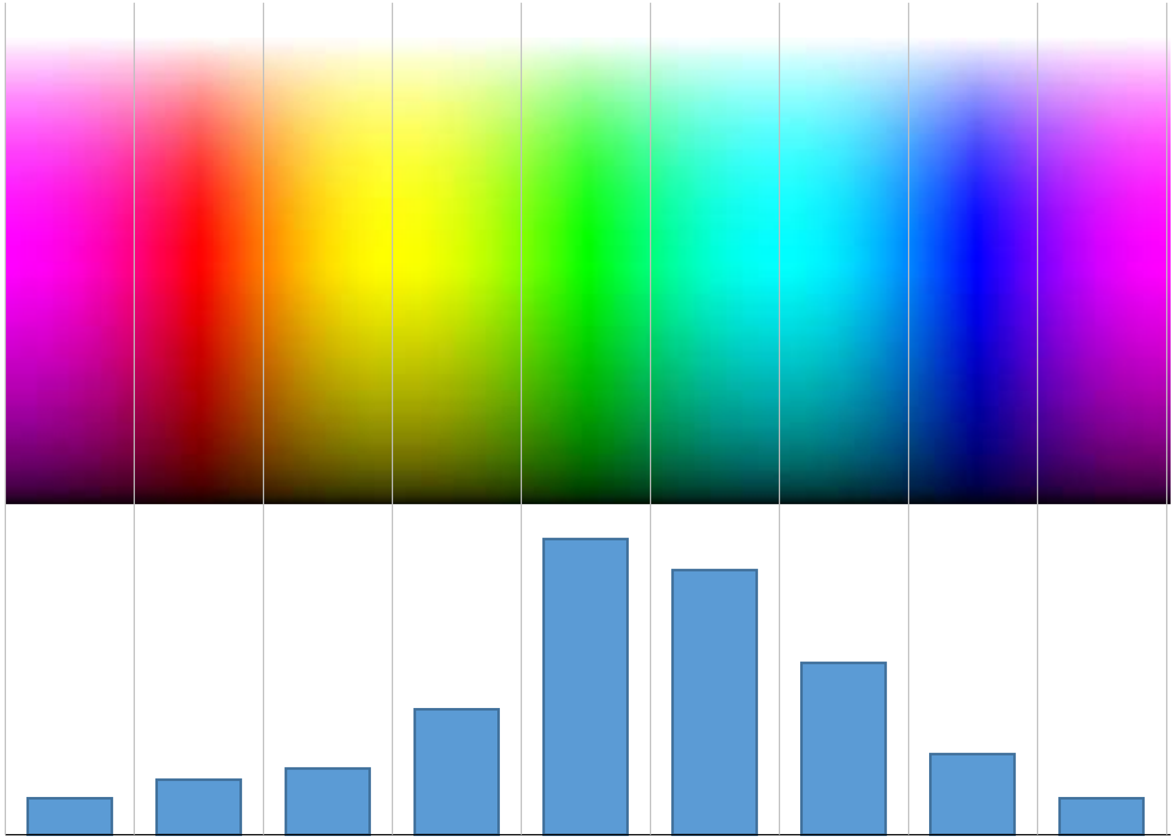
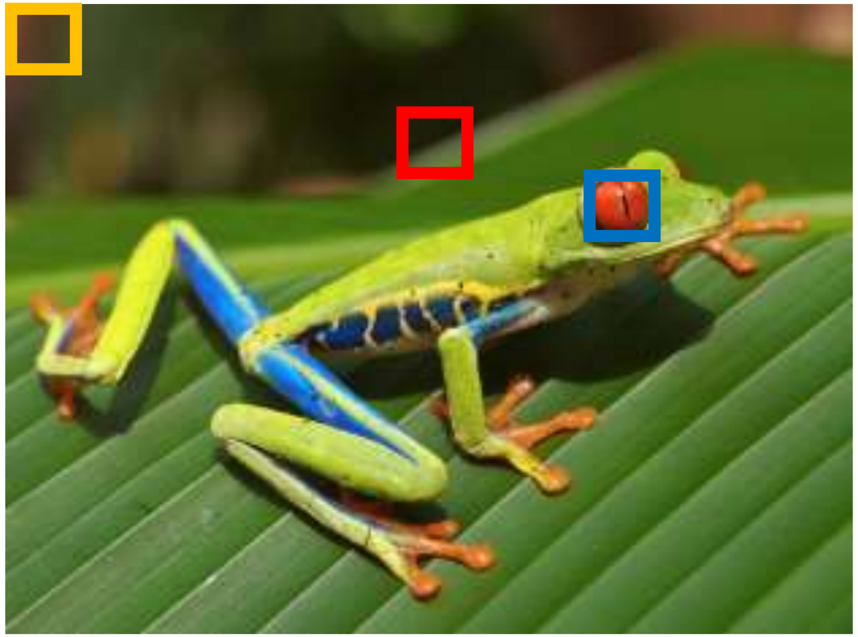


Image Features: Color Histogram



Ignores texture,
spatial positions

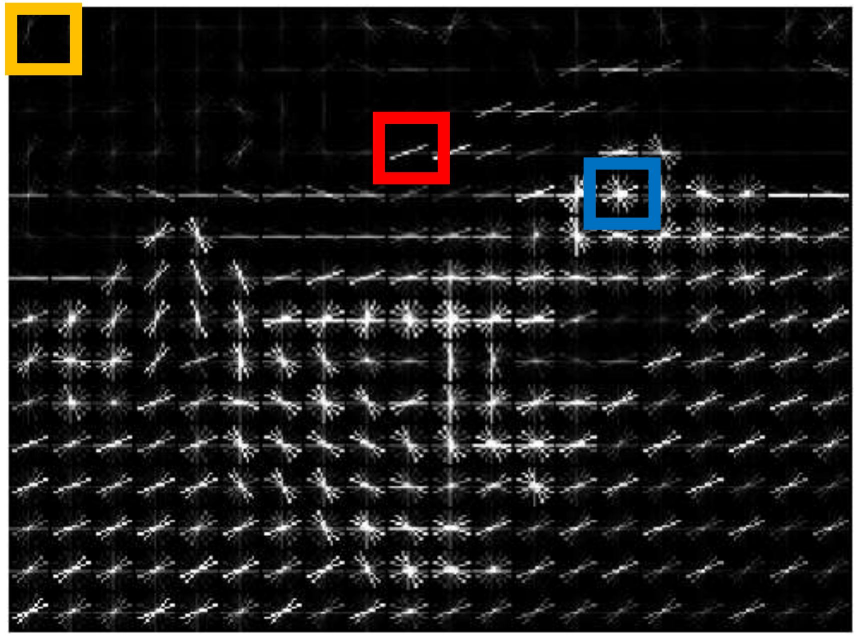
Image Features: Histogram of Oriented Gradients (HoG)



Weak edges

Strong diagonal edges

Edges in all directions



1. Compute edge direction / strength at each pixel
2. Divide image into 8x8 regions
3. Within each region compute a histogram of edge directions weighted by edge strength

Captures texture and position, robust to small image changes

Example: 320x240 image gets divided into 40x30 bins; 8 directions per bin; feature vector has 30*40*9 = 10,800 numbers

Lowe, “Object recognition from local scale-invariant features”, ICCV 1999
Dalal and Triggs, "Histograms of oriented gradients for human detection," CVPR 2005

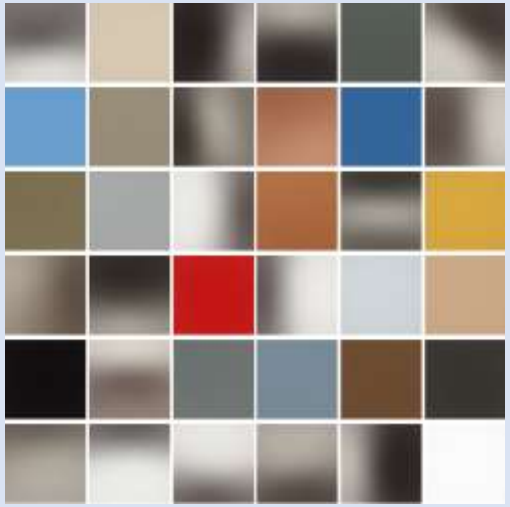
Image Features: Bag of Words (Data-Driven)

Step 1: Build codebook

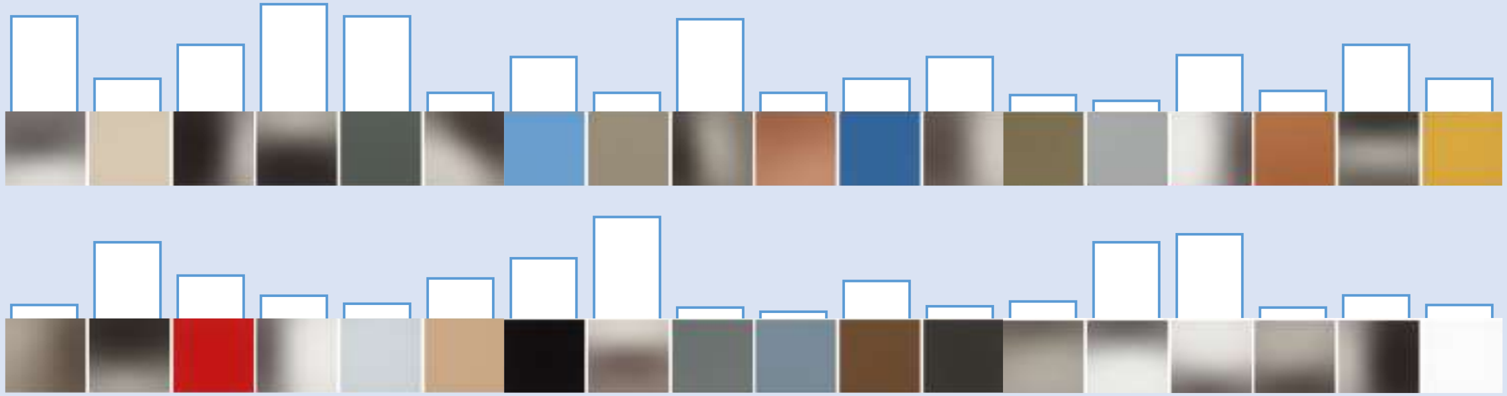
Extract random patches

from all images

Cluster patches to form "codebook" of "visual words"



Step 2: Encode images



Fei-Fei and Perona, "A bayesian hierarchical model for learning natural scene categories", CVPR 2005

Image Features

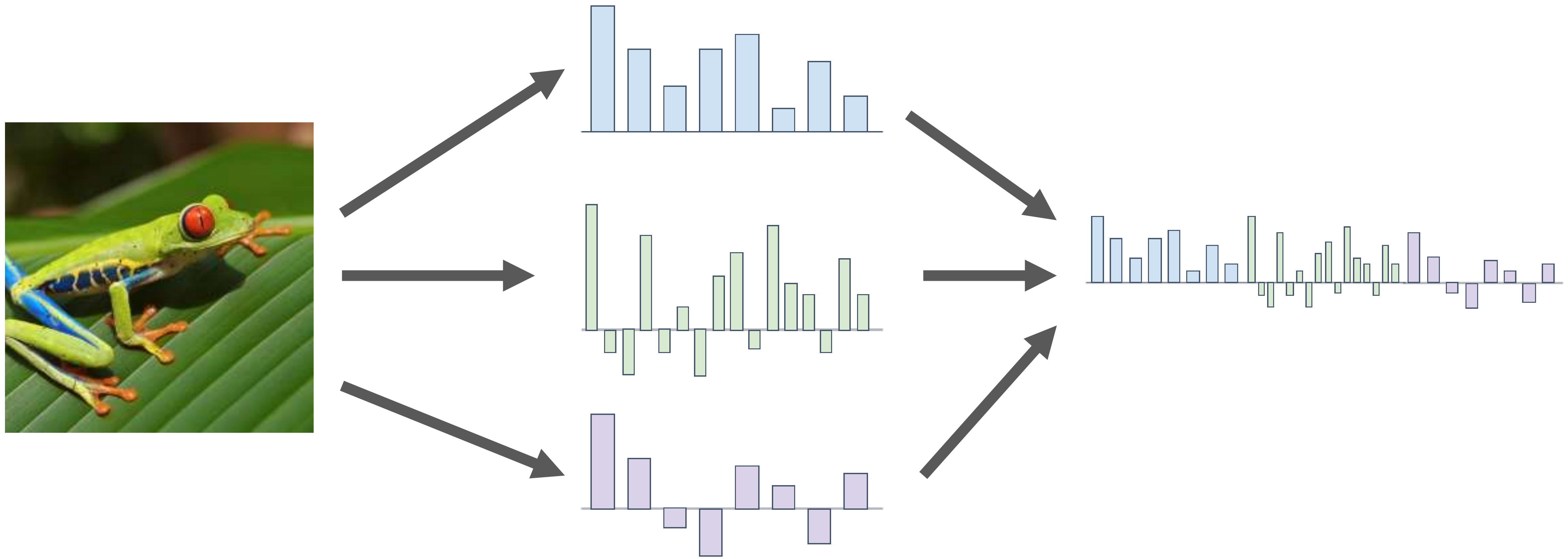
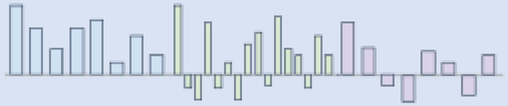


Image Features

Feature Extraction



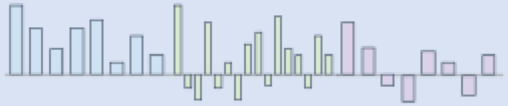
f

10 numbers giving scores for classes

training

Image Features vs Neural Networks

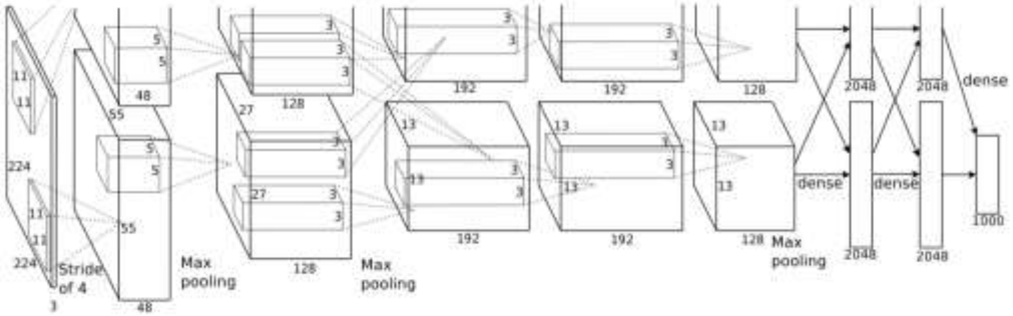
Feature Extraction



f

10 numbers giving scores for classes

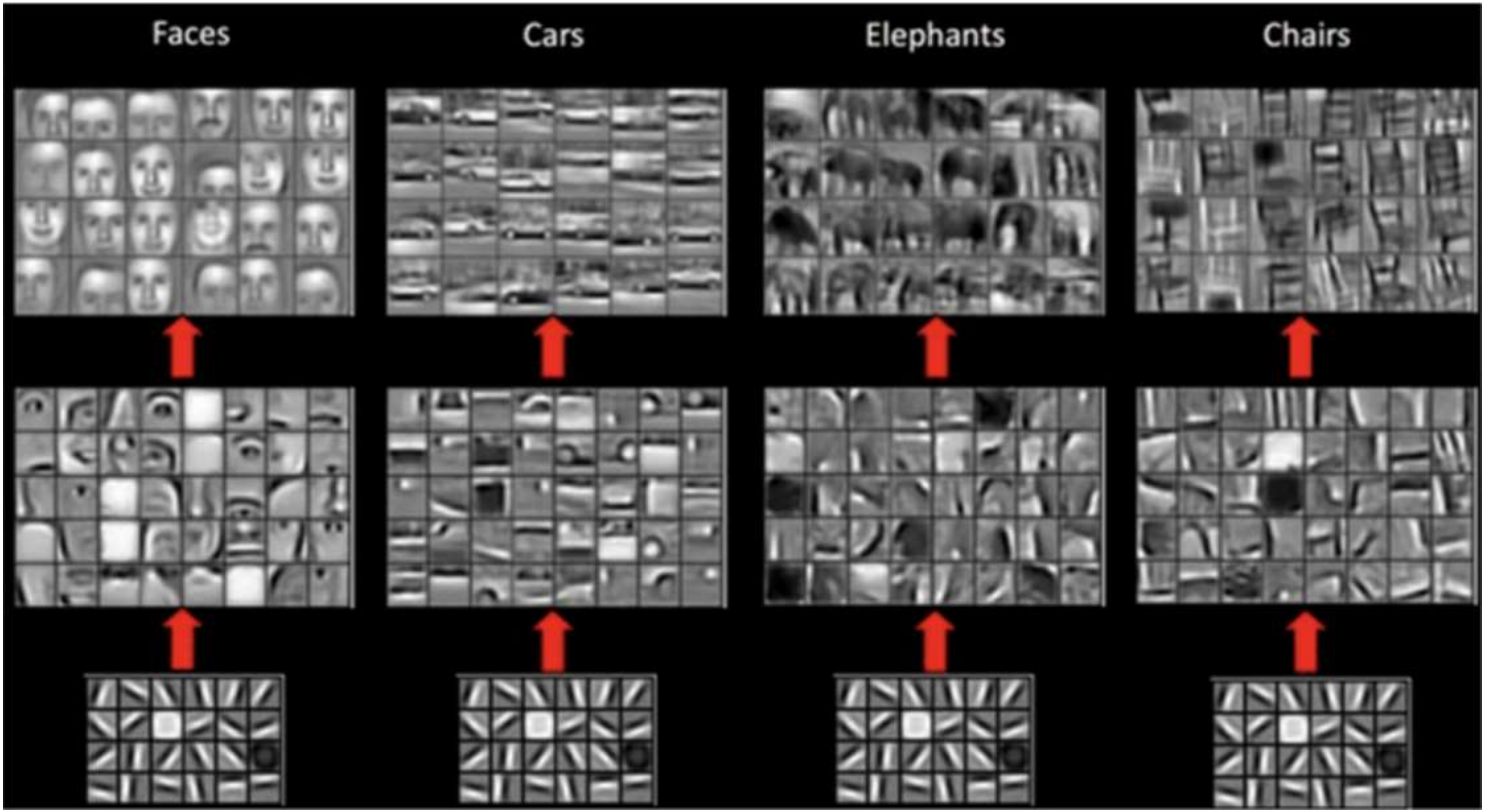
training



Krizhevsky, Sutskever, and Hinton, "Imagenet classification with deep convolutional neural networks", NIPS 2012.
Figure copyright Krizhevsky, Sutskever, and Hinton, 2012.

10 numbers giving scores for classes

training



ITXOTIC intel

Activation Function

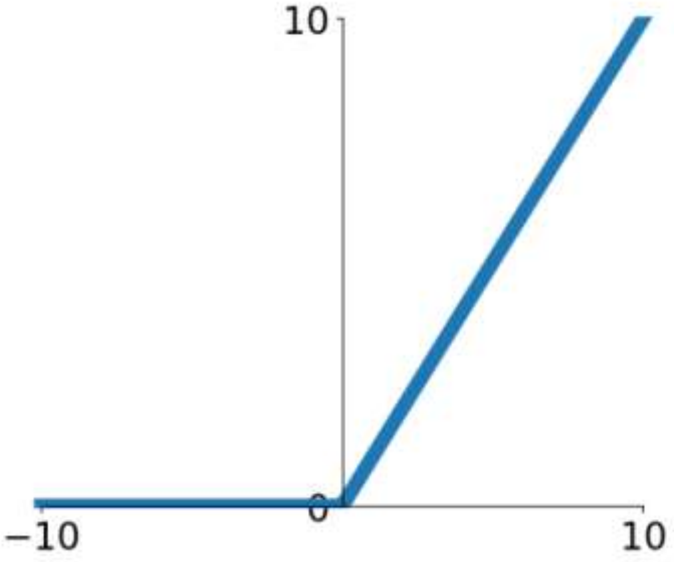
ReLU is a good default choice for most problems

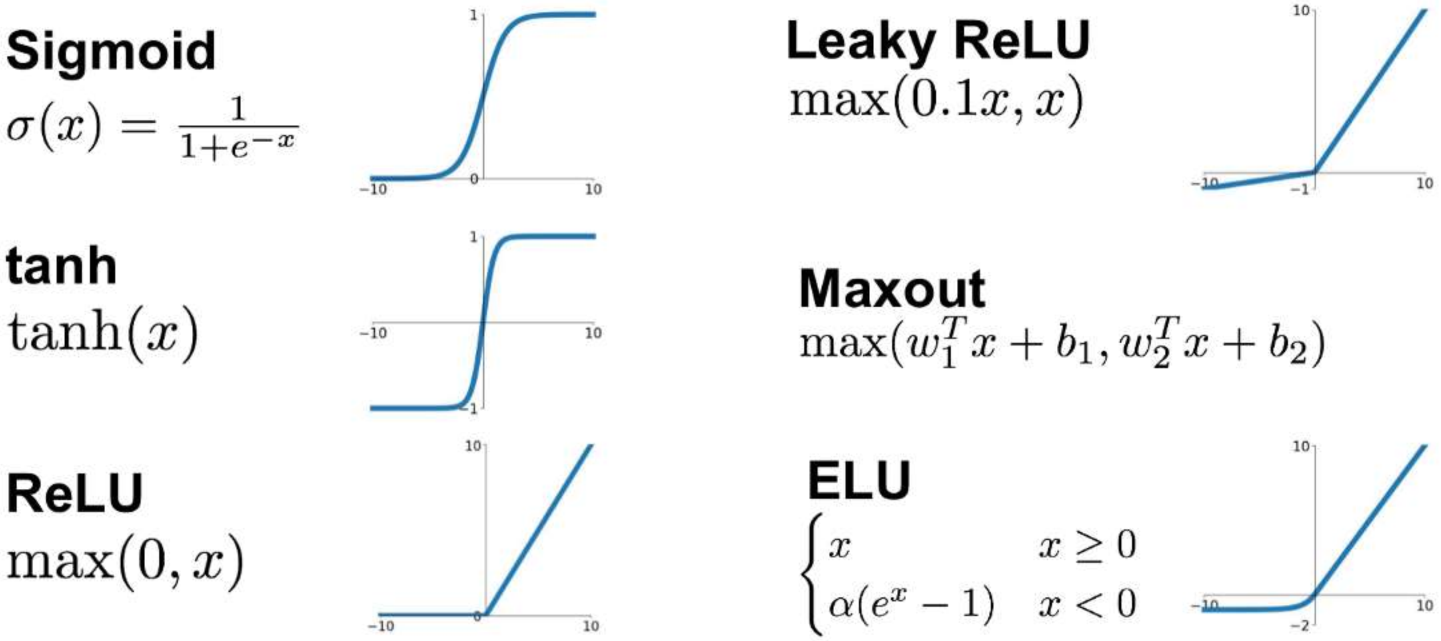
- A function that defines the output of a neuron given the input

The function

$$\mathrm{ReLU}(z) = \max(0, z)$$

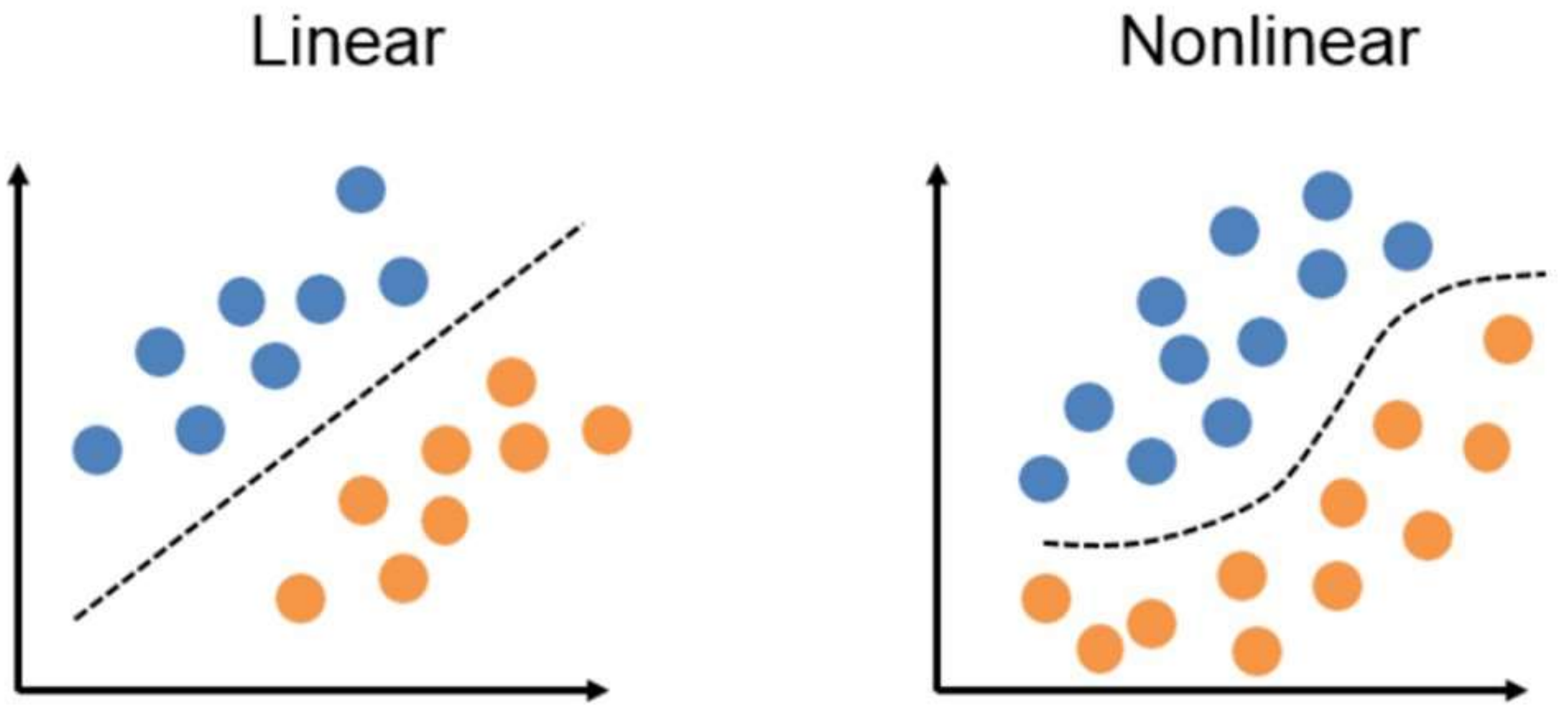
is called "Rectified Linear Unit"



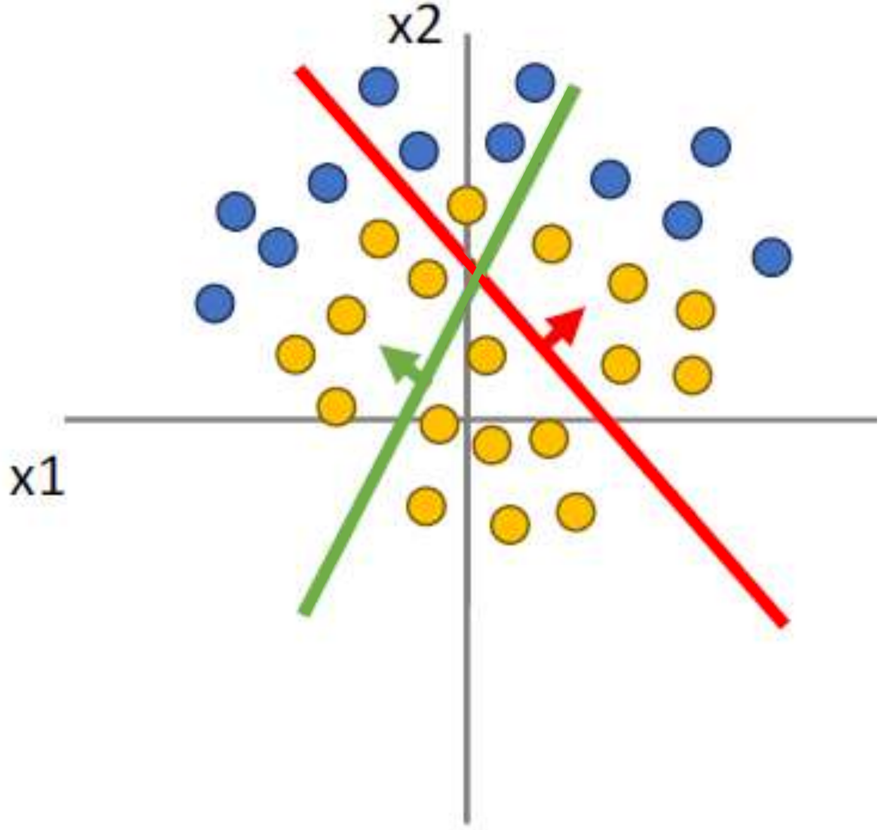


ITXOTIC intel.

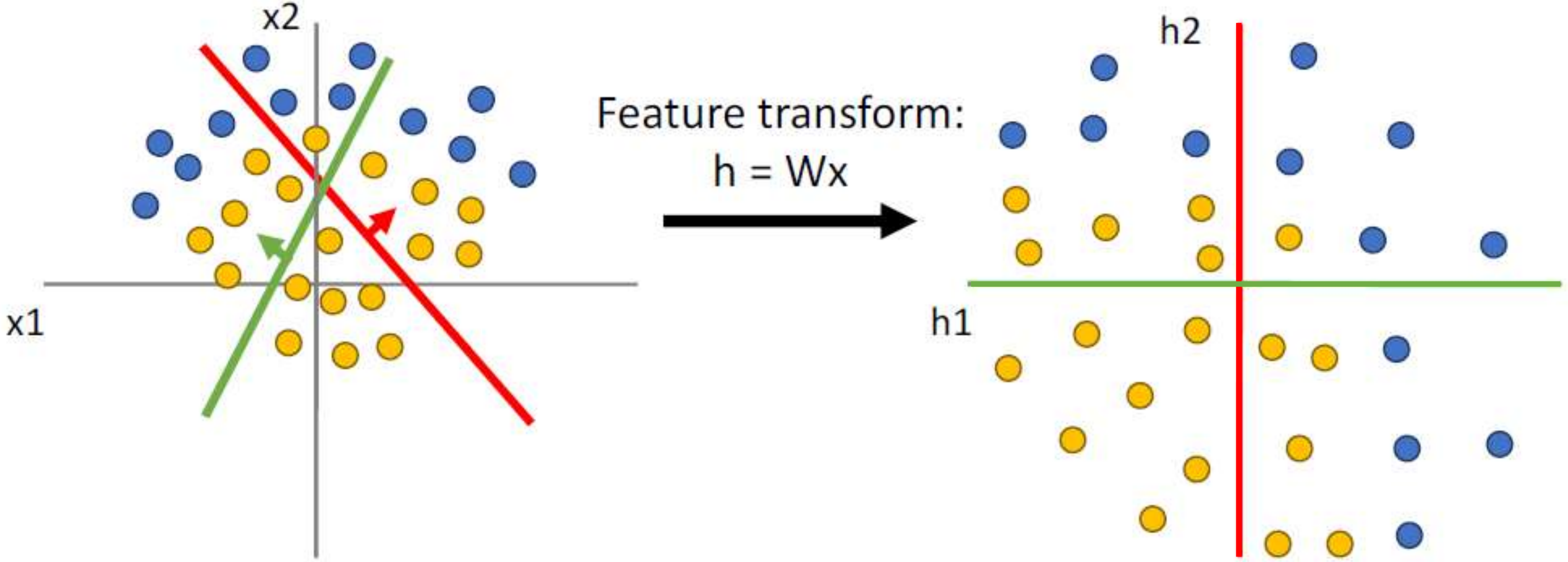
Non-linear curve with activation function

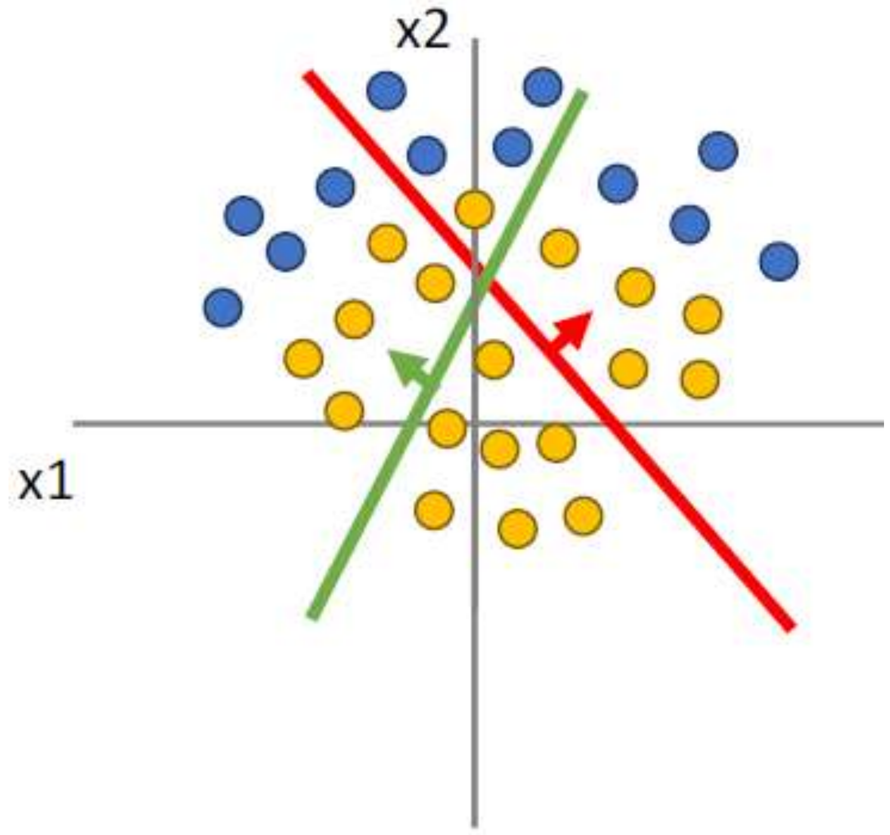


Points not linearly
separable in original space



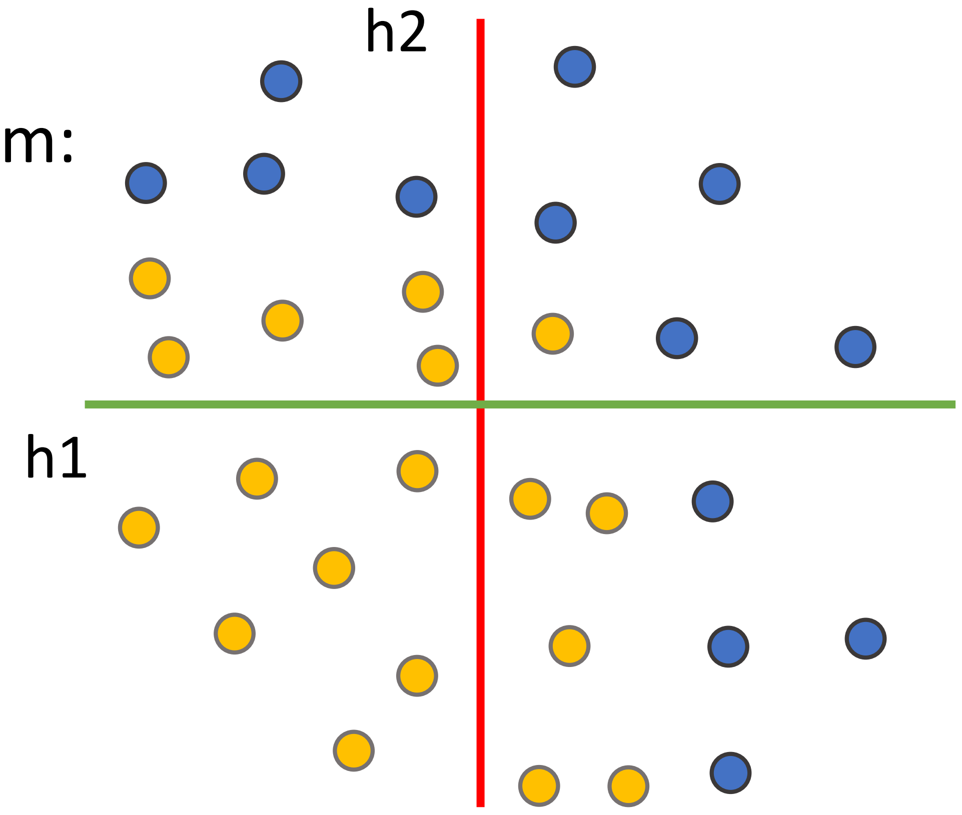
Points not linearly separable in original space





Feature transform:

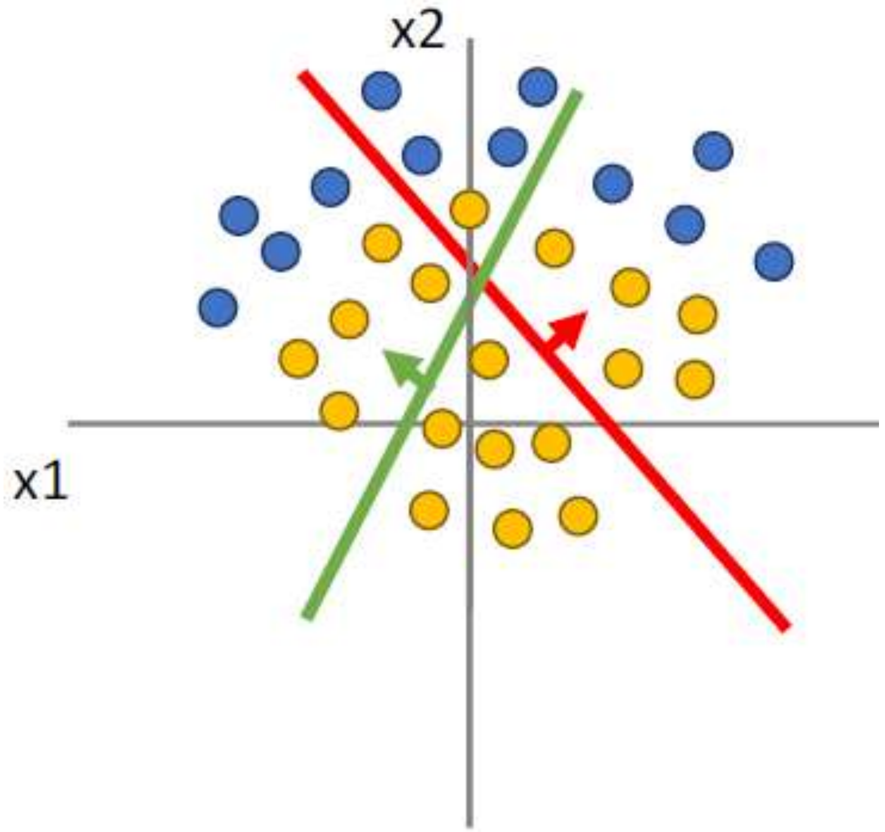
$h = Wx$



Consider a neural net hidden layer:

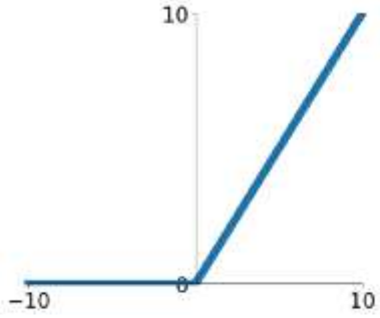
$h = \mathrm{ReLU}(Wx) = \max(0, Wx)$

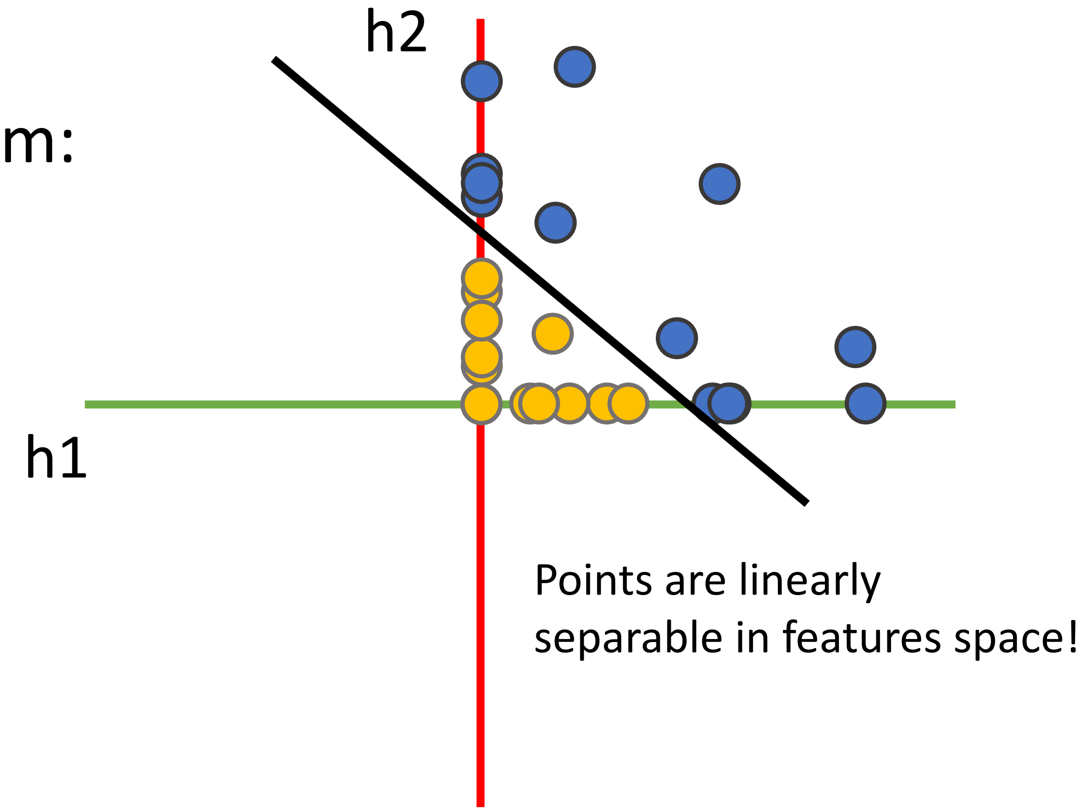
Where x, h are both 2-dimensional



Feature transform:

$h = \mathrm{ReLU}(Wx)$

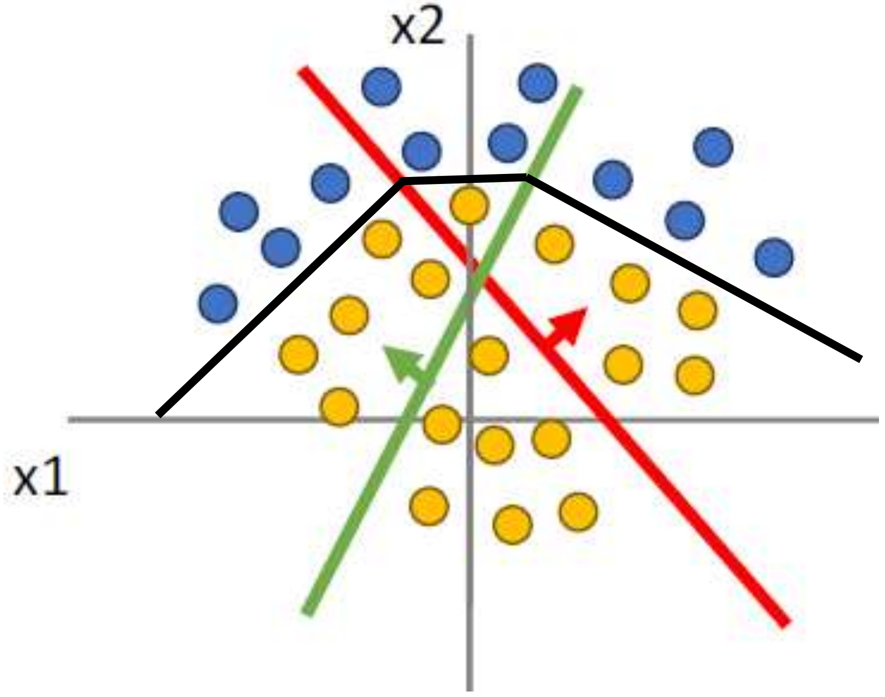




Consider a neural net hidden layer:

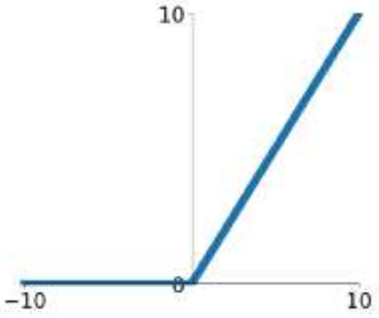
h = ReLU(Wx) = max(0, Wx)

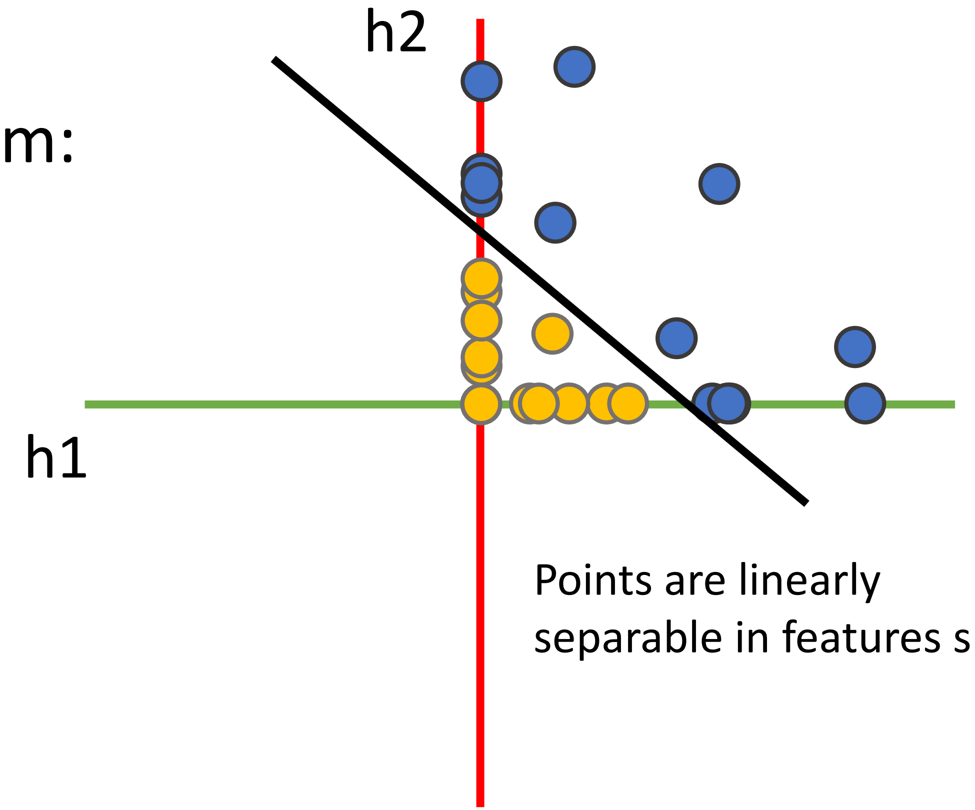
Where x, h are both 2-dimensional



Feature transform:

h = ReLU(Wx)

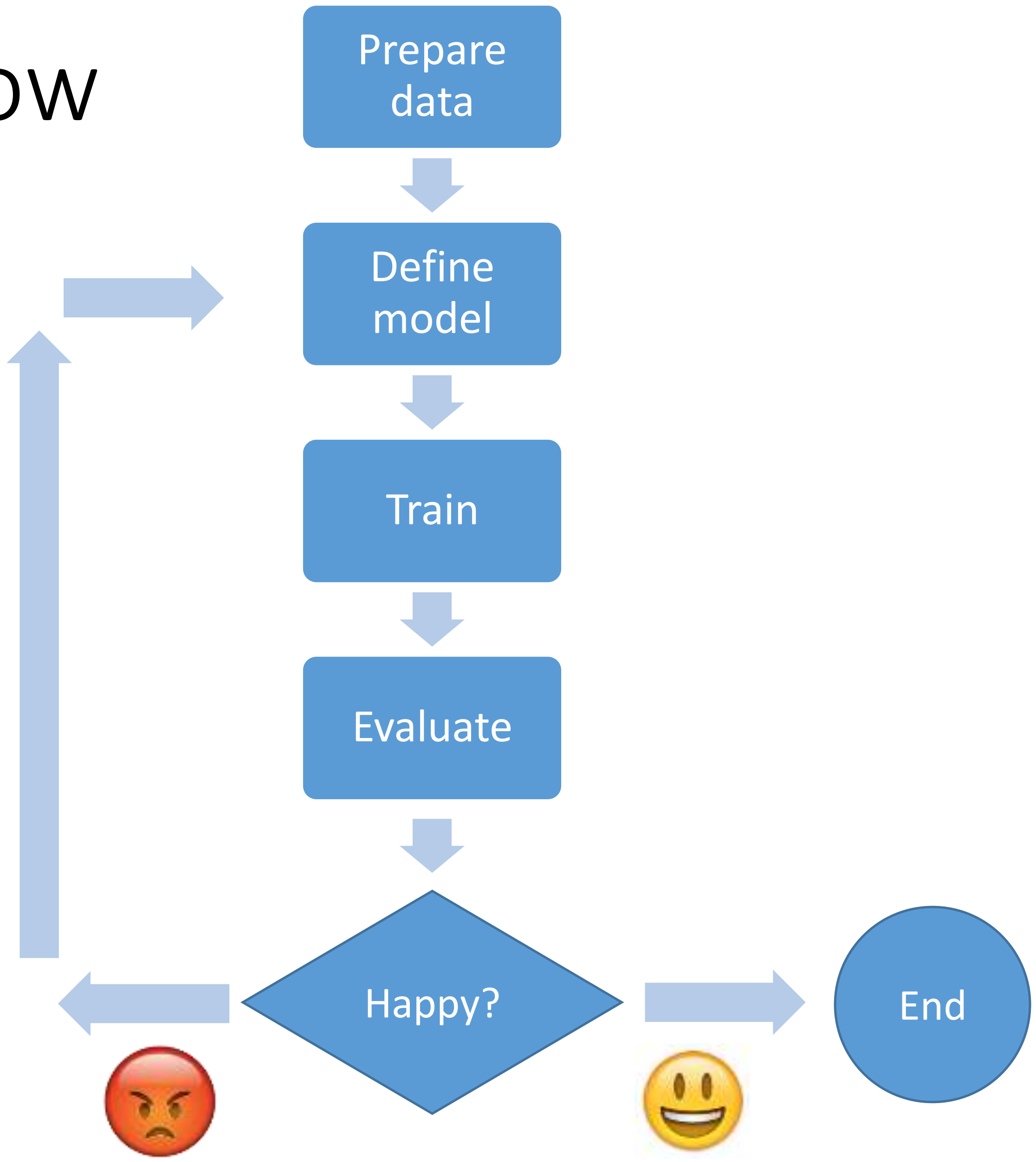




Points are linearly separable in features space!

Linear classifier in feature space gives nonlinear classifier in original space

Typical DL workflow

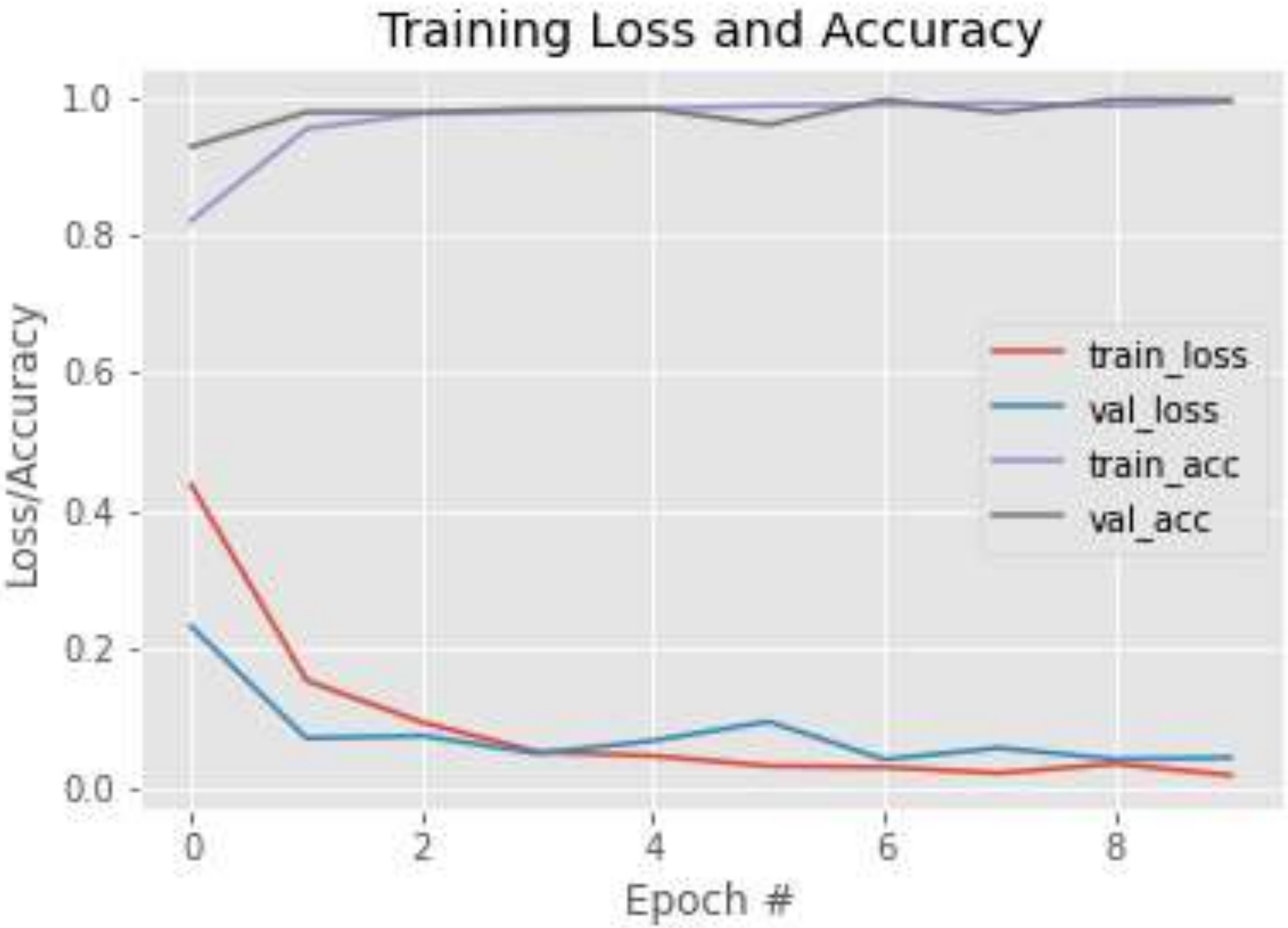


- Train the best classifier on the MNIST dataset
- Tune the following hyperparameters
  - # filters
  - Kernel size
  - # neurons
  - Activation function: 'relu', 'sigmoid', 'tanh'
  - # epochs
  - # layers

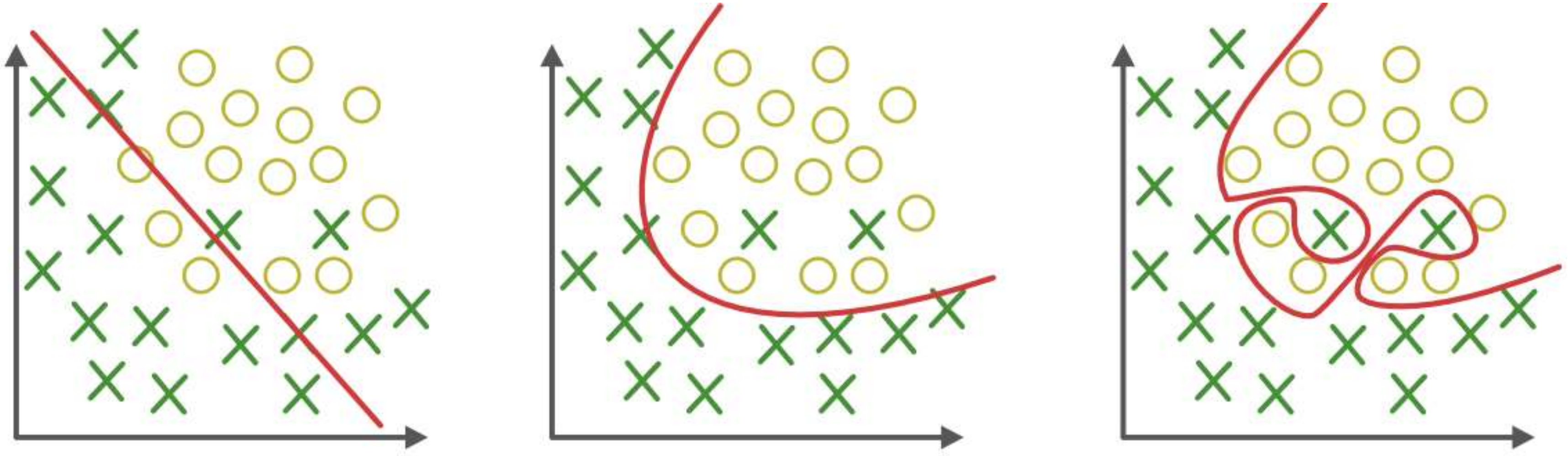
Overfitting

- Data Augmentation
- Use Dropout layer
- Transfer Learning

Learning curve



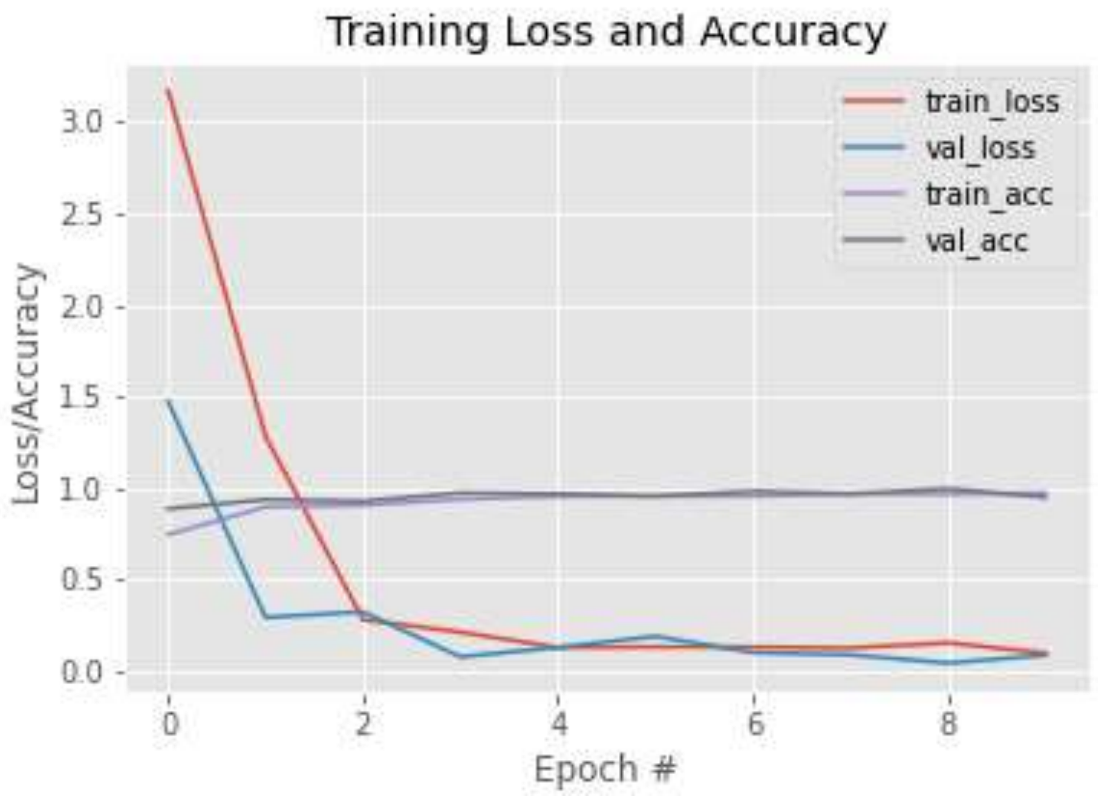
Overfitting



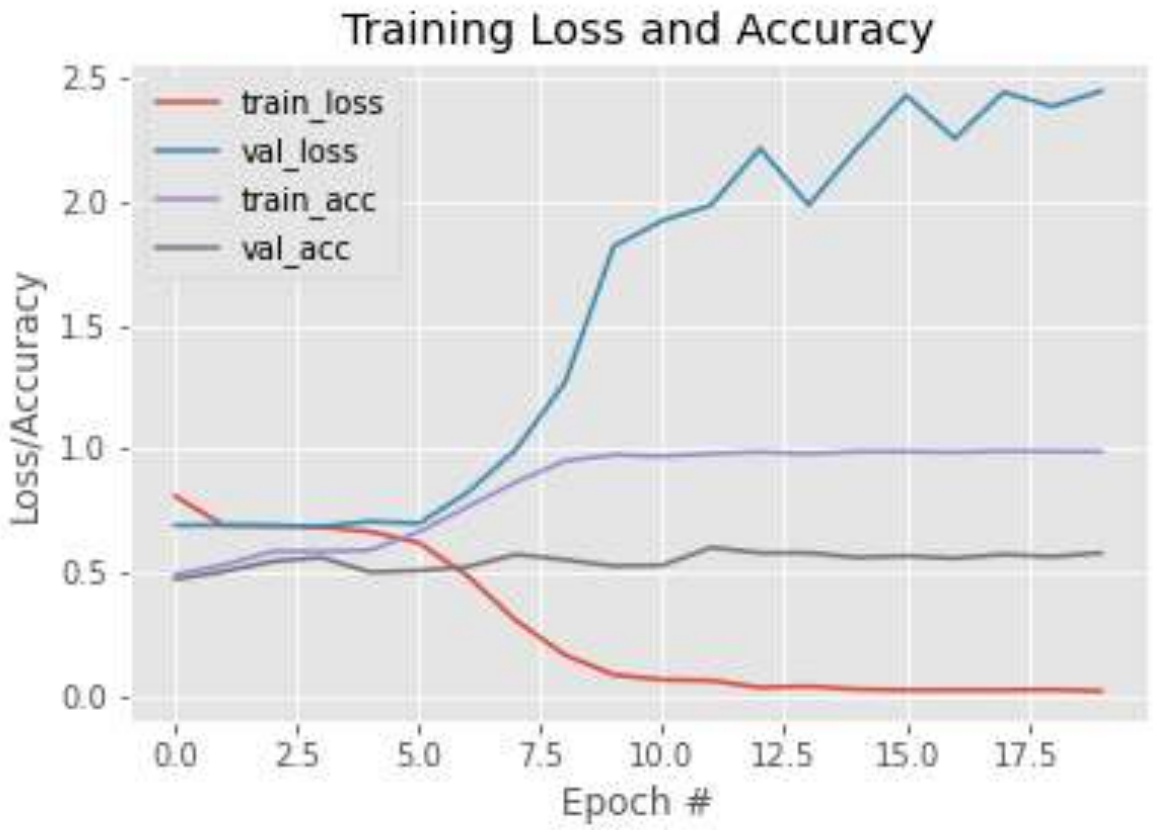
Underfit Optimal Overfit

Signs of overfitting

- High training accuracy, low validation/test accuracy



Normal

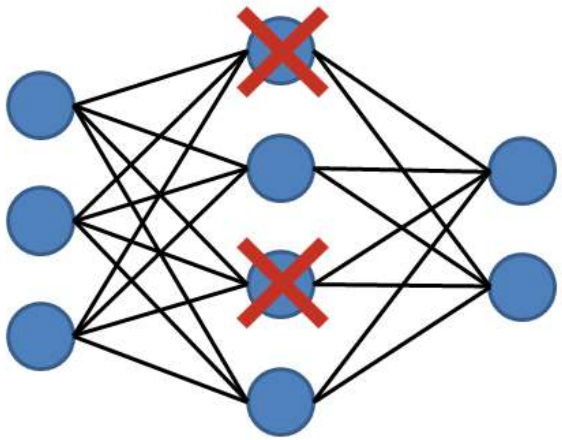


Overfit

ITXOTIC intel

Dropout Regularization

```
model = keras.Sequential(
    [
        keras.Input(shape=input_shape),
        layers.Conv2D(32, kernel_size=(3, 3), activation="relu"),
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Conv2D(64, kernel_size=(3, 3), activation="relu"),
        layers.MaxPooling2D(pool_size=(2, 2)),
        layers.Flatten(),
        layers.Dense(64, activation='relu'),
        layers.Dropout(0.2),
        layers.Dense(num_classes, activation="softmax"),
    ]
)
```

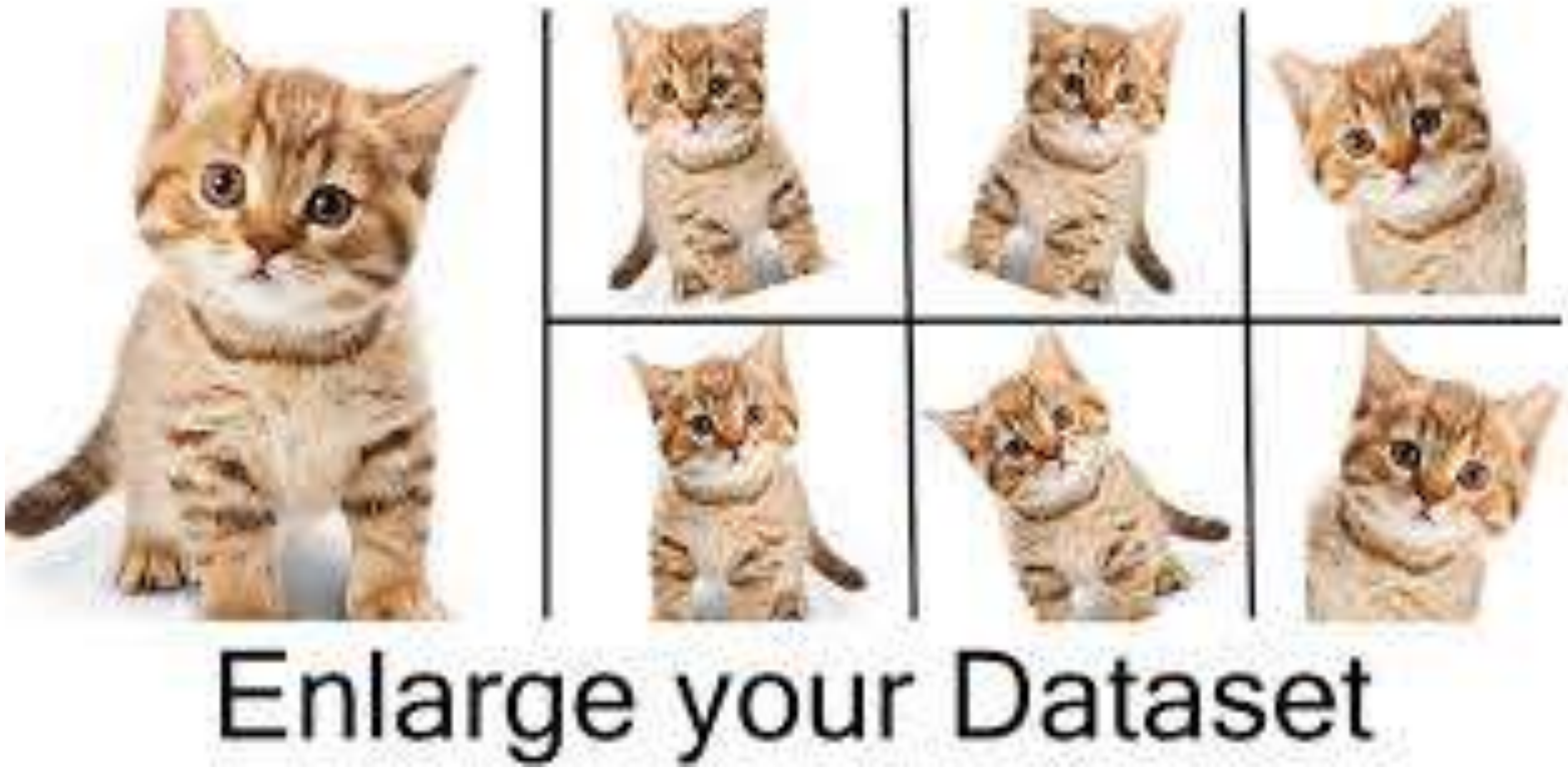


Dropout randomly sets input units to 0, which helps prevent overfitting.

Data Augmentation

More data variation helps reduce overfitting

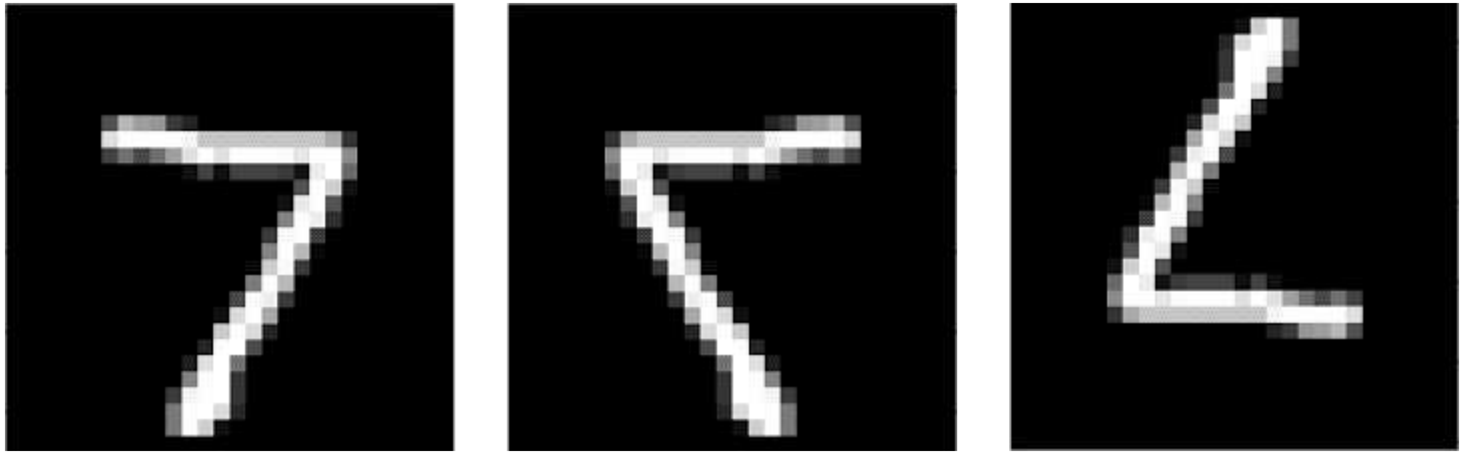
- Rotation
- Translation
- Flipping
- Zoom/Pan



Data Augmentation

More data variation helps reduce overfitting

- Rotation
- Translation
- Flipping
- Zoom/Pan



ITXOTIC intel

Data Augmentation

Mirror + Rotation

Rotation

Data Augmentation

Mirror + Noise

Translation

CHALLENGE

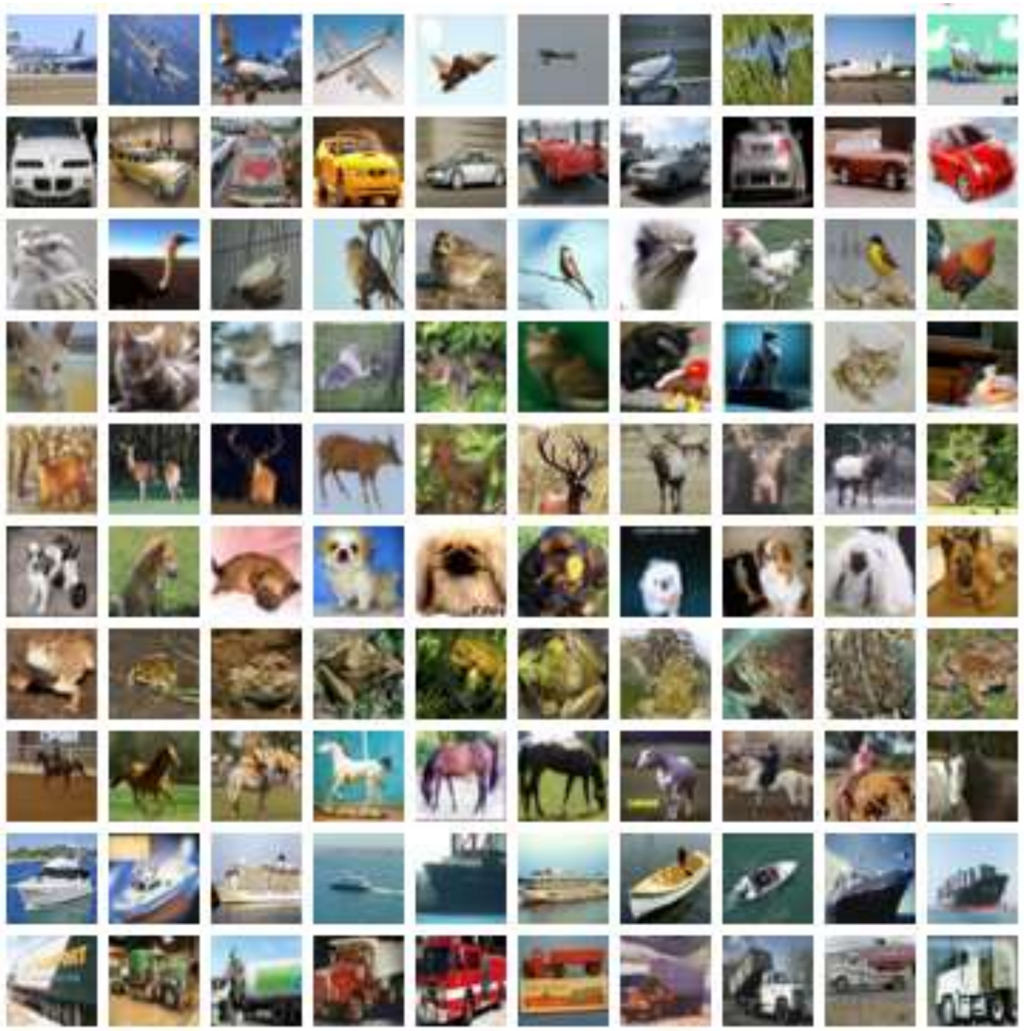
- Implement regularization techniques.
- Tune hyperparameters to yield the best accuracy.
- Try on other datasets.

CIFAR10

Datasets

1. MNIST
2. CIFAR10
3. CIFAR100
4. ImageNet
5. MIT Places

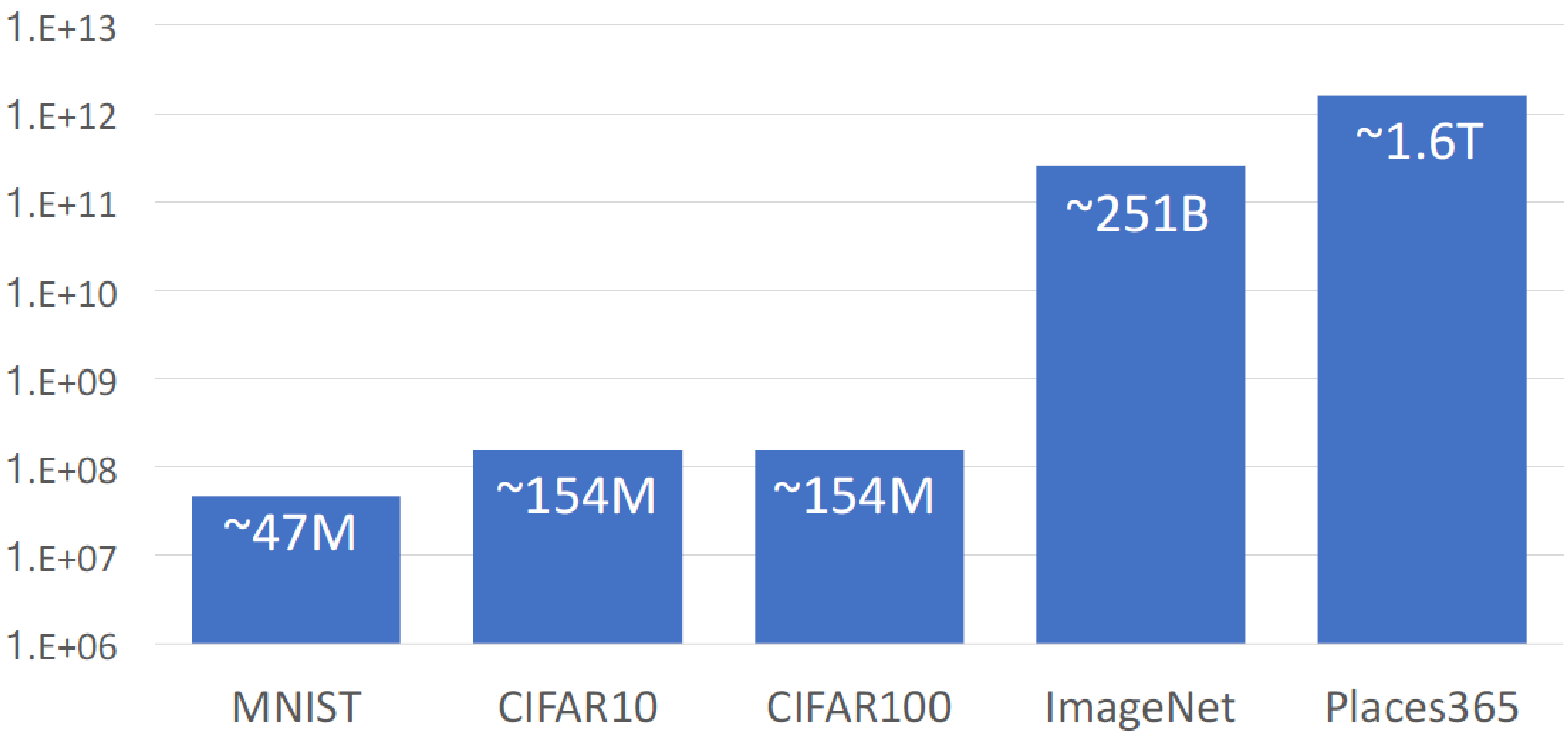
airplane
automobile
bird
cat
deer
dog
frog
horse
ship
truck



50,000 training images
Each image is **32x32x3**

10,000 test images.

Datasets: Number of Training Pixels

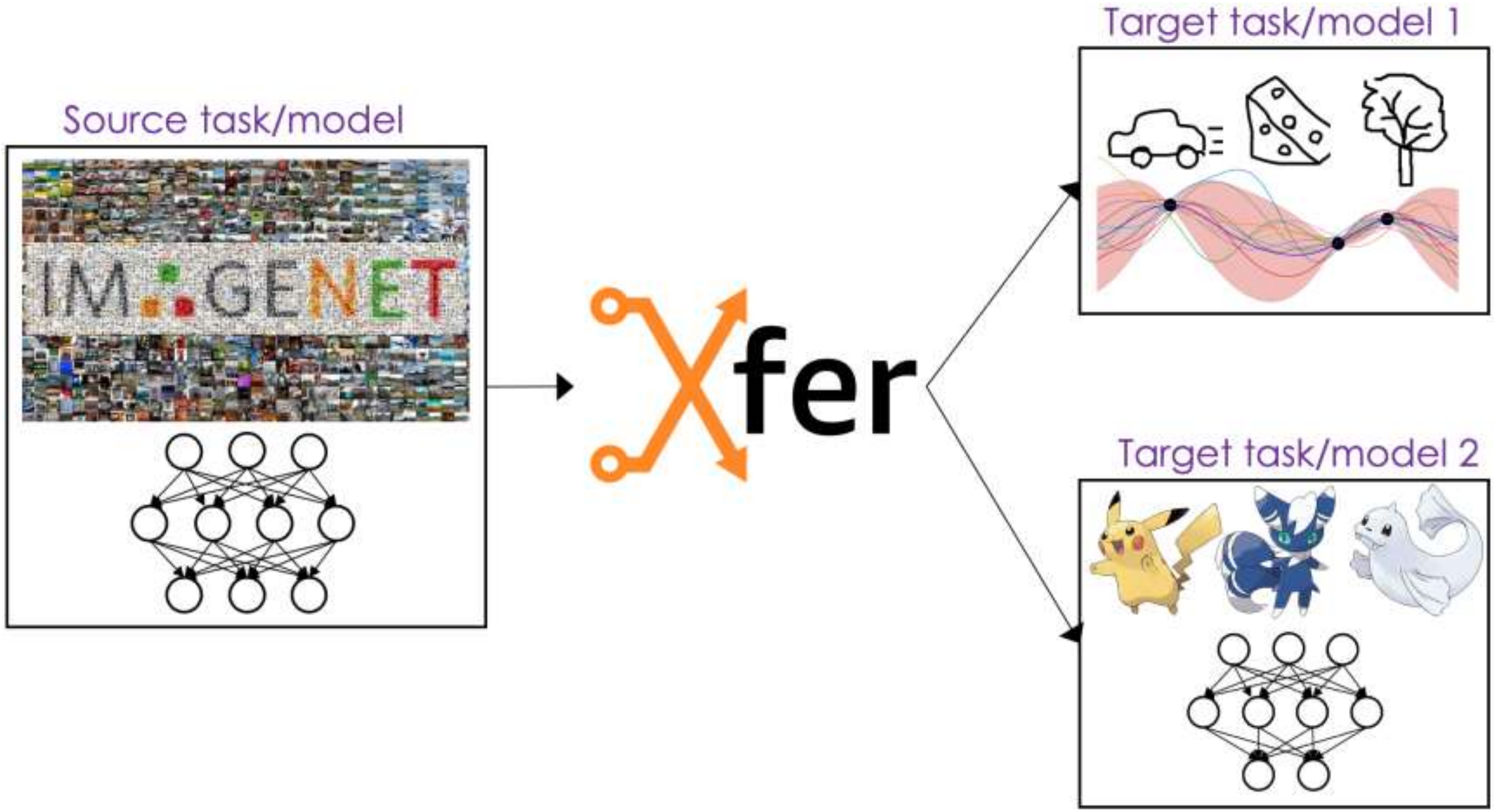


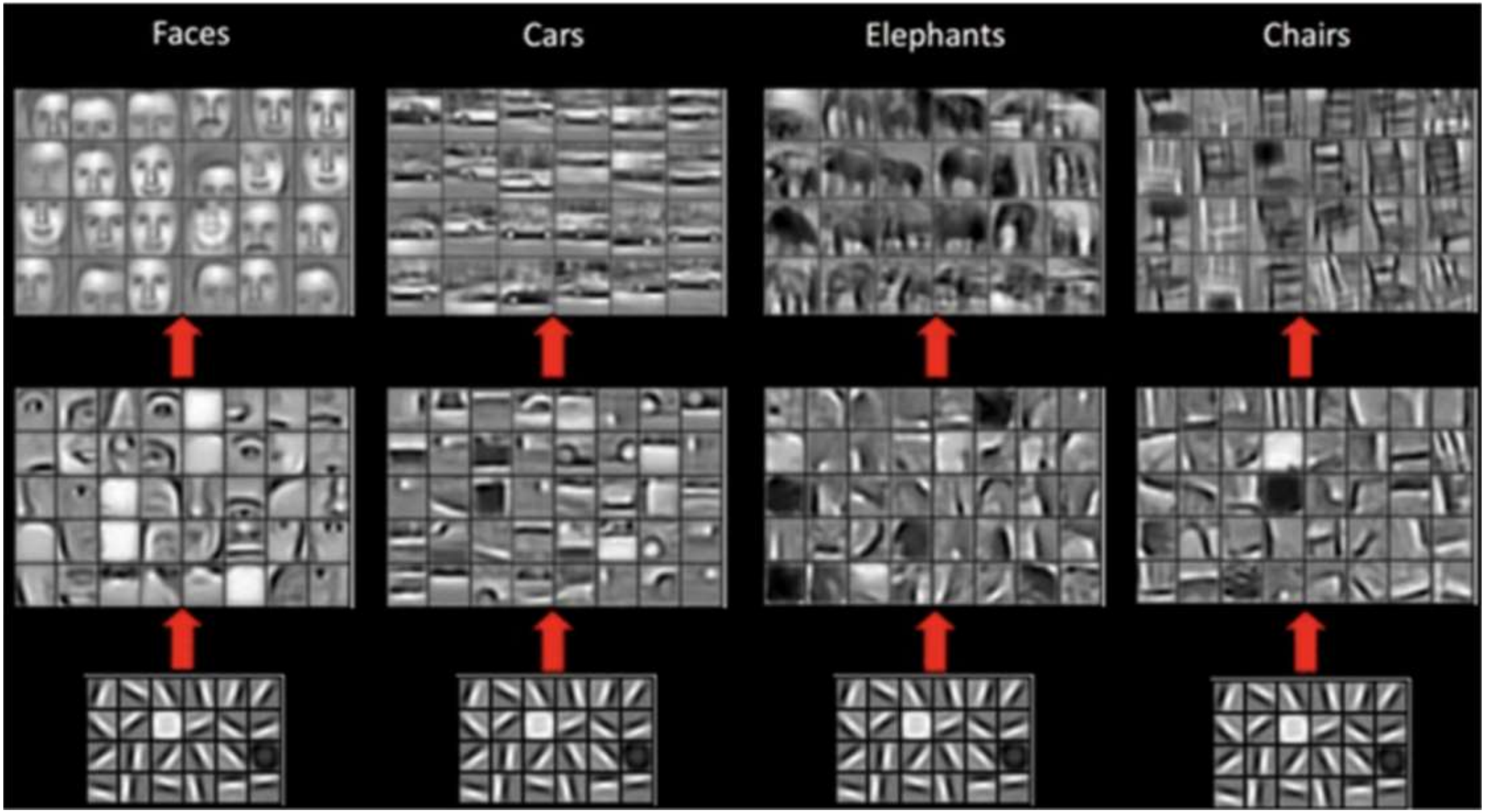
Transfer Learning

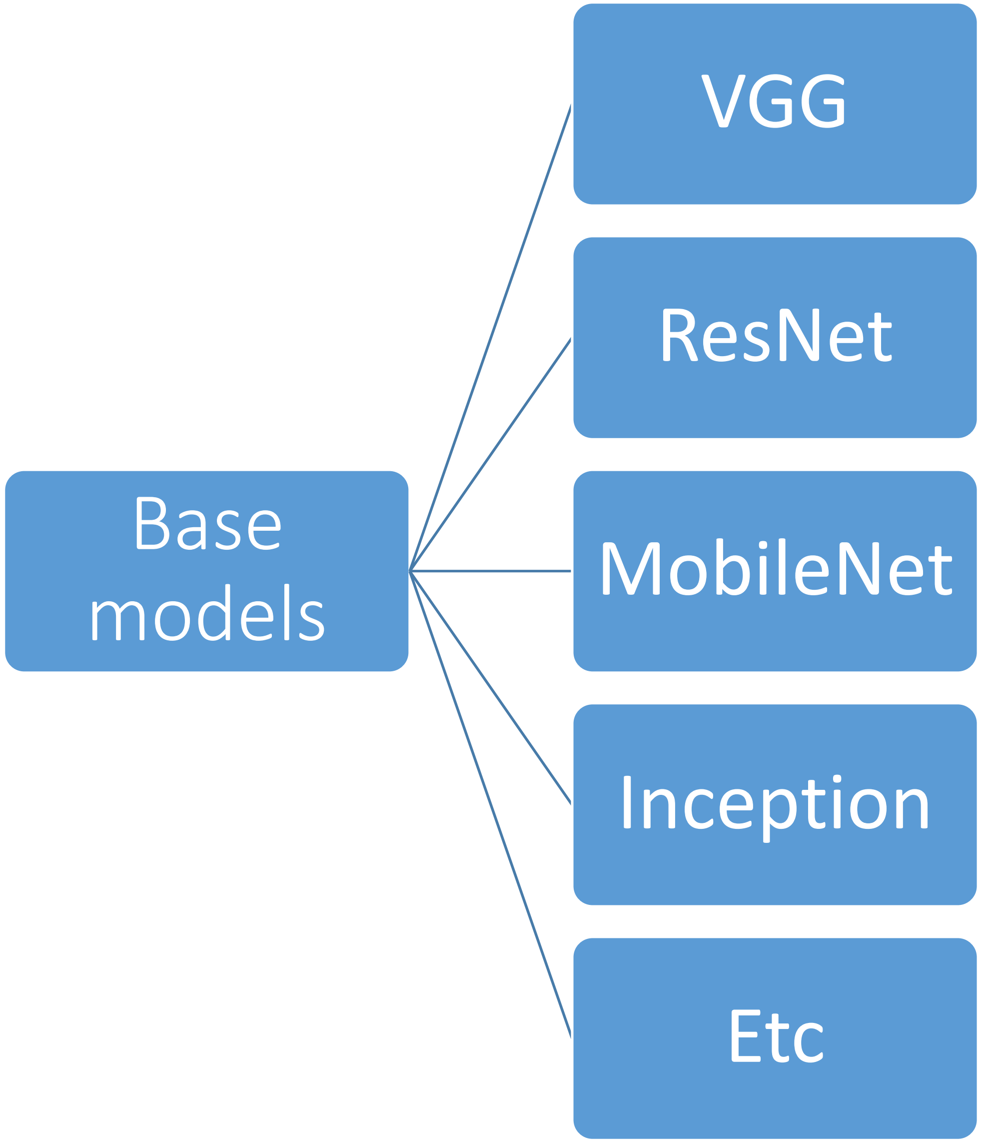
Use the **knowledge** gained while solving **one problem** and **applying** it to a **different** but related problem.

Start with a **pre-trained model** that are good at one task, lets you train far more **quickly** and with **less data** than if you were to train from scratch.

Transfer Learning

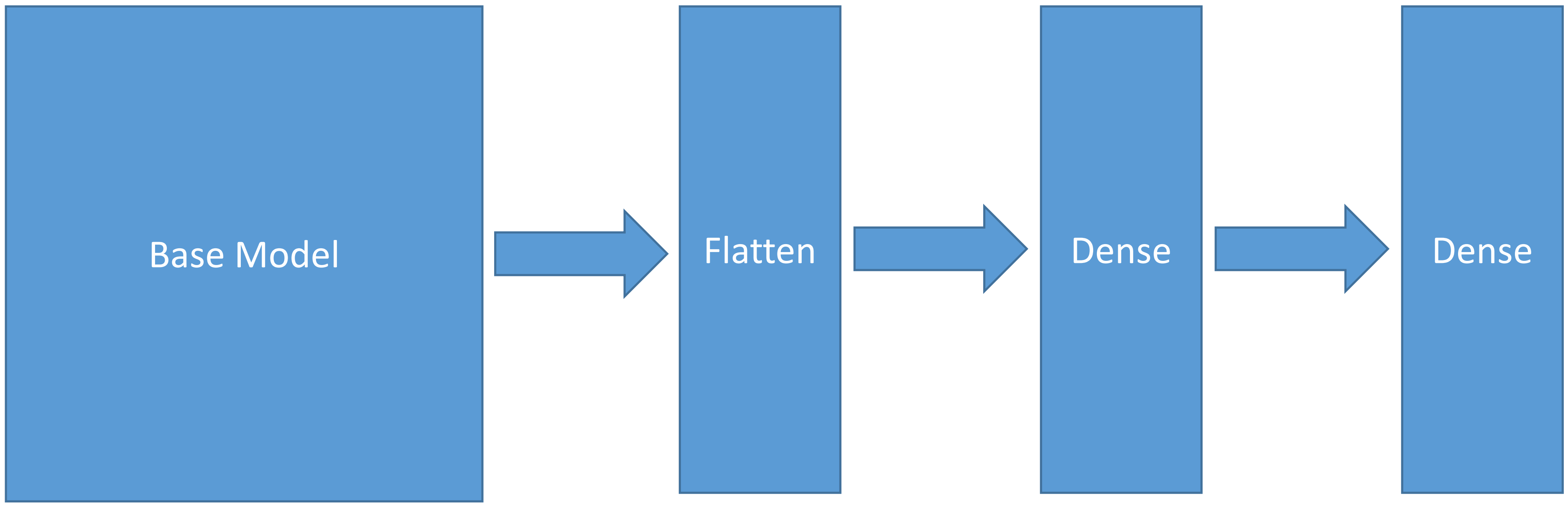



Faces
Cars
Elephants
Chairs



ITXOTIC intel

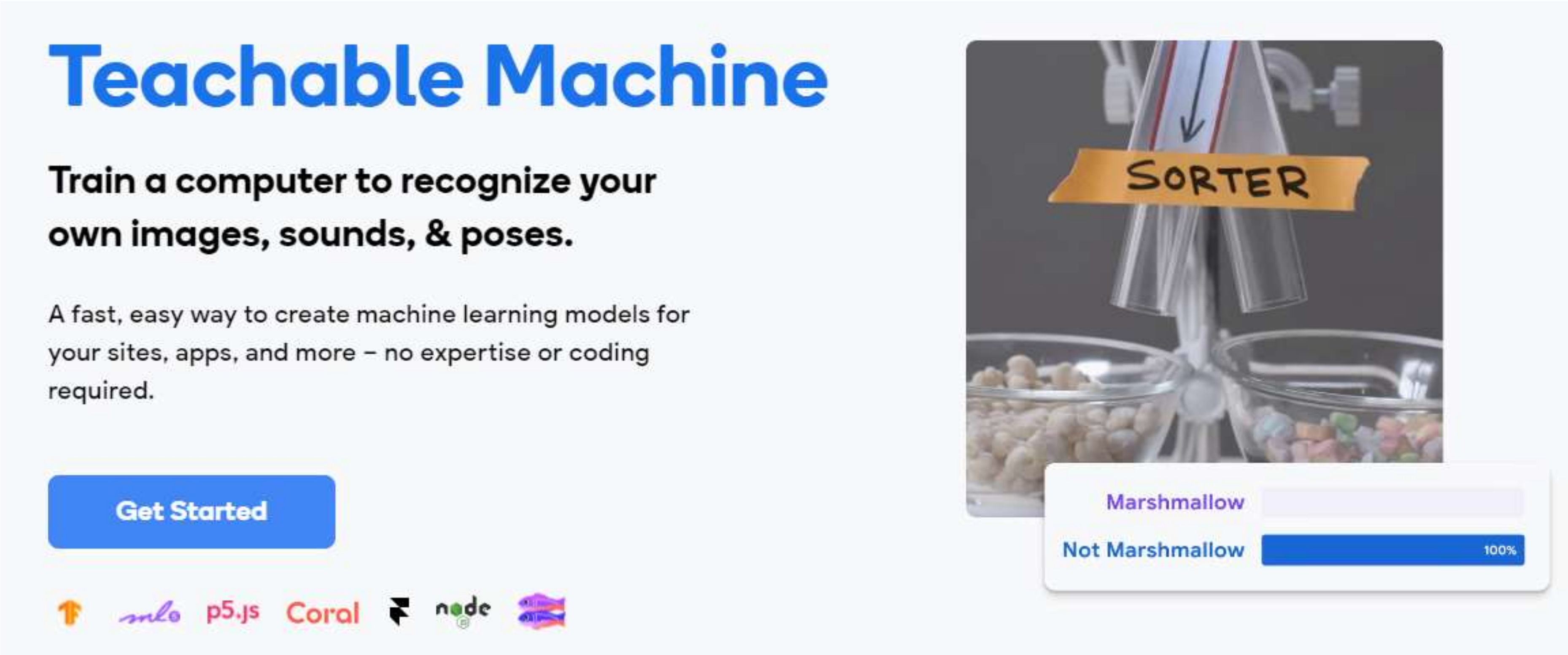
Transfer learning model



- Train the best classifier on the cats dogs dataset with transfer learning.
- Tune the base model and other hyperparameters for best results.
- With TL, you can get away training with very little data. But how little?

Lab 3: Teachable Machine

Customizable image classifier on webcam images



ITXOTIC intel.

MobileNet Model

- **MobileNet**: **Efficient** Convolutional Neural Networks for Mobile Vision Applications(paper).

Model Loaded

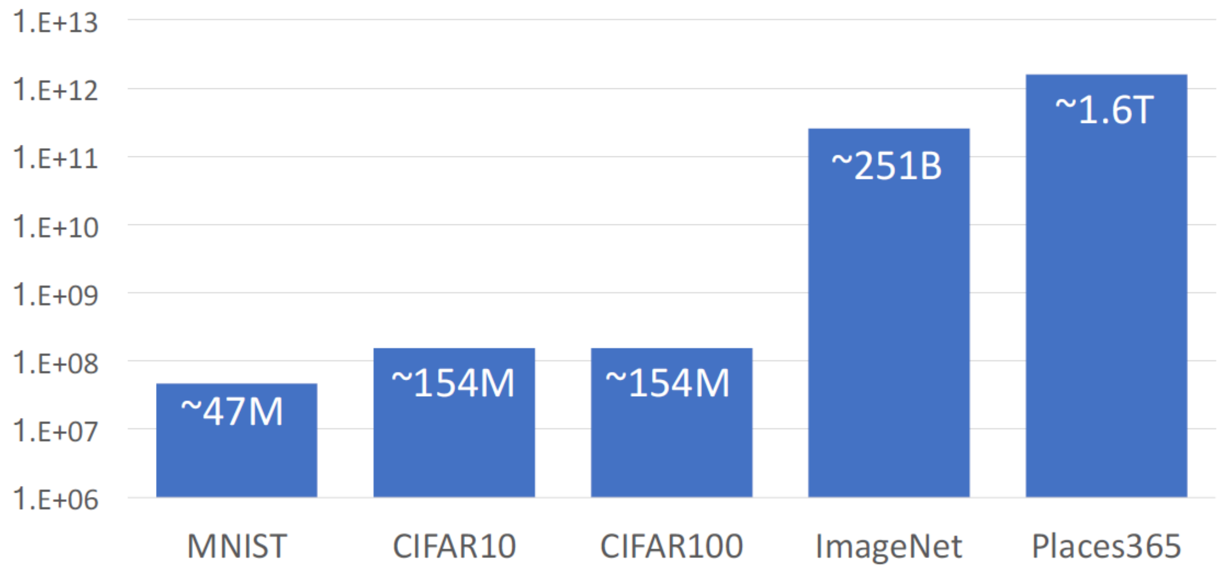
The MobileNet model labeled this as robin, American robin, Turdus migratorius, with a confidence of 0.99.

class name	probability	imagenet class id
Egyptian cat	0.478	285
tabby, tabby cat	0.300	281
tiger cat	0.167	282
remote control, remote	0.016	761
Siamese cat, Siamese	0.008	284

ITXOTIC intel

MobileNet Model

- **Small**, **fast, accurate**. A model that is trained on **ImageNet**.



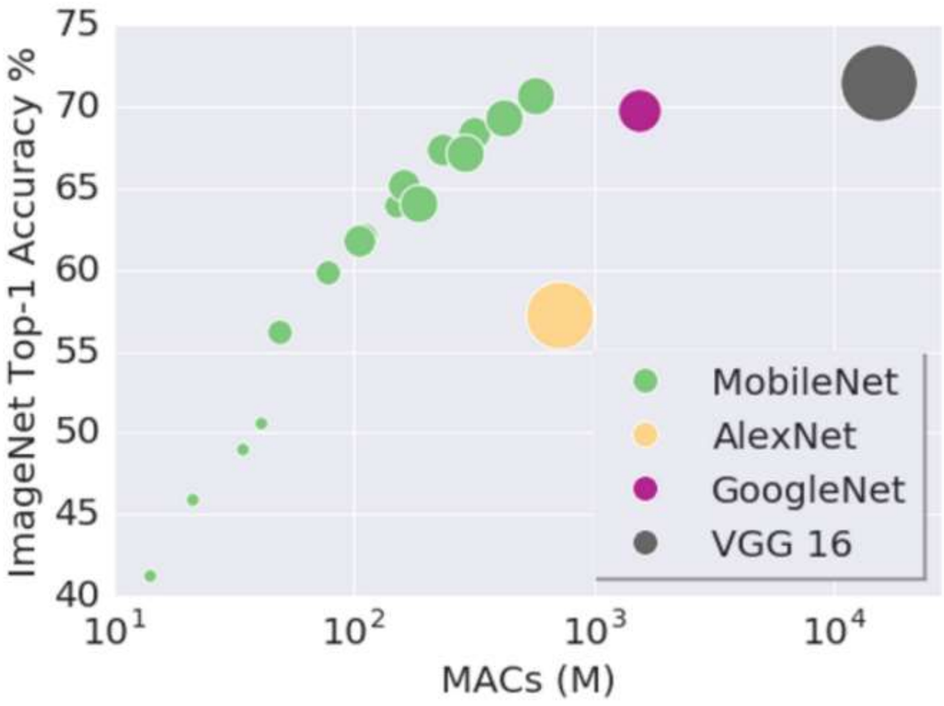


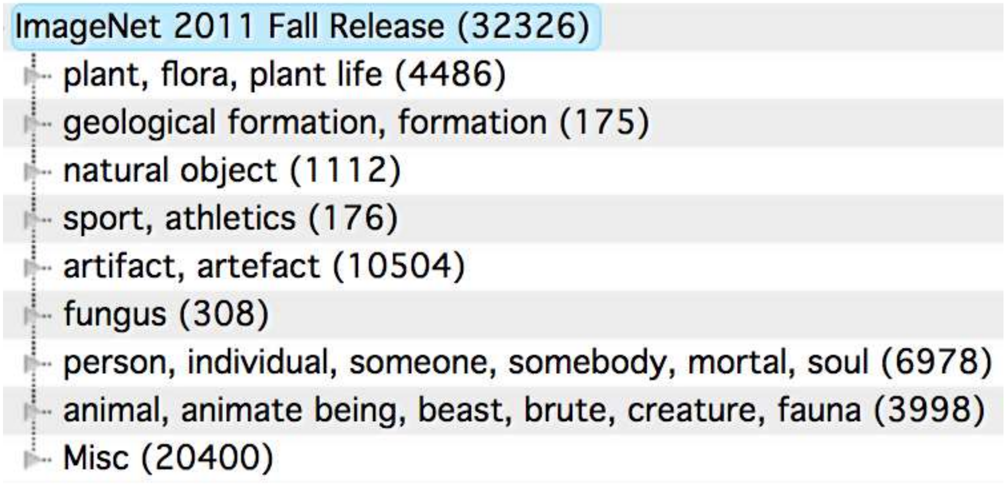
Table 9. Smaller MobileNet Comparison to Popular Models

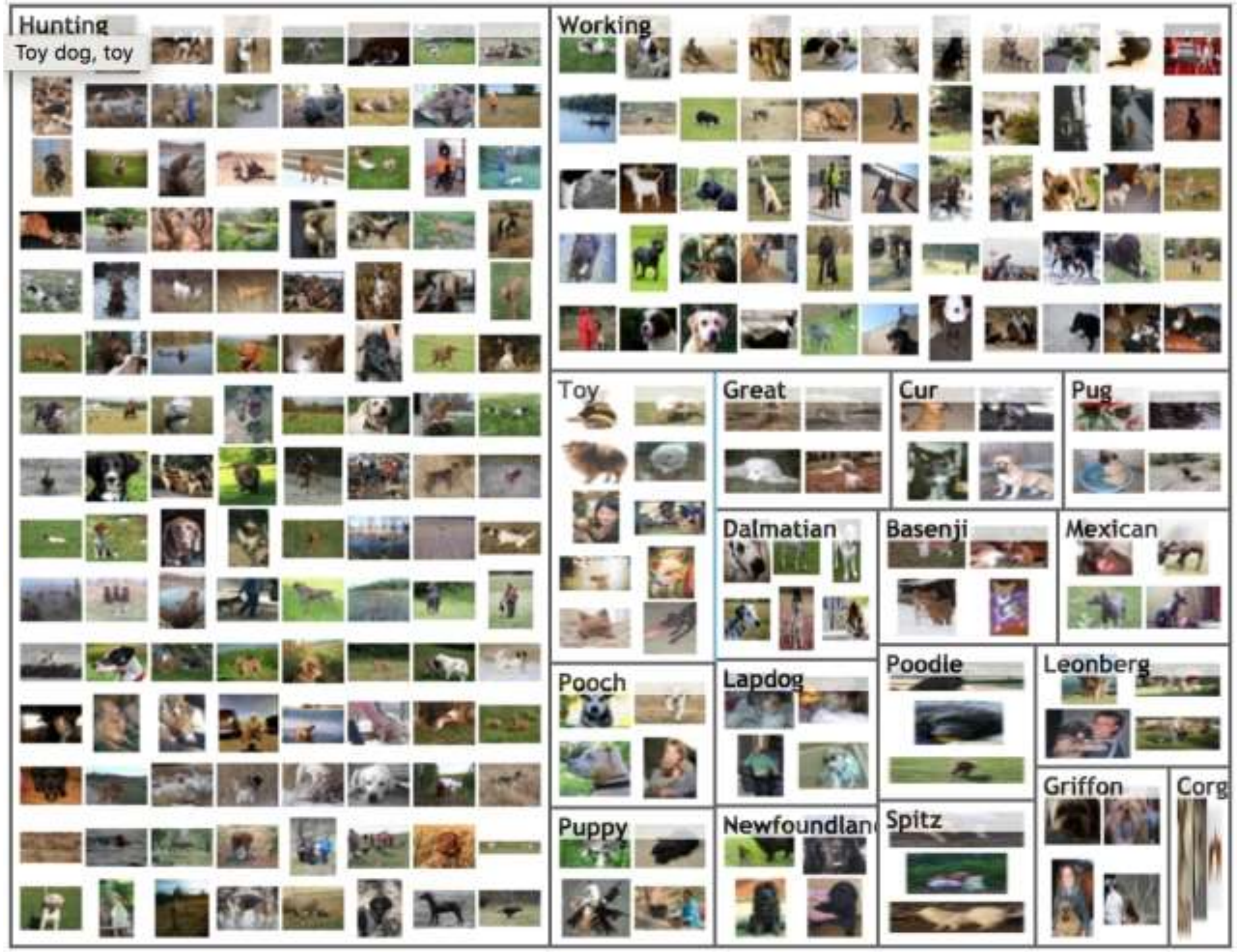
Model	ImageNet Accuracy	Million Mult-Adds	Million Parameters
0.50 MobileNet-160	60.2%	76	1.32
Squeezenet	57.5%	1700	1.25
AlexNet	57.2%	720	60

* Multiply-Accumulates (MACs) measures the number of fused Multiplication and Addition operations

ImageNet Dataset

ImageNet: a **dataset** of **millions** of images with labels for **1000** different classes of **objects**, like dogs, cats, and fruits.





Transfer Learning

Use the knowledge gained while recognizing the **Imagenet classes** could apply when trying to recognize **user customized classes**.

Benefits:

- **Short** training time
- Needs very **little data**
- All in the **browser**.

Image from Webcam → MobileNet(Second to last layer) → NN Classifier → Output

(High-level semantic features of the image)

Introduction to Machine Learning & Deep Learning Concepts