

# Practical Applications of Deep Learning

*A hands-on MATLAB workshop*



**Pitambar Dayal**  
**Abhijit Bhattacharjee**

**Product Marketing Manager**  
**Application Engineer**

↻ Internet

Retweeted

**Computer Facts**

@computerfact



concerned parent: if all your friends  
jumped off a bridge would you  
follow them?

2:20 PM · Mar 15, 2018

# Deep Learning Demo

## Image Classification

# Agenda

Introduction



**Exercise 1:** Deep learning in 5 lines of code

Deep learning fundamentals



**Exercises 2 and 3:** Exploring pretrained networks/Classifying handwritten digits

Transfer learning



**Exercise 4:** Creating a food classifier

Deploying deep neural networks

Conclusion

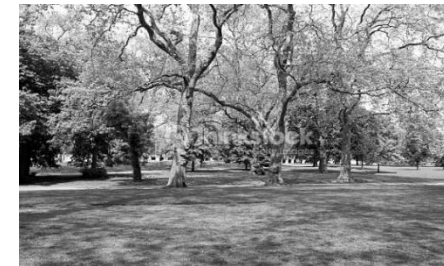
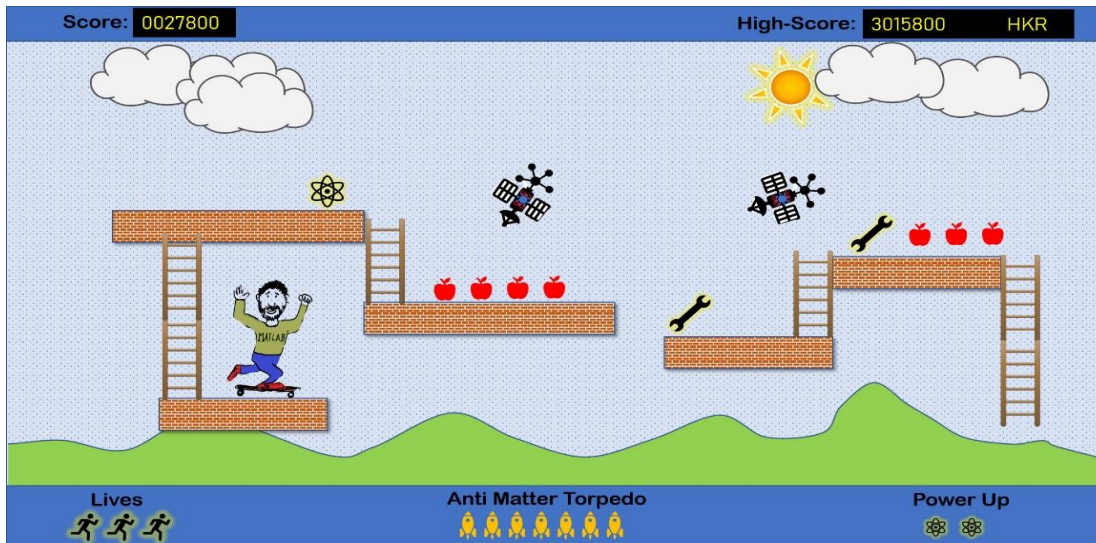


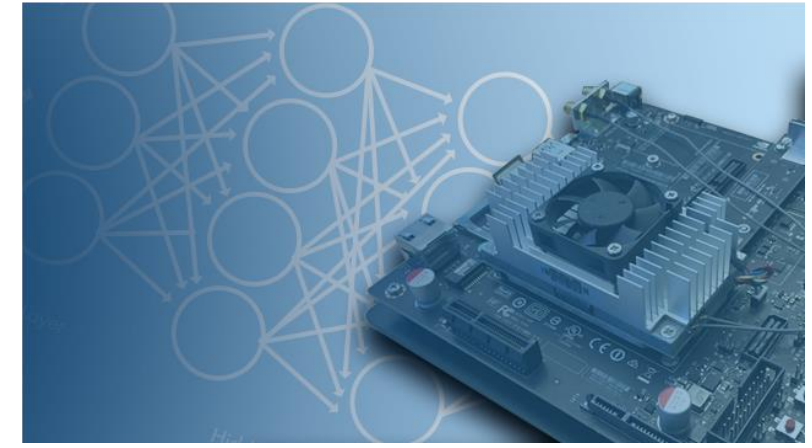
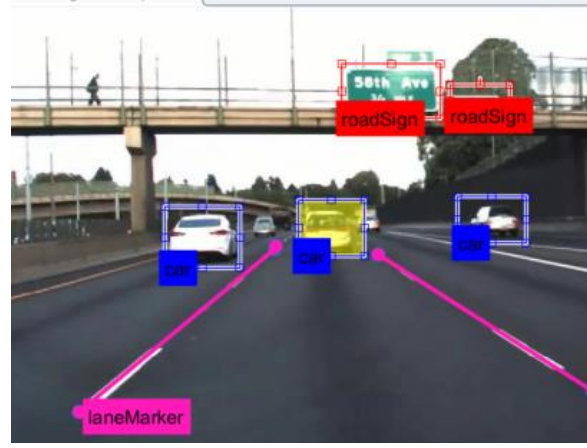
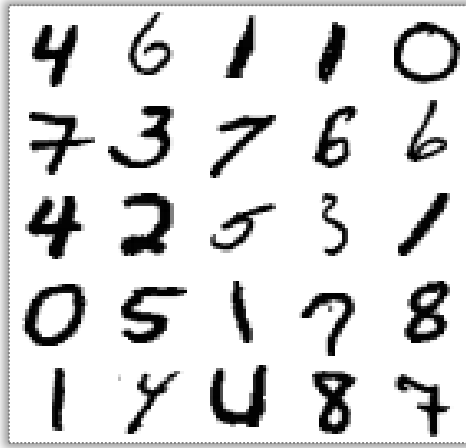
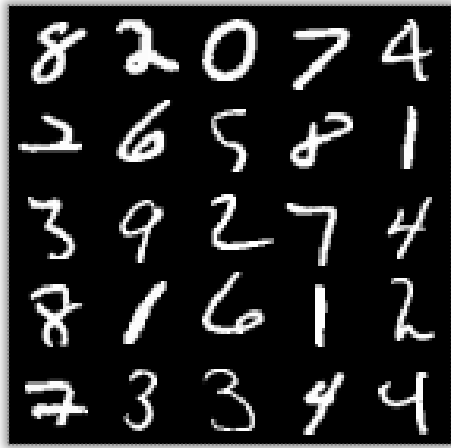
# Deep Learning Applications

Voice assistants (speech to text)

Teaching character to beat video game

Automatically coloring black and white images





# What is Deep Learning?



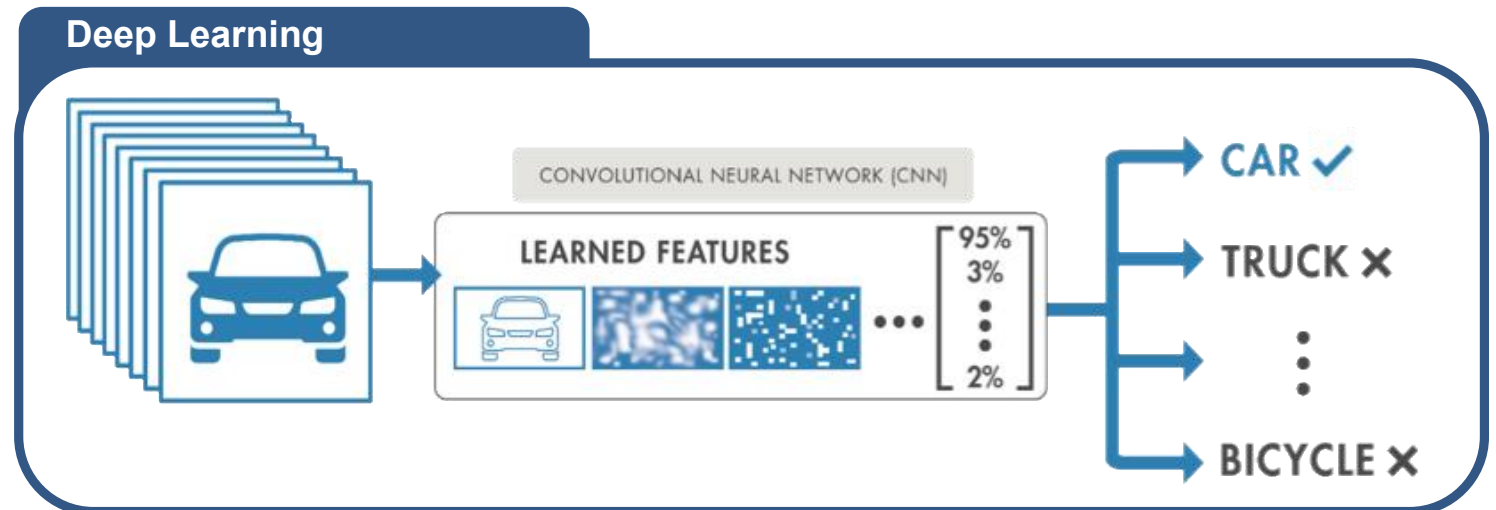
12 40.0%	0 0.0%	100% 0.0%
0 0.0%	18 60.0%	100% 0.0%
100% 0.0%	100% 0.0%	100% 0.0%

# What is Deep Learning?

- Subset of machine learning with **automatic feature extraction**
  - Learns features and tasks directly from data
  - More Data = better model

**Machine  
Learning**

**Deep  
Learning**





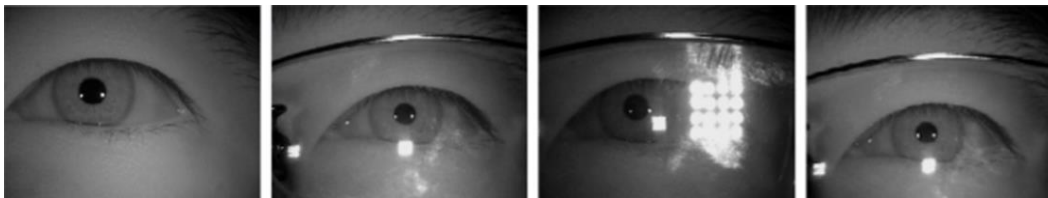
# DL Applications include **image classification**, speech recognition, autonomous driving, etc.



*Detection of cars and road in autonomous driving systems*



*Rain Detection and Removal<sup>1</sup>*

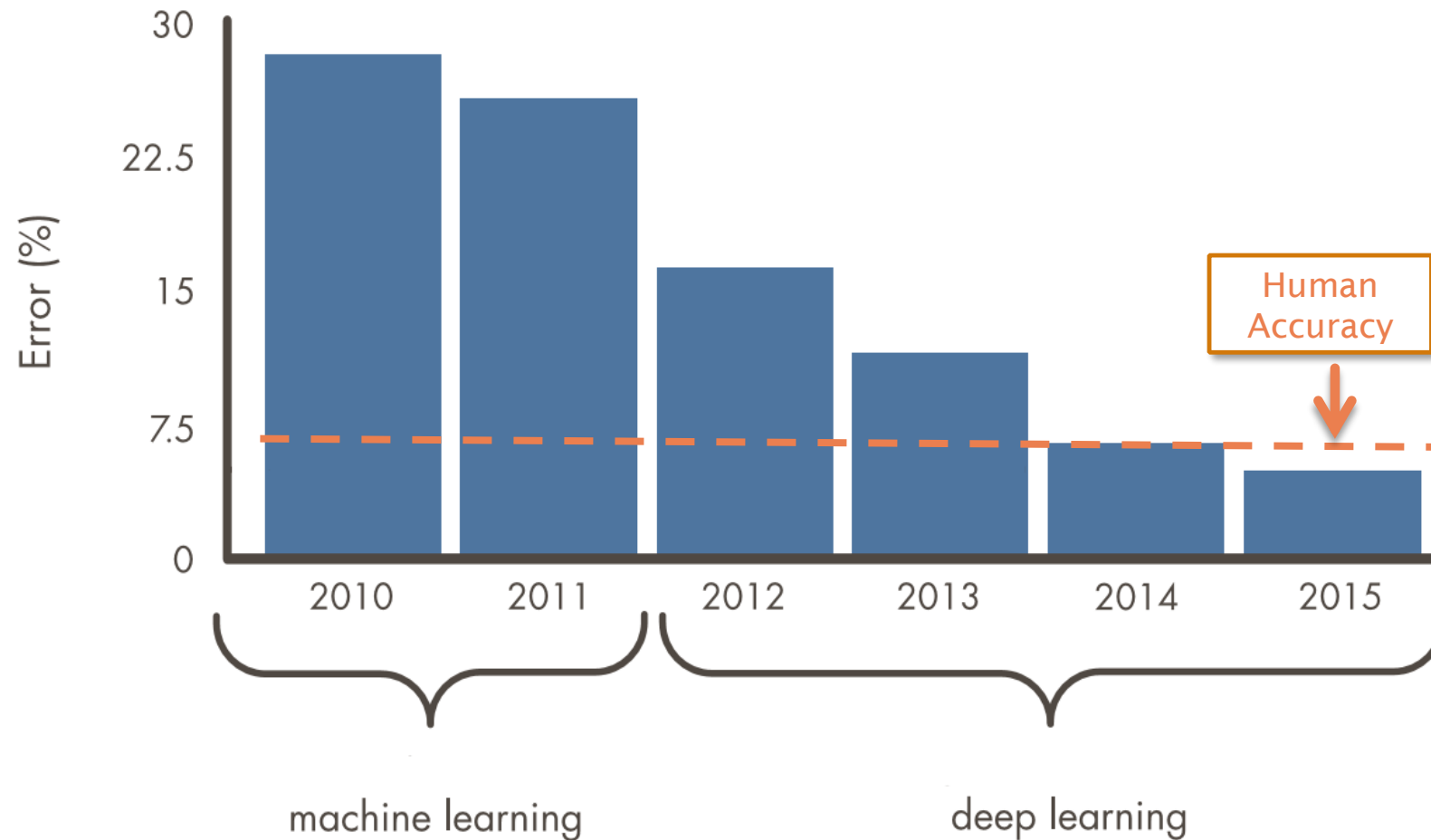


*Iris Recognition – 99.4% accuracy<sup>2</sup>*

1. "Deep Joint Rain Detection and Removal from a Single Image" Wenhan Yang, Robby T. Tan, Jiashi Feng, Jiaying Liu, Zongming Guo, and Shuicheng Yan
2. Source: An experimental study of deep convolutional features for iris recognition Signal Processing in Medicine and Biology Symposium (SPMB), 2016 IEEE Shervin Minaee ; Amirali Abdolrashidiy ; Yao Wang; An experimental study of deep convolutional features for iris recognition

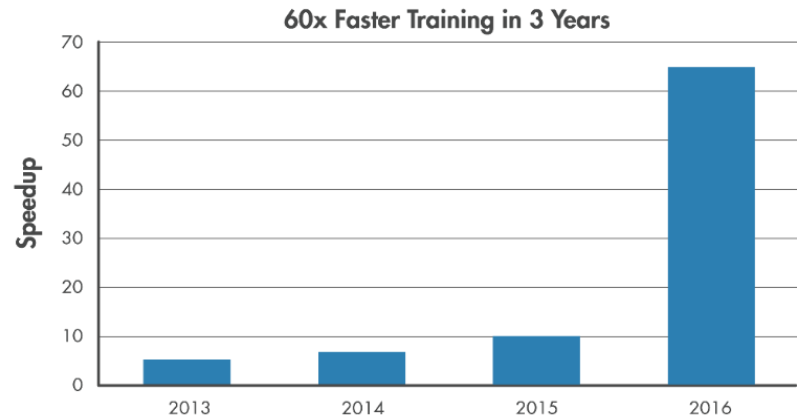


# Deep Learning Models can Surpass Human Accuracy.

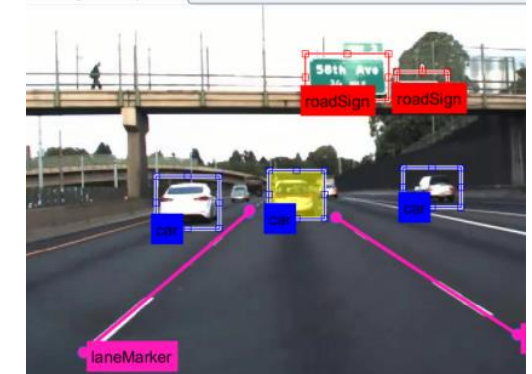
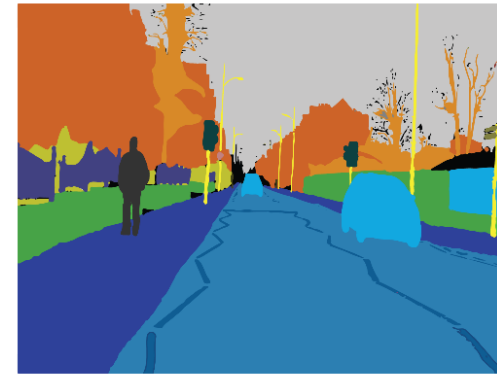


# Deep Learning Enablers

## Increased GPU acceleration



## Labeled public datasets



## World-class models

**AlexNet**

PRETRAINED MODEL

**VGG-16/19**

PRETRAINED MODEL

**Caffe**

MODEL IMPORTER

**ResNet**

PRETRAINED MODEL

**GoogLeNet**

PRETRAINED MODEL

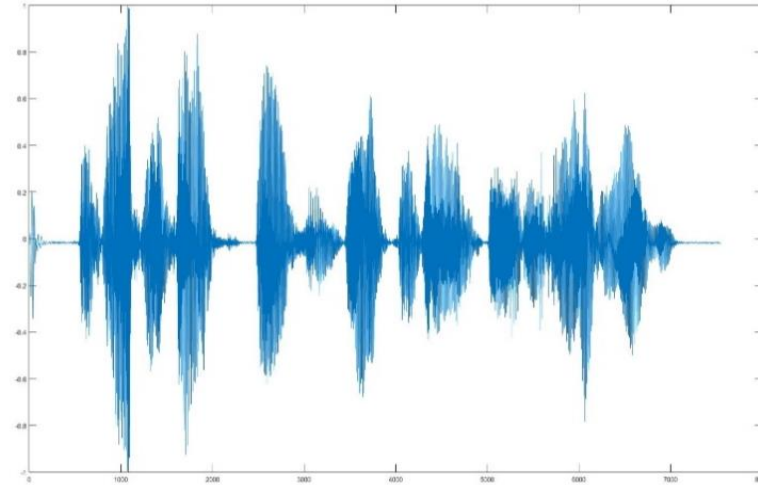
**TensorFlow-Keras**

MODEL IMPORTER

# Image



# Signal



# Numeric

AgeCat	WeightQ	GroupCount	mean_BloodPressure	
Under 30	Q1	6	123.17	79.667
Under 30	Q2	3	120.33	79.667
Under 30	Q3	2	127.5	86.5
Under 30	Q4	4	122	78
30-39	Q1	12	121.75	81.75
30-39	Q2	9	119.56	82.556
30-39	Q3	9	121	83.222
30-39	Q4	11	125.55	87.273
Over 40	Q1	7	122.14	84.714
Over 40	Q2	13	123.38	79.385
Over 40	Q3	14	123.07	84.643
Over 40	Q4	10	124.6	85.1

# Text



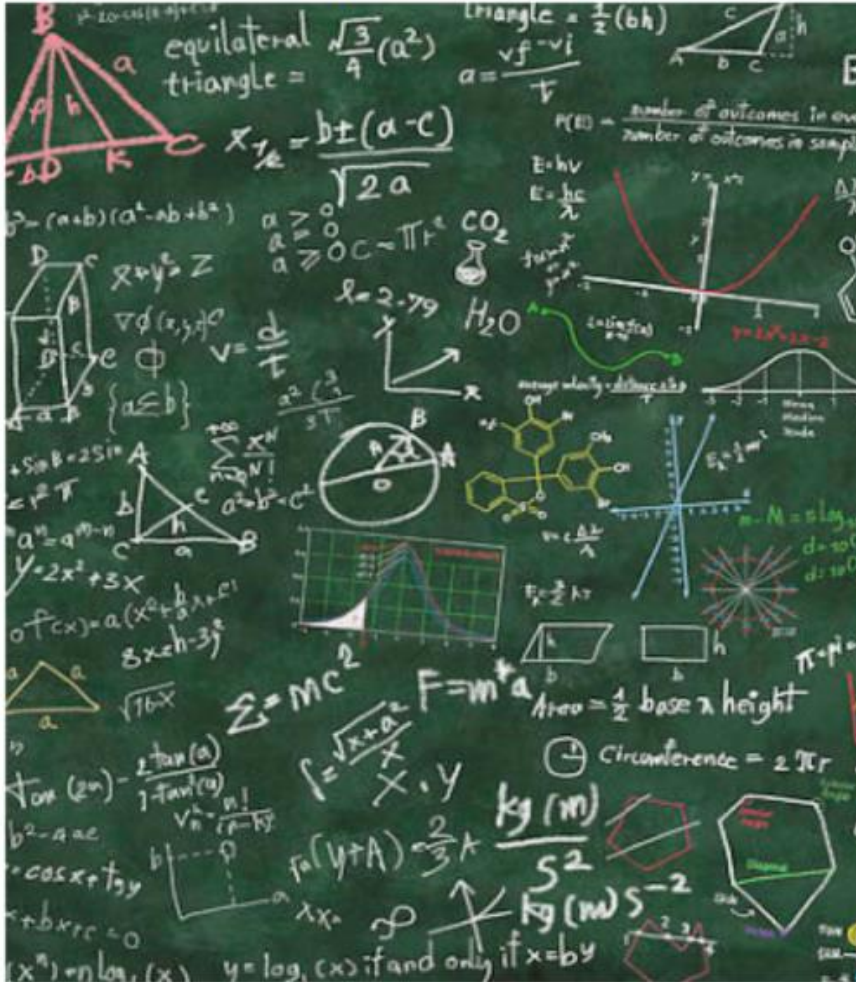


# Let's try it out!

Open: ***DeepLearningIn5Lines.mlx***  
in folder 01-DeepLearningIn5Lines

# Deep learning is not complicated. It can be easy!

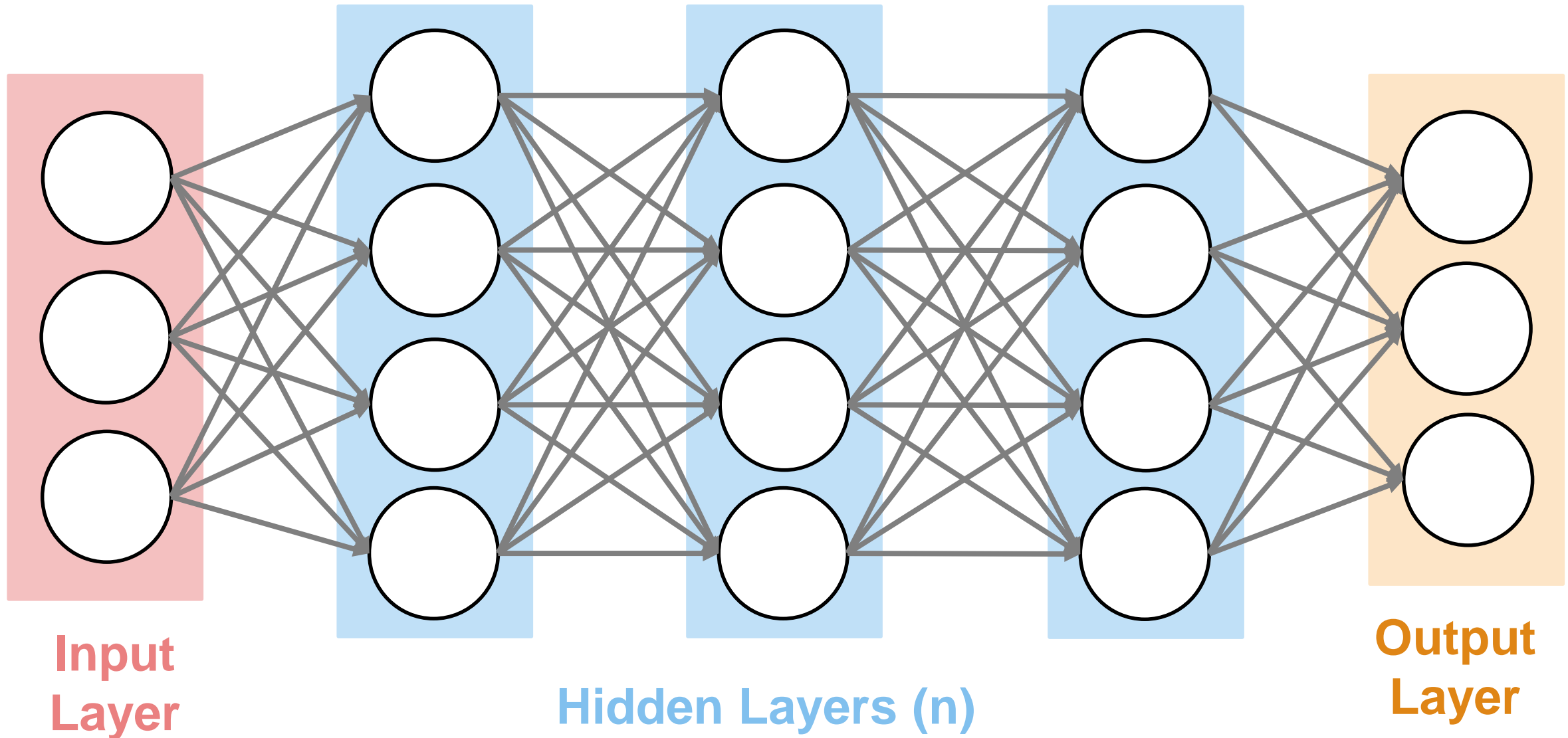
## Expectation



## Reality



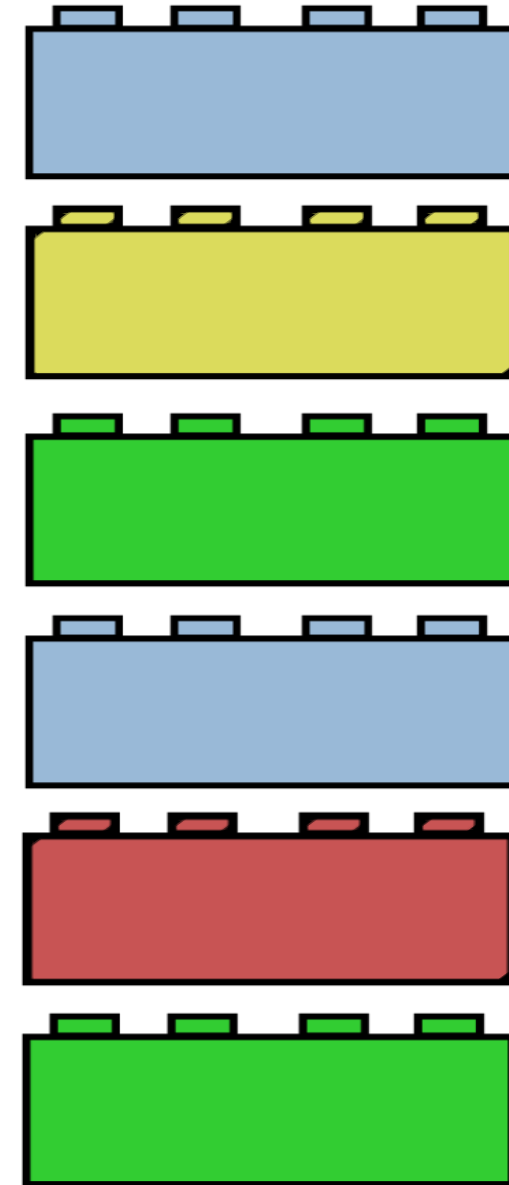
# Deep Learning Uses a Neural Network Architecture





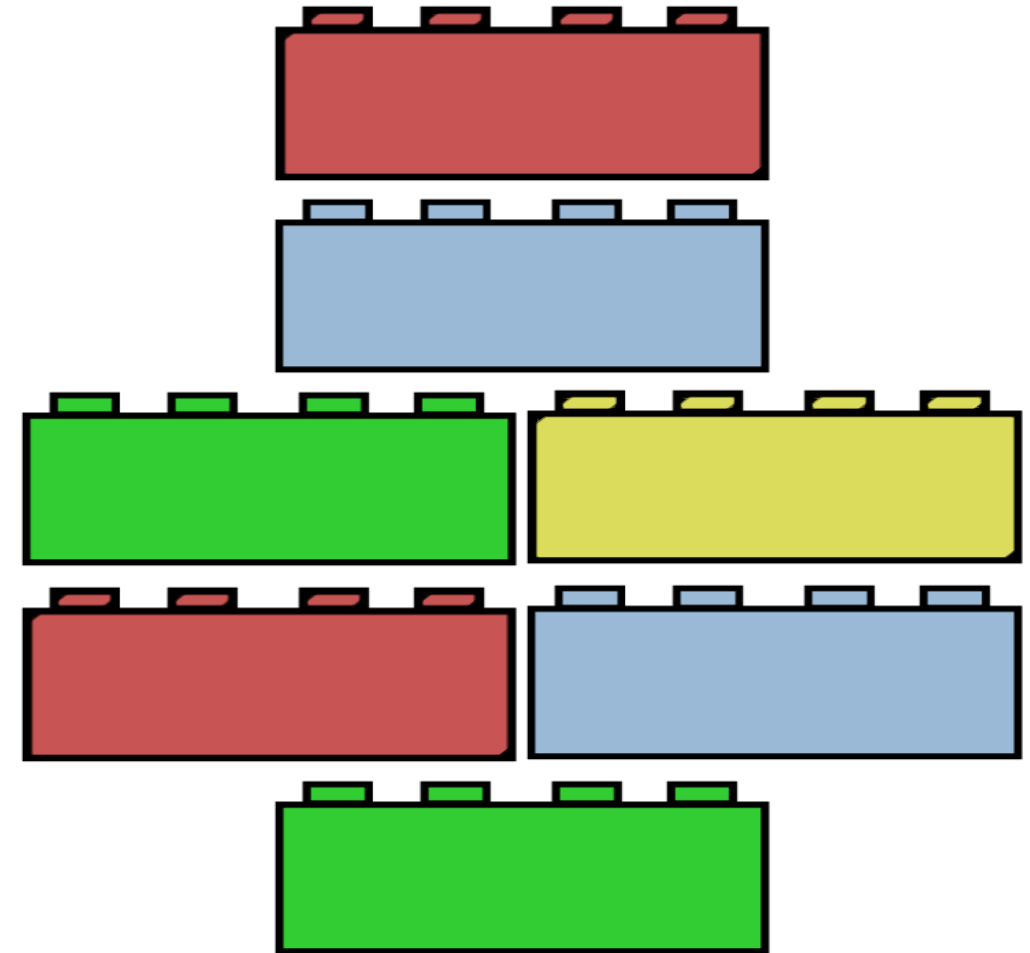
# Thinking about Layers

- Layers are like blocks
  - Stack on top of each other
  - Replace one block with a different one
- Each hidden layer processes the information from the previous layer



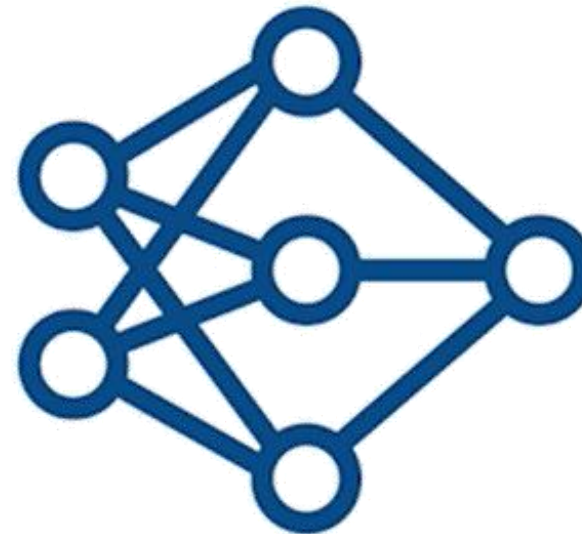
# Thinking about Layers

- Layers are like blocks
  - Stack on top of each other
  - Replace one block with a different one
- Each hidden layer processes the information from the previous layer
- Layers can be ordered in different ways



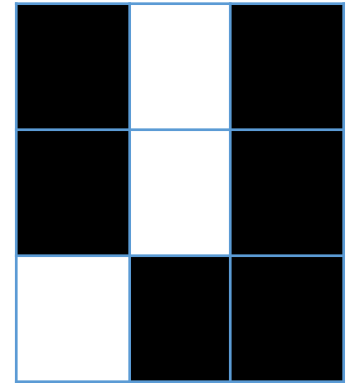
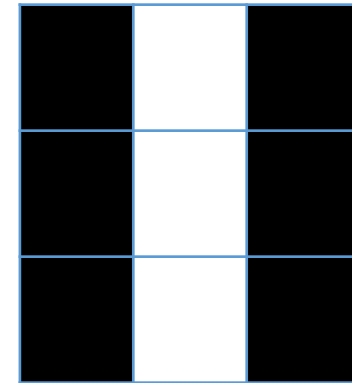
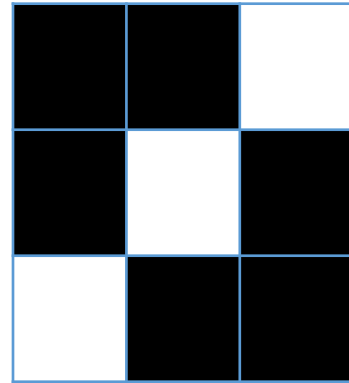
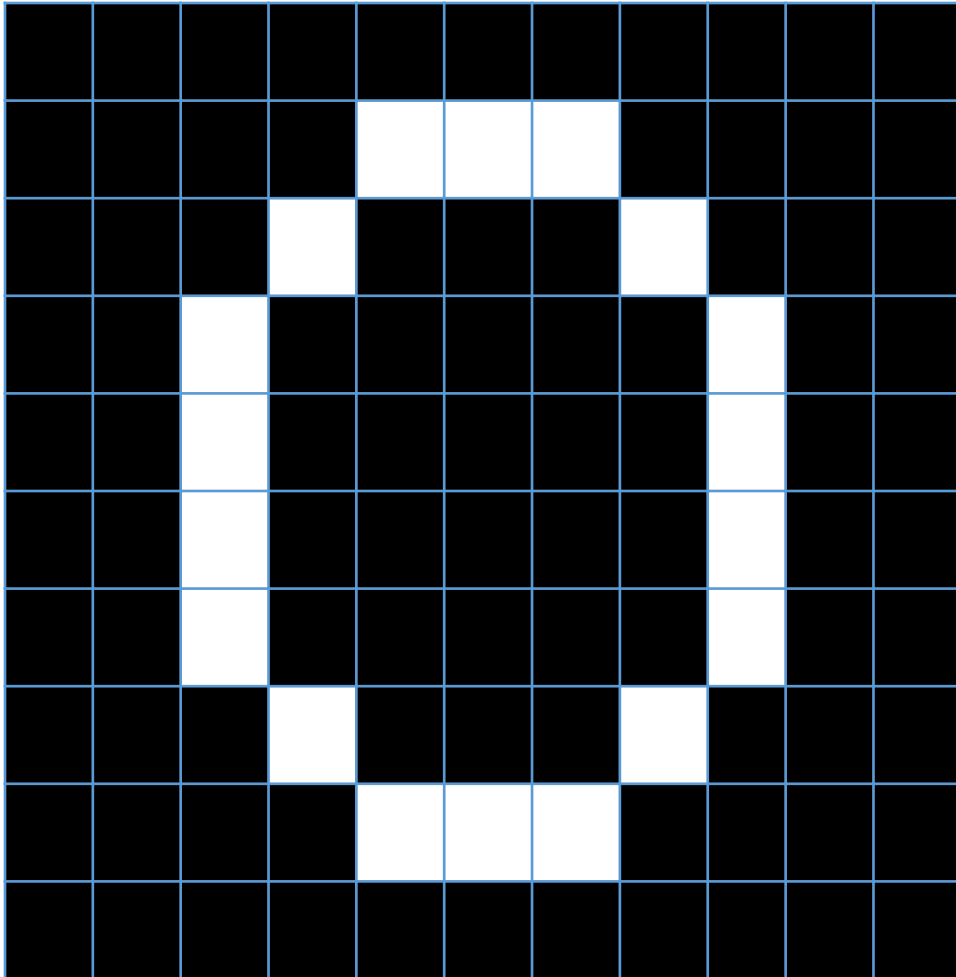
# Convolutional Neural Networks (CNNs)

- Special layer combinations that make them great for image classification
  - Convolution Layer
  - Max Pooling Layer
  - ReLU Layer

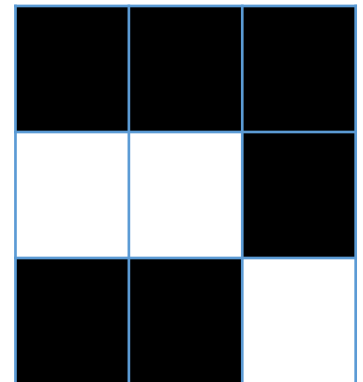
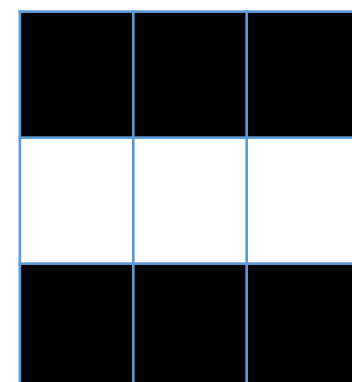
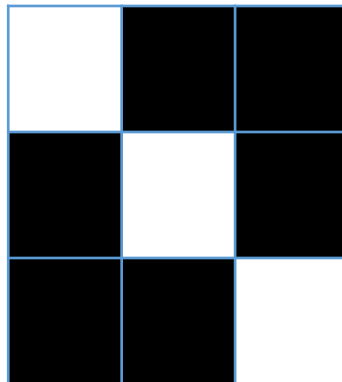




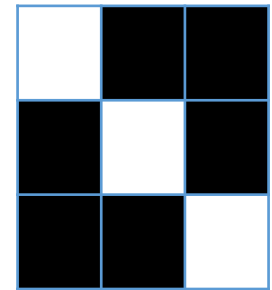
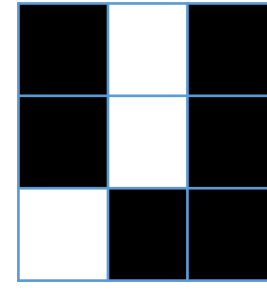
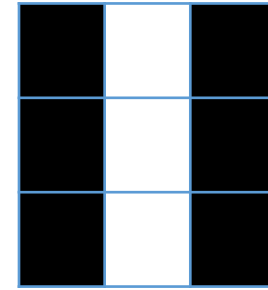
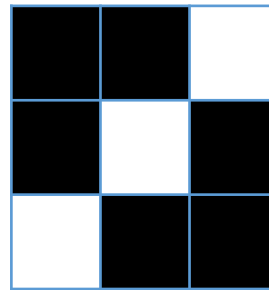
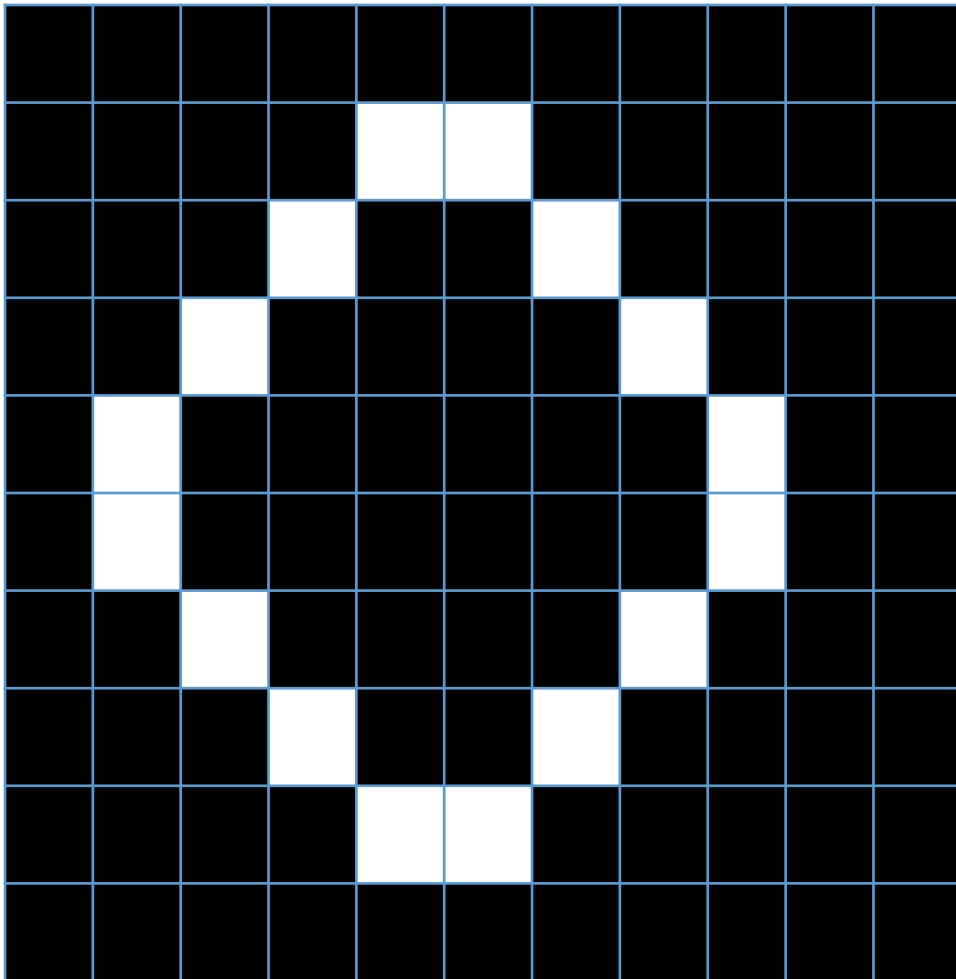
# Convolution Layers Search for Patterns



These patterns would be common in the number 0



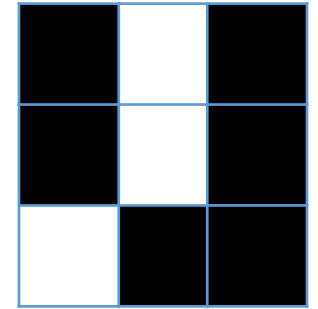
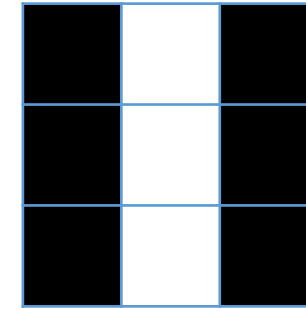
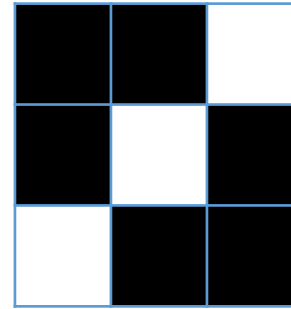
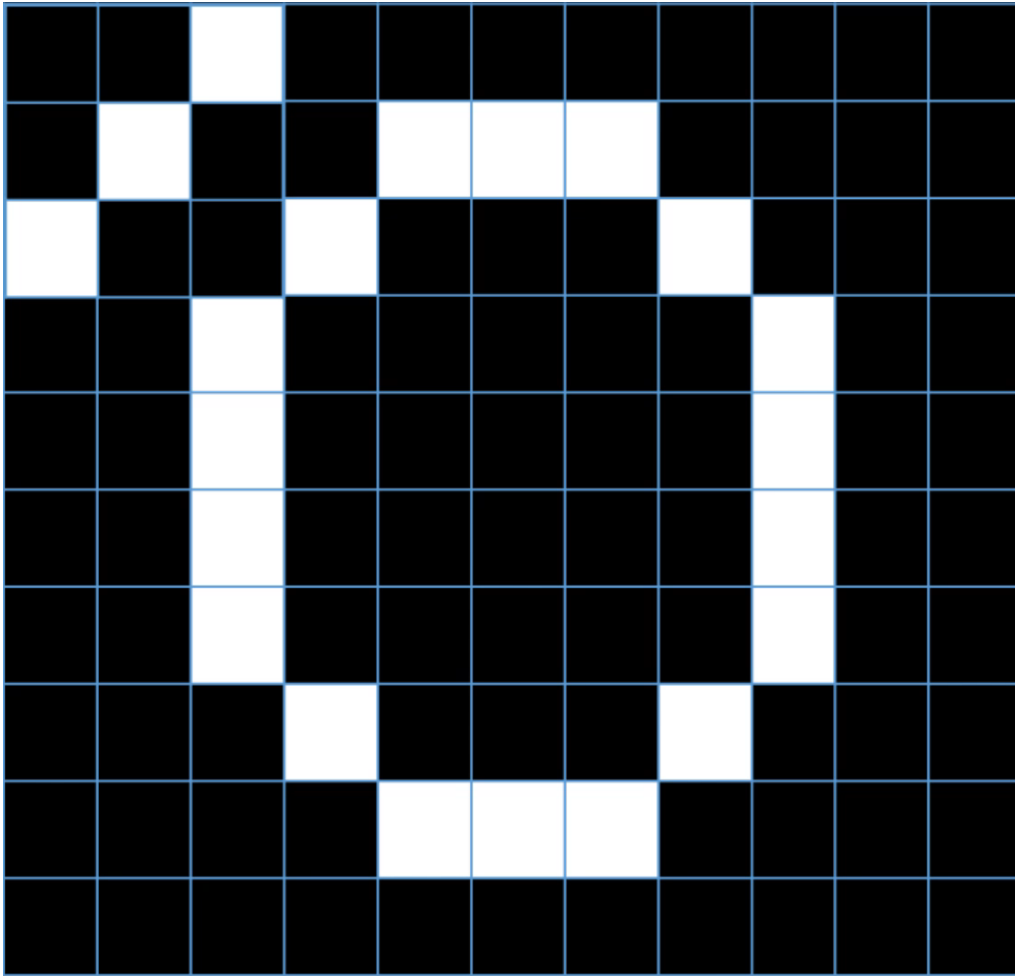
# All patterns are compared to the patterns on a new image.



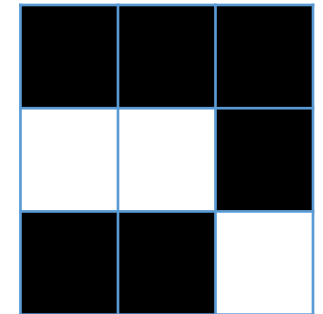
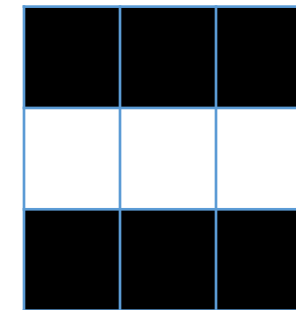
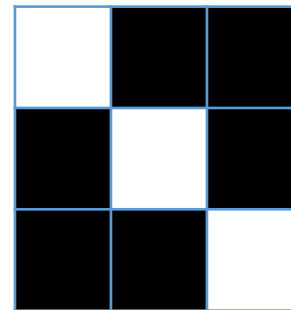
...

- Pattern starts at left corner  
Perform comparison  
Slide over one pixel
- Reach end of image
- Repeat for next pattern

# Convolution Layers Search for Patterns

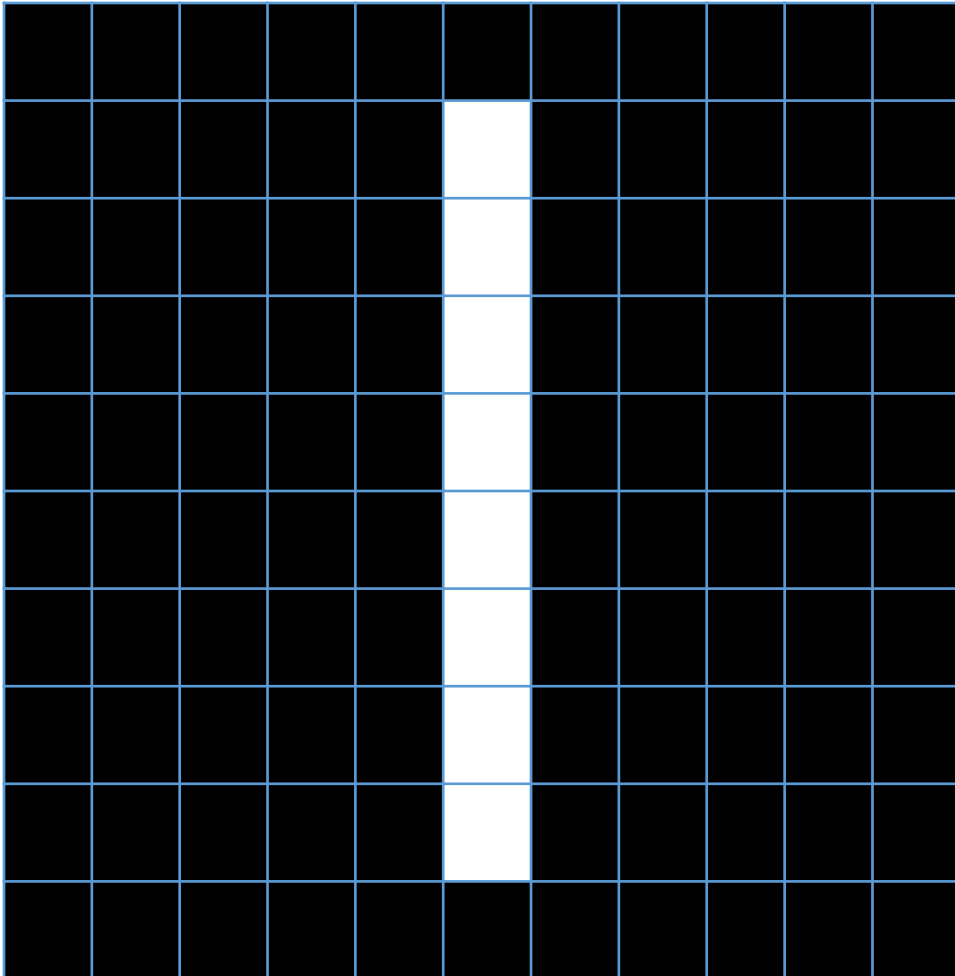


These patterns would be common in the number 0

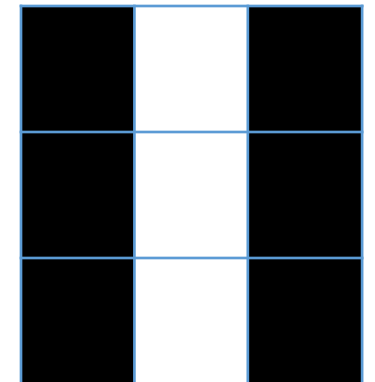




# Good pattern matching in convolution improves chances that object will classify properly

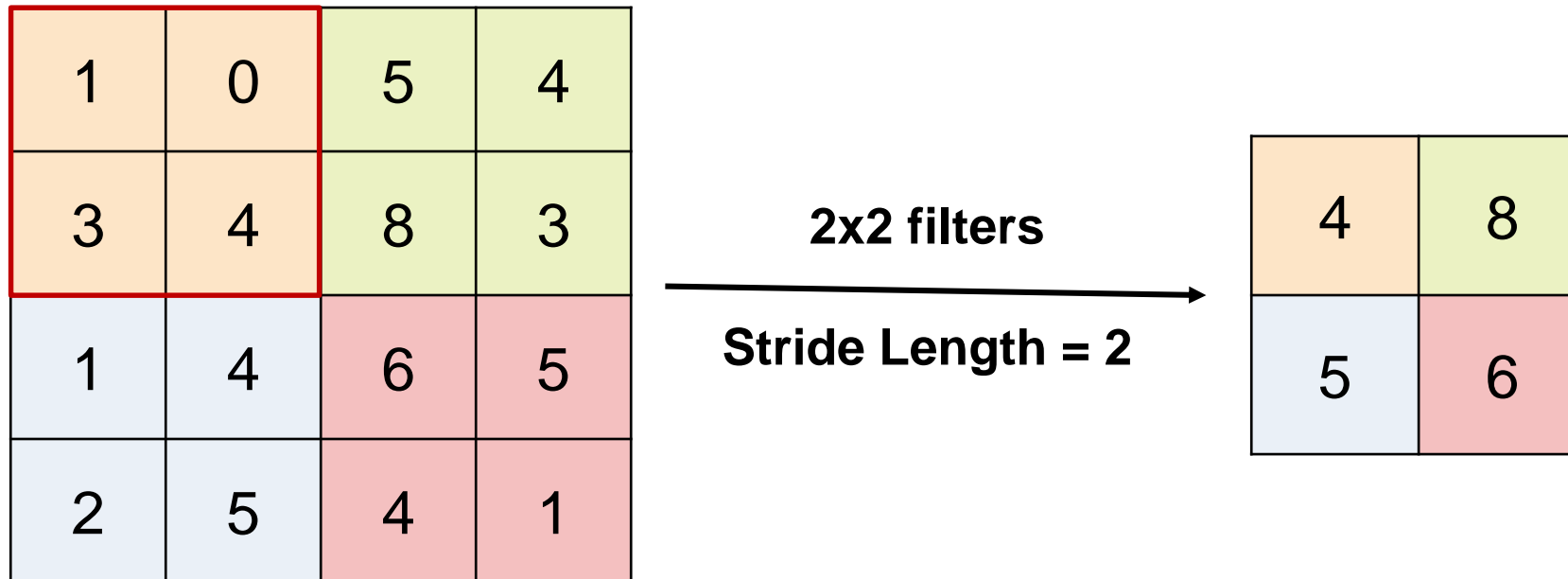


- This image would not match well against the patterns for the number zero
- It would only do very well against this pattern



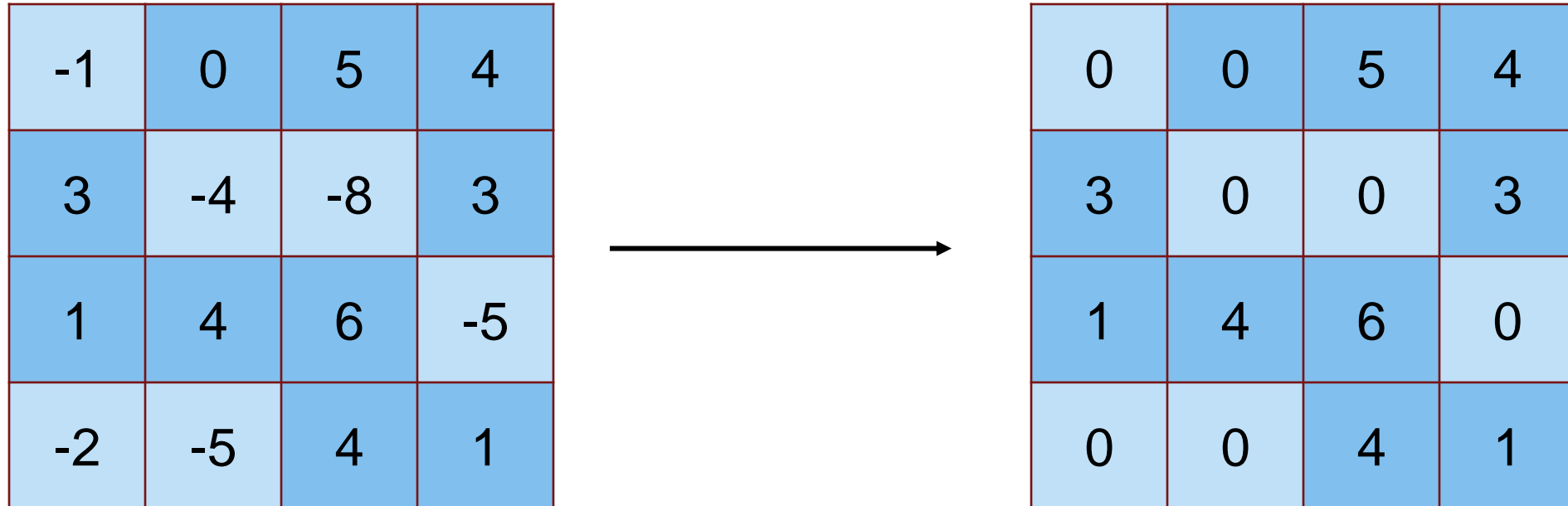
# Max Pooling is a down-sampling operation

Shrink large images while preserving important information



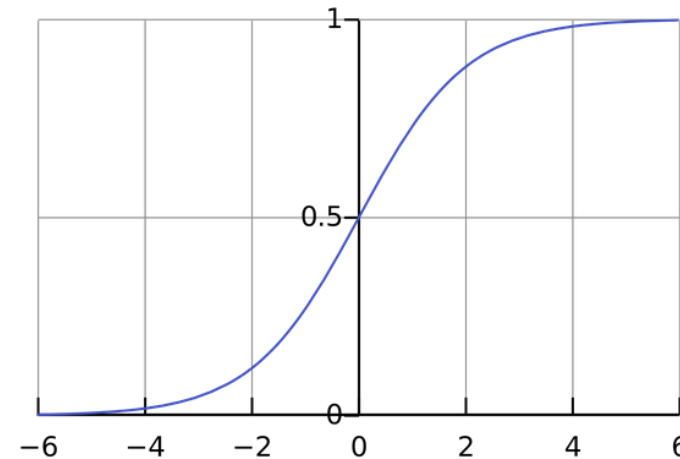
# Rectified Linear Units Layer (ReLU)

Typically converts negative numbers to zero



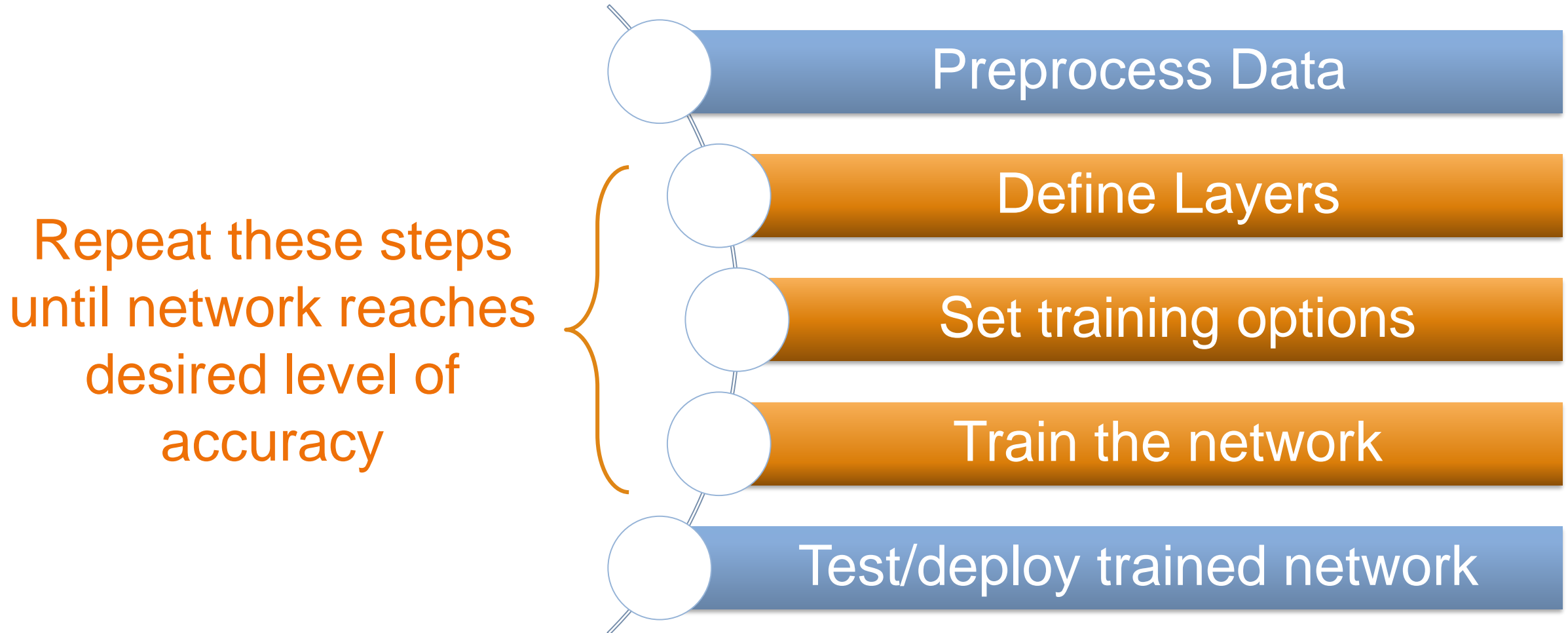
# CNNs typically end with 3 Layers

- Fully Connected Layer
  - Looks at which high-level features correspond to a specific category
  - Calculates scores for each category (highest score wins)
- Softmax Layer
  - Turns scores into probabilities.
- Classification Layer
  - Categorizes image into one of the classes that the network is trained on





# Deep Learning Workflow



# Pretrained Networks

- Researchers created network architecture for classifying hundreds of objects
- MATLAB makes it easy to import these networks
  - AlexNet, GoogLeNet, ResNet, VGG, Caffe models, TensorFlow-Keras models
- This is what we did for the peppers example! (AlexNet)

**AlexNet**

PRETRAINED MODEL

**VGG-16/19**

PRETRAINED MODEL

**Caffe**

MODEL IMPORTER

**ResNet**

PRETRAINED MODEL

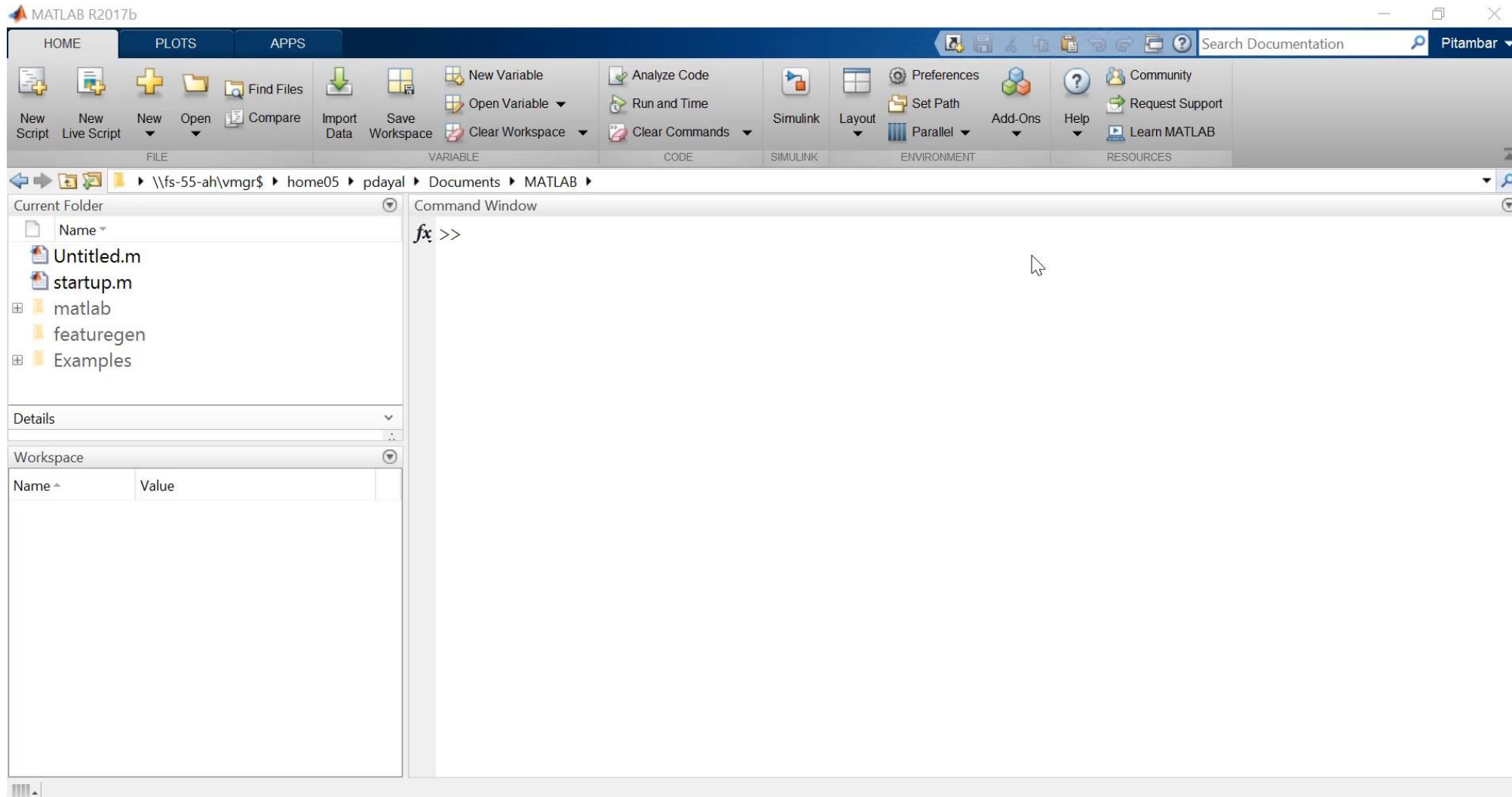
**GoogLeNet**

PRETRAINED MODEL

**TensorFlow-Keras**

MODEL IMPORTER

# Import Models from Keras-Tensorflow and Caffe



# Questions?





# Let's try it out!

*Exercise: **Work\_ExploringPretrainedNetworks.mlx**  
in folder 02-PretrainedModelExercise*

*Exercise: **MNIST\_HandwritingRecognition.mlx**  
in folder 03-MNISTExercise*

# Takeaways

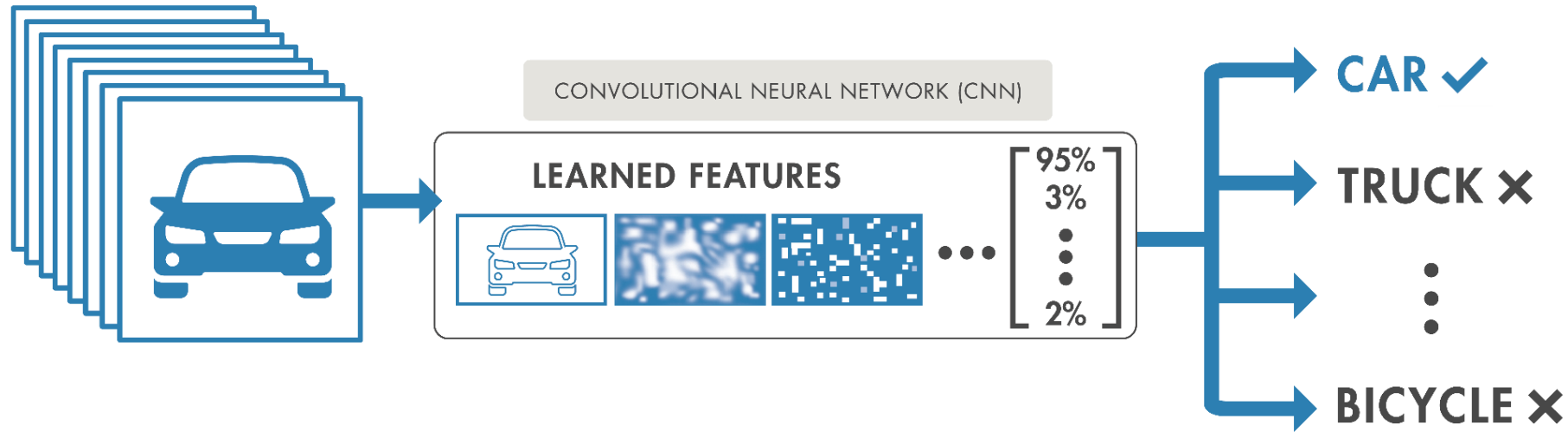
- Pre-trained networks have a pre-determined layer order that makes them effective for classifying images
  - Typically trained to classify lots of images
- Different networks yield different results
- Great starting point, but not consistently accurate
  - We'll fix this later with transfer learning!

# Takeaways

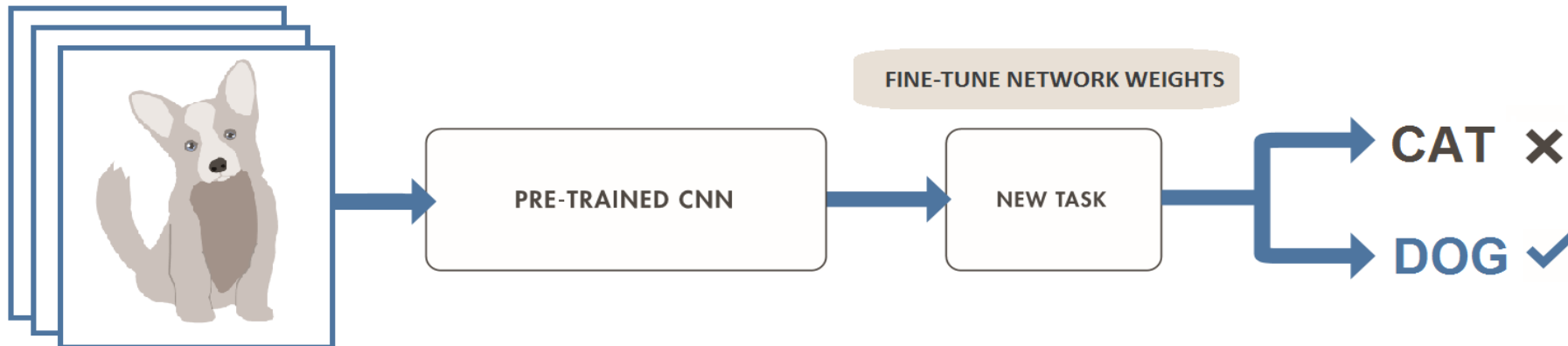
- Deep learning for image classification uses CNNs
- CNNs can have different combinations of initial layers but usually end with:
  - Fully Connected Layer
  - Softmax Layer
  - Classification Layer
- Important factors that affect accuracy and training time
  - Network architecture
  - Initial learning rate

# Two Approaches for Deep Learning

## 1. Train a Deep Neural Network from Scratch



## 2. Fine-tune a pre-trained model (transfer learning)



# Transfer Learning Workflow

## Load pretrained network

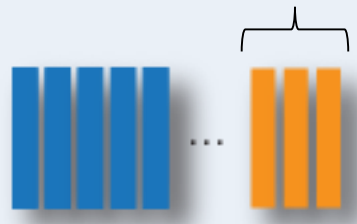
Early layers that learned low-level features (edges, blobs, colors)      Last layers that learned task specific features



1 million images  
1000s classes

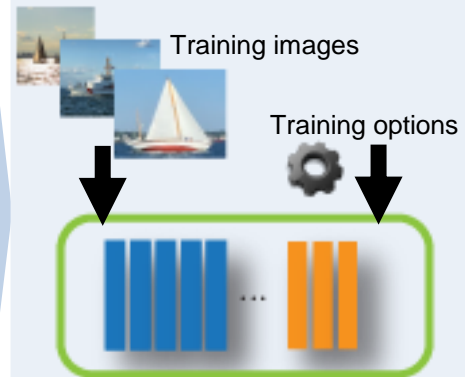
## Replace final layers

New layers to learn features specific to your data



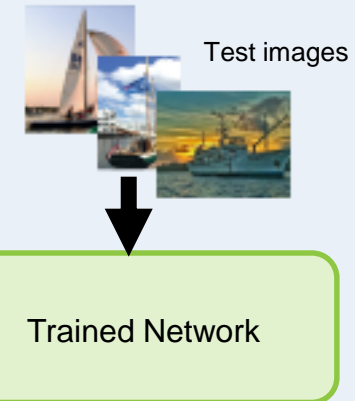
Fewer classes  
Learn faster

## Train network

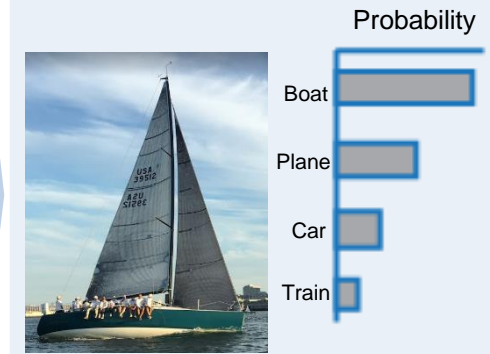


100s images  
10s classes

## Predict and assess network accuracy



## Deploy results

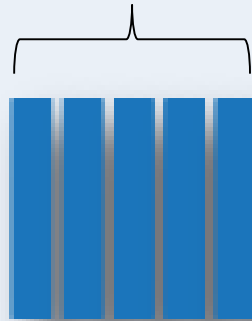




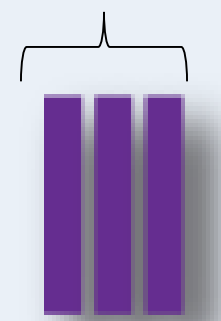
# Transfer Learning Workflow – Step 1

## Load pretrained network

Early layers learn low-level features (edges, blobs, colors)



Last layers learn task-specific features



...

1 million images  
1000s classes

# Transfer Learning Workflow – Step 2

## Load pretrained network

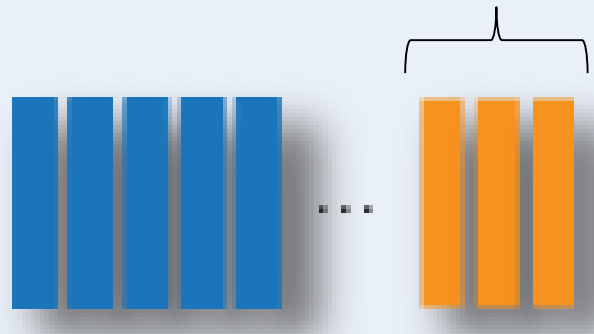
Early layers that learned low-level features (edges, blobs, colors)      Last layers that learned task specific features



1 million images  
1000s classes

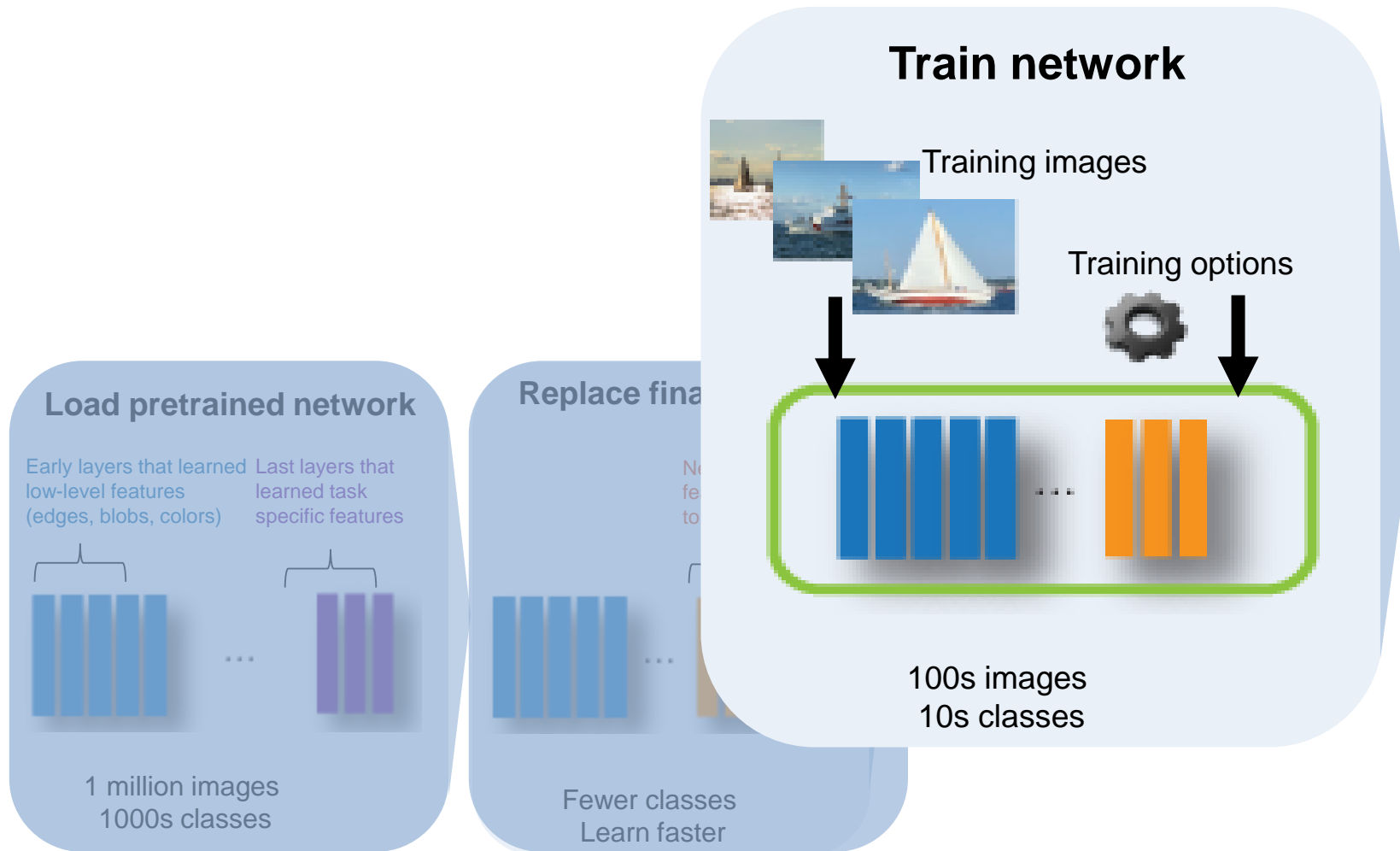
## Replace final layers

New layers learn features specific to your data

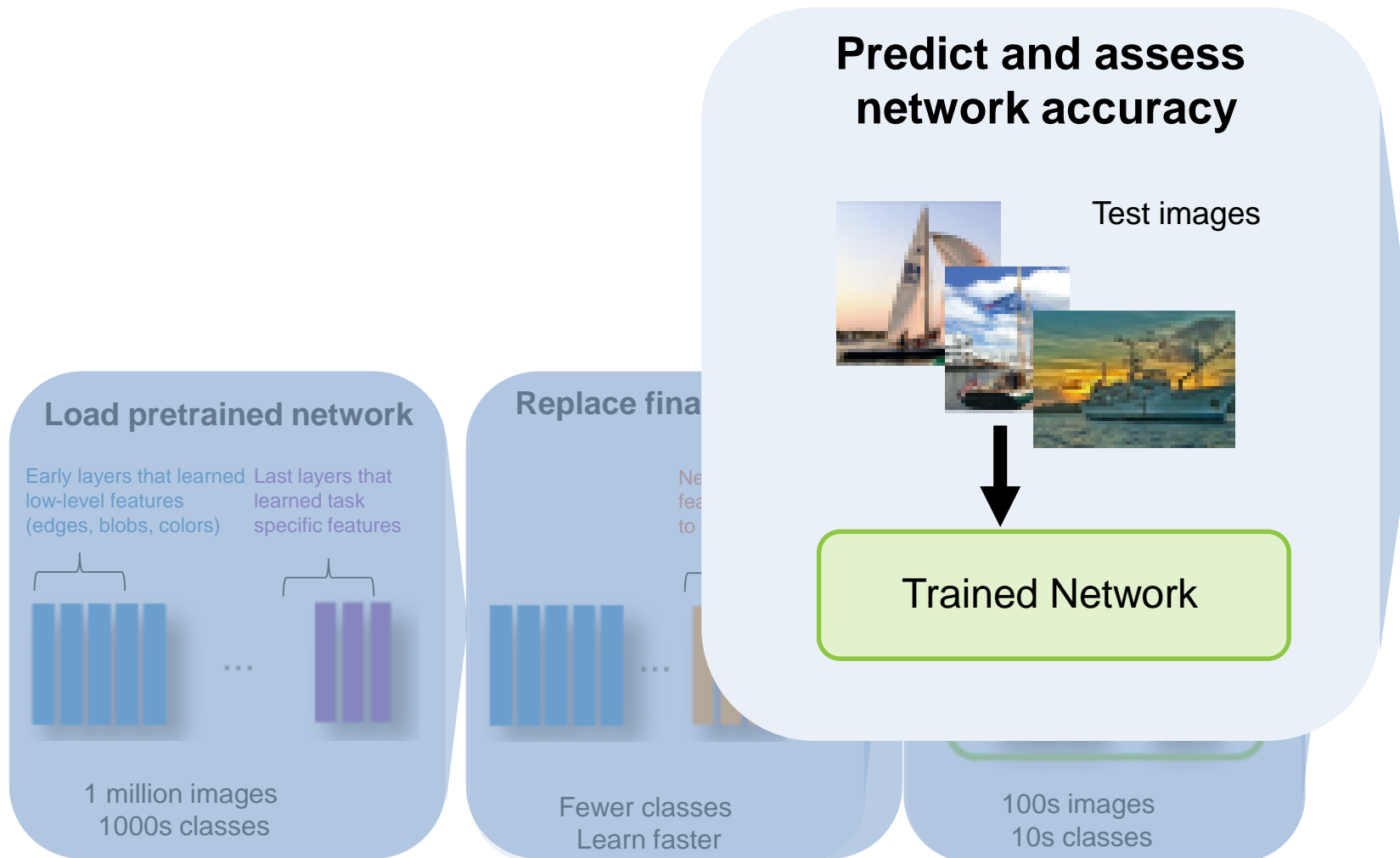


Fewer classes  
Learn faster

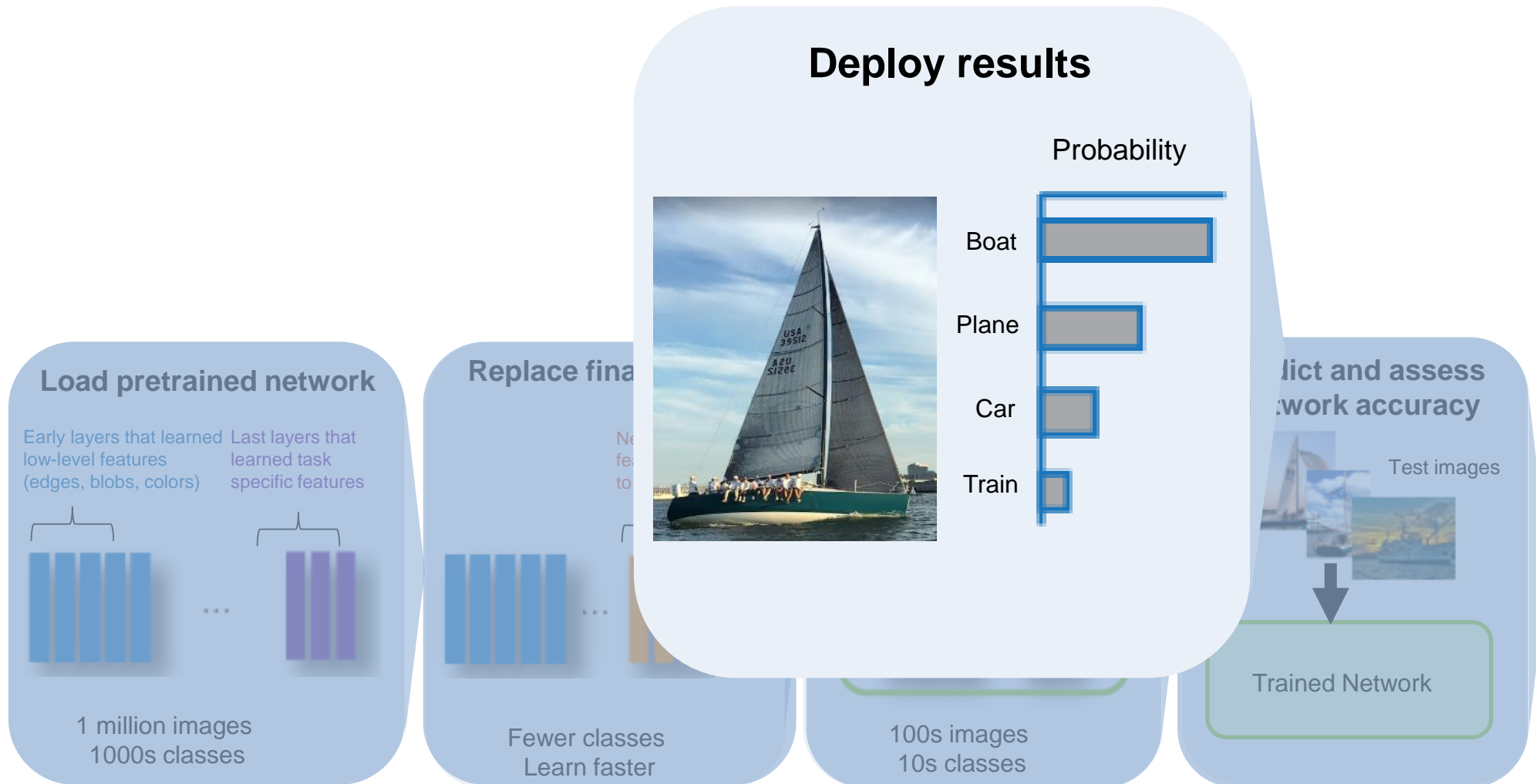
# Transfer Learning Workflow – Step 3



# Transfer Learning Workflow – Step 4



# Transfer Learning Workflow – Step 5





# Transfer Learning Workflow

## Load pretrained network

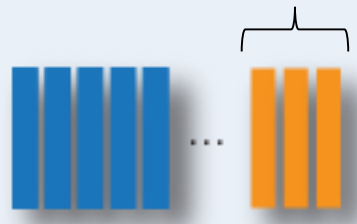
Early layers that learned low-level features (edges, blobs, colors)      Last layers that learned task specific features



1 million images  
1000s classes

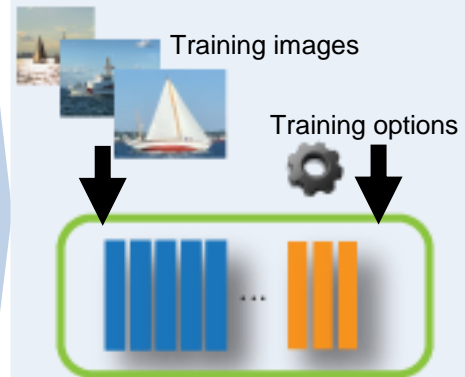
## Replace final layers

New layers to learn features specific to your data



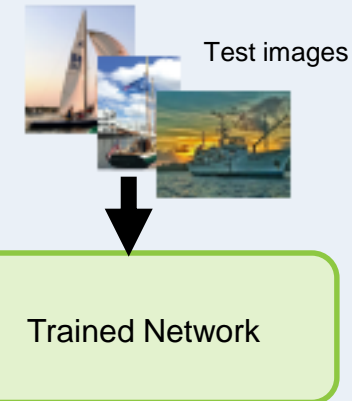
Fewer classes  
Learn faster

## Train network

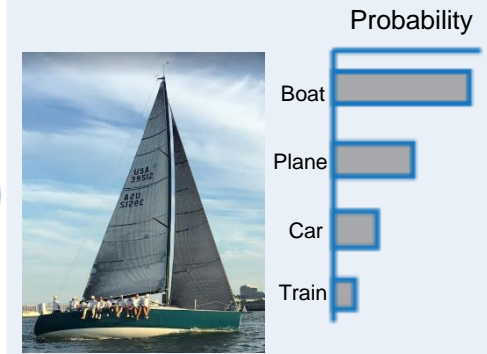


100s images  
10s classes

## Predict and assess network accuracy



## Deploy results



# Let's try it out!

*Exercise: **Work\_SeeFoodTransferLearning.mlx***  
in folder 04-TransferLearningExercise

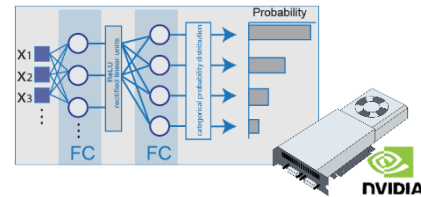
# Takeaways – Transfer Learning

- Replace last layers with our own layers
- Efficient way to modify pre-trained models to our needs
- Use an Image datastore when working with lots of images
- MATLAB lets you visualize activations in a network

# Deep Learning Workflow Extends Beyond Training

DEVELOP PREDICTIVE  
MODELS

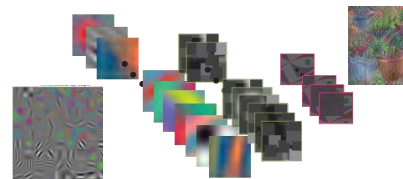
## Hardware-Accelerated Training



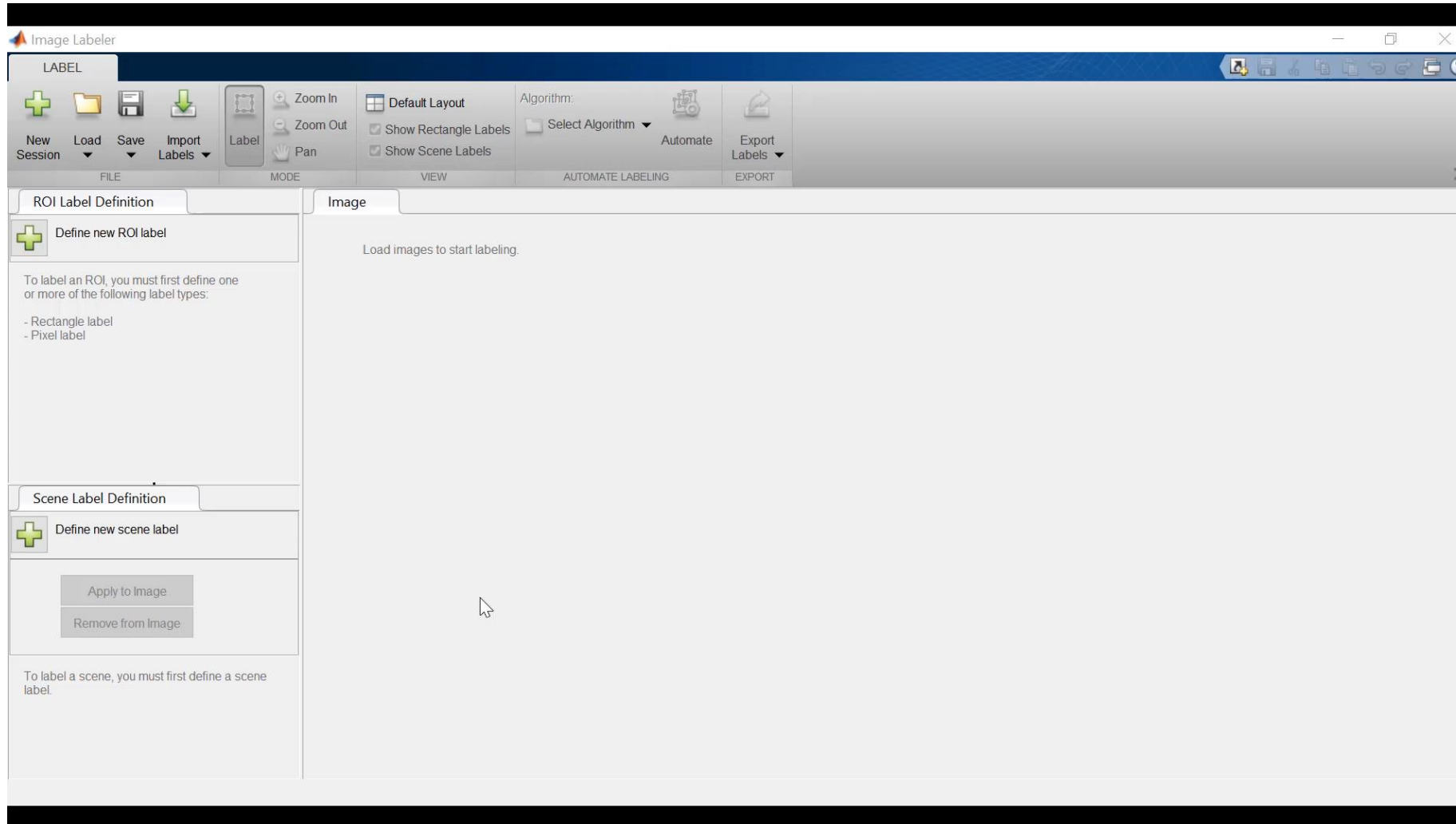
## Hyperparameter Tuning



## Network Visualization



# Automated Object Detection

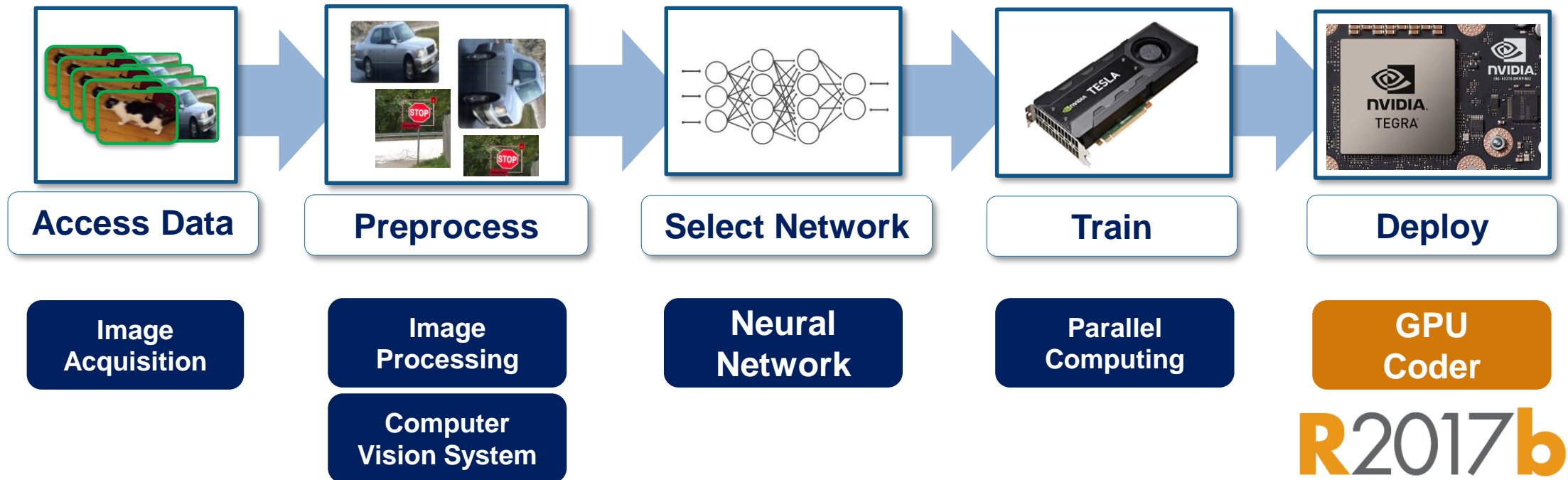


# Labeling Data

- Image Labeler App
  - Object Detection
  - Pixel Labeling
- Ground-truth Labeler App
  - Videos for automated driving applications
- 3D Point Cloud Labeling with Semantic Segmentation
- Labeling Big Images



# One Step Left – Deployment!

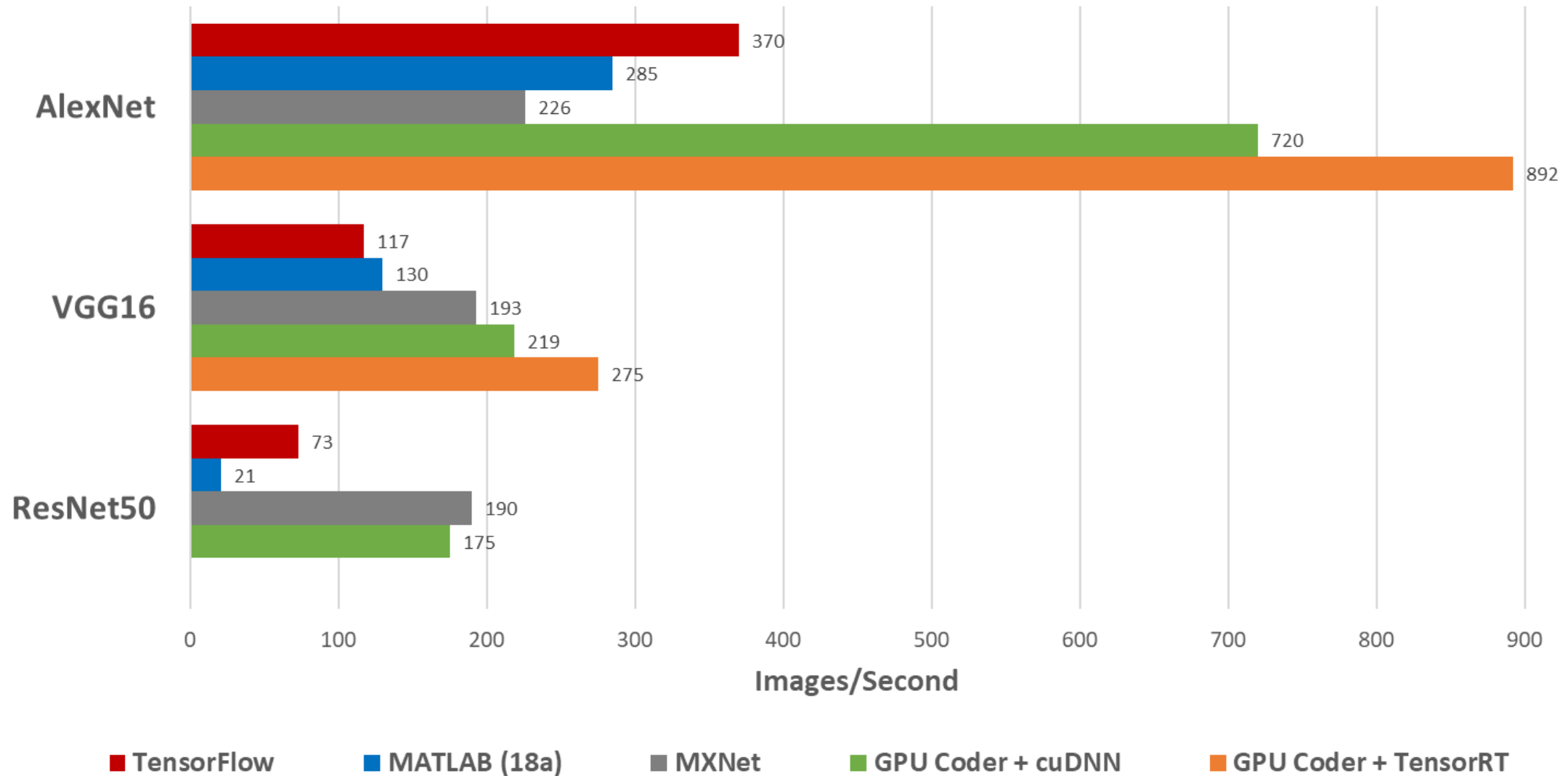


# GPU Coder

- **Automatically** generates CUDA Code from MATLAB Code
  - can be used on NVIDIA GPUs



## Single Image Inference (Titan XP, Linux) - Higher is Better



# How fast is GPU Coder? Vision Algorithms compared to C on CPU



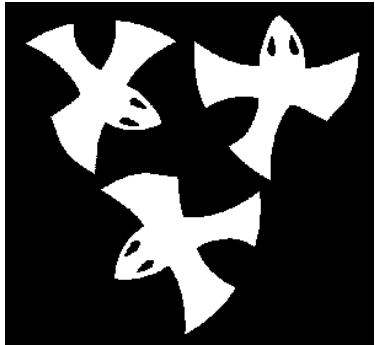
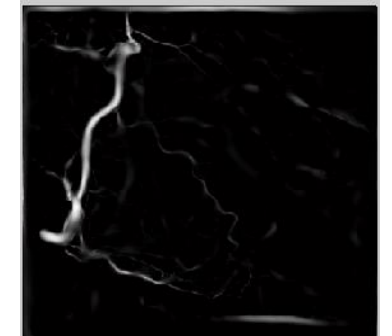
Fog removal

5x speedup



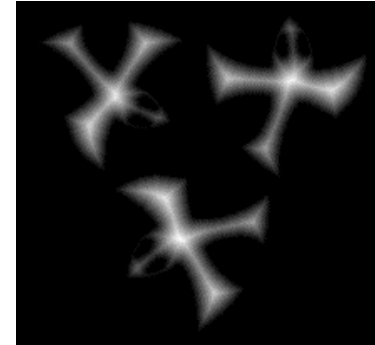
Frangi filter

3x speedup



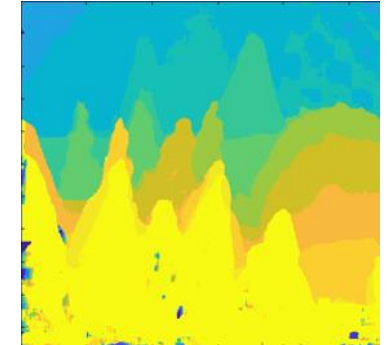
Distance transform

8x speedup



Stereo disparity

50x speedup



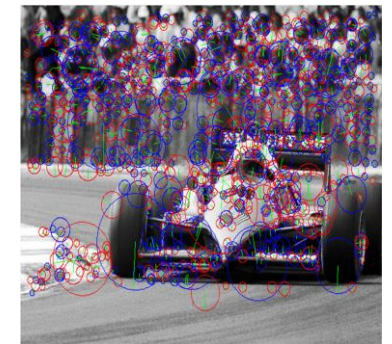
Ray tracing

18x speedup



SURF feature extraction

700x speedup



# GPU Coder Demo

*Deploying our deep network on a GPU*

Open: **GenerateGPUCode.mlx**

in folder 05-GPUCoder

# Takeaways – GPU Coder

- MATLAB supports entire deep learning workflow
- Generate code for various targets from MATLAB code
- GPU Coder+TensorRT fastest for series networks
- GPU Coder very fast for DAG networks



# Deep Learning OnRamp

[Contact Us](#) [How to Buy](#) [Log In](#)[MathWorks®](#) [Products](#) [Solutions](#) [Academia](#) [Support](#) [Community](#) [Events](#)

## MATLAB and Simulink Training

[Overview](#) | [Course Offerings](#) | [Course Schedule](#) | [Self-Paced Training](#) | [Training At Your Facility](#) | [Certification](#) | [More ▾](#) | [Contact Training](#)[Course Schedule](#)

### Prerequisites

[MATLAB Onramp](#)

## Deep Learning Onramp

<https://matlabacademy.mathworks.com/>

This free self-paced course provides an interactive introduction to practical deep learning. It focuses on using MATLAB® to apply deep learning methods to perform image recognition. The course consists of hands-on exercises and short videos. In the exercises, you will enter commands in an online version of MATLAB and receive contextual feedback that will help you correct common mistakes. Topics include:

- Convolutional neural networks
- Preprocessing images
- Using pretrained networks
- Transfer learning
- Evaluating network performance

Self-paced FREE course  
Hands-on experience  
Everything done in the browser

### Course Schedule

Results 1 - 1 of 1

Dates	Location	Language	Price	Register
On Demand	Self-Paced	English	Free	<a href="#">Launch</a>

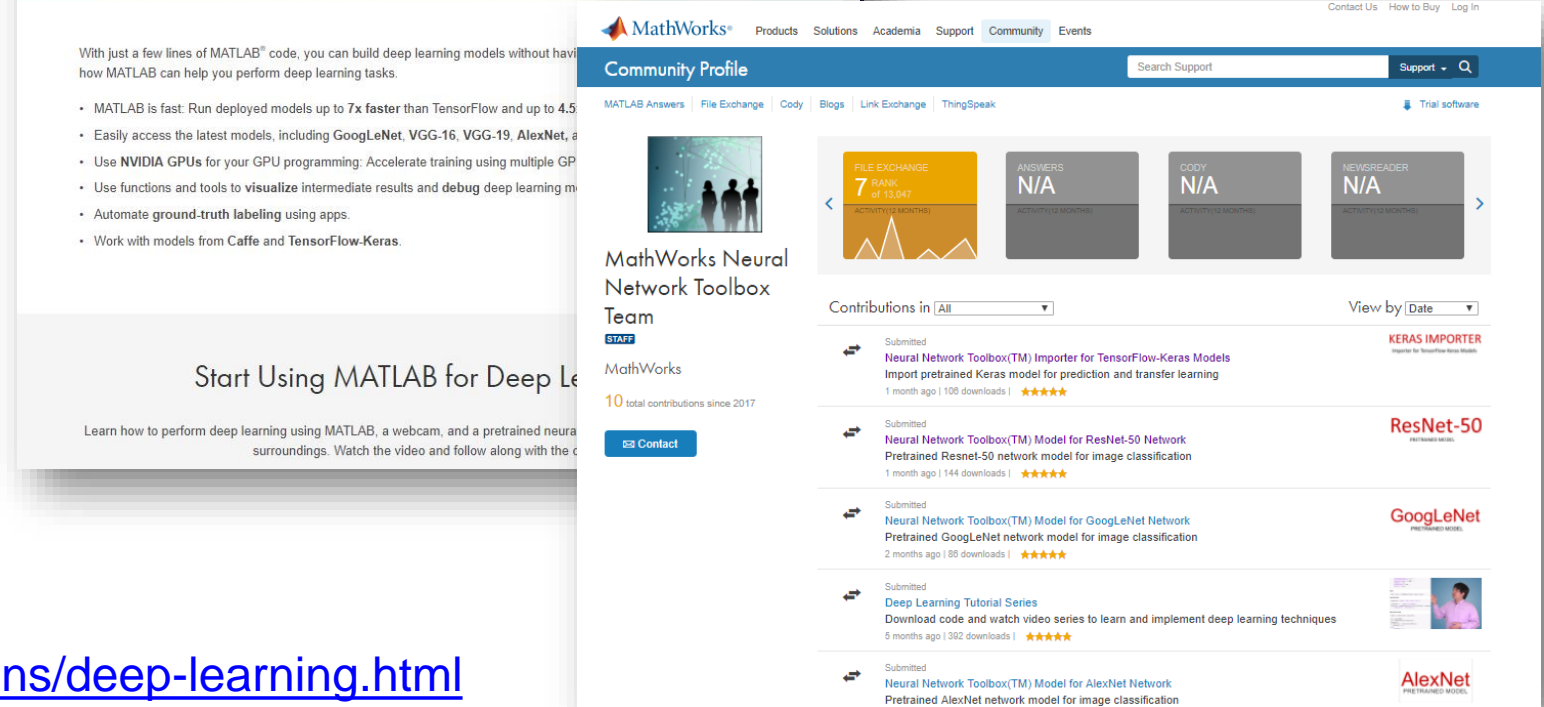
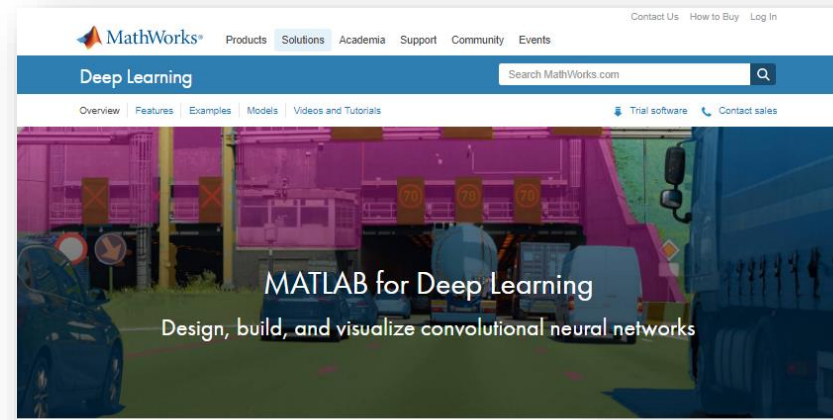
Results 1 - 1 of 1

The pricing applies for purchase and use in United States. For pricing in other regions [Contact Sales](#). The product price does not include sales, use, excise, value-added, or other taxes. Any applicable taxes, duties, levies, assessments and governmental charges payable in connection with this purchase will be assessed on the order. Refer to [Training Policies](#) for more information.

# Resources

- Web
- Documentation
- Community
  - File Exchange/GitHub
- Deep Learning OnRamp

<https://www.mathworks.com/solutions/deep-learning.html>



# POP QUIZ

*Results will be reported to your manager*

# What is the difference between Machine Learning and Deep Learning?

- A. Deep learning is machine learning done really far underground.
- B. I don't know, I didn't pay attention, I actually don't even work here, I just show up to these things.
- C. Machine learning requires manual feature extraction while deep learning automatically extracts features making it end-to-end learning

# What is the difference between Machine Learning and Deep Learning?

- A. Deep learning is machine learning done really far underground.
- B. I don't know, I didn't pay attention, I actually don't even work here, I just show up to these things.
- C. Machine learning requires manual feature extraction while deep learning automatically extracts features making it end-to-end learning

# Which of the following is not an application of deep learning?

- A. Image classification
- B. Speech recognition
- C. Automated driving
- D. Filtering applications like rain removal
- E. Recognizing people's faces on your phone's photo app
- F. Building a hotdog/not-hotdog classifier
- G. None of the above

# Which of the following is not an application of deep learning?

- A. Image classification
- B. Speech recognition
- C. Automated driving
- D. Filtering applications like rain removal
- E. Recognizing people's faces on your phone's photo app
- F. Building a hotdog/not-hotdog classifier
- G. None of the above



# Which of the following is NOT a layer in deep networks?

- A. Fully Connected Layer
- B. Softmax Layer
- C. Classification Layer
- D. Convolution Layer
- E. ReLu Layer
- F. MaxPooling Layer
- G. Banana Layer (classifies all objects as Banana)

# Which of the following is NOT a layer in deep networks?

- A. Fully Connected Layer
- B. Softmax Layer
- C. Classification Layer
- D. Convolution Layer
- E. ReLu Layer
- F. MaxPooling Layer
- G. Banana Layer (classifies all objects as Banana)

# What does the Fully Connected Layer do?

- A. Calculates a score for each category
- B. Plays a full game of Connect Four
- C. Saves you 15% or more on car insurance

# What does the Fully Connected Layer do?

- A. Calculates a score for each category
- B. Ensures your layered sandwiches stay Fully Connected
- C. Saves you 15% or more on car insurance

# How do we perform transfer learning?

- A. Change every other layer of our network to a softmax layer
- B. Transfer all data from the CPU to the GPU
- C. Load in a pre-trained network, modify the last few layers, and train it on our data.

# How do we perform transfer learning?

- A. Change every other layer of our network to a softmax layer
- B. Transfer all data from the CPU to the GPU
- C. Load in a pre-trained network, modify the last few layers, and train it on our data.

**What are three hyperparameters that have a major impact on training time and accuracy?**

- A. Network Architecture
- B. Mini Batch Size
- C. Learning Rate
- D. Flux Capacitor



# What are three hyperparameters that have a major impact on training time and accuracy?

- A. Network Architecture
- B. Mini Batch Size
- C. Learning Rate
- D. Flux Capacitor

# What is loss?

- A. The opposite of a win
- B. The state or feeling of grief when deprived of someone or something of value
- C. A measurement of error between predicted labels and actual labels. Loss has an inverse relationship with score, and our goal is to minimize loss.
- D. All of the above

# What is loss?

- A. The opposite of a win
- B. The state or feeling of grief when deprived of someone or something of value
- C. A measurement of error between predicted labels and actual labels. Loss has an inverse relationship with score, and our goal is to minimize loss.
- D. All of the above

# Which of the following statements is false?

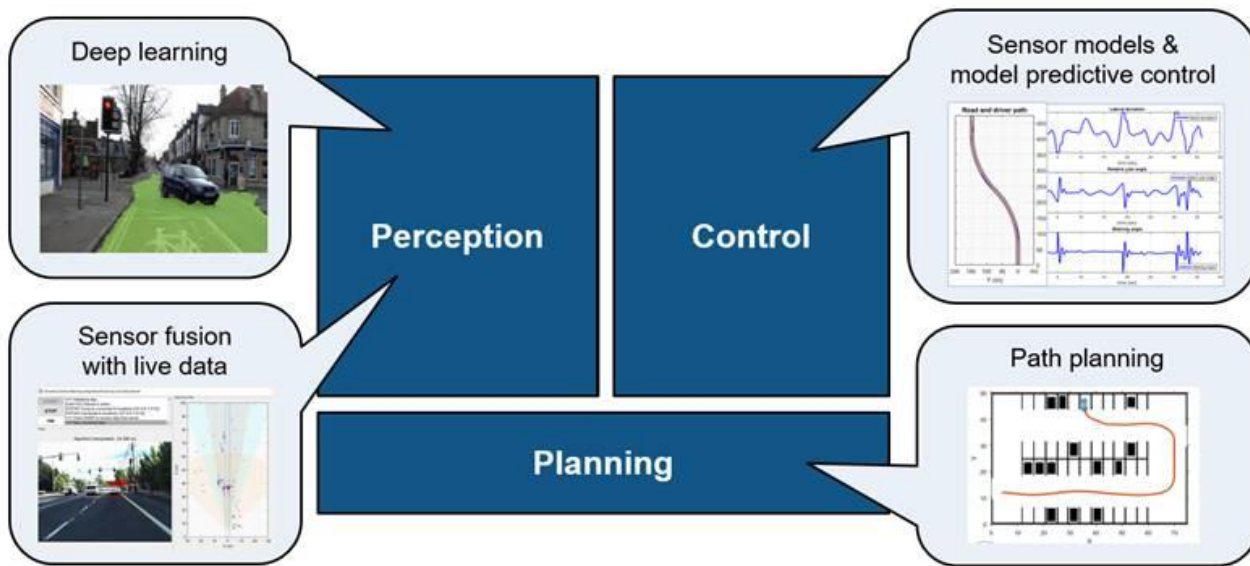
- A. MATLAB makes it easy to import pre-trained models through add-ons and model importers
- B. MATLAB supports the entire deep learning workflow including labeling, training, and deployment
- C. MATLAB has visual training plots that allow you to see accuracy and loss during training
- D. We do a great job of **subtly** marketing MATLAB's deep learning capabilities

# Which of the following statements is false?

- A. MATLAB makes it easy to import pre-trained models through add-ons and model importers
- B. MATLAB supports the entire deep learning workflow including labeling, training, and deployment
- C. MATLAB has visual training plots that allow you to see accuracy and loss during training
- D. We do a great job of **subtly** marketing MATLAB's deep learning capabilities

# Free Seminar: ADAS and Automated Driving Development Using MATLAB and Simulink

Examples of how you can use MATLAB and Simulink to develop automated driving algorithms



Location	Venue	Start Date	End Date
Santa Clara, CA	MathWorks Office (Mission Towers, Floor 1)	29 Mar 2018 - 9:00 AM	29 Mar 2018 - 12:00 PM

## Overview

Do you have an **open and flexible visualization** tool to gain insight from vision, radar, and LiDAR data? How quickly can you apply the latest **deep learning** research to vision perception development? Are you able to design **sensor fusion and control** in simulation, before going to the test vehicle?

In this seminar, MathWorks engineers will demonstrate several new technologies to accelerate ADAS and automated driving development with MATLAB and Simulink.

## Highlights

They will introduce the latest ADAS and automated driving development tools from MathWorks, including

- Visualize recorded and live sensor data
- Framework for sensor fusion algorithm design and test
- Deep learning for LiDAR and camera processing
- Control design in simulation

They will demonstrate new products and ADAS-extensions of existing MathWorks products, including

- Automated Driving System Toolbox
- Model Predictive Control Toolbox
- Vehicle Network Toolbox
- Robotics Systems Toolbox
- GPU Coder

## Who Should Attend

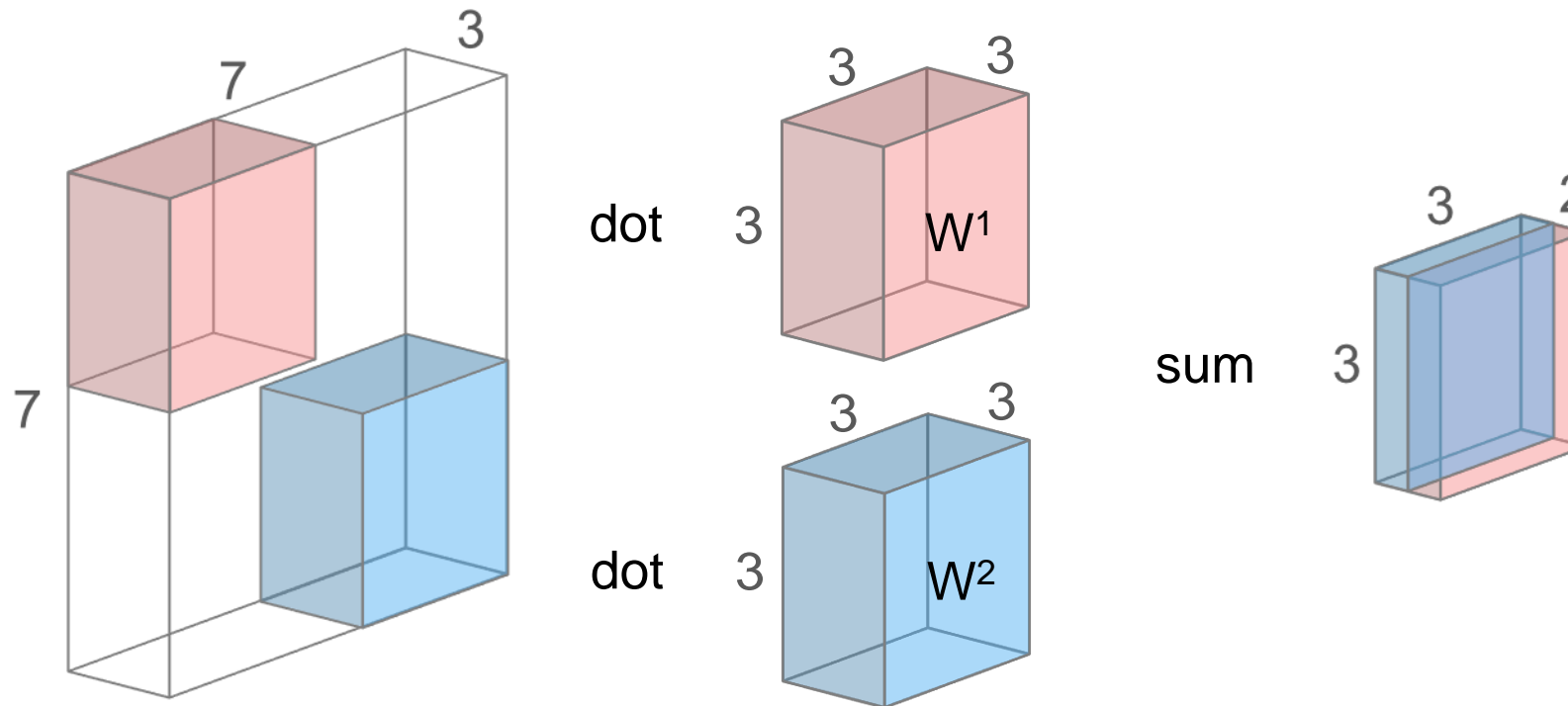
Engineers and managers working on ADAS and automated driving system, algorithm, and software development.

# Questions?



# Convolution Layer

- Core building block of a CNN
- Convolve the filters sliding them across the input, computing the dot product

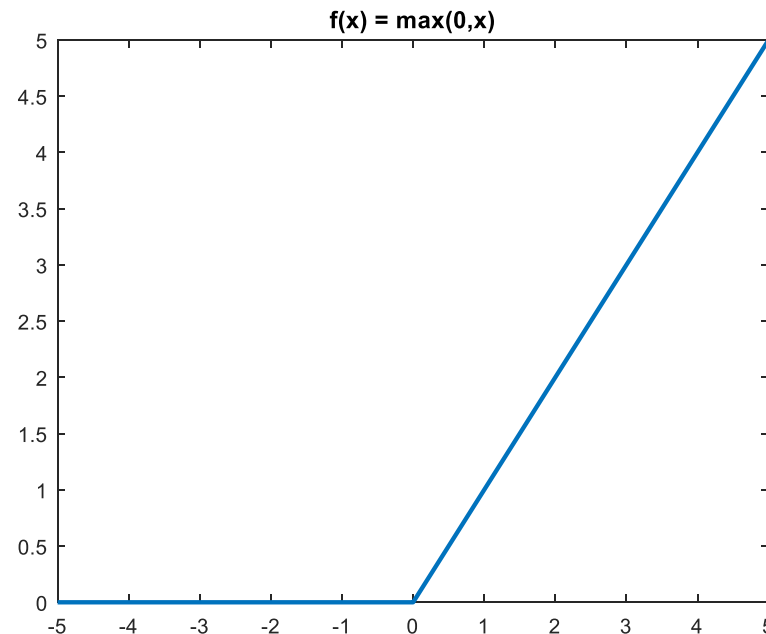


- Intuition: learn filters that activate when they “see” some specific feature



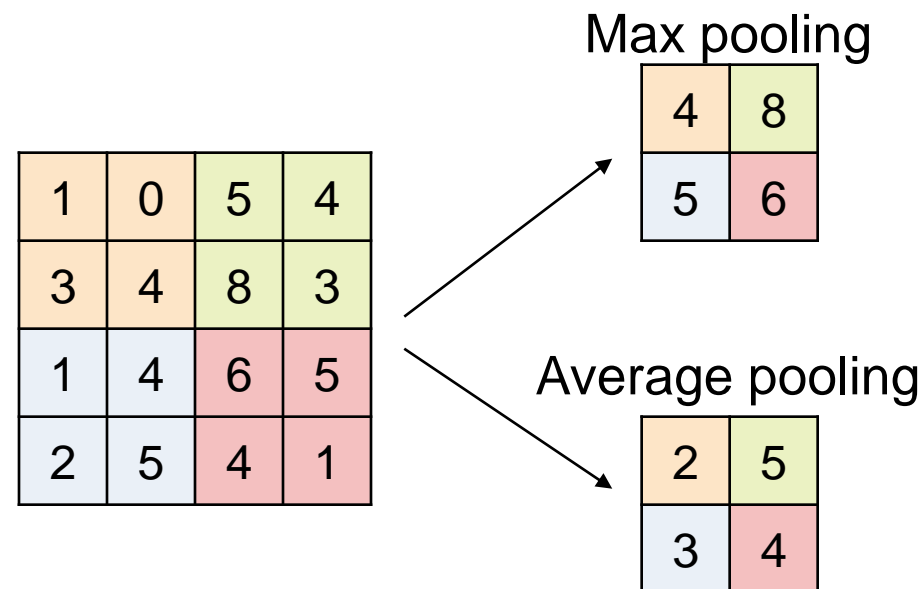
# Rectified Linear Unit (ReLU) Layer

- Frequently used in combination with Convolution layers
- Do not add complexity to the network
- Most popular choice:  $f(x) = \max(0, x)$ , activation is thresholded at 0



# Pooling Layer

- Perform a **downsampling** operation across the spatial dimensions
- Goal: progressively decrease the size of the layers
- Max pooling and average pooling methods
- Popular choice: Max pooling with 2x2 filters, Stride = 2



# Convolution

0	0	0	0	0	0	0	0	0	0
0	-8	-7	-9	4	4	-8	8	2	0
0	6	-1	3	-5	-7	0	9	-2	0
0	5	0	-9	-6	2	-3	3	1	0
0	8	3	-2	4	3	-8	1	2	0
0	0	-7	6	1	3	2	5	-3	0
0	3	9	4	0	9	7	2	6	0
0	-1	-5	-4	2	5	6	-4	-9	0
0	7	-2	8	9	1	2	-7	5	0
0	0	0	0	0	0	0	0	0	0

1	0	0
0	1	0
0	0	1


# Convolution

0	0	0	0	0	0	0	0	0	0
0	-8	-7	-9	4	4	-8	8	2	0
0	6	-1	3	-5	-7	0	9	-2	0
0	5	0	-9	-6	2	-3	3	1	0
0	8	3	-2	4	3	-8	1	2	0
0	0	-7	6	1	3	2	5	-3	0
0	3	9	4	0	9	7	2	6	0
0	-1	-5	-4	2	5	6	-4	-9	0
0	7	-2	8	9	1	2	-7	5	0
0	0	0	0	0	0	0	0	0	0

1	0	0
0	1	0
0	0	1


# Convolution

0	0	0	0	0	0	0	0	0	0
0	-8	-7	-9	4	4	-8	8	2	0
0	6	-1	3	-5	-7	0	9	-2	0
0	5	0	-9	-6	2	-3	3	1	0
0	8	3	-2	4	3	-8	1	2	0
0	0	-7	6	1	3	2	5	-3	0
0	3	9	4	0	9	7	2	6	0
0	-1	-5	-4	2	5	6	-4	-9	0
0	7	-2	8	9	1	2	-7	5	0
0	0	0	0	0	0	0	0	0	0

1	0	0
0	1	0
0	0	1

-9							

# Convolution

0	0	0	0	0	0	0	0	0	0
0	-8	-7	-9	4	4	-8	8	2	0
0	6	-1	3	-5	-7	0	9	-2	0
0	5	0	-9	-6	2	-3	3	1	0
0	8	3	-2	4	3	-8	1	2	0
0	0	-7	6	1	3	2	5	-3	0
0	3	9	4	0	9	7	2	6	0
0	-1	-5	-4	2	5	6	-4	-9	0
0	7	-2	8	9	1	2	-7	5	0
0	0	0	0	0	0	0	0	0	0

1	0	0
0	1	0
0	0	1

-9	-4						

# Convolution

0	0	0	0	0	0	0	0	0	0
0	-8	-7	-9	4	4	-8	8	2	0
0	6	-1	3	-5	-7	0	9	-2	0
0	5	0	-9	-6	2	-3	3	1	0
0	8	3	-2	4	3	-8	1	2	0
0	0	-7	6	1	3	2	5	-3	0
0	3	9	4	0	9	7	2	6	0
0	-1	-5	-4	2	5	6	-4	-9	0
0	7	-2	8	9	1	2	-7	5	0
0	0	0	0	0	0	0	0	0	0

1	0	0
0	1	0
0	0	1

-9	-4	-14	3	4	1	6	2
6	-18	-10	-12	-6	7	2	6
8	4	-6	-6	-11	-9	5	10
1	14	-1	-2	-7	-1	-5	5
9	5	9	8	14	1	3	-2
-2	-5	-1	11	16	6	-5	11
-3	6	14	7	7	8	8	-7
7	-3	3	5	3	7	-1	1

# Convolution

0	0	0	0	0	0	0	0	0	0
0	-8	-7	-9	4	4	-8	8	2	0
0	6	-1	3	-5	-7	0	9	-2	0
0	5	0	-9	-6	2	-3	3	1	0
0	8	3	-2	4	3	-8	1	2	0
0	0	-7	6	1	3	2	5	-3	0
0	3	9	4	0	9	7	2	6	0
0	-1	-5	-4	2	5	6	-4	-9	0
0	7	-2	8	9	1	2	-7	5	0
0	0	0	0	0	0	0	0	0	0

1	0	0
0	1	0
0	0	1

-9	-4	-14	3	4	1	6	2
6	-18	-10	-12	-6	7	2	6
8	4	-6	-6	-11	-9	5	10
1	14	-1	-2	-7	-1	-5	5
9	5	9	8	14	1	3	-2
-2	-5	-1	11	16	6	-5	11
-3	6	14	7	7	8	8	-7
7	-3	3	5	3	7	-1	1



# Activation

-9	-4	-14	3	4	1	6	2
6	-18	-10	-12	-6	7	2	6
8	4	-6	-6	-11	-9	5	10
1	14	-1	-2	-7	-1	-5	5
9	5	9	8	14	1	3	-2
-2	-5	-1	11	16	6	-5	11
-3	6	14	7	7	8	8	-7
7	-3	3	5	3	7	-1	1

ReLU



-9	-4	-14	3	4	1	6	2
6	-18	-10	-12	-6	7	2	6
8	4	-6	-6	-11	-9	5	10
1	14	-1	-2	-7	-1	-5	5
9	5	9	8	14	1	3	-2
-2	-5	-1	11	16	6	-5	11
-3	6	14	7	7	8	8	-7
7	-3	3	5	3	7	-1	1

# Activation

-9	-4	-14	3	4	1	6	2
6	-18	-10	-12	-6	7	2	6
8	4	-6	-6	-11	-9	5	10
1	14	-1	-2	-7	-1	-5	5
9	5	9	8	14	1	3	-2
-2	-5	-1	11	16	6	-5	11
-3	6	14	7	7	8	8	-7
7	-3	3	5	3	7	-1	1

ReLU



0	0	0	3	4	1	6	2
6	0	0	0	0	7	2	6
8	4	0	0	0	0	5	10
1	14	0	0	0	0	0	5
9	5	9	8	14	1	3	0
0	0	0	11	16	6	0	11
0	6	14	7	7	8	8	0
7	0	3	5	3	7	0	1

# Activation

-9	-4	-14	3	4	1	6	2
6	-18	-10	-12	-6	7	2	6
8	4	-6	-6	-11	-9	5	10
1	14	-1	-2	-7	-1	-5	5
9	5	9	8	14	1	3	-2
-2	-5	-1	11	16	6	-5	11
-3	6	14	7	7	8	8	-7
7	-3	3	5	3	7	-1	1

ReLU



0	0	0	3	4	1	6	2
6	0	0	0	0	7	2	6
8	4	0	0	0	0	5	10
1	14	0	0	0	0	0	5
9	5	9	8	14	1	3	0
0	0	0	11	16	6	0	11
0	6	14	7	7	8	8	0
7	0	3	5	3	7	0	1

# Activation

-9	-4	-14	3	4	1	6	2
6	-18	-10	-12	-6	7	2	6
8	4	-6	-6	-11	-9	5	10
1	14	-1	-2	-7	-1	-5	5
9	5	9	8	14	1	3	-2
-2	-5	-1	11	16	6	-5	11
-3	6	14	7	7	8	8	-7
7	-3	3	5	3	7	-1	1

ReLU



0	0	0	3	4	1	6	2
6	0	0	0	0	7	2	6
8	4	0	0	0	0	5	10
1	14	0	0	0	0	0	5
9	5	9	8	14	1	3	0
0	0	0	11	16	6	0	11
0	6	14	7	7	8	8	0
7	0	3	5	3	7	0	1

# Pooling

0	0	0	3	4	1	6	2
6	0	0	0	0	7	2	6
8	4	0	0	0	0	5	10
1	14	0	0	0	0	0	5
9	5	9	8	14	1	3	0
0	0	0	11	16	6	0	11
0	6	14	7	7	8	8	0
7	0	3	5	3	7	0	1


# Pooling

0	0	0	3	4	1	6	2
6	0	0	0	0	7	2	6
8	4	0	0	0	0	5	10
1	14	0	0	0	0	0	5
9	5	9	8	14	1	3	0
0	0	0	11	16	6	0	11
0	6	14	7	7	8	8	0
7	0	3	5	3	7	0	1


# Pooling

0	0	0	3	4	1	6	2
<b>6</b>	0	0	0	0	7	2	6
8	4	0	0	0	0	5	10
1	14	0	0	0	0	0	5
9	5	9	8	14	1	3	0
0	0	0	11	16	6	0	11
0	6	14	7	7	8	8	0
7	0	3	5	3	7	0	1

<b>6</b>			

# Pooling

0	0	0	3	4	1	6	2
6	0	0	0	0	7	2	6
8	4	0	0	0	0	5	10
1	14	0	0	0	0	0	5
9	5	9	8	14	1	3	0
0	0	0	11	16	6	0	11
0	6	14	7	7	8	8	0
7	0	3	5	3	7	0	1

6	3		



# Pooling

0	0	0	<b>3</b>	4	1	<b>6</b>	2
<b>6</b>	0	0	0	0	<b>7</b>	2	6
8	4	0	<b>0</b>	<b>0</b>	0	5	<b>10</b>
1	<b>14</b>	0	0	0	0	0	5
<b>9</b>	5	9	8	14	1	3	0
0	0	0	<b>11</b>	<b>16</b>	6	0	<b>11</b>
0	6	<b>14</b>	7	7	<b>8</b>	<b>8</b>	0
<b>7</b>	0	3	5	3	7	0	1

<b>6</b>	<b>3</b>	<b>7</b>	<b>6</b>
<b>14</b>	<b>0</b>	<b>0</b>	<b>10</b>
<b>9</b>	<b>11</b>	<b>16</b>	<b>11</b>
<b>7</b>	<b>14</b>	<b>8</b>	<b>8</b>