ARTIFICIAL INTELLIGENCE 501

Lesson 6

Deep Learning

Legal Notices and Disclaimers

This presentation is for informational purposes only. INTEL MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS SUMMARY.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. Check with your system manufacturer or retailer or learn more at intel.com.

Learning Objectives

You will be able to:

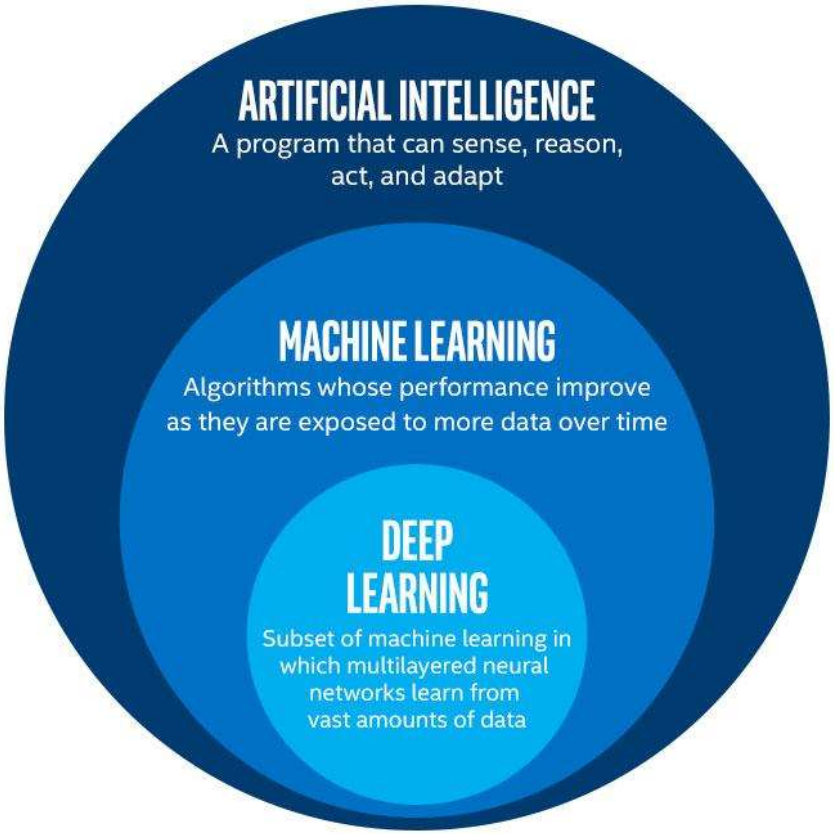
- Identify the types of problems Deep Learning resolves
- Describe the steps in building a neural network model
- Describe a convolutional neural network
- Explain transfer learning and why it's useful
- Identify common Deep Learning architectures

DEEP LEARNING

Deep Learning

“Machine learning that involves using very complicated models called “deep neural networks”." (Intel)

Models determine best representation of original data; in classic machine learning, humans must do this.



Deep Learning Differences

Classic Machine Learning

Step 1: Determine features.
Step 2: Feed them through model.

Feature Detection

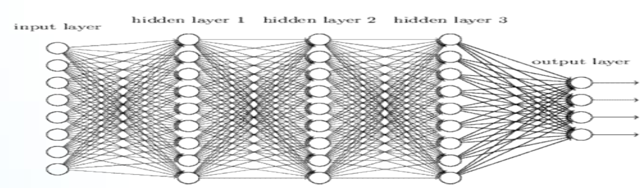
Machine Learning Classifier Algorithm

“Arjun"

Deep Learning

Steps 1 and 2 are combined into 1 step.

Neural Network



“Arjun"

Deep Learning Problem Types

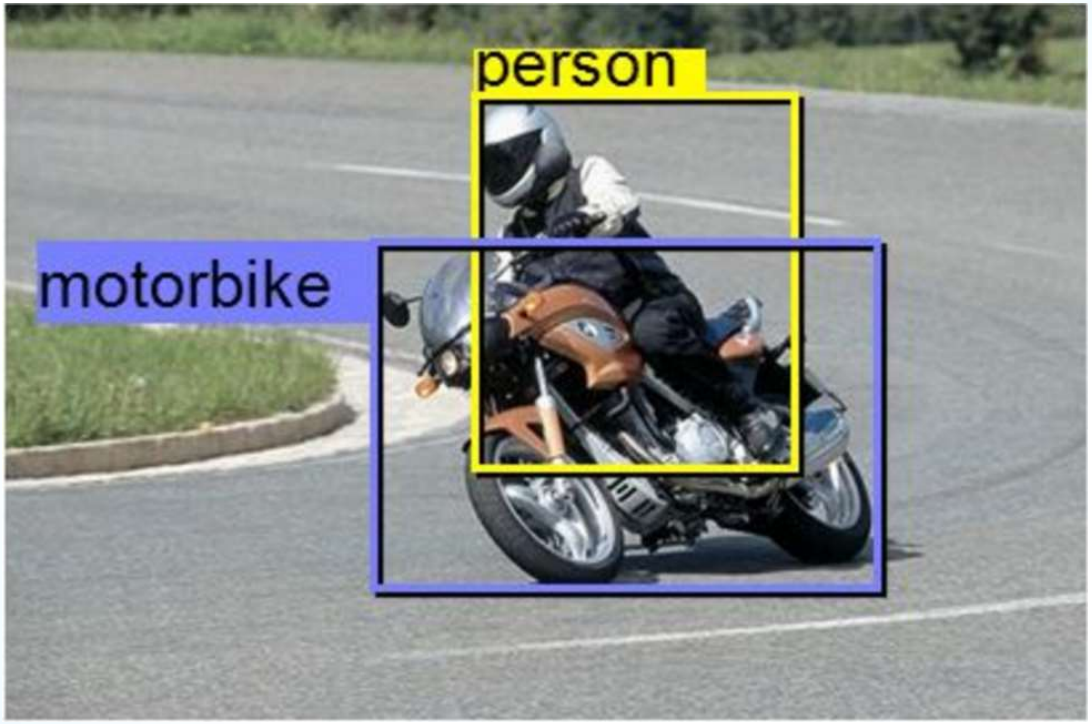
Deep Learning can solve multiple supervised and unsupervised problems.

- The majority of its success has been when working with images, natural language, and audio data.
- Image classification and detection.
- Semantic segmentation.
- Natural language object retrieval.
- Speech recognition and language translation.

Classification and Detection

Detect and label the image

- Person
- Motor Bike



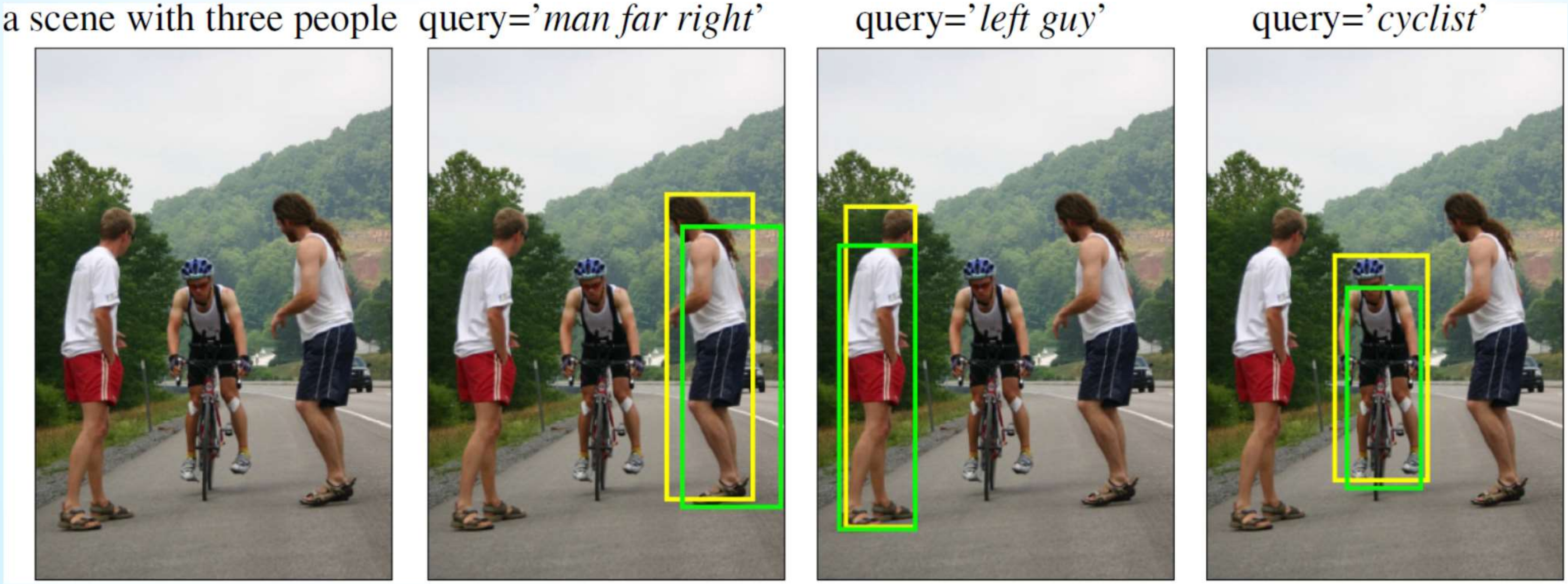
https://people.eecs.berkeley.edu/~jhoffman/talks/lsda-baylearn2014.pdf

Semantic Segmentation

Label every pixel

https://people.eecs.berkeley.edu/~jhoffman/talks/lsda-baylearn2014.pdf

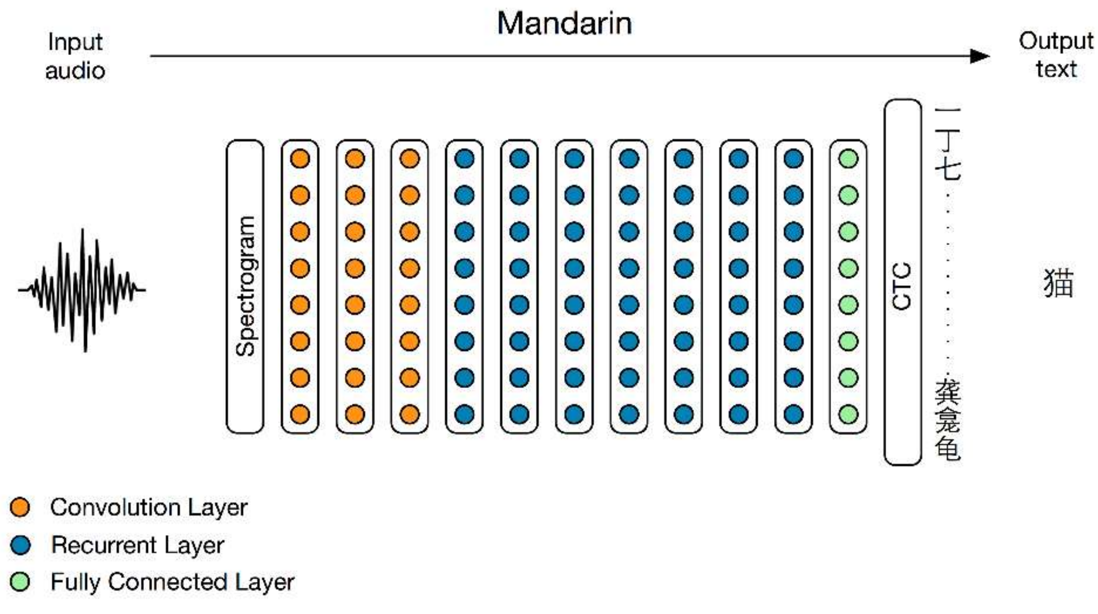
Natural Language Object Retrieval



http://arxiv.org/pdf/1511.04164v3.pdf

Speech Recognition and Language Translation

The same architecture can be used for speech recognition in English, or in Mandarin Chinese.



http://svail.github.io/mandarin/

FULLY CONNECTED NETWORK

Formulating Supervised Learning Tools

For a **supervised learning** problem:

- Collect a labeled dataset (features and target labels).
- Choose the model.
- Choose an evaluation metric:

  "What to use to measure performance."

- Choose an optimization method:[1]

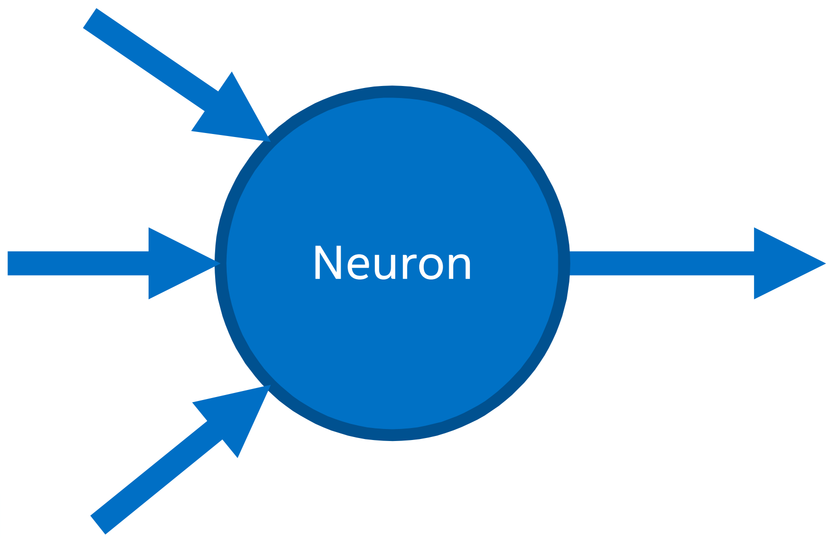
  "How to find the model configuration that gives the best performance."

[1] There are standard methods to use for different models and metrics.

Which Model?

There are many models that represent the problem and make decisions in different ways, each with their own advantages and disadvantages.

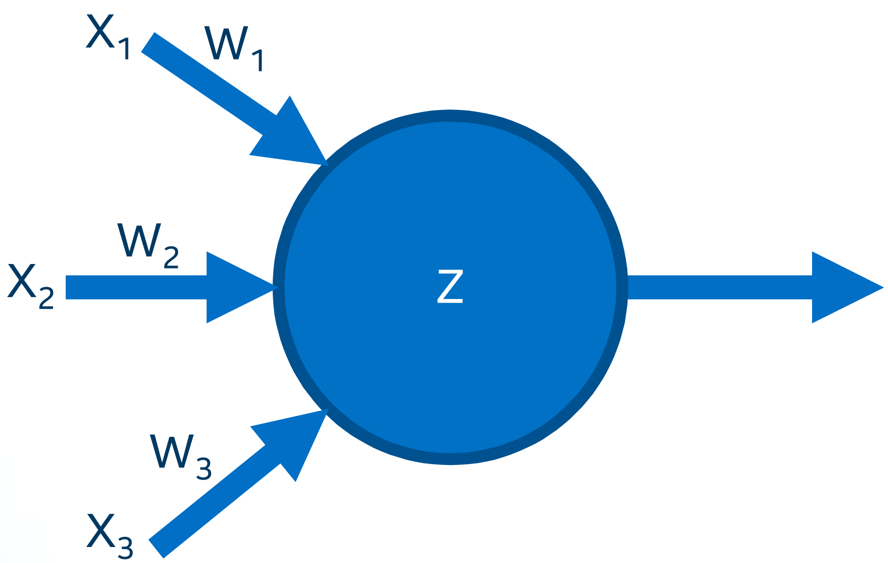
- DL models are biologically inspired.
- The main building block is a **neuron**.



Neuron

A neuron multiplies each feature by a **weight** and then adds these values together.

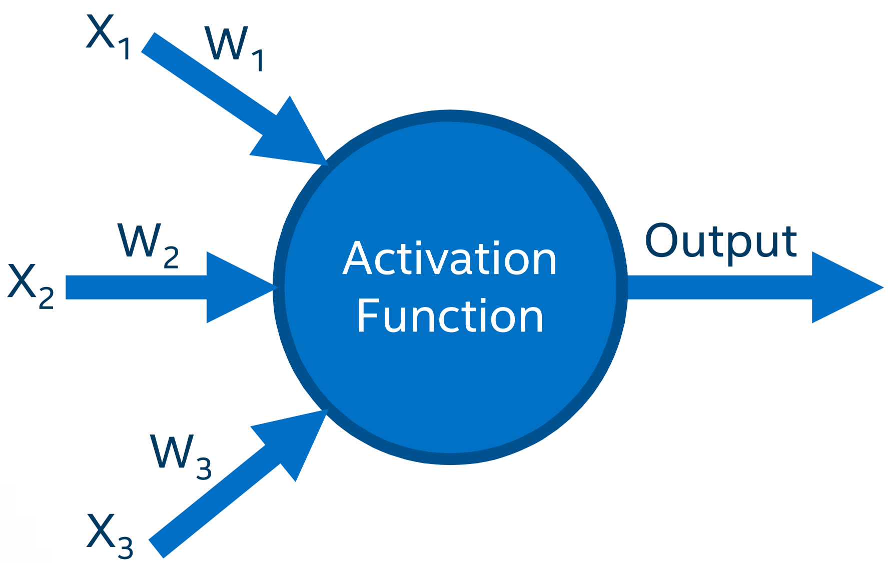
- $Z = X_1W_1 + X_2W_2 + X_3W_3$



Neuron

This value is then put through a function called the **activation function.**

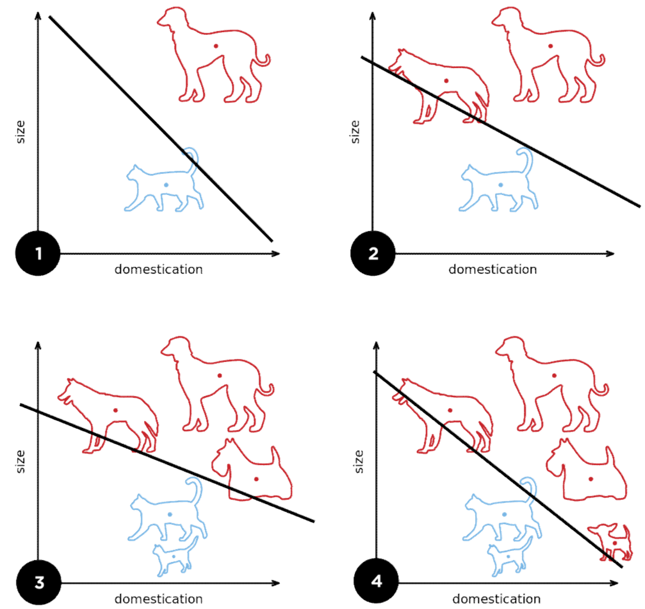
- There are several activation functions that can be used.
- The output of the neuron is the output of the activation function.



Perceptron

A neuron with a simple activation function can solve **linearly separable** problems.

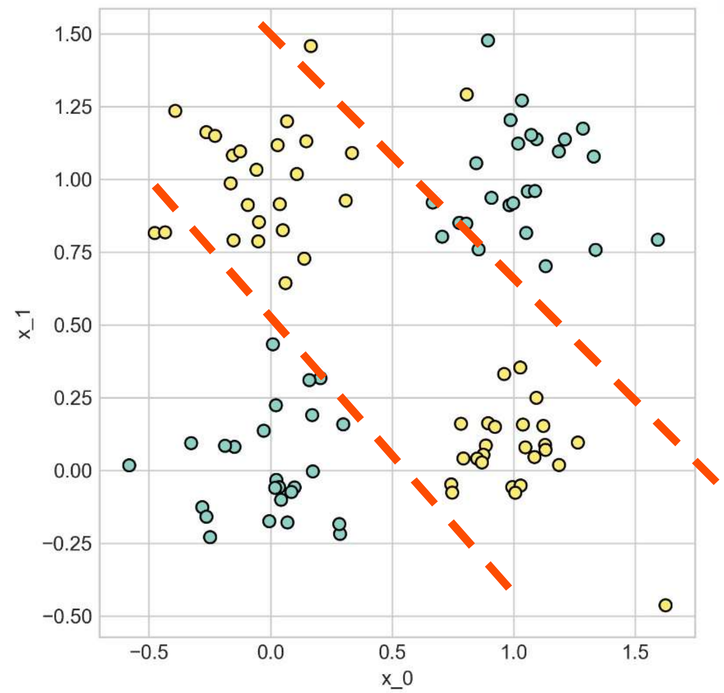
- These are problems where the different classes can be separated by a line.
- **The Perceptron**: one of the earliest neural network models that used neurons with simple activation functions.



Perceptron

Problems where the labels cannot be separated by a single line are not solvable by a single neuron.

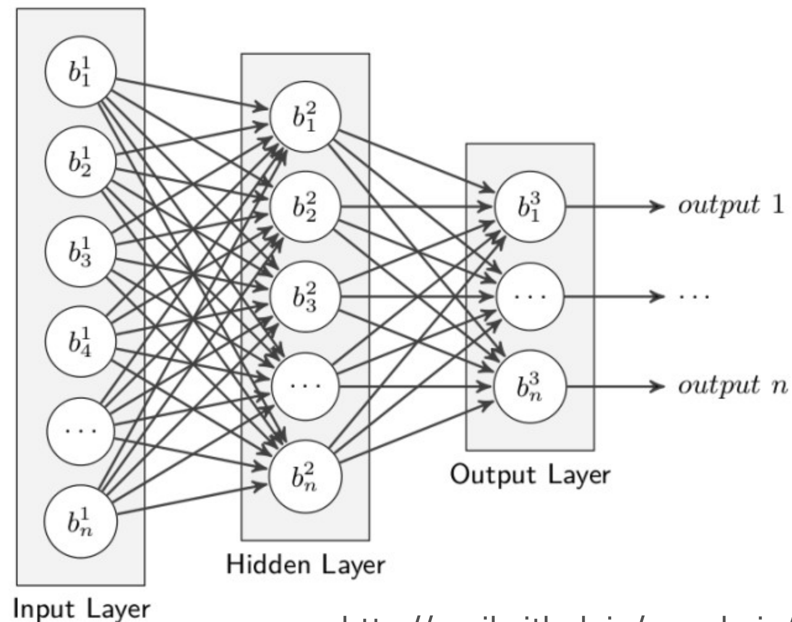
- This is a major limitation, and one of the reasons for the first AI winter, that was discussed in lesson 1.



Fully Connected Network

More complicated problems can be solved by connecting multiple neurons together and using more complicated activation functions.

- Organized into **layers** of neurons.
- Each neuron is connected to every neuron in the previous layer.
- Each layer transforms the output of the previous layer and then passes it on to the next.
- Every connection has a separate weight.

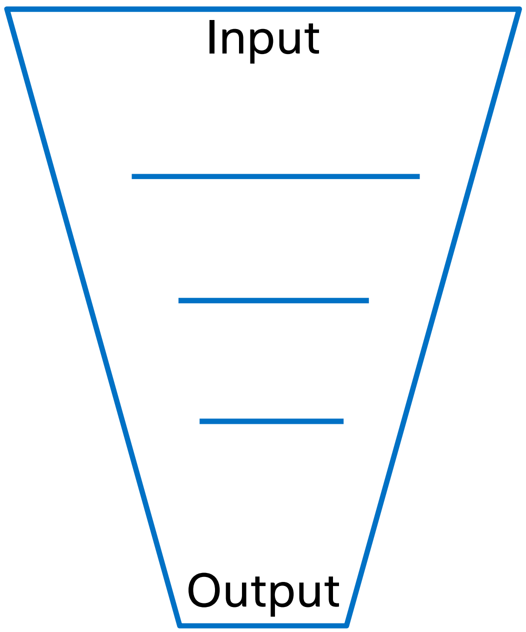


http://svail.github.io/mandarin/

Deep Learning

Deep Learning refers to when many layers are used to build **deep networks**.

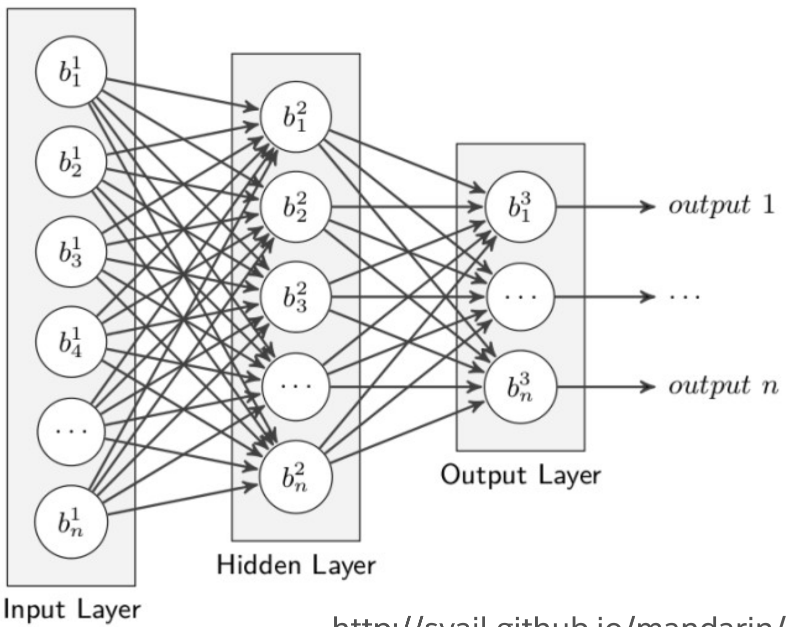
- State-of-the-art models use hundreds of layers.
- Deep layers tend to decrease in width.
- Successive layers transform inputs with two effects:
- **Compression**: each layer is asked to summarize the input in a way that best serves the task.
- **Extraction**: the model succeeds when each layer extracts task-relevant information.



Steps in Building a Fully Connected Network

To build a fully connected network a user needs to:

- Define the network architecture.
- How many layers and neurons?
- Define what activation function to use for each neuron.
- Define an evaluation metric.
- The values for the weights are learned during model training.



http://svail.github.io/mandarin/

Evaluation Metric

The metric used will depend on the problem being solved. Some examples include:

- Regression
  - Mean Squared Error
- Classification
  - Categorical Cross-Entropy
- Multi-Label classification
  - Binary Cross-Entropy

Fully Connected Network Problems

Not optimal for detecting features.

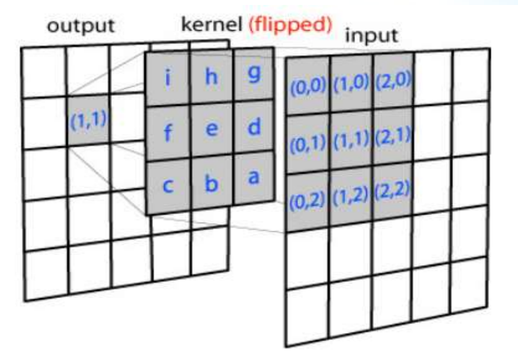
- Computationally intensive – heavy memory usage.

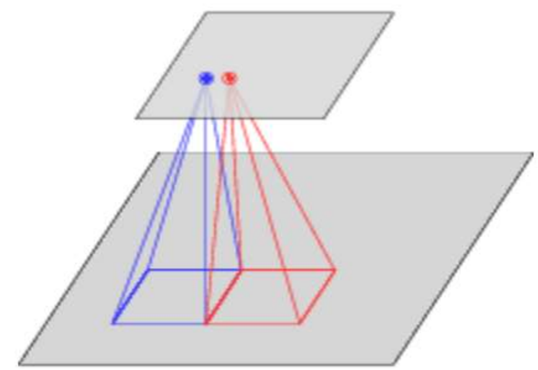
CONVOLUTIONAL NEURAL NETWORK

Convolutional Neural Network

Convolutional neural networks reduce the required computation and are good for detecting features.

- Each neuron is connected to a small set of nearby neurons in the previous layer.
- The same set of weights are used for each neuron.
- Ideal for spatial feature recognition.
    - Example: image recognition
- Cheaper on resources due to fewer connections.





http://svail.github.io/mandarin/

Convolutions as Feature Detectors

Convolutions can be thought of as “local feature detectors”.

Vertical Line Detector

-1	1	-1
-1	1	-1
-1	1	-1

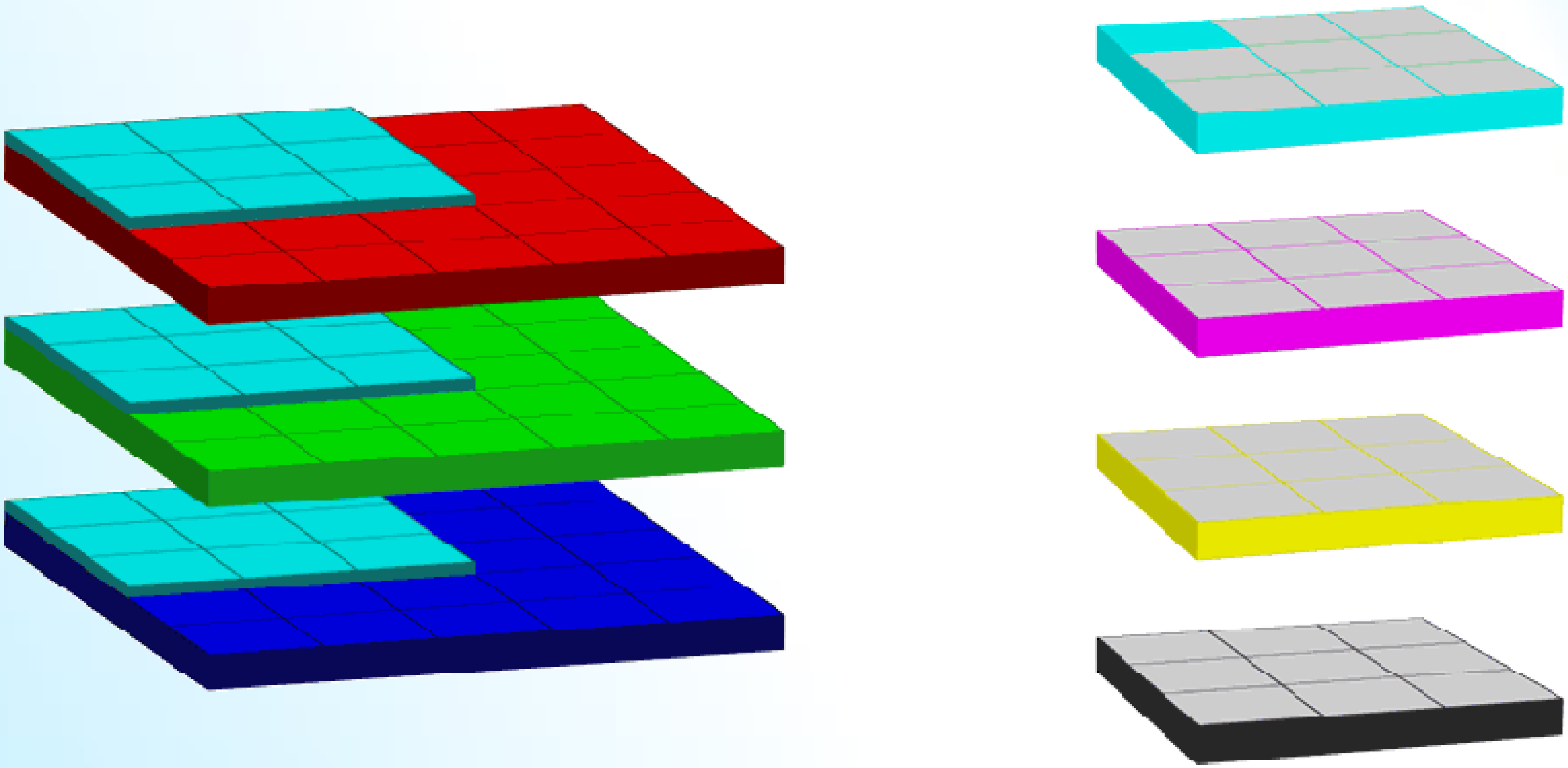
Horizontal Line Detector

-1	-1	-1
1	1	1
-1	-1	-1

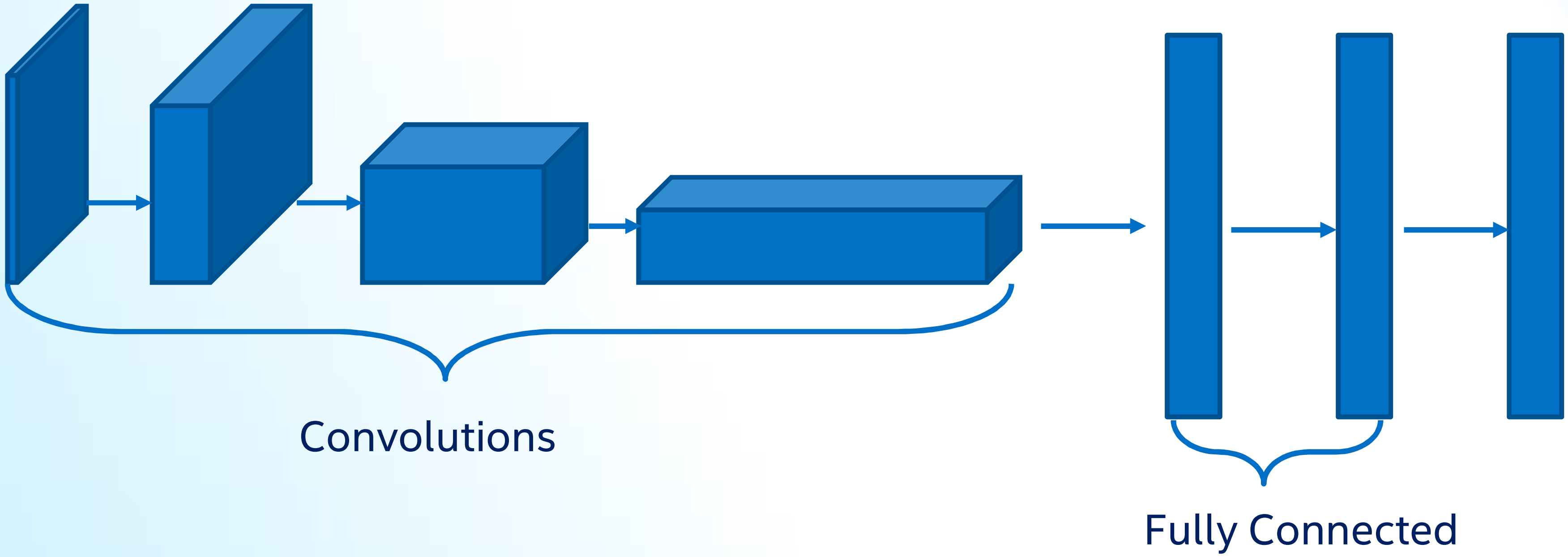
Corner Detector

-1	-1	-1
-1	1	1
-1	1	1

Convolutions



Convolutional Neural Network



TRANSFER LEARNING

Transfer Learning

There are difficulties with building convolutional neural networks.

- They require huge datasets.
- There is a large amount of required computation.
- A large amount of time is spent experimenting to get the hyper-parameters correct.
  - It would be very difficult to train a famous, competition-winning model from scratch

Transfer Learning

We can take advantage of existing DL models.

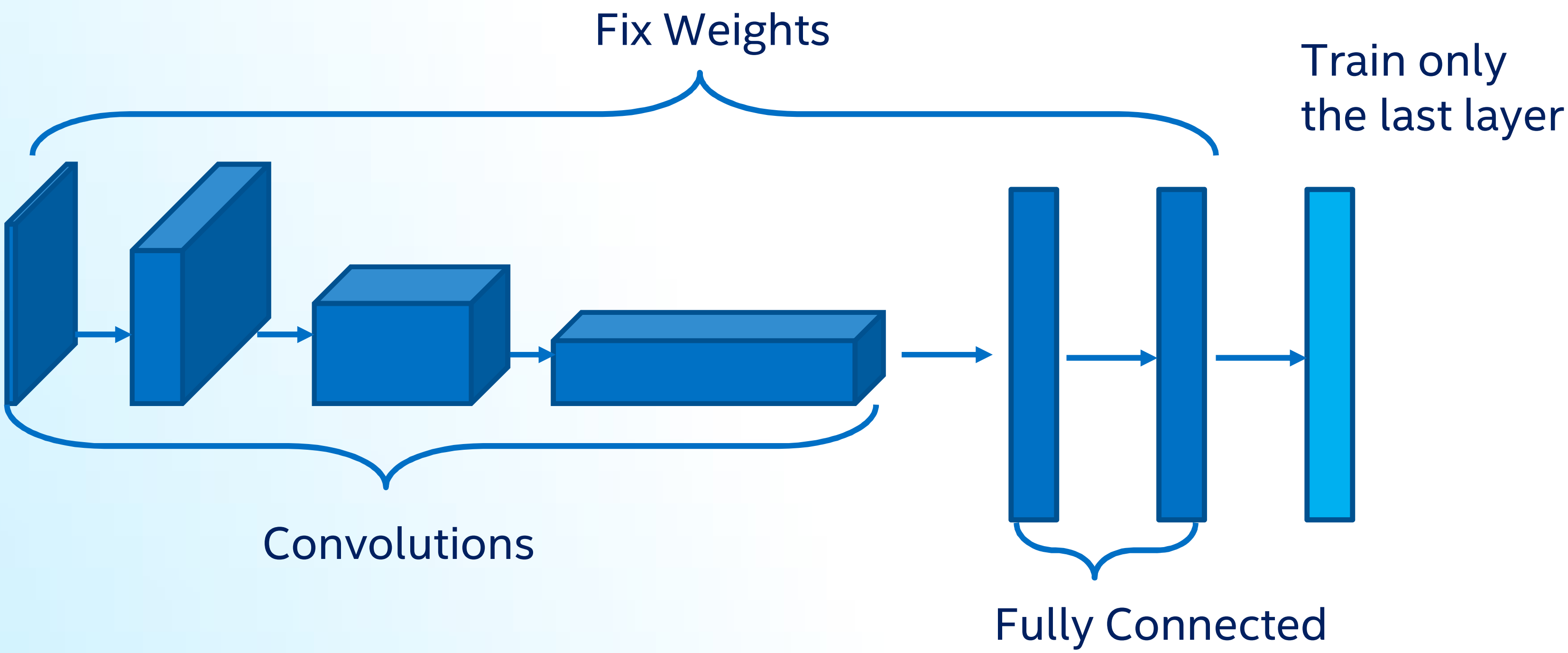
- **Early layers** in a neural network are the hardest (slowest) to train.
- These ”primitive” features should be general across many image classification tasks.
- **Later layers** in the network capture features that are more particular to the specific image classification problem.
- Later layers are easier (quicker) to train since adjusting their weights has a more immediate impact on the final result.

Transfer Learning

Keeping the early layers of a pre-trained network, and re-training the later layers for a specific application, is called ***transfer learning***.

- Results of the training are weights (numbers) that are easy to store.
- Using pre-trained layers reduces the amount of required data.

Convolutional Neural Network



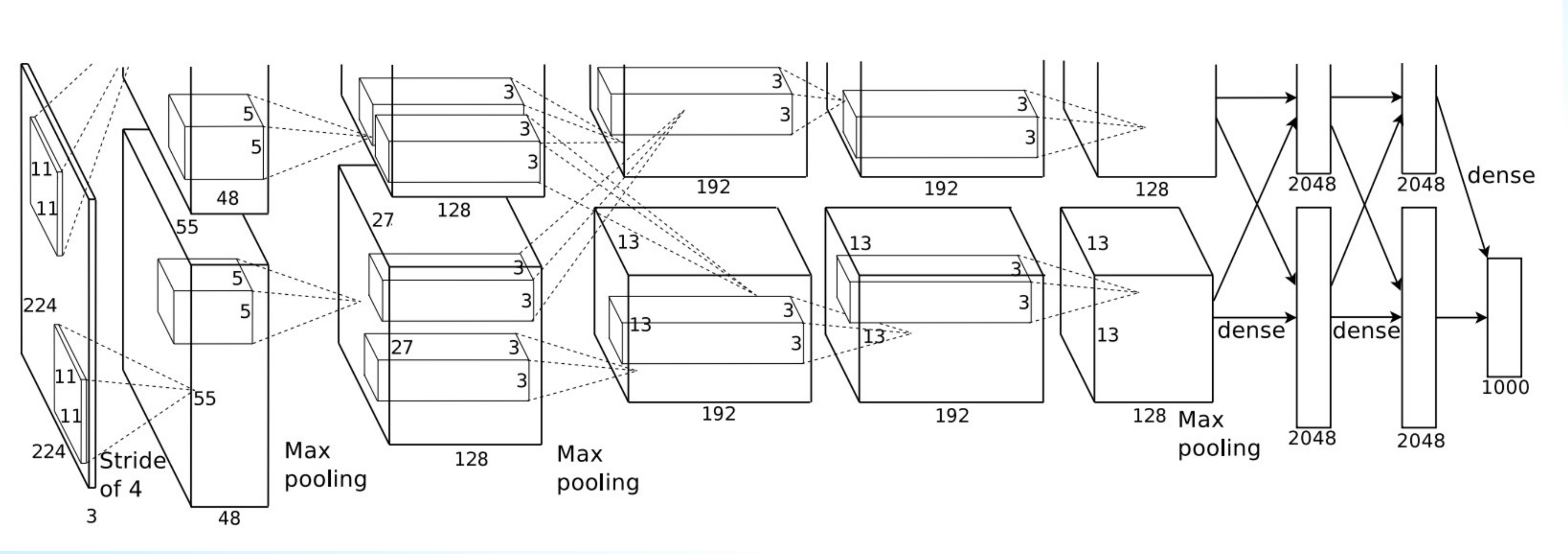
Transfer Learning Options

The additional training of a pre-trained network on a specific new dataset is referred to as “fine-tuning”.

- Choose “how much” and “how far back” to fine-tune.
- Should I train just the very last layer, or go back a few layers?
- Re-train the entire network (from the starting-point of the existing network)?

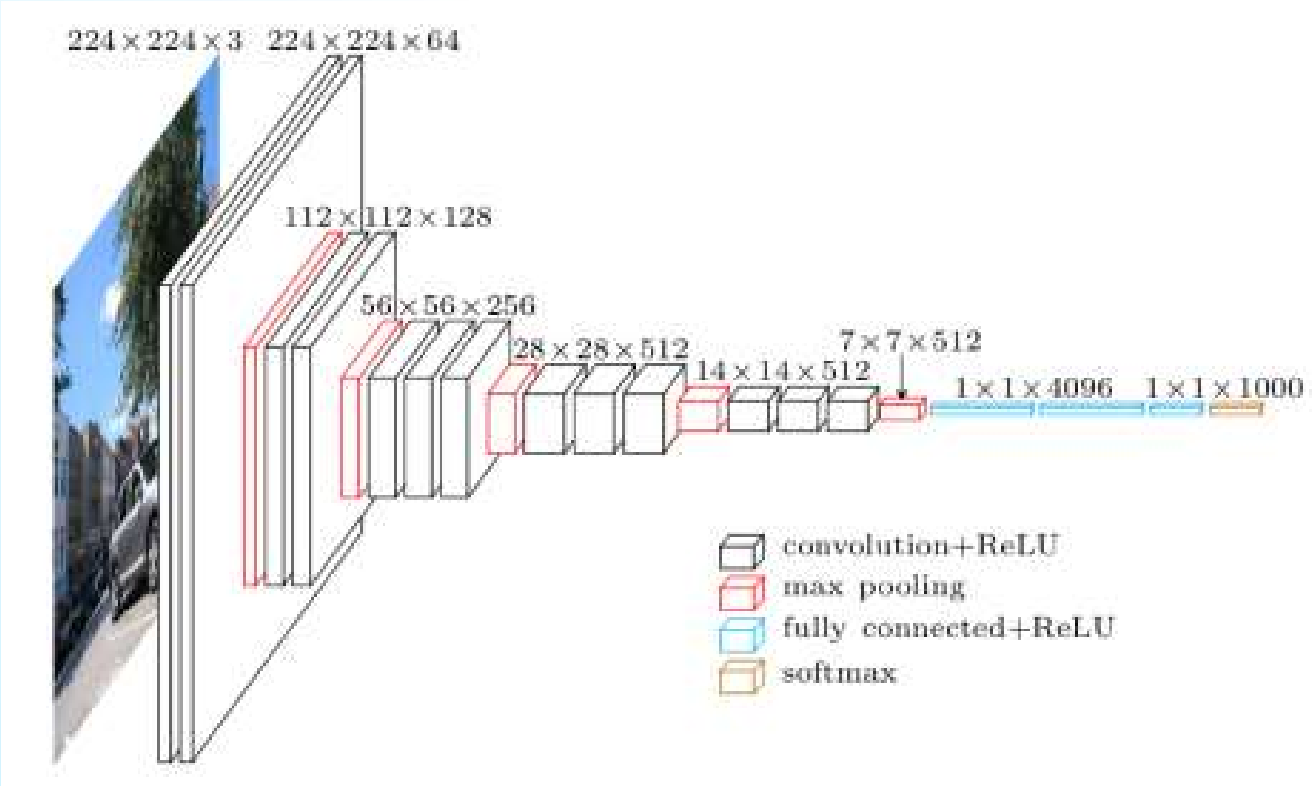
COMMON ARCHITECTURES

AlexNet



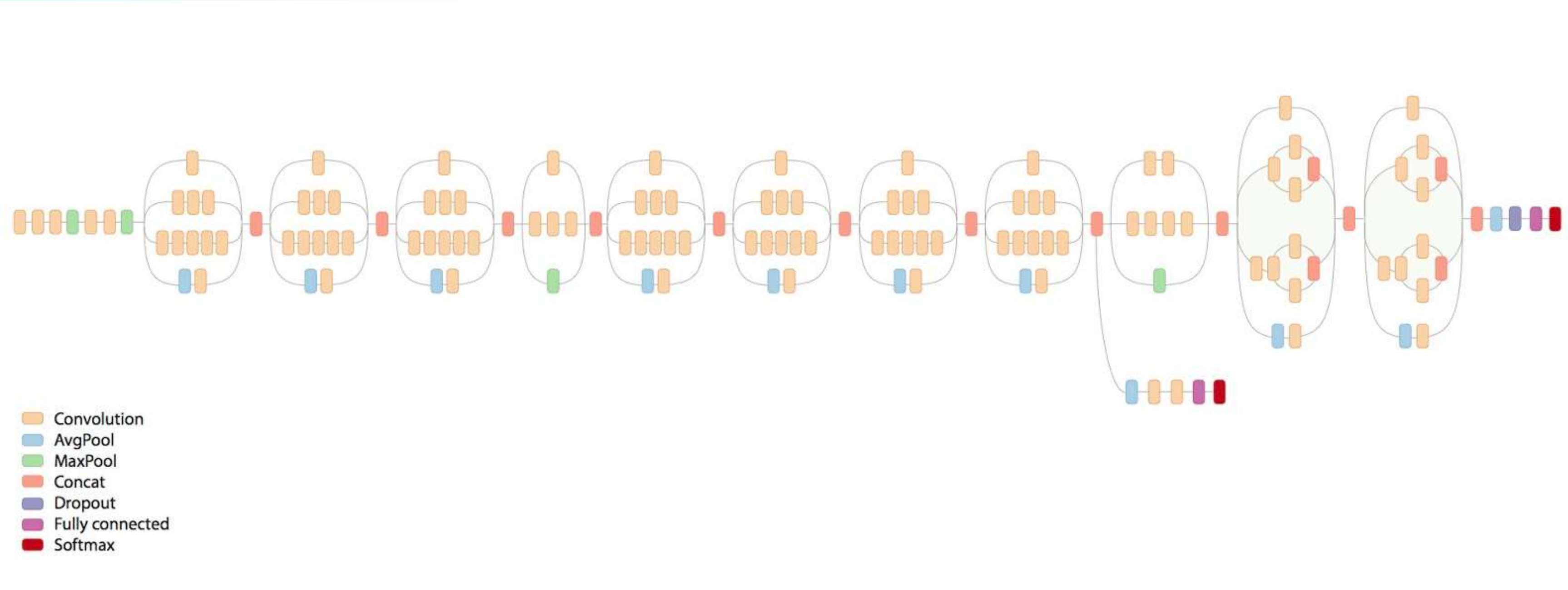
224
224
3
11
11
11
11
Stride of 4
55
55
48
48
5
5
5
5
Max pooling
27
27
128
128
3
3
3
3
3
3
Max pooling
13
13
192
192
13
13
192
192
13
13
128
128
Max pooling
dense
dense
dense
2048
2048
2048
2048
1000

VGG 16



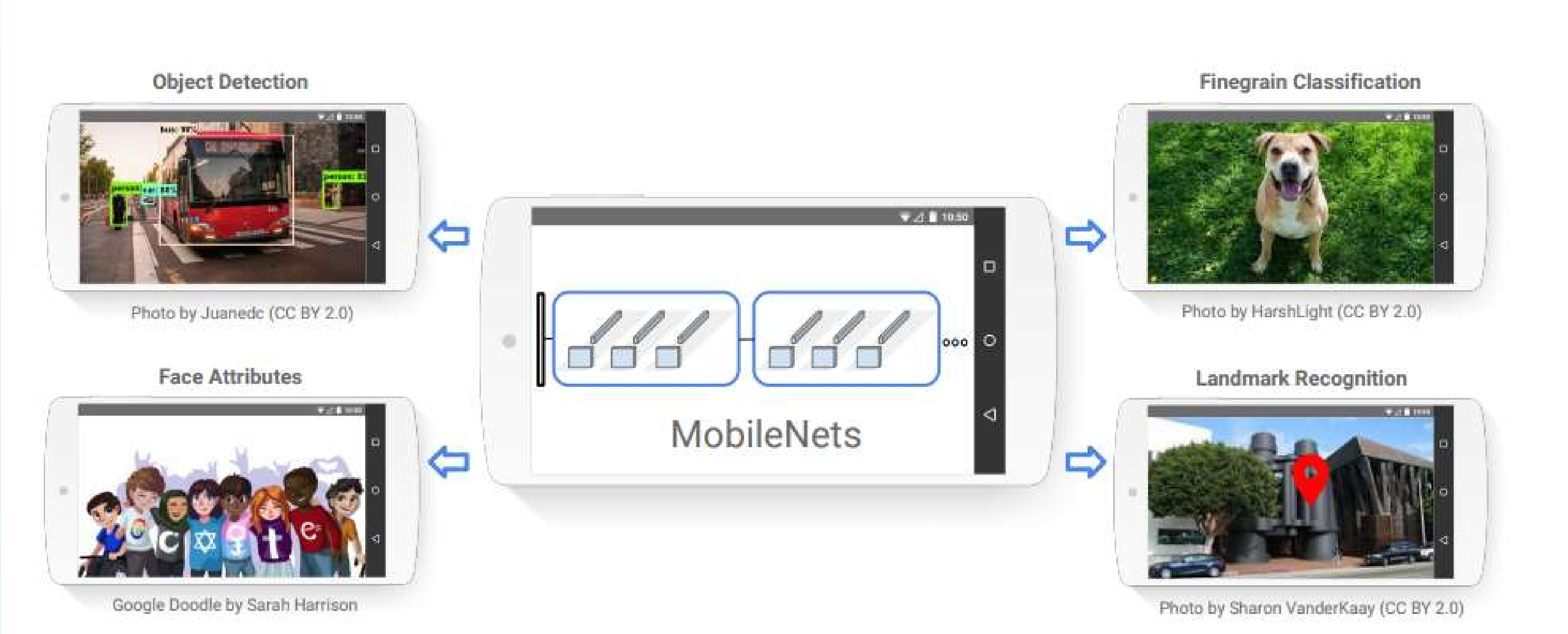
224 × 224 × 3
224 × 224 × 64
112 × 112 × 128
56 × 56 × 256
28 × 28 × 512
14 × 14 × 512
7 × 7 × 512
1 × 1 × 4096
1 × 1 × 1000
convolution+ReLU
max pooling
fully connected+ReLU
softmax

Inception


Convolution
AvgPool
MaxPool
Concat
Dropout
Fully connected
Softmax

MobileNets

Efficient models for mobile and embedded vision applications.



MobileNet models can be applied to various recognition tasks for efficient device intelligence.

Learning Objectives

In this session we worked to:

- Identify the types of problems Deep Learning resolves.
- Describe the steps in building a neural network model.
- Describe a convolutional neural network.
- Explain transfer learning and why it’s useful.
- Identify common Deep Learning architectures.

experience
what's inside™