



Are you at the right place ?

20 July, 1pm - 4pm AEST: Deep Learning with Images and MATLAB

Overview

Please join MathWorks and learn how to get started with MATLAB for Deep Learning with Images. In this hands-on workshop, we will introduce you to fundamentals of Deep Learning with Images. You'll have the opportunity to try out specific examples using MATLAB tools. The hands-on component of the workshop will be run via MATLAB Online – so attendees do NOT need to have MATLAB locally installed on their computers.

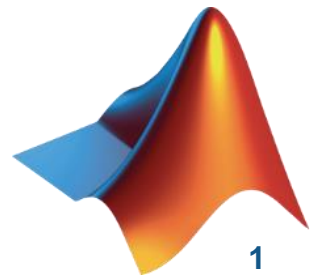
Highlights

- Learn the Deep Learning image classification workflow in MATLAB
- Image Data management
- Network assembly, training
- Experiment management
- Create a Convolution Neural Network (CNN) from scratch
- Programmatically and using APPs
- Explore how to access and adjust pretrained models (transfer learning)
- Explore how to evaluate the network and improve its accuracy

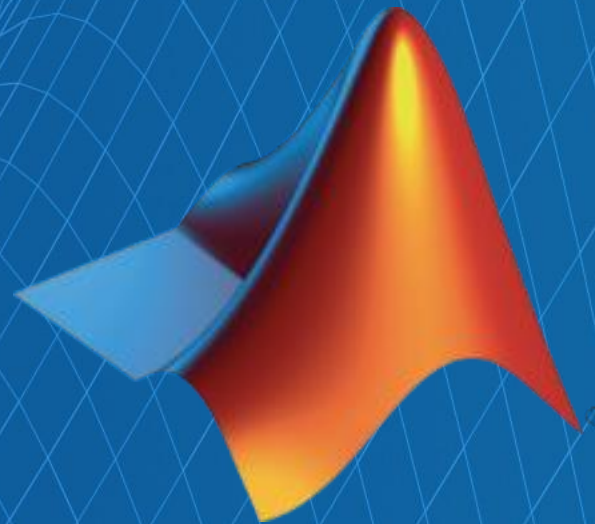
Pework:

- Attendees will need to bring their own laptops **(MATLAB does NOT need to be installed)**
- This workshop is for NCI users who have a MATLAB license. For more information about MATLAB license, please check our webpage about MATLAB license supported groups. The workshop will be held at ANU campus. **See the following FAQ for instructions on how to create a MathWorks account: <https://www.mathworks.com/videos/create-a-mathworks-account-using-a-matlab-portal-1600159919958.html>**
- Attendees should have a basic level of understanding of MATLAB syntax. The FREE online course MATLAB OnRamp would be a recommended prerequisite for any new users of MATLAB.

Any issues doing this ?



Workshop Setup



***Should take
5 minutes !***

Setup overview

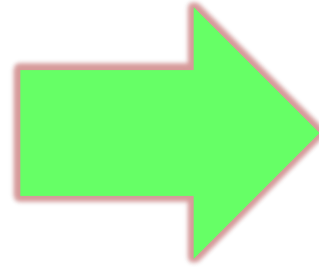
- **Background:**
 - Today we'll be using MATLAB Online
 - A version of MATLAB that runs in your web Browser
 - But please use the special MATLAB Online link that I will share with you shortly.
 - This gives access to Cloud GPUs

- **STEPS**
 1. Confirm you are using a supported Browser
 2. Make sure you have a MathWorks Account
 3. Use the special MATLAB Online link
 4. Copy the Workshop files
 5. Confirm you have access to a cloud GPU

*Should take
5-10 minutes*

A Word on Browser support

Highly recommended to use Google chrome



Eg: our **deepNetworkDesigner** APP is only supported in Chrome

Stop Stare bookmark



MathWorks - License Center - Access

mathworks.com/licensecenter/classroom/DL_3467201/

MathWorks® Products Solutions Academia Support Community Events

MATLAB & Simulink

Access MATLAB for your Deep Learning Workshop

MathWorks is pleased to provide a special license to you as a course participant to use for your Deep Learning Workshop. This is a limited license for the duration of your course and is intended to be used only for course work and not for government, research, commercial, or other organization use.

Course Name:	ANU Hands-on MATLAB workshop on Deep Learning with Images
Organization:	MathWorks Deep Learning
Ending:	20 Jul 2021

Access MATLAB Online

https://www.mathworks.com/licensecenter/classroom/DL_3467201/

Download the Setup PDF



<https://drive.matlab.com/sharing/56c9e7cb-400c-4fb0-93c8-72f438d7fe46>

Set-Up Instructions – part 1 of 3

Step 1: MATLAB Drive - Login with your MathWorks Account used to register for the event

Login to your **MATLAB Drive** at <https://drive.matlab.com/login>

- Use the email address that you submitted to register for the event.

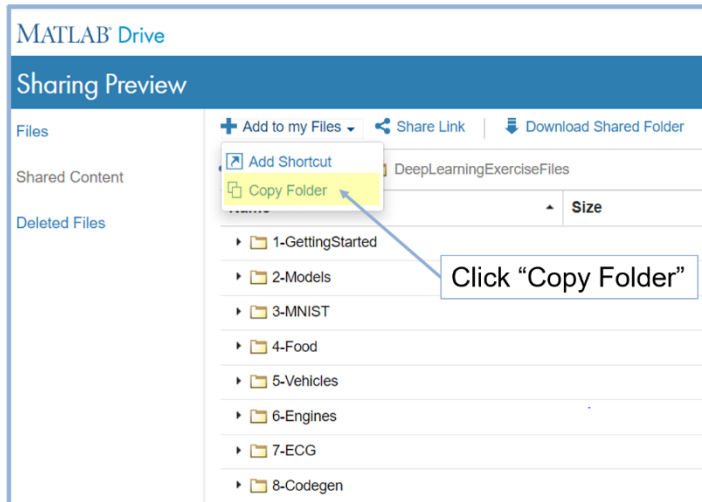
If creating a new account, visit <https://www.mathworks.com/mwaccount/register>

Step 2: Copy Workshop Files

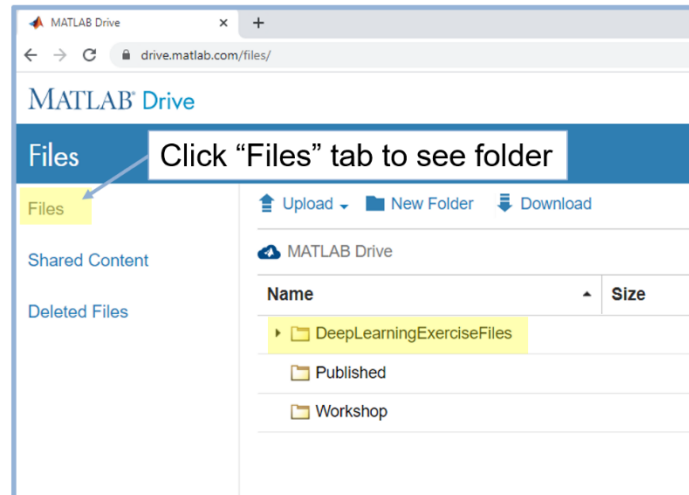
- Click [HERE](#) and accept the shared files from Pitambar Dayal.
 - FYI: the full address is:
 - <https://drive.matlab.com/sharing/e1d60207-94f1-4af3-ae4-8174370eb421>

Note: If you are unable to access the above link, wait 30 minutes and try again.

Step 2a

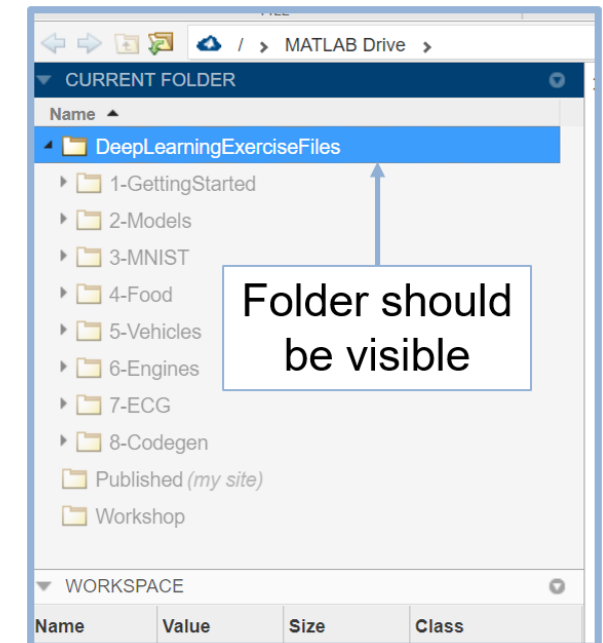


Step 2b



Step 3: Log into the Workshop MATLAB Online and Confirm Web Browser

- Visit the following URL and login to access MATLAB Online
 - https://www.mathworks.com/licensecenter/classroom/DL_3467201/
- **Note:** If you are unable to login or access the above link, wait a few minutes and try again. If having issues with your browser, Chrome has been tested and usually works well.



- Your current folder browser should have the folder you copied over.

Set-Up Instructions – part 2 of 3

3 – Navigate into the **DeepLearningExerciseFiles** folder

The screenshot shows the MATLAB Online R2021a interface. The browser address bar is `workshop-matlab.mathworks.com`. The navigation pane shows the current folder path as `MATLAB Drive > DeepLearningExerciseFiles`, which is highlighted with a red box. Below the path, a list of folders is displayed:

- 1-GettingStarted
- 2-Models
- 3-MNIST
- 4-Food
- 5-Vehicles
- 6-Engines
- 7-ECG
- 8-Codegen

A red bracket on the right side of the folder list is accompanied by the text: "You should see something like this".

4 – Confirm you have a reserved GPU

The screenshot shows the MATLAB Command Window with the command `>> gpuDevice` entered and highlighted by a red box and a green arrow. The output is as follows:

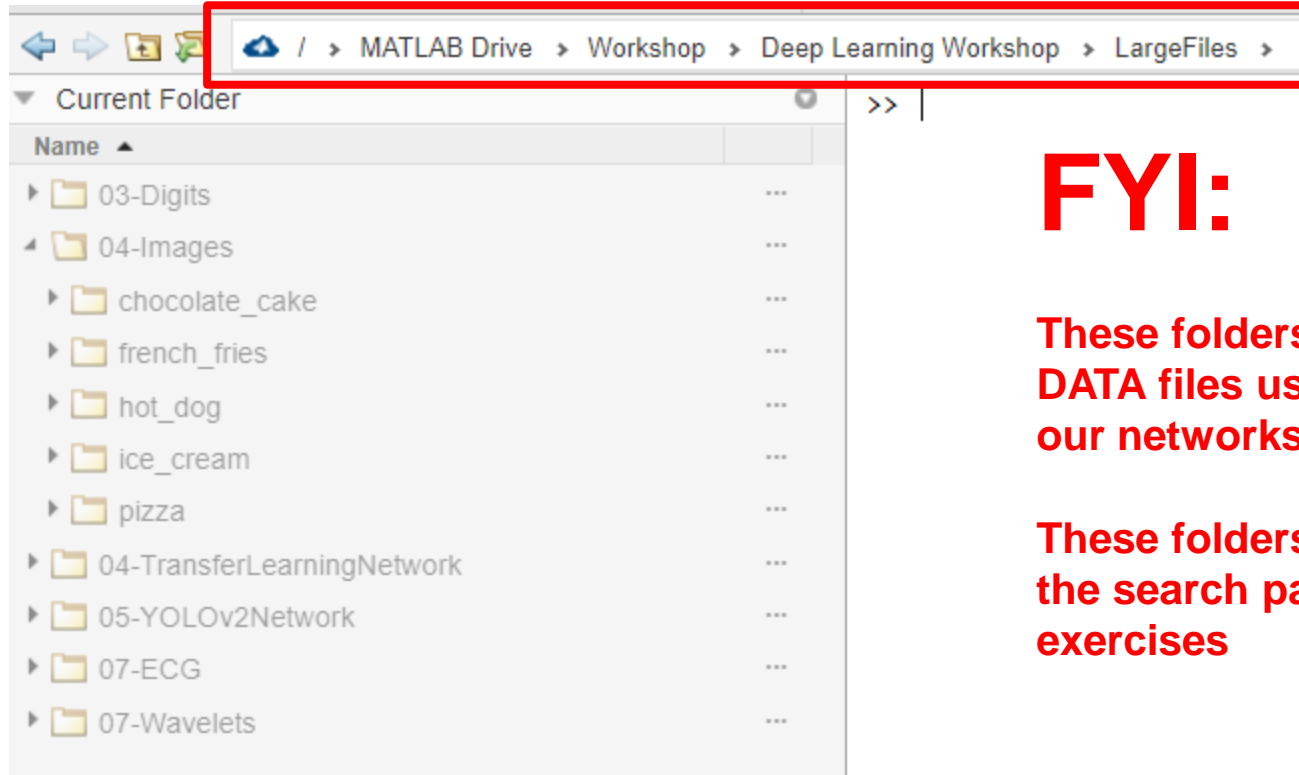
```
ans =

  CUDADevice with properties:

        Name: 'Tesla T4'
        Index: 1
  ComputeCapability: '7.5'
  SupportsDouble: 1
    DriverVersion: 11
    ToolkitVersion: 11
  MaxThreadsPerBlock: 1024
  MaxShmemPerBlock: 49152
  MaxThreadBlockSize: [1024 1024 64]
        MaxGridSize: [2.1475e+09 65535 65535]
        SIMDwidth: 32
        TotalMemory: 1.5844e+10
  AvailableMemory: 1.5582e+10
  MultiprocessorCount: 40
        ClockRateKHz: 1590000
        ComputeMode: 'Default'
```

A red bracket on the right side of the output is accompanied by the text: "You should see something like this".

Set-Up Instructions – part 3 of 3



FYI:

These folders contain the DATA files used for training our networks.

These folders will be added to the search path during the exercises

Deep Learning Toolbox

172 shipping examples to explore

230 functions/classes

Documentation

Search Help

CONTENTS

« Documentation Home

« Examples

Category

Deep Learning Toolbox

- Get Started with Deep Learning Toolbox 8
- Deep Learning with Images 17
- Deep Learning with Time Series, Sequences, and Text 20
- Deep Learning Tuning and Visualization 21
- Deep Learning in Parallel and in the Cloud 8
- Deep Learning Applications 50
- Deep Learning Import, Export, and Customization 23
- Deep Learning Data Preprocessing 9
- Deep Learning Code Generation 21

Type

- All 172
- MATLAB 165
- Simulink 7

Deep Learning Toolbox — Examples

Get Started with Deep Learning Toolbox

Classify Webcam Images Using Deep Learning

Classify images from a webcam in real time using the pretrained deep convolutional neural network GoogLeNet.

[Open Script](#)

Train Deep Learning Network to Classify New Images

Use transfer learning to retrain a convolutional neural network to classify a new set of images.

[Open Live Script](#)

Time Series Forecasting Using Deep Learning

Forecast time series data using a long short-term memory (LSTM) network.

[Open Live Script](#)

Reuse Pretrained Network

Use final layers | Train network | Predict

file:///C:/MATLAB/R2020a/help/deeplearning/gs/classify-image-using-pretrained-network.html

Documentation

Search Help

CONTENTS

« Documentation Home

« Functions

Category

By Category | [Alphabetical List](#)

Deep Learning Toolbox

- Deep Learning with Images 70
- Deep Learning with Time Series, Sequences, and Text 27
- Deep Learning Tuning and Visualization 14
- Deep Learning in Parallel and in the Cloud 1
- Deep Learning Applications 1
- Deep Learning Import, Export, and Customization 48
- Deep Learning Code Generation 4
- Function Approximation, Clustering, and Control 65

Extended Capability

- C/C++ Code Generation 21
- GPU Code Generation 52
- Automatic Parallel Support 1
- GPU Arrays 22

Deep Learning Toolbox — Functions

Deep Learning with Images

trainingOptions	Options for training deep learning neural network
trainNetwork	Train neural network for deep learning
analyzeNetwork	Analyze deep learning network architecture
squeezeNet	SqueezeNet convolutional neural network
googLeNet	GoogLeNet convolutional neural network
inceptionv3	Inception-v3 convolutional neural network
densenet201	DenseNet-201 convolutional neural network
mobilenetv2	MobileNet-v2 convolutional neural network
resnet18	ResNet-18 convolutional neural network
resnet50	ResNet-50 convolutional neural network
resnet101	ResNet-101 convolutional neural network
xception	Xception convolutional neural network
inceptionresnetv2	Pretrained Inception-ResNet-v2 convolutional neural network
nasnetlarge	Pretrained NASNet-Large convolutional neural network
nasnetmobile	Pretrained NASNet-Mobile convolutional neural network
shuffleNet	Pretrained ShuffleNet convolutional neural network
darknet19	DarkNet-19 convolutional neural network
darknet53	DarkNet-53 convolutional neural network
alexnet	AlexNet convolutional neural network
vgg16	VGG-16 convolutional neural network

file:///C:/MATLAB/R2020a/help/deeplearning/ref/nasnetmobile.html

Fun fact:

The **MATLAB Campus License** is almost at every university in AU/NZ !

AU: 33 out of 40

NZ: 6 out of 8

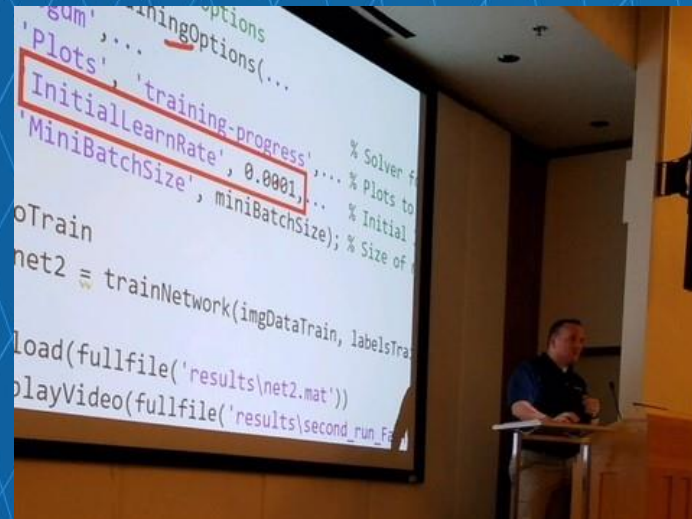
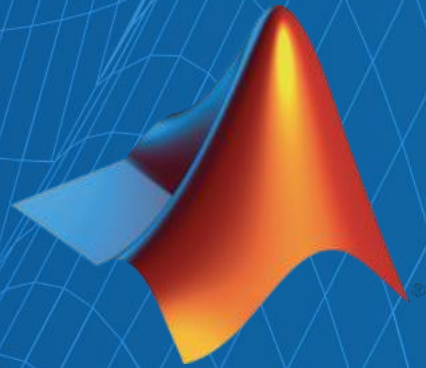
So ?

- Every student
- **Every Product**
- Every Computer(campus)
- Every Computer(personal)



MATLAB	Global Optimization Toolbox	Polyspace Code Prover	Simulink Coder
Simulink	GPU Coder	Powertrain Blockset	Simulink Compiler
5G Toolbox	HDL Coder	Predictive Maintenance Toolbox	Simulink Control Design
Aerospace Blockset	HDL Verifier	Radar Toolbox	Simulink Coverage
Aerospace Toolbox	Image Acquisition Toolbox	Reinforcement Learning Toolbox	Simulink Design Optimization
Antenna Toolbox	Image Processing Toolbox	RF Blockset	Simulink Design Verifier
Audio Toolbox	Instrument Control Toolbox	RF Toolbox	Simulink Desktop Real-Time
Automated Driving Toolbox	Lidar Toolbox	Risk Management Toolbox	Simulink PLC Coder
AUTOSAR Blockset	LTE Toolbox	RoadRunner	Simulink Real-Time
Bioinformatics Toolbox	Mapping Toolbox	RoadRunner Asset Library	Simulink Report Generator
Communications Toolbox	MATLAB Coder	Robotics System Toolbox	Simulink Requirements
Computer Vision Toolbox	MATLAB Compiler	Robust Control Toolbox	Simulink Test
Control System Toolbox	MATLAB Compiler SDK	ROS Toolbox	SoC Blockset
Curve Fitting Toolbox	MATLAB Parallel Server	Satellite Communications Toolbox	Spreadsheet Link
Data Acquisition Toolbox	MATLAB Production Server	Sensor Fusion and Tracking Toolbox	Stateflow
Database Toolbox	MATLAB Report Generator	SerDes Toolbox	Statistics and Machine Learning Toolbox
Datafeed Toolbox	MATLAB Web App Server	Signal Processing Toolbox	Symbolic Math Toolbox
DDS Blockset	Mixed-Signal Blockset	SimBiology	System Composer
Deep Learning HDL Toolbox	Model Predictive Control Toolbox	SimEvents	System Identification Toolbox
Deep Learning Toolbox	Model-Based Calibration Toolbox	Simscape	Text Analytics Toolbox
DSP System Toolbox	Motor Control Blockset	Simscape Driveline	UAV Toolbox
Econometrics Toolbox	Navigation Toolbox	Simscape Electrical	Vehicle Dynamics Blockset
Embedded Coder	OPC Toolbox	Simscape Fluids	Vehicle Network Toolbox
Filter Design HDL Coder	Optimization Toolbox	Simscape Multibody	Vision HDL Toolbox
Financial Instruments Toolbox	Parallel Computing Toolbox	Simulink 3D Animation	Wavelet Toolbox
Financial Toolbox	Partial Differential Equation Toolbox	Simulink Check	Wireless HDL Toolbox
Fixed-Point Designer	Phased Array System Toolbox	Simulink Code Inspector	WLAN Toolbox
Fuzzy Logic Toolbox	Polyspace Bug Finder		

Hands-on Deep Learning Workshop



DEMOs

The Experiment Manager APP

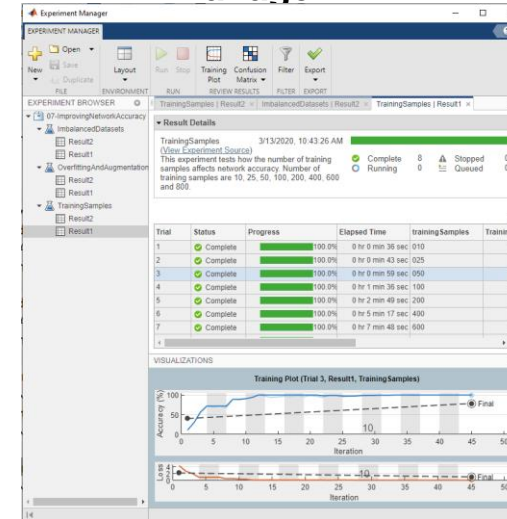
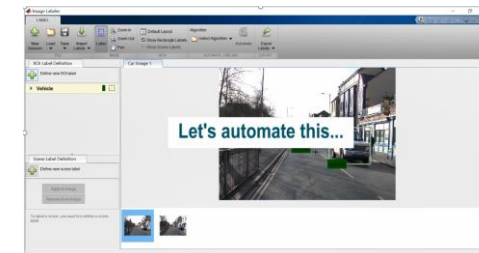


Image Labeller APP



Agenda

Gentle "Warm up" stretches

interesting bits

Exercise 01

Exercise 02

Exercise 03

Exercise 04

Exercise 05

Exercise 07

Deep Learning in 6 lines

Managing Data files

Digit classification

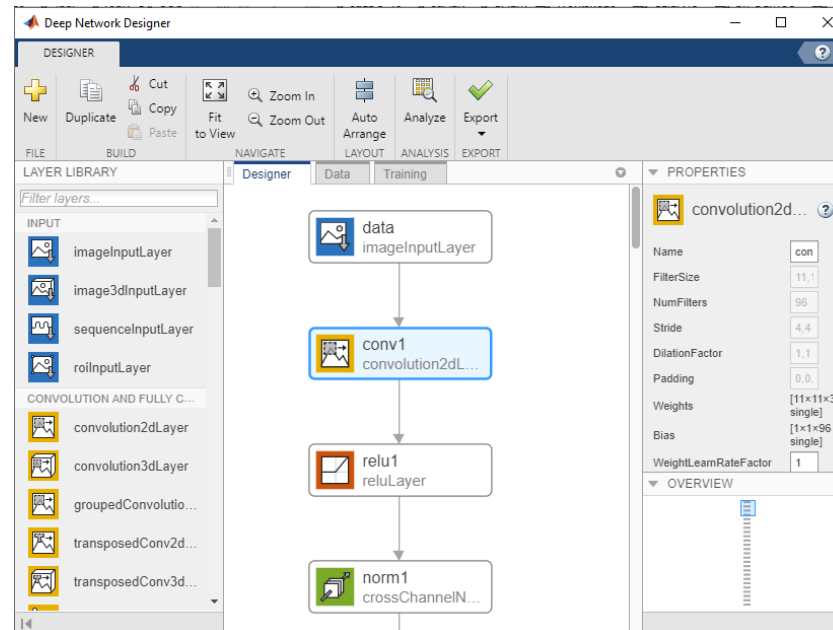
Transfer Learning

YOLO Object detector

ECG Heart Condition classifier

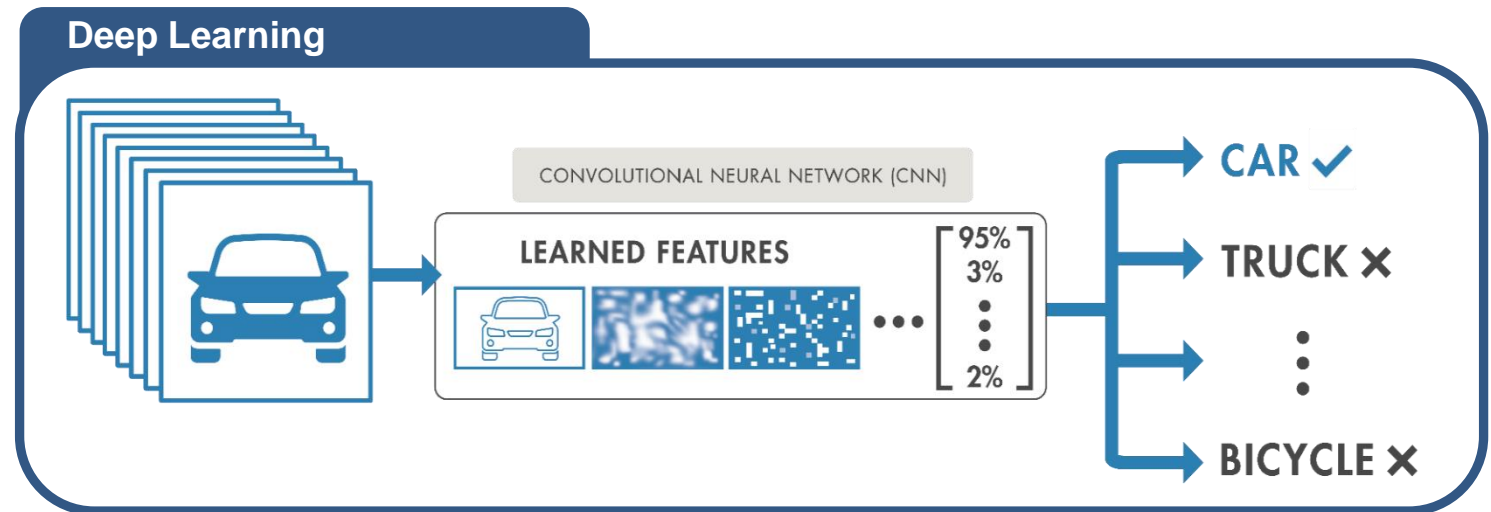
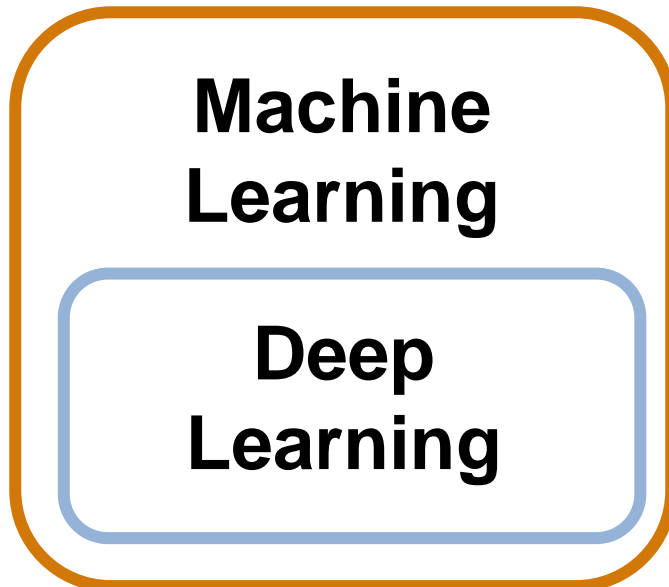
WORKFLOW#1: Deep Network Designer APP

WORKFLOW#2: Manual Coding

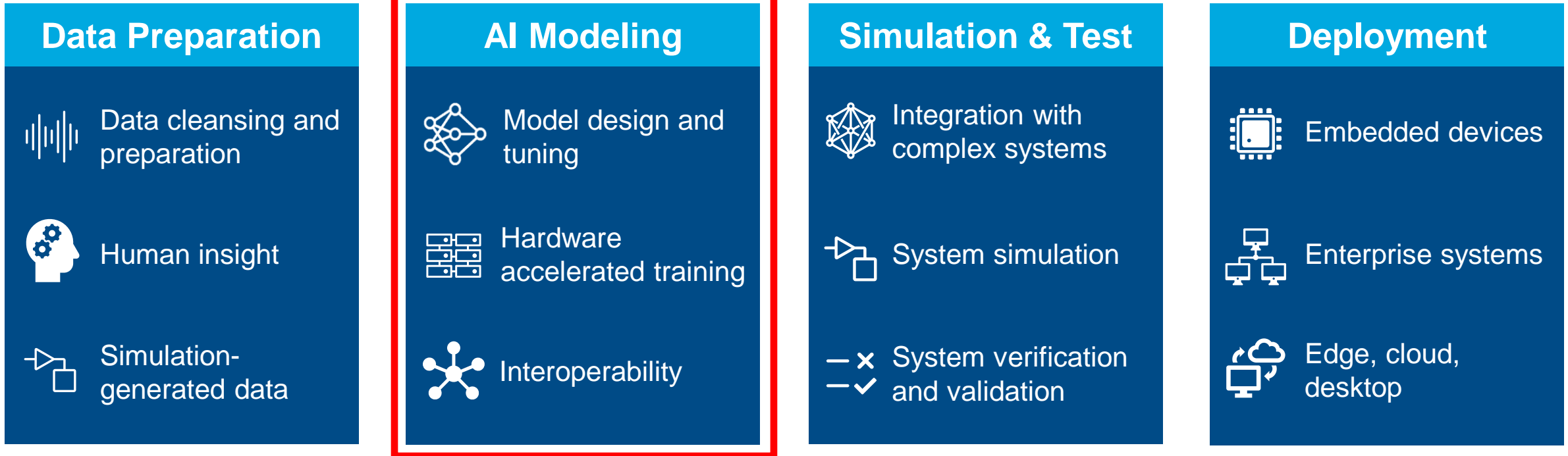


What is Deep Learning?

- Subset of machine learning with **automatic feature extraction**
 - Learns features and tasks directly from data
- Accuracy can surpass traditional ML Algorithms



Deep Learning Workflow



Today we'll focus on this part

Deep Learning and AI in Industry



Oversteering Detection



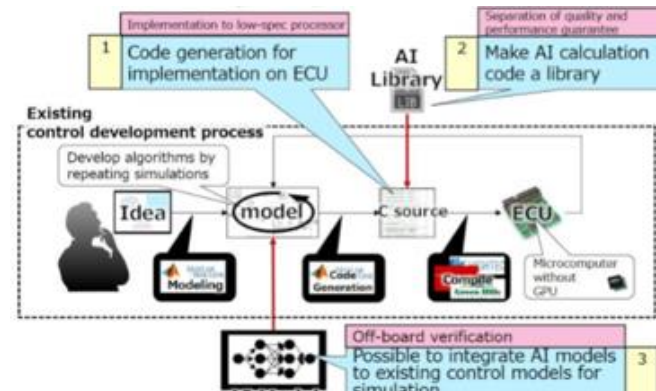
Digital Twins of Compressors



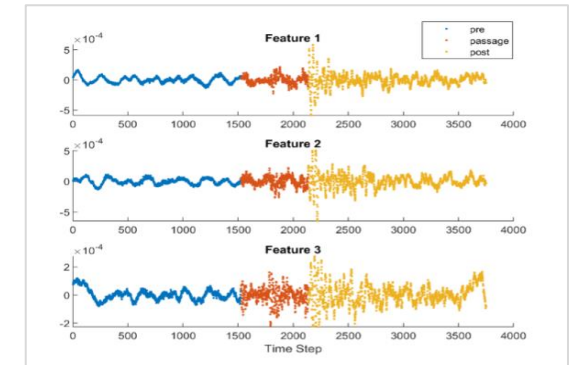
Energy use optimization



Automatic Defect Detection



ECU Vehicle Control **DENSO**

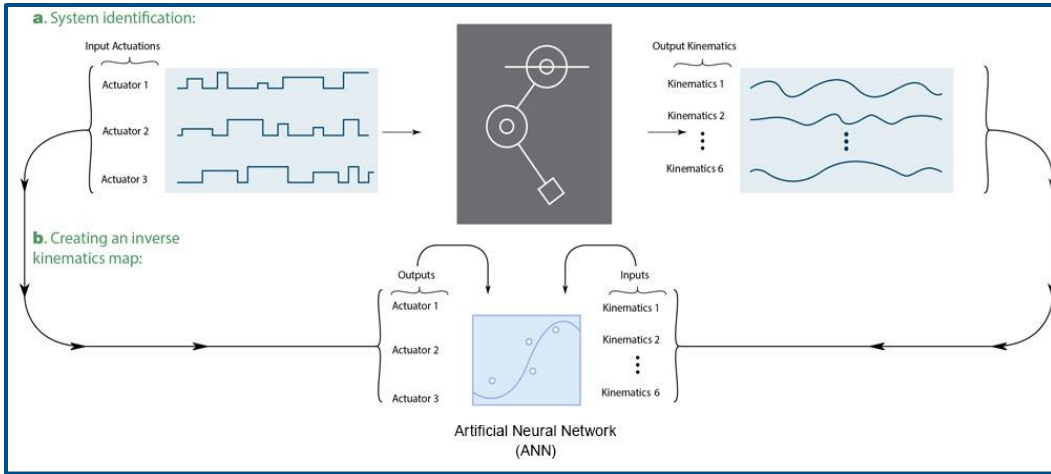


Seismic Event Detection

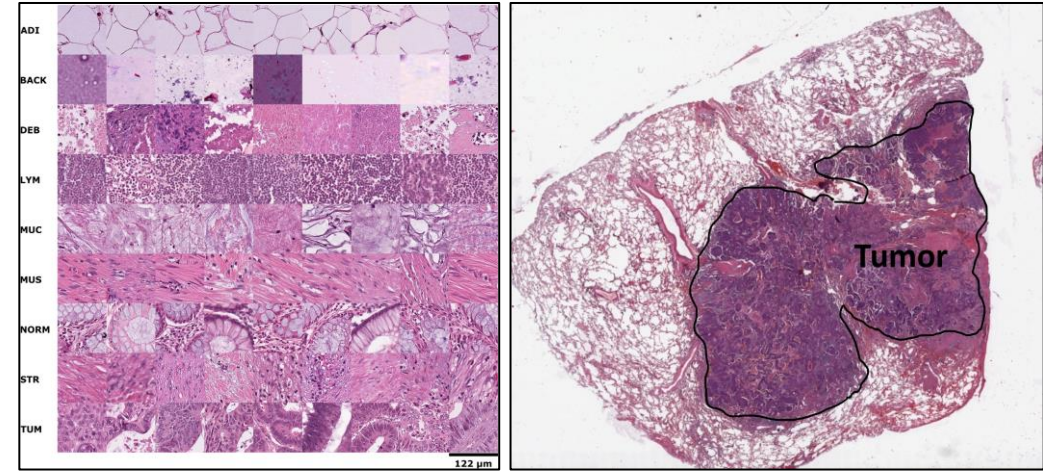


Deep Learning and AI in Research

University of Southern California



Reinforcement Learning for Robotic Arm



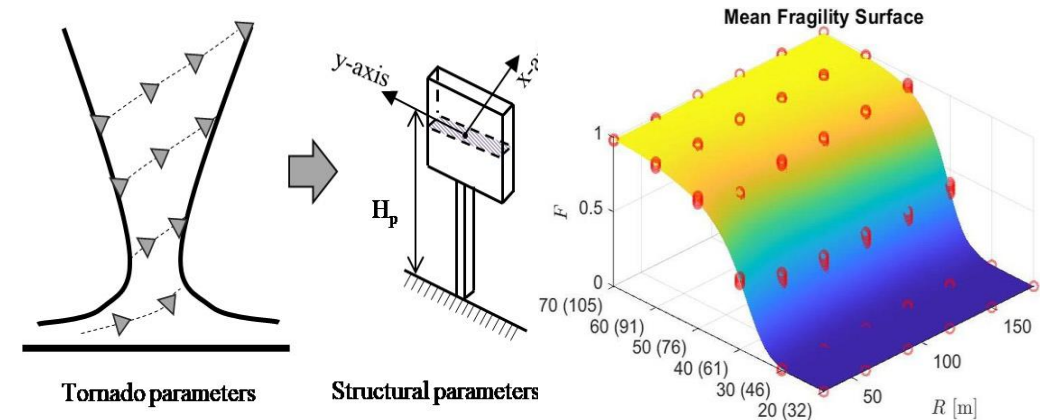
Deep Learning for Tumor Detection

DKFZ Heidelberg

University of Twente



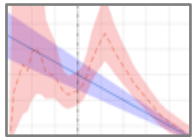
Augmented Reality of blood flow



Neural Networks simulate tornadic wind load

Northeast University

MathWorks Focus on Deep Learning and AI for Engineering and Science



Predictive Maintenance

- [Bearing Prognosis](#)
- [Pump Fault Diagnosis](#)

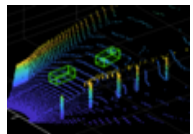
Predictive Maintenance
Toolbox™



Land-Use Classification

- [Semantic Segmentation for Multispectral Images](#)

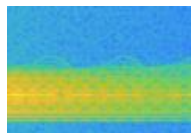
Image Processing
Toolbox™



Lidar

- [Lidar Point Cloud Semantic Segmentation](#)
- [3-D Object Detection Using PointPillars](#)

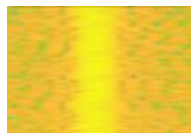
Lidar
Toolbox™



Radar

- [Radar Waveform Classification](#)
- [Pedestrian and Bicyclist Classification](#)

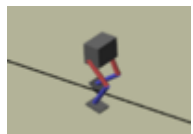
Phased Array
System Toolbox™



Wireless Communications

- [Modulation Classification](#)
- [Detect WLAN Router Impersonation](#)

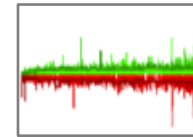
Communications
Toolbox™



Reinforcement Learning

- [Train Biped Robot to Walk](#)
- [PMSM Motor Control](#)

Reinforcement
Learning Toolbox™



Computational Finance

- [Machine Learning for Statistical Arbitrage](#)

Financial
Toolbox™



Robotics

- [Avoid Obstacles using Reinforcement Learning](#)

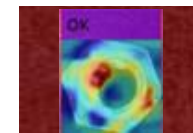
Robotics System
Toolbox™



Automated Driving

- [Deep Learning Vehicle Detector](#)
- [Occupancy Grid with Semantic Segmentation](#)

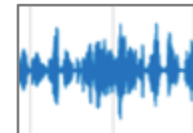
Automated
Driving Toolbox™



Visual Inspection

- [Manufacturing Defect Detection](#)
- [Anomaly Detection for Cloth Manufacturing](#)

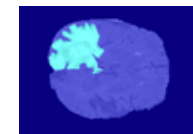
Image Processing
Toolbox™



Audio

- [Speech Command Recognition](#)
- [Cocktail Party Source Separation](#)

Audio
Toolbox™



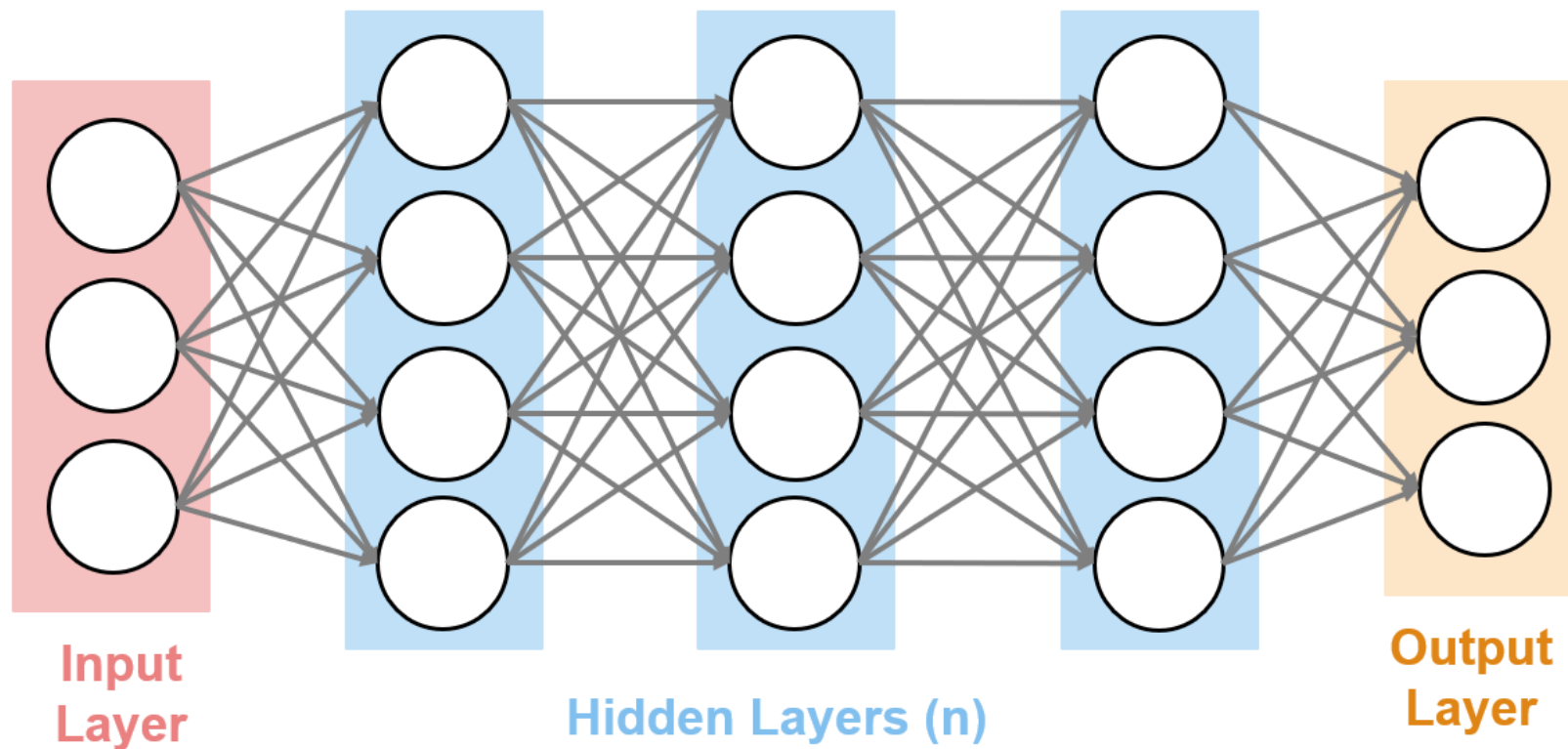
Medical Imaging

- [3-D Brain Tumor Segmentation](#)
- [Breast Cancer Tumor Classification](#)

Image Processing
Toolbox™

Deep Learning Models are Neural Networks

- Deep neural networks have many layers
- Data is passed through the network, and the layer parameters are updated (training)

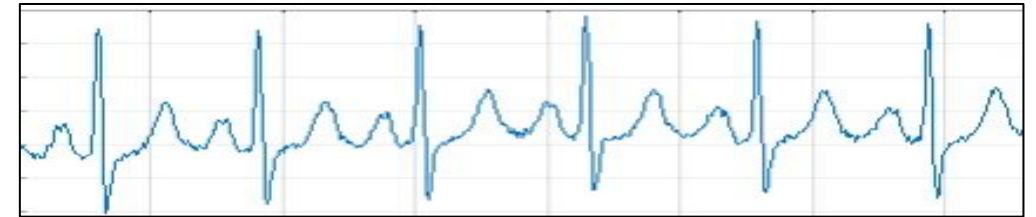


Deep Learning Networks Take in Numeric Data



199	206	208	201	188	178	165	164	180
202	205	202	188	176	169	178	186	183
203	206	189	178	181	183	182	154	87
203	192	184	186	177	167	153	181	192
191	182	176	166	153	141	136	180	227
166	165	154	154	138	137	169	170	211
158	150	145	183	144	156	158	154	179
143	51	98	144	129	130	143	178	123
107	50	33	95	152	173	192	159	87
104	100	84	120	132	172	131	64	94
119	101	97	81	90	109	87	106	111
127	122	110	97	108	120	133	131	134
111	117	108	119	131	143	146	141	156
126	122	113	119	139	142	155	161	151
129	126	130	111	103	130	149	149	156
138	128	136	144	136	129	134	122	145
154	133	134	141	168	150	126	127	151

Images are a numeric matrix



131 158 185 212 239 266 293 320 347 374 376

Signals are numeric vectors

The Bird Flies = [0 13 5 6]
 The Leaf Is Brown = [13 3 11 2]

Text is processed as numeric vectors

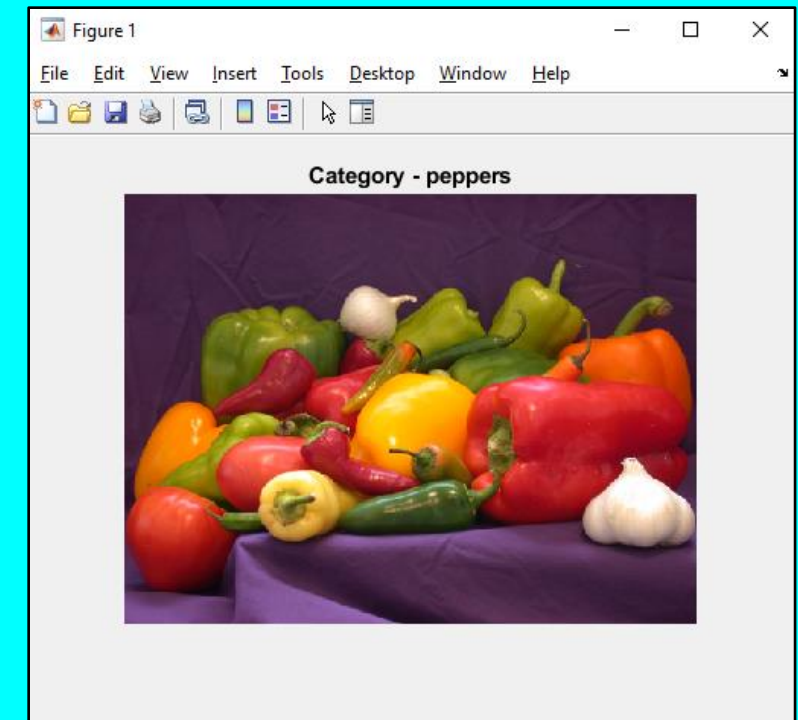
Exercise 1 – Deep Learning in 6 Lines of Code

Purpose:

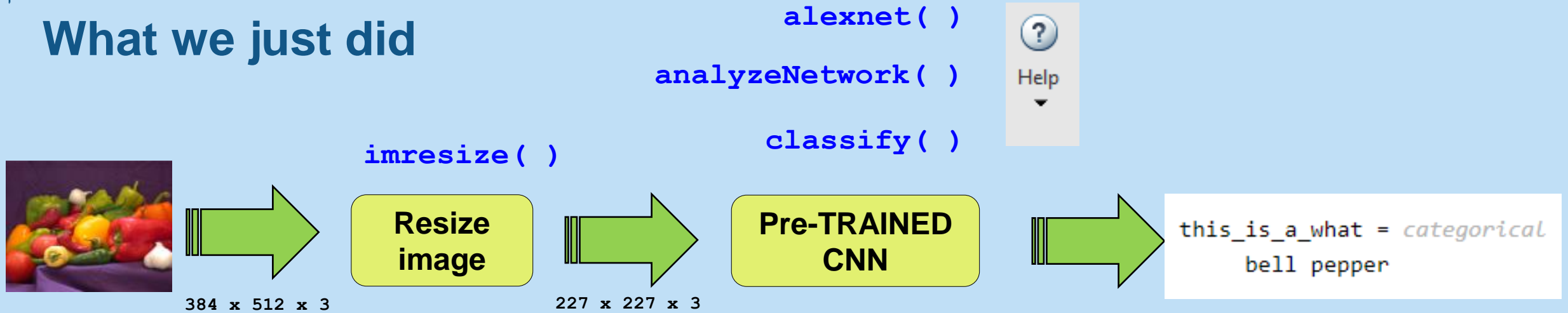
- Use a **PRETRAINED** neural network to classify an image
- Introduction to some basic functions

To Do:

1. Open **Work_GettingStarted.mlx**
2. Follow along with instructor



What we just did

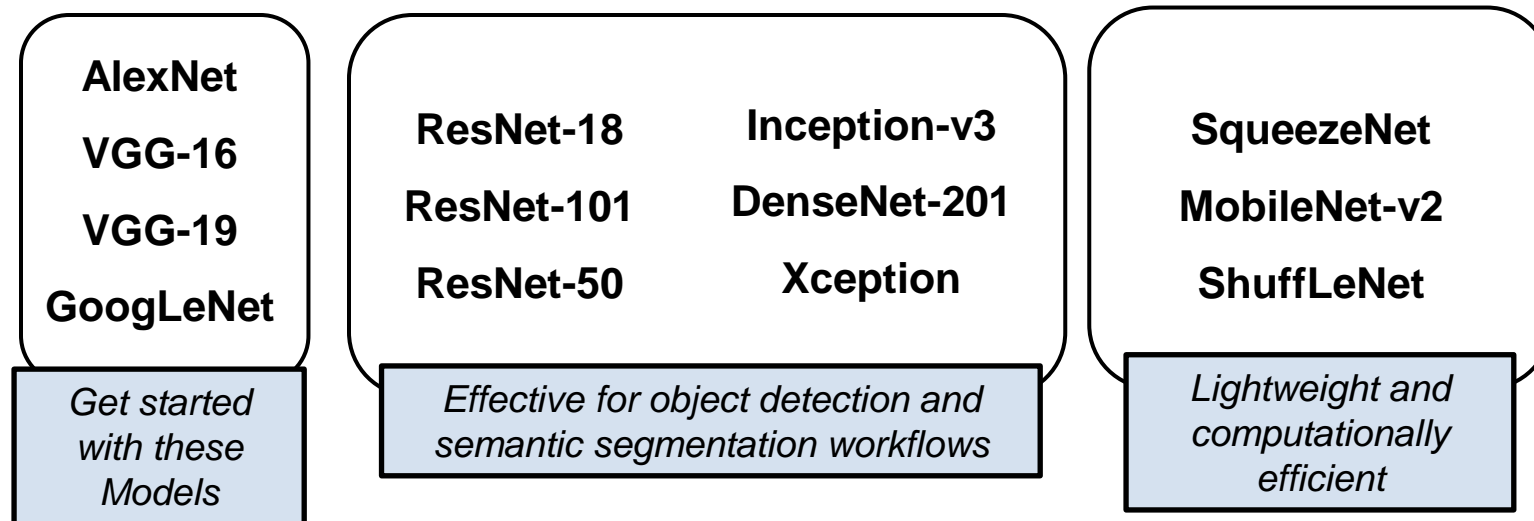


Let's discuss the following

1. What is a convolutional Neural Network (CNN) ?
2. How do you assemble them ?
3. How do you train them ?
4. How do I assess their performance ?
 - Is it good at what it does ?

We Can Build Networks from Scratch or Use Pretrained Models

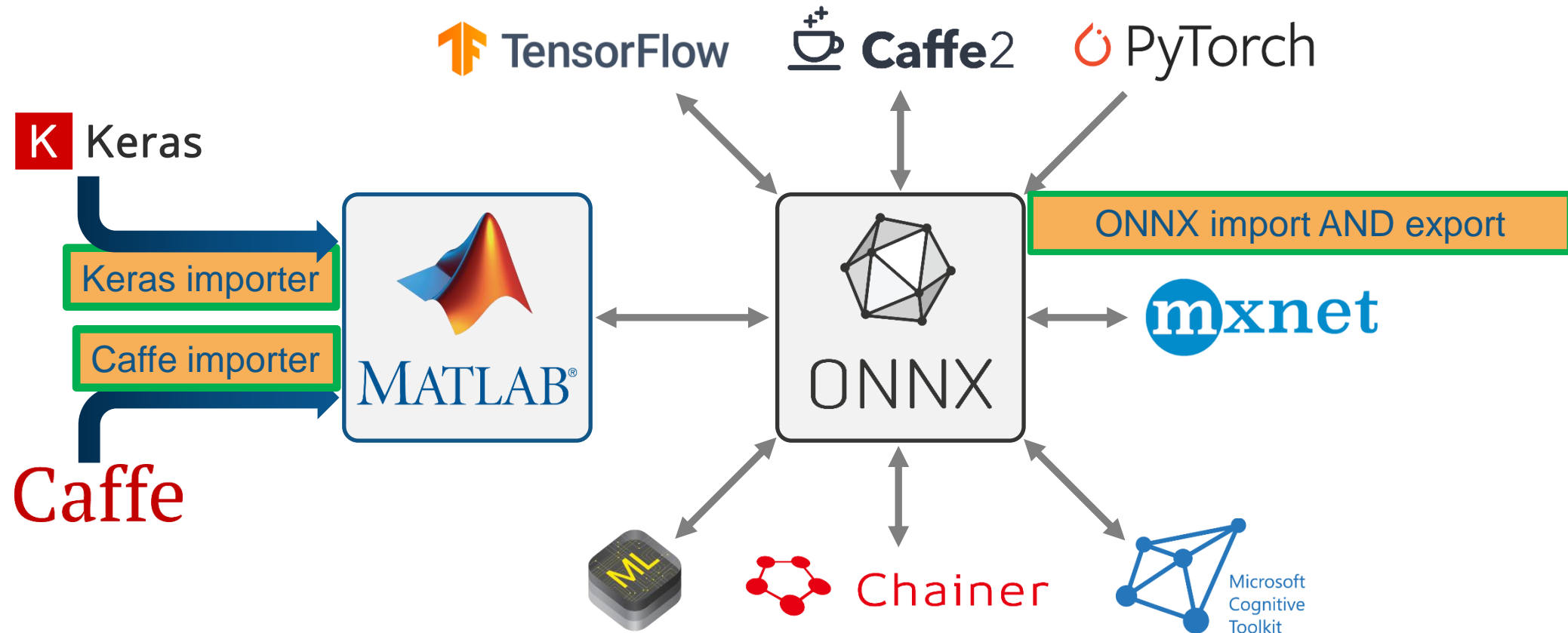
- Pretrained models have predefined layer orders and parameter values
- Can be used directly for inference (AlexNet Example)



Full list of models available [HERE](#)

Access Pretrained Models from Within MATLAB or Import from the Web

<https://www.mathworks.com/help/releases/R2021a/deeplearning/deep-learning-import-export-and-customization.html>



Exercise 2 – Models

Purpose:

- Classify Images using pretrained models.
- See how different network architectures affect results.
- Use **datastores** to access data efficiently

To Do:

- Open **Work_Models.mlx**

Pick a network to load in.

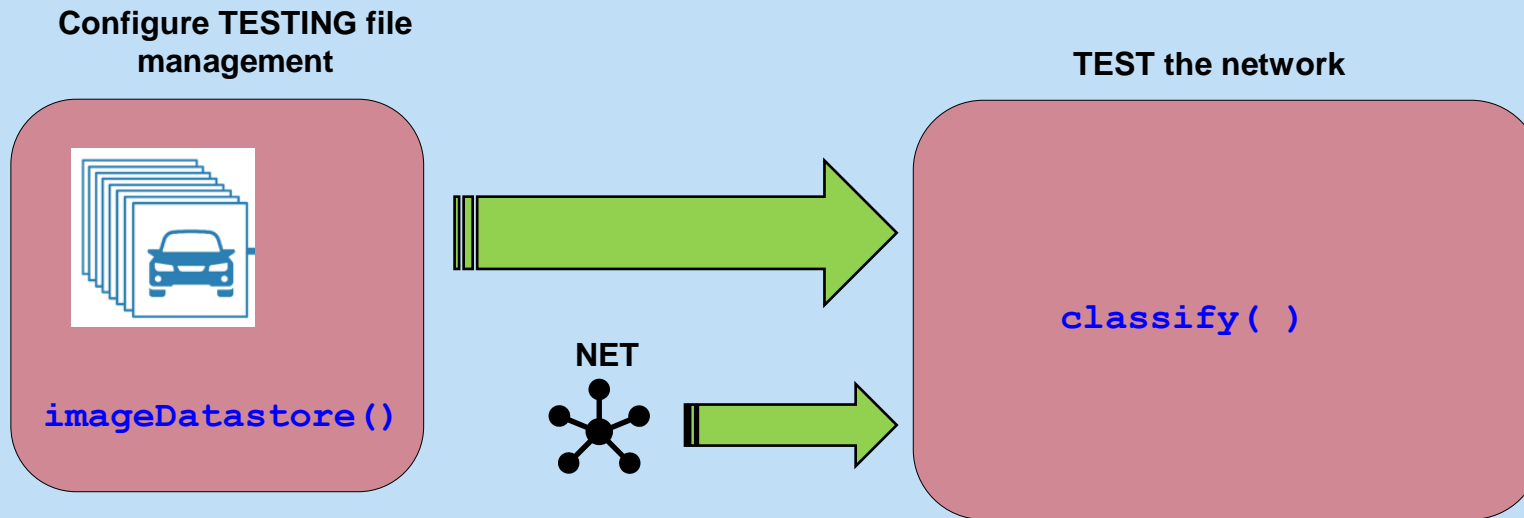
net = googlenet

alexnet
googlenet
vgg16
resnet50
inception
inceptionv3
squeezeenet
nasnetlarge

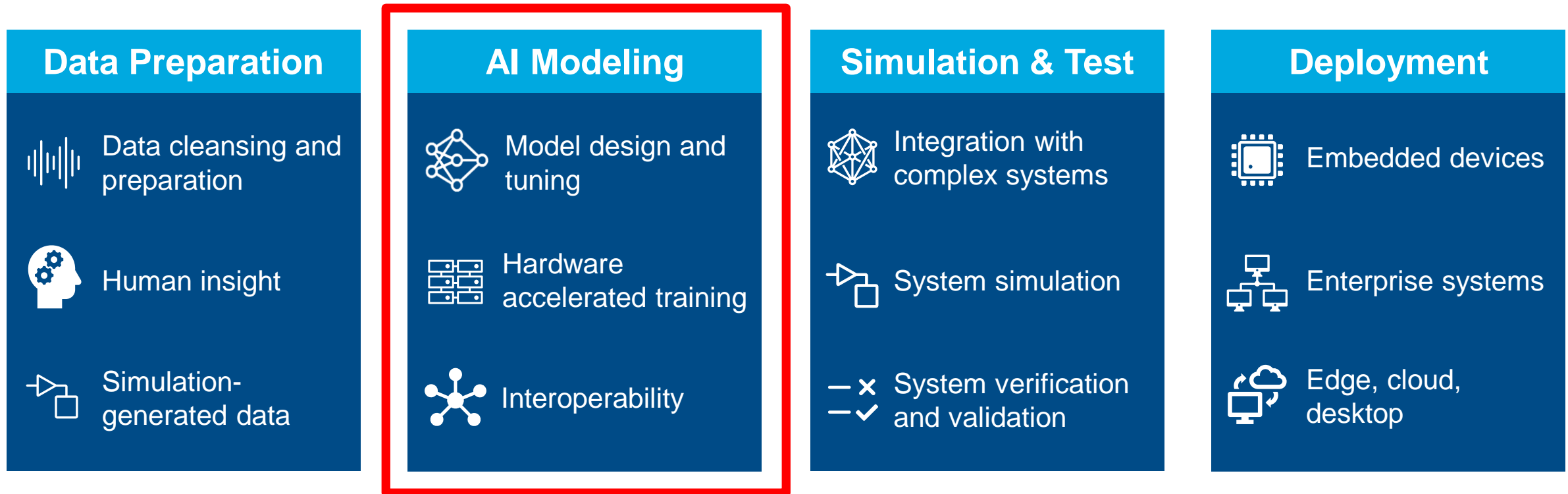
% Insp
% analy
net.La
% Insp
% Look
class

data
conv1-7x7_s2
conv1-relu_7x7
pool1-3x3_s2
pool1-norm1
conv2-3x3_r...
conv2-relu_3...
conv2-3x3
conv2-relu_3x3
conv2-norm2
pool2-3x3_s2
inception_3a-...
inception_3a-...
inception_3a-...
inception_3a-...
inception_3a-...
inception_3a-...
inception_3b-...
inception_3b-...
inception_3b-...
inception_3b-...
inception_3b-...
inception_3b-...

What we just did



Pretrained models aren't always enough. We may have to build and train networks from scratch

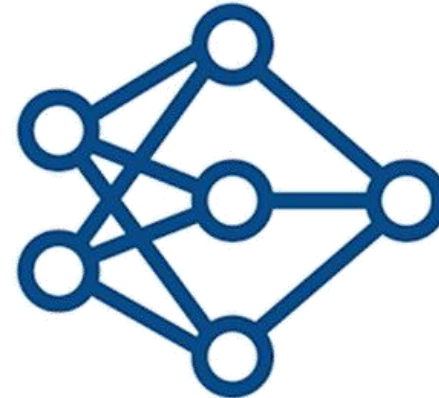


.... So let's talk about building and training models from scratch

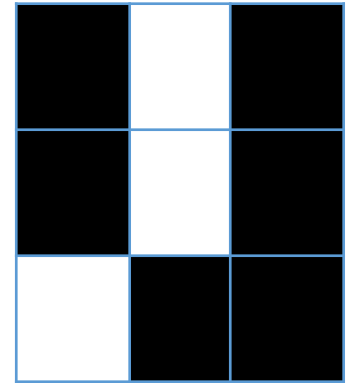
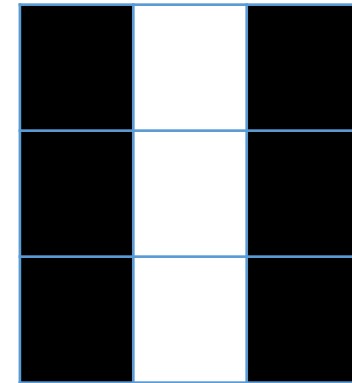
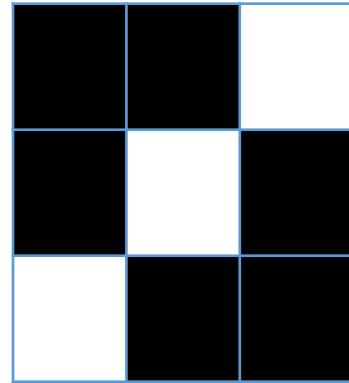
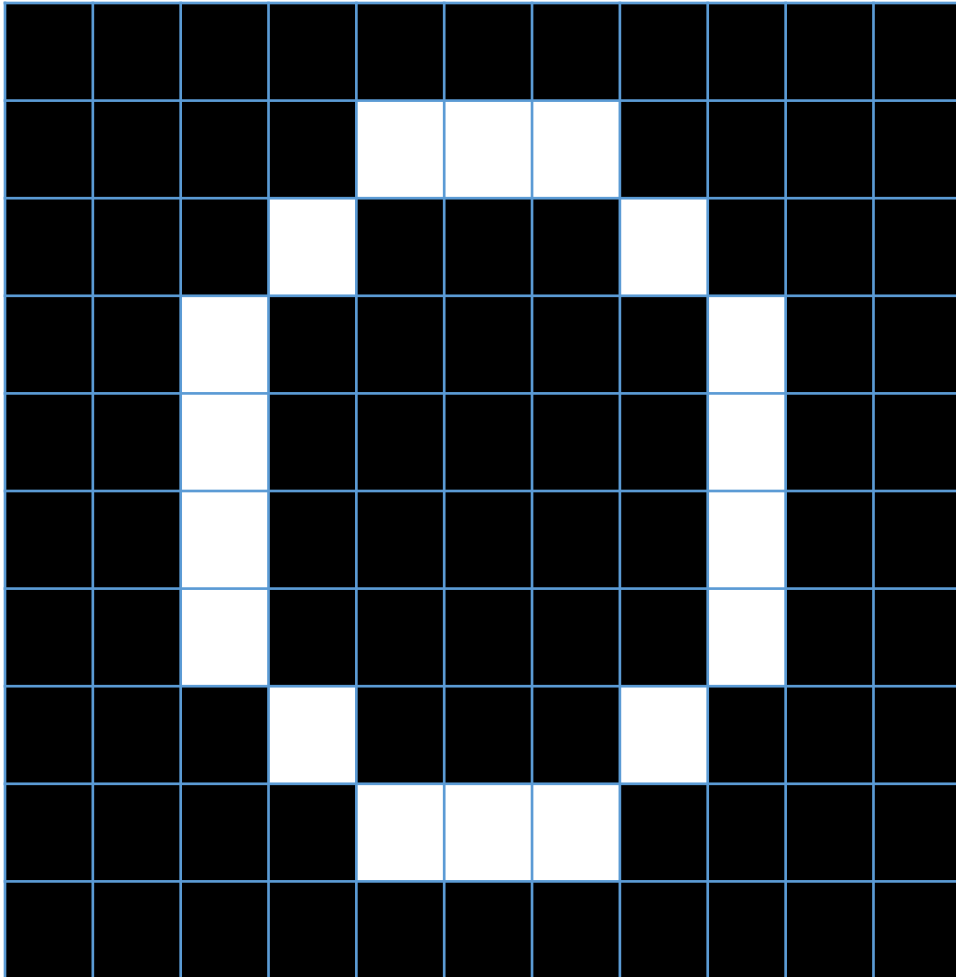
Creating Layer Architectures

- Convolution Neural Networks – CNN
- Special layer combinations that make them adept at classifying images

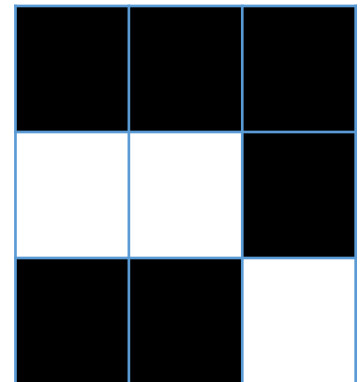
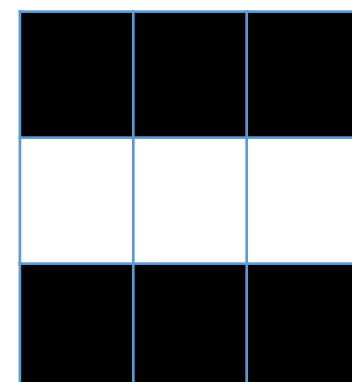
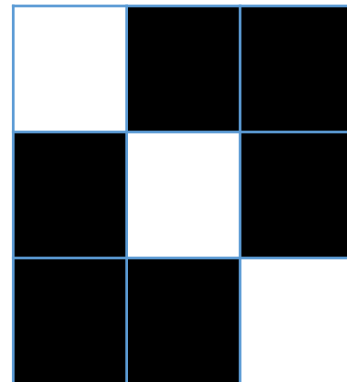
- Convolution Layer
- ReLU Layer
- Max Pooling Layer



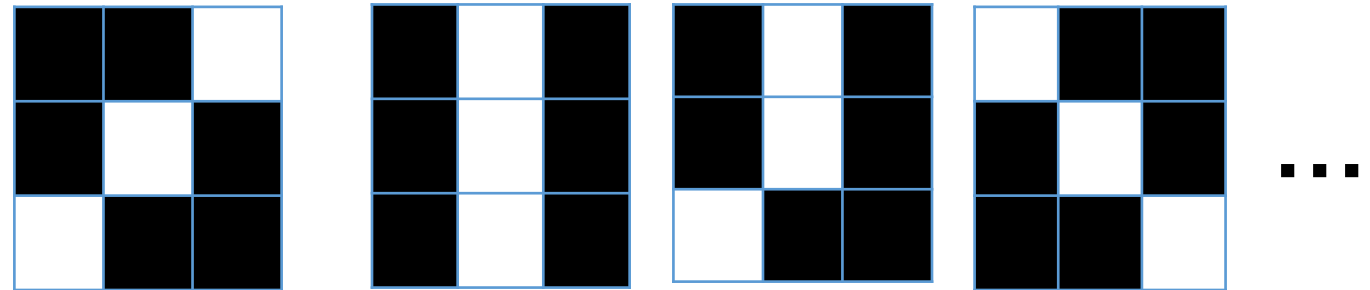
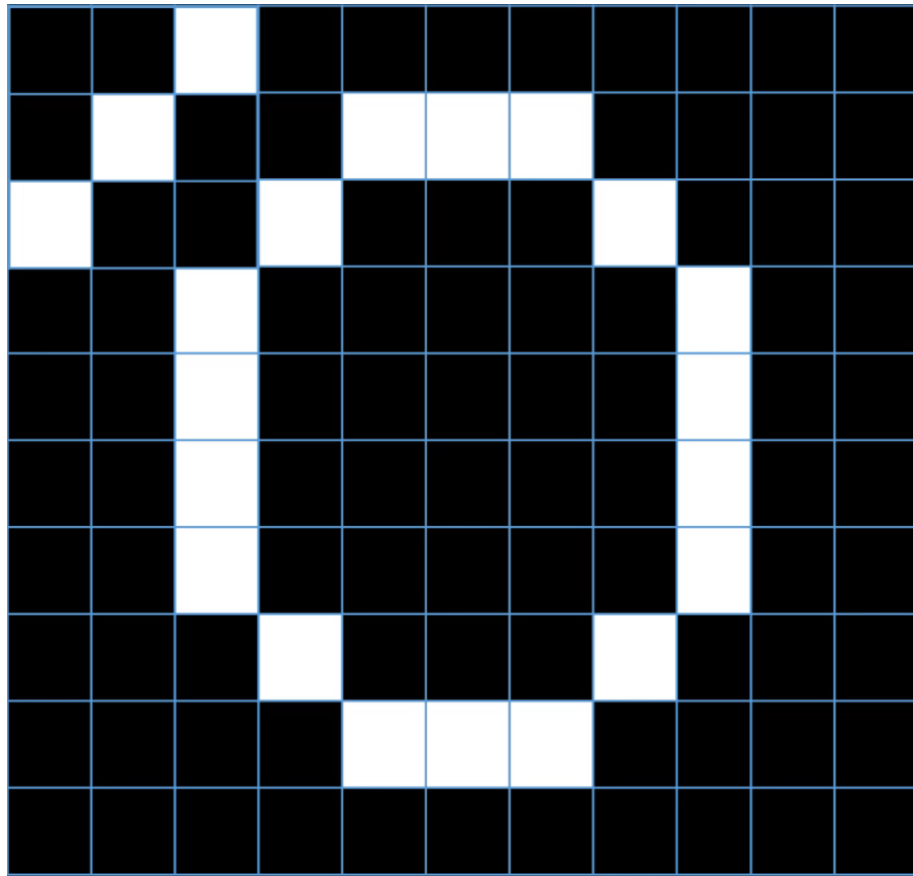
Convolution Layers Search for Patterns



These patterns would be common in the number 0



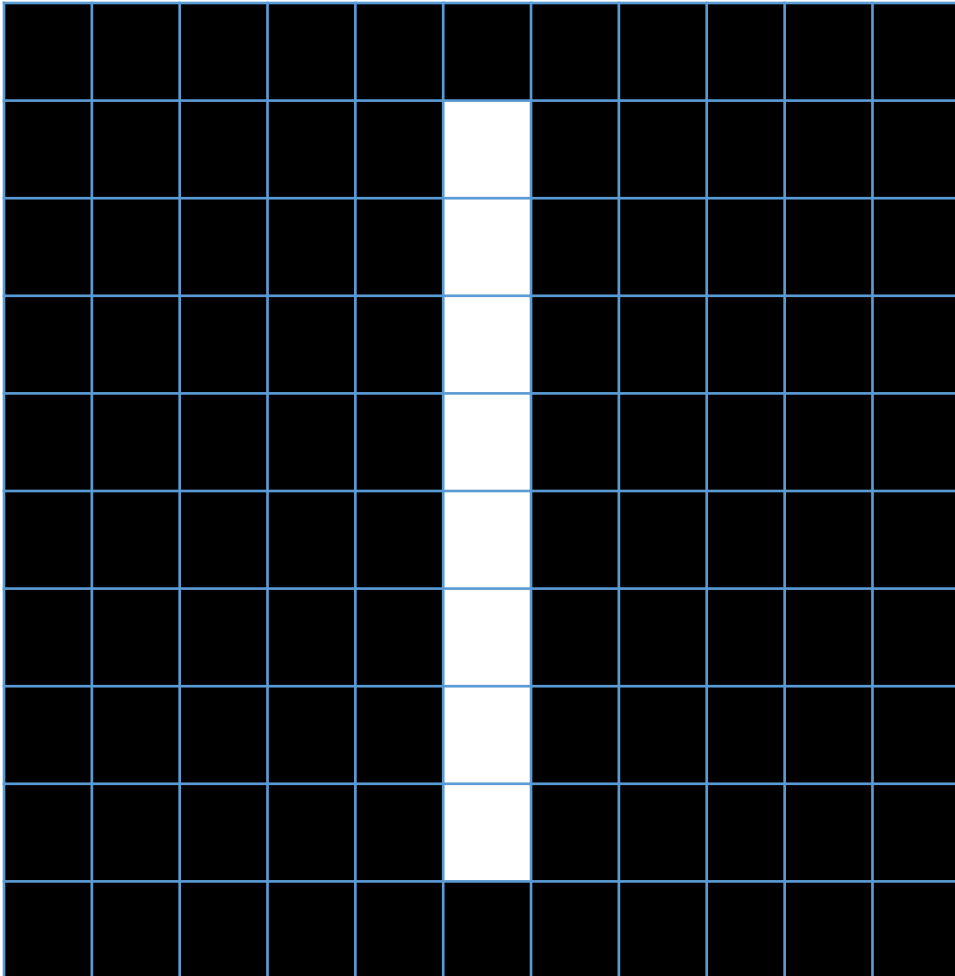
All patterns are compared to the patterns on a new image.



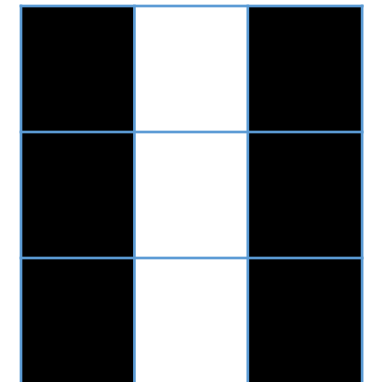
- Pattern starts at left corner
Perform comparison
Slide over one pixel
- Reach end of image
- Repeat for next pattern

Play the video !

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

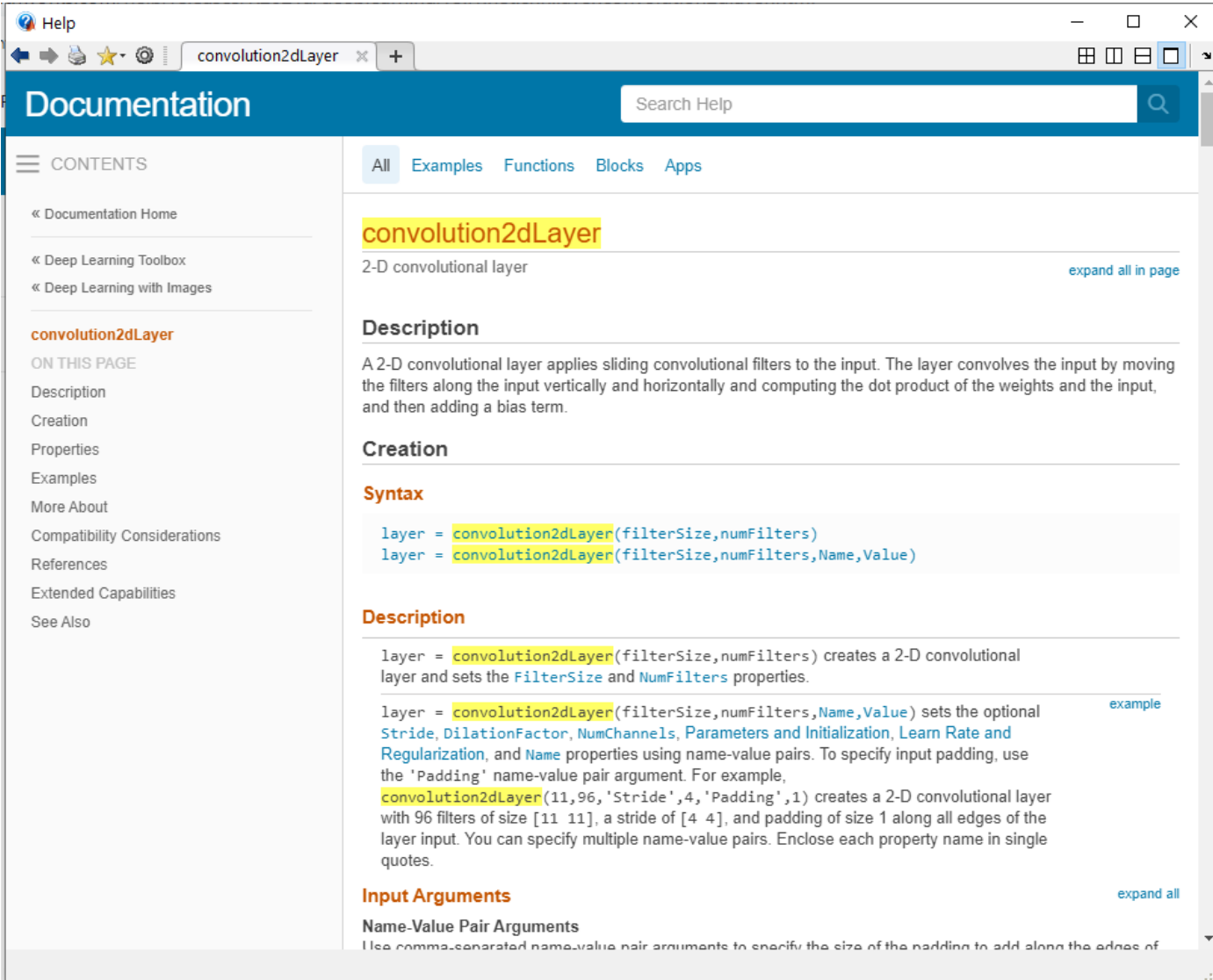


- This image would not match well against the patterns for the number zero
- It would only do very well against this pattern



Convolution Layers attributes

convolution2dLayer ()



The screenshot shows the MATLAB documentation page for the `convolution2dLayer` function. The page is titled "convolution2dLayer" and is part of the "Deep Learning Toolbox" documentation. The main content area is titled "convolution2dLayer" and describes it as a "2-D convolutional layer". The "Description" section explains that the layer applies sliding convolutional filters to the input, moving them vertically and horizontally, and computing the dot product of the weights and the input, then adding a bias term. The "Creation" section provides the syntax for creating the layer: `layer = convolution2dLayer(filterSize,numFilters)` and `layer = convolution2dLayer(filterSize,numFilters,Name,Value)`. The "Description" section further details the parameters: `filterSize` and `numFilters` are required, while `Stride`, `DilationFactor`, `NumChannels`, `Parameters and Initialization`, `Learn Rate and Regularization`, and `Name` are optional. An example is provided: `convolution2dLayer(11,96,'Stride',4,'Padding',1)` creates a 2-D convolutional layer with 96 filters of size [11 11], a stride of [4 4], and padding of size 1 along all edges of the layer input. The "Input Arguments" section is partially visible, mentioning "Name-Value Pair Arguments".

Properties

Convolution

- > **FilterSize** — Height and width of filters
vector of two positive integers
- > **NumFilters** — Number of filters
positive integer
- > **Stride** — Step size for traversing input
[1 1] (default) | vector of two positive integers
- > **DilationFactor** — Factor for dilated convolution
[1 1] (default) | vector of two positive integers
- > **PaddingSize** — Size of padding
[0 0 0 0] (default) | vector of four nonnegative integers
- > **PaddingMode** — Method to determine padding size
'manual' (default) | 'same'
- > **Padding** — Size of padding
[0 0] (default) | vector of two nonnegative integers
- > **PaddingValue** — Value to pad data
0 (default) | scalar | 'symmetric-include-edge' | 'symmetric-exclude-edge'
- > **NumChannels** — Number of channels for each filter
'auto' (default) | positive integer

Parameters and Initialization

- > **WeightsInitializer** — Function to initialize weights

Rectified Linear Units Layer (ReLU)

Converts negative numbers to zero

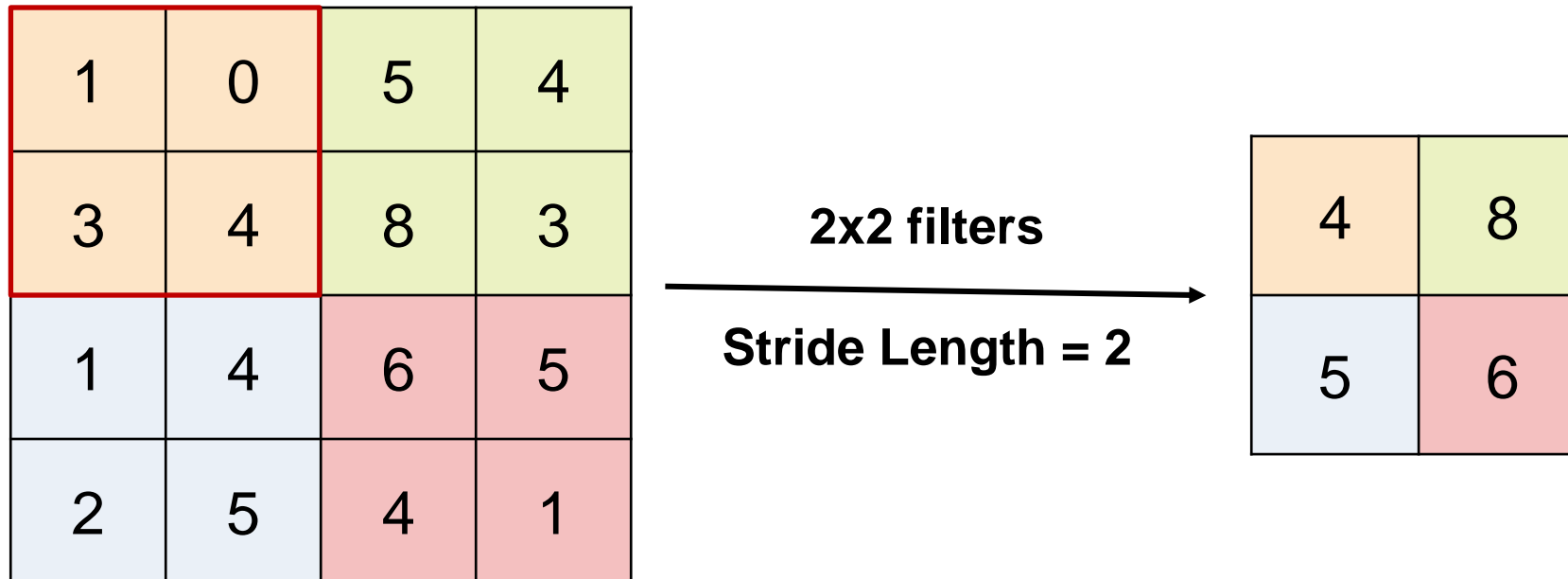
-1	0	5	4
3	-4	-8	3
1	4	6	-5
-2	-5	4	1

→

0	0	5	4
3	0	0	3
1	4	6	0
0	0	4	1

Max Pooling is a down-sampling operation

Shrink large images while preserving important information



Last layers

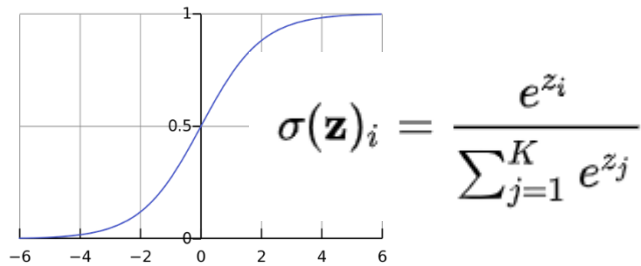
Classification problems end with 3 Layers

A Fully Connected Layer

- looks at which high-level features correspond to a specific category
- calculates scores for each category (highest score wins)
- “flattens” the matrix into a column vector

B Softmax Layer

- turns scores into probabilities

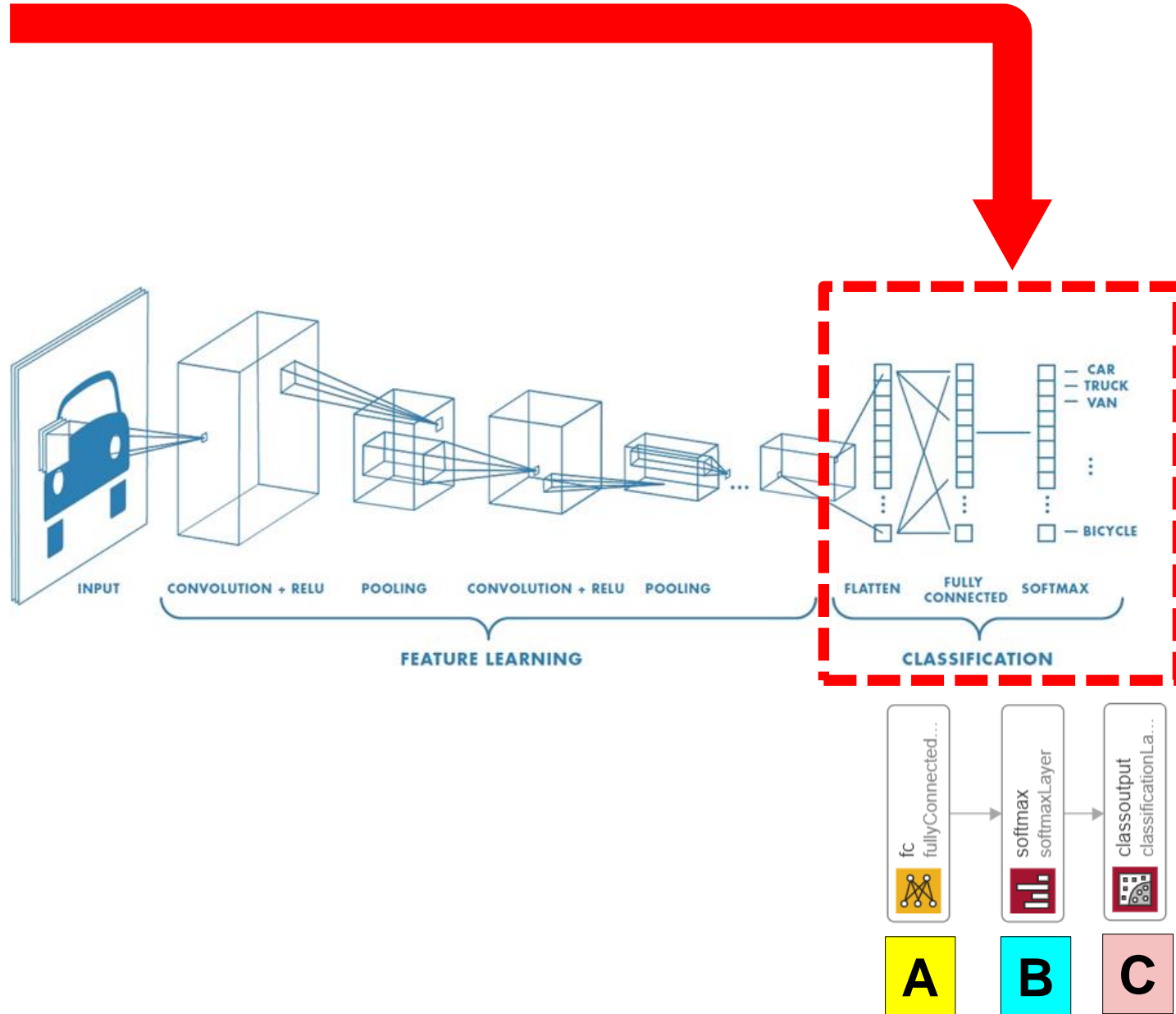


C Classification Layer

- categorizes image into one of the classes that the network is trained on

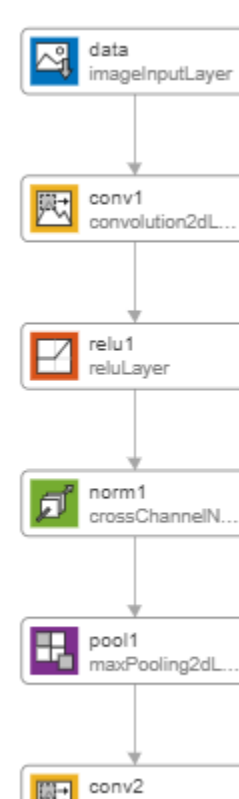
Note: regression problems end with 2 layers

- Fully Connected Layer
- Regression Layer



How does Deep Learning work?

- Hyperparameters** are set manually
 - are not updated during training
 - number of hidden layers, learning rate, minibatch size, epochs, activation function, etc.
- Learnable Parameters** are chosen by the network
 - calculated within the neurons
 - adjusted during training by comparing the predicted (final) output with the actual output
- The deeper the network, the more information is processed
 - multiple types of neural networks with different structure (e. g. CNN, RNN, GAN, and many more)
 - only considering specific regions (e. g. receptive fields in CNN) or specific outputs



```

net.Layers(2)
Convolution2DLayer with properties:
Name: 'conv1'

Hyperparameters
• FilterSize: [11 11]
• NumChannels: 3
• NumFilters: 96
• Stride: [4 4]
• DilationFactor: [1 1]
• PaddingMode: 'manual'
• PaddingSize: [0 0 0 0]

Learnable Parameters
• Weights: [11×11×3×96 single]
• Bias: [1×1×96 single]
  
```

Deep Learning Network Analyzer

net 25 layers 0 warnings 0 errors

Analysis date: 21-Jun-2021 14:18:12

	Activations	Learnables
Image Input	227×227×3	-
Convolution	55×55×96	Weights 11×11×3×96 Bias 1×1×96
ReLU	55×55×96	-
Cross Channel Normalization	55×55×96	-
Pooling	27×27×96	-
Grouped Convolution	27×27×256	Weights 5×5×48×128... Bias 1×1×128×2
ReLU	27×27×256	-
Cross Channel Normalization	27×27×256	-
Pooling	13×13×256	-
Convolution	13×13×384	Weights 3×3×256×384 Bias 1×1×384
ReLU	13×13×384	-

How Do I know Which Layers to Use?

Feature Extraction - Images

- 2D and 3D convolution
- Transposed convolution (...)

Activation Functions

- ReLU
- Tanh (...)

Sequence Data

Signal, Text, Numeric

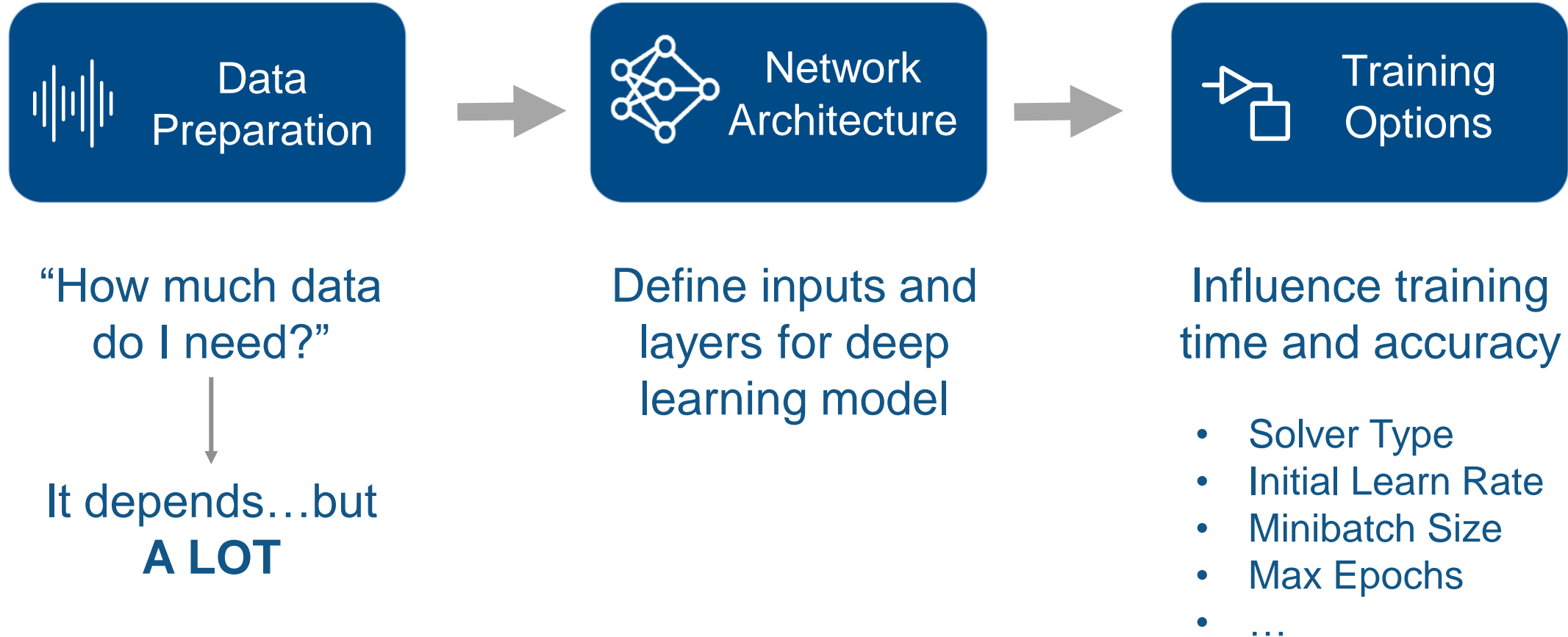
- LSTM
- BiLSTM
- Word Embedding (...)

Normalization

- Dropout
- Batch normalization
- (...)

Research papers and [doc examples](#) can provide guidelines for creating architecture.

3 Components to Train any Network



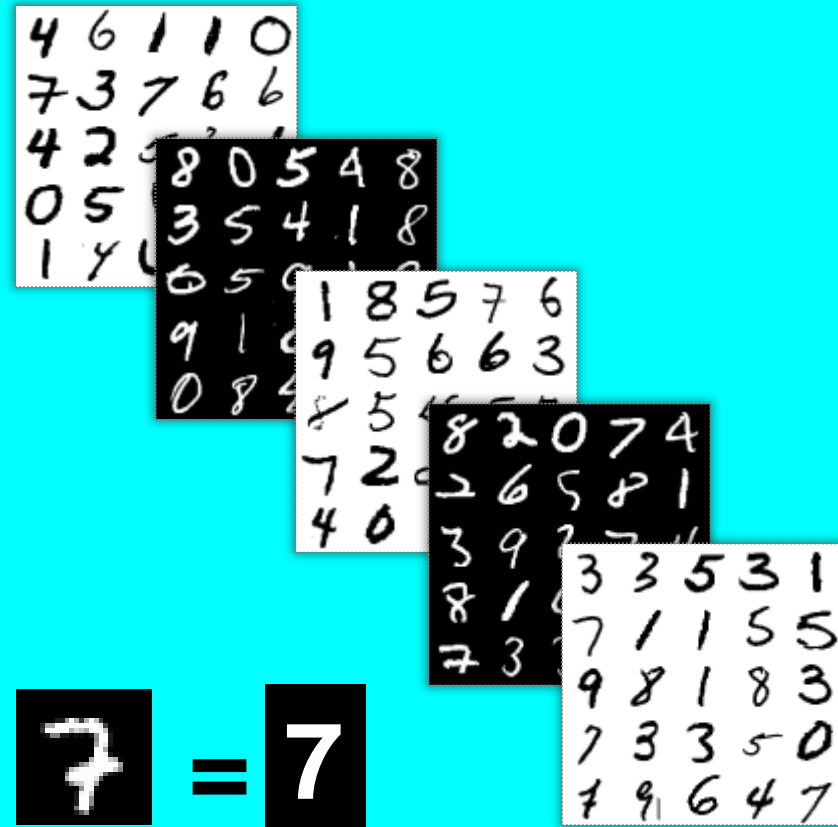
Exercise 3 - MNIST

Purpose:

- Learn how to create and train deep neural network
- Use MATLAB's **Deep Network Designer**
- Explore hyperparameters

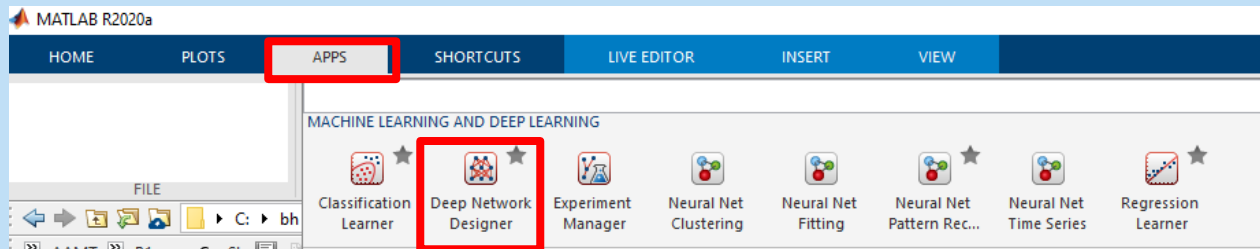
Details

- Dataset consists of handwritten digits 0-9
- 60,000 training images
- 10,000 test images

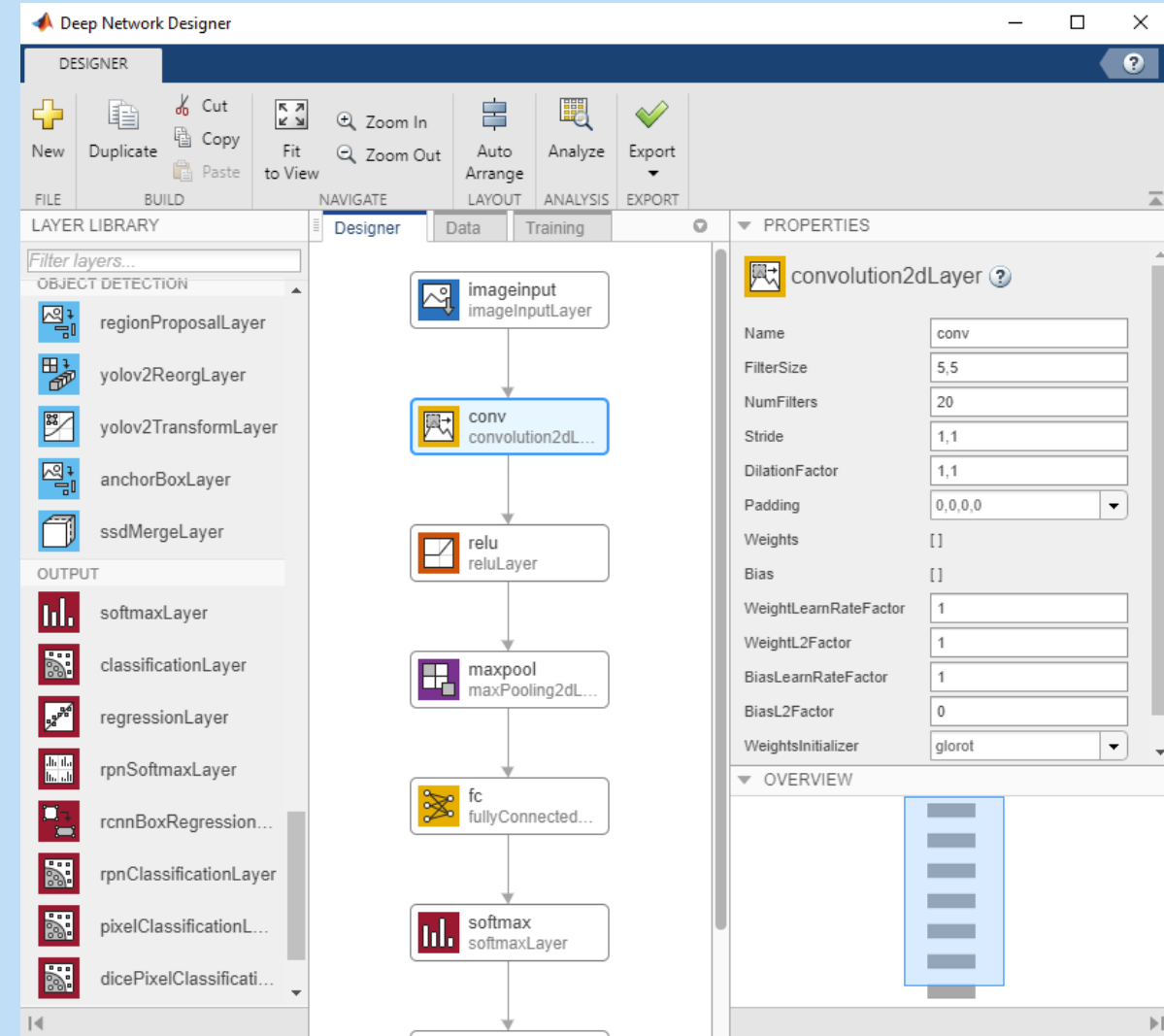


Sources: [MNIST handwritten digit database](#), Yann LeCun, Corinna Cortes and Chris Burges

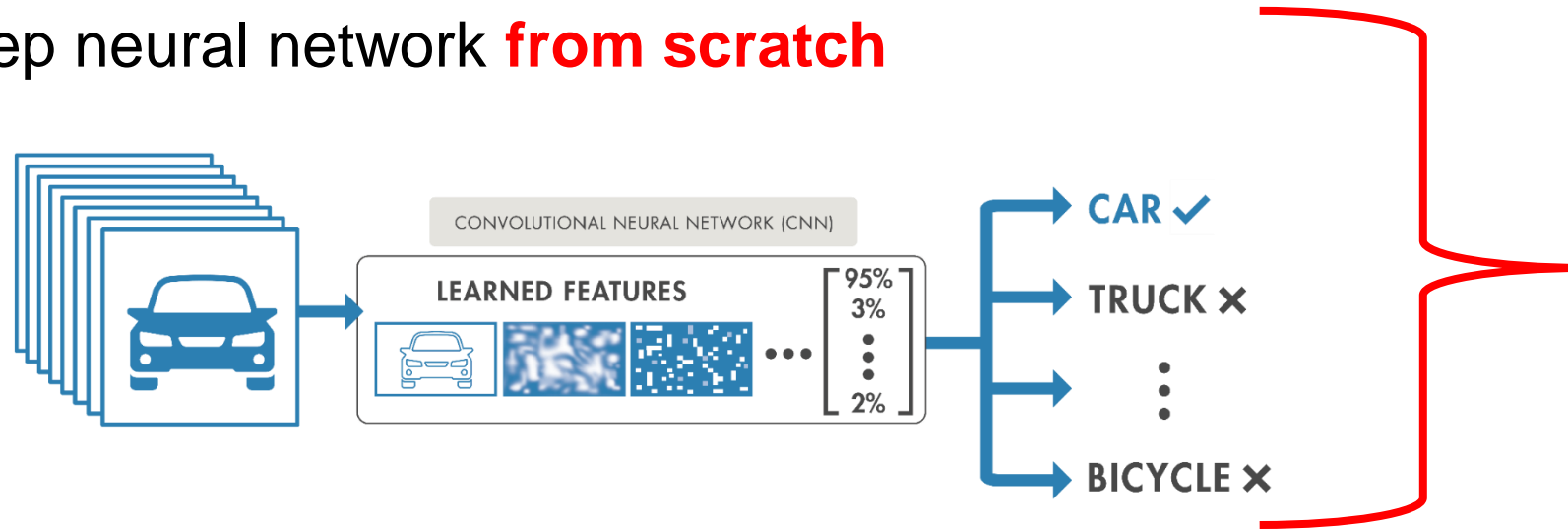
What we just did



The Deep Network Designer



- Train a deep neural network **from scratch**



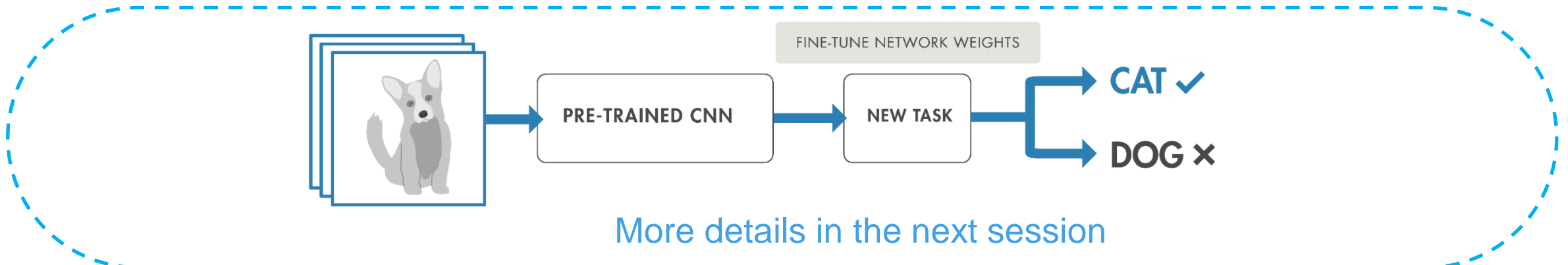
We did this in the MNIST exercise

But WHY start from scratch ?

Can we start with an already “partially” trained model

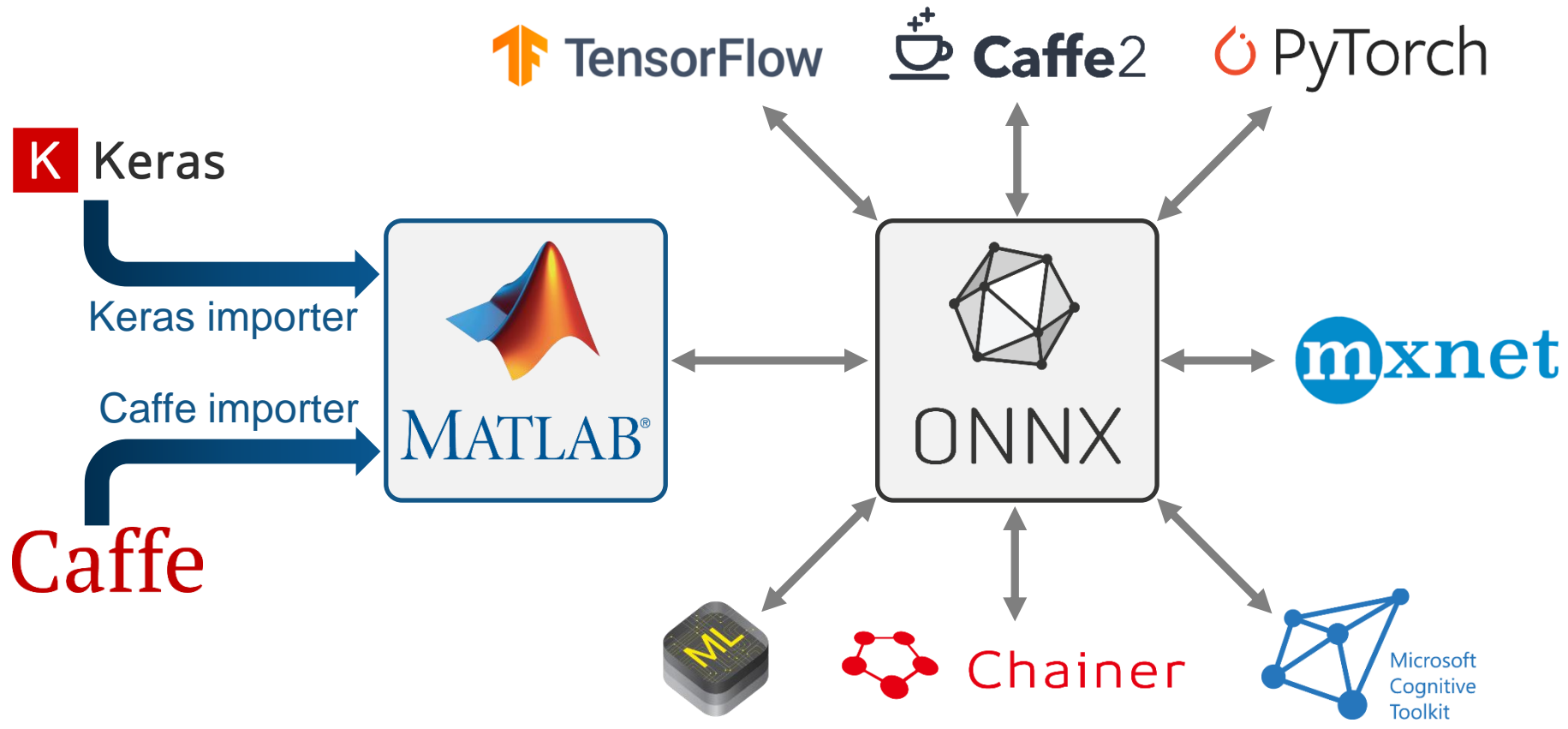
Transfer Learning

- Use a pretrained model – Transfer Learning

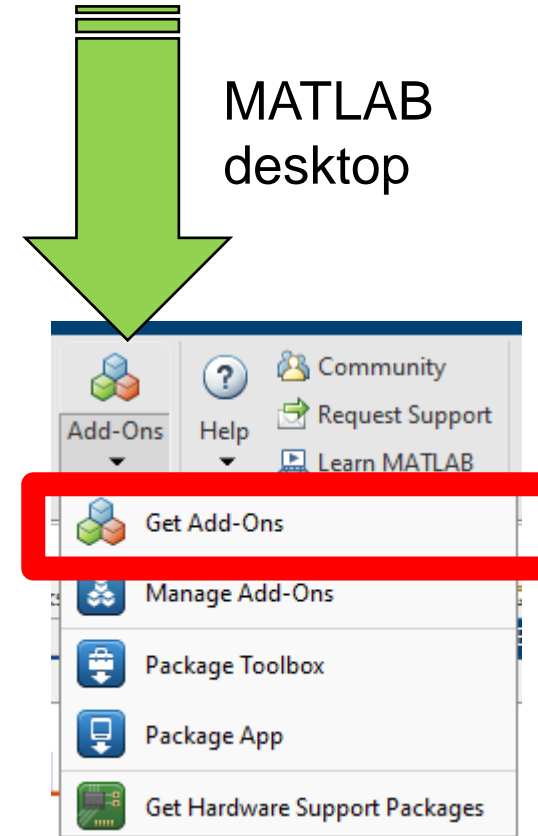


More details in the next session

Access Pretrained Models from Within MATLAB or Import from the Web



How

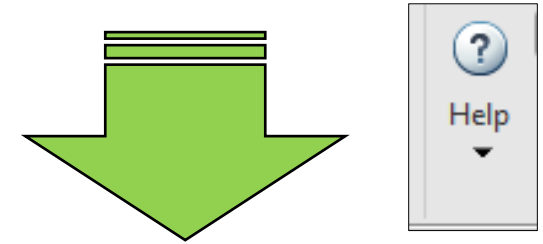


Search:
Onnx, Keras, etc

Pretrained Models

- Pretrained models have predefined layer orders and parameter values
- Can be used for inference without training

Full list of models available [HERE](#)



AlexNet
VGG-16
VGG-19
GoogLeNet

Get started with these Models

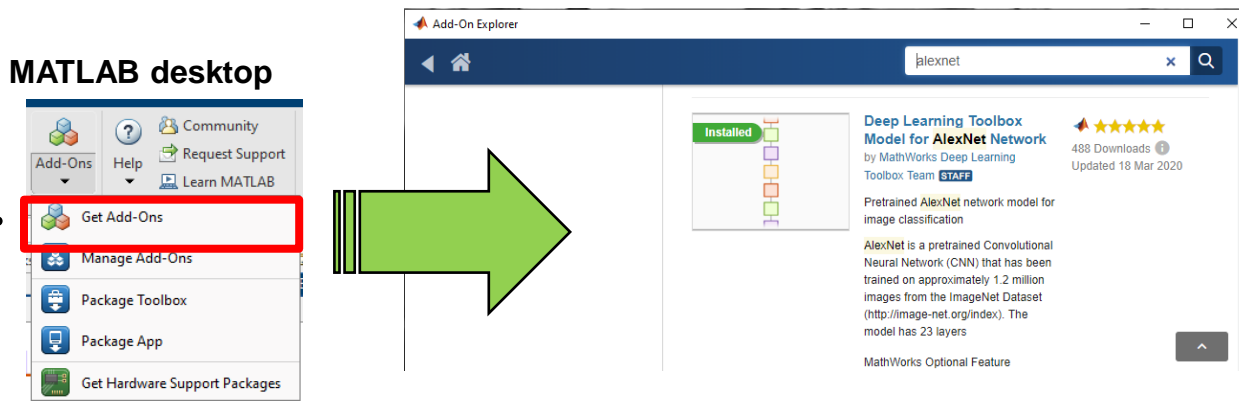
ResNet-18 **Inception-v3**
ResNet-101 **DenseNet-201**
ResNet-50 **Xception**

Effective for object detection and semantic segmentation workflows

SqueezeNet
MobileNet-v2
ShuffleNet

Lightweight and computationally efficient

How

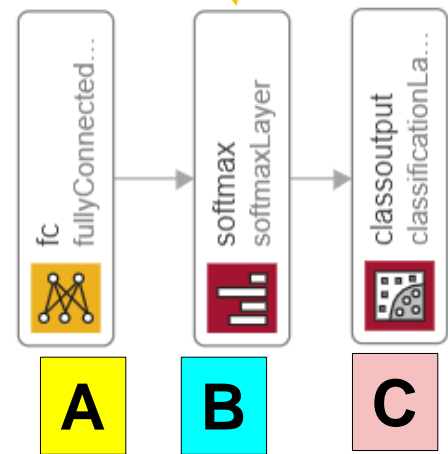
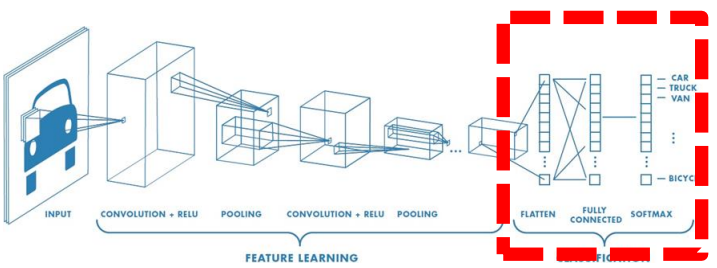
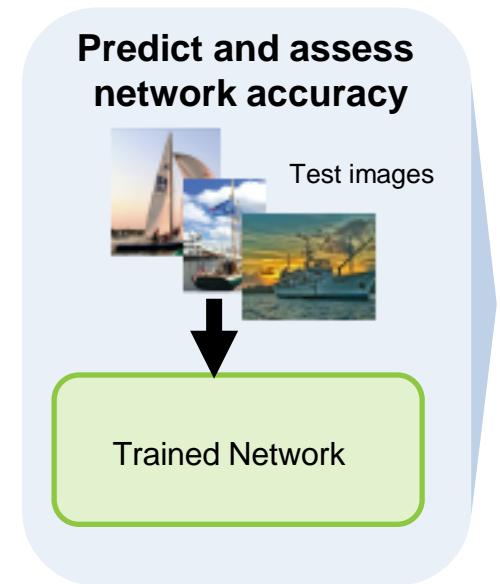
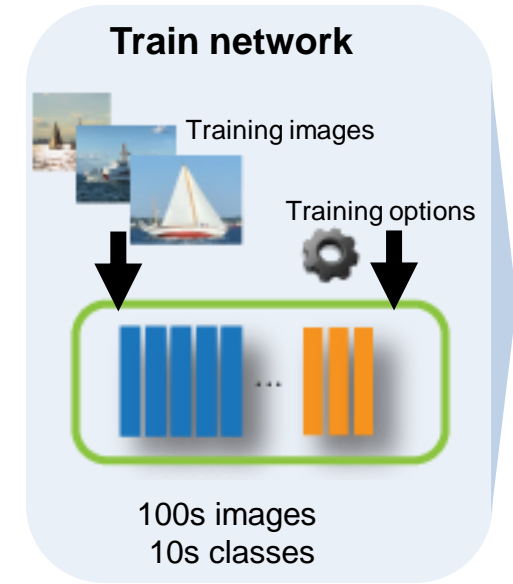
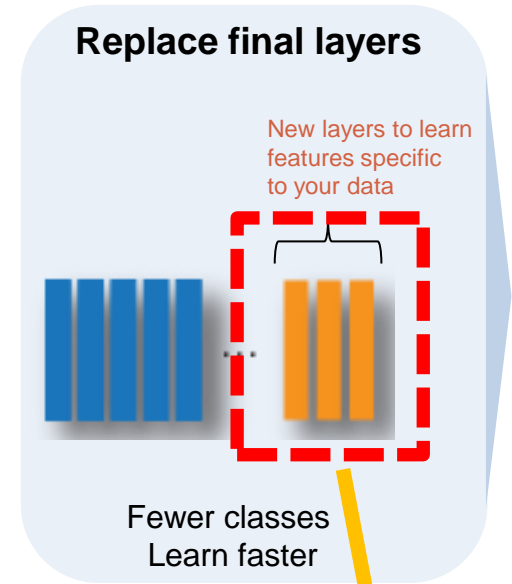
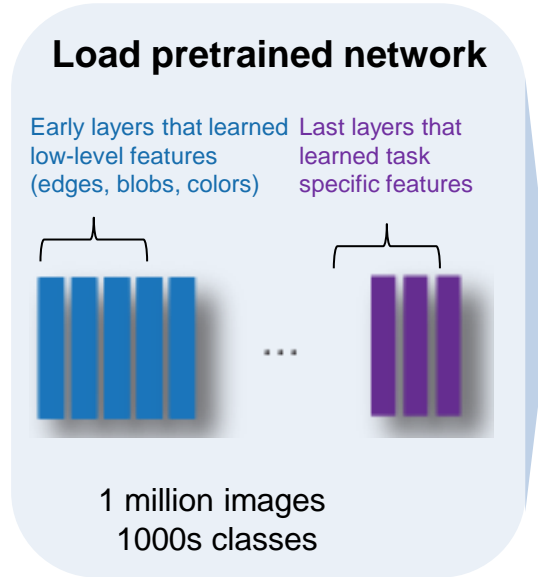


Purpose	Description
Classification	Apply pretrained networks directly to classification problems. To classify a new image, use <code>classify</code> . For an example showing how to use a pretrained network for classification, see Classify Image Using GoogLeNet .
Feature Extraction	Use a pretrained network as a feature extractor by using the layer activations as features. You can use these activations as features to train another machine learning model, such as a support vector machine (SVM). For more information, see Feature Extraction . For an example, see Extract Image Features Using Pretrained Network .
Transfer Learning	Take layers from a network trained on a large data set and fine-tune on a new data set. For more information, see Transfer Learning . For a simple example, see Get Started with Transfer Learning . To try more pretrained networks , see Train Deep Learning Network to Classify New Images .

Compare Pretrained Networks

Pretrained networks have different characteristics that matter when choosing a **network** to apply to your problem. The most important characteristics are **network** accuracy, speed, and size. Choosing a **network** is generally a tradeoff between these characteristics. Use the plot below to compare the ImageNet validation accuracy with the time required to make a prediction using the **network**.

Transfer Learning Workflow – model assembly



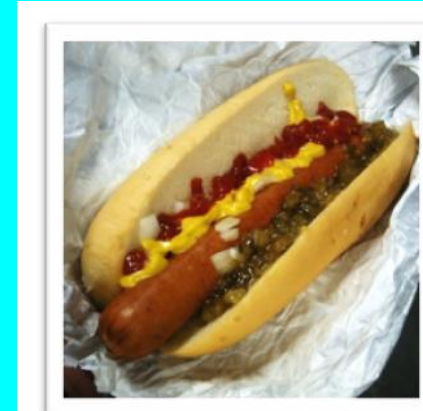
Exercise 4 – Food

Purpose:

- Use transfer learning to leverage a pretrained model to classify 5 types of food
- Visualize activations within a network

To Do:

1. Open **Work_Food.mlx**



What we just did

Configure TRAINING file management



`imageDatastore ()`
`splitEachLabel ()`
`augmentedImageDatastore ()`

Configure TRAINING options

`trainingOptions ()`

TRAIN the network

`trainNetwork ()`

Define Network Layers



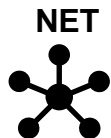
Configure TESTING file management



`imageDatastore ()`
`augmentedImageDatastore ()`

TEST the trained network

`classify ()`



DEMO – Experiment Manager

- Run, Track, and Analyze Multiple Deep Learning Experiments

The screenshot shows the Experiment Manager interface with a toolbar at the top containing icons for New, Save, Duplicate, Layout, Run, Stop, Training Plot, Confusion Matrix, Filter, and Export. The left sidebar shows a tree view of experiments under 'DigitsClassifier', including 'Baseline Establishment' and 'Baseline Tuning'. The main area displays 'Result Details' for 'Baseline Tuning' with a progress bar and a summary table. Below this is a detailed table of 16 trials.

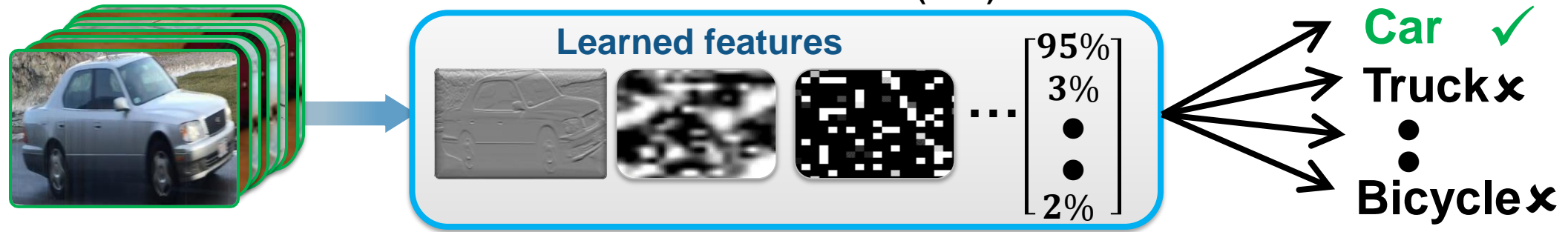
Trial	Status	Progress	Elapsed Time	myInitialLearn...	convFilterSize	Training Accu...	Training Loss	Validation Ac..
1	Complete	100.0%	0 hr 0 min 16 sec	1.0000e-6	3.0000	12.5000	2.6441	10.
2	Complete	100.0%	0 hr 0 min 15 sec	1.0000e-5	3.0000	25.7813	2.1228	20.
3	Complete	100.0%	0 hr 0 min 14 sec	0.0001	3.0000	64.8438	1.0878	42.
4	Complete	100.0%	0 hr 0 min 16 sec	0.0005	3.0000	90.6250	0.4648	49.
5	Complete	100.0%	0 hr 0 min 15 sec	1.0000e-6	4.0000	11.7188	2.4967	6.
6	Complete	100.0%	0 hr 0 min 15 sec	1.0000e-5	4.0000	23.4375	2.1213	14.
7	Complete	100.0%	0 hr 0 min 17 sec	0.0001	4.0000	72.6563	1.0283	39.
8	Running	30.7%	0 hr 0 min 4 sec	0.0005	4.0000			
9	Queued	0.0%		1.0000e-6	5.0000			
10	Queued	0.0%		1.0000e-5	5.0000			
11	Queued	0.0%		0.0001	5.0000			
12	Queued	0.0%		0.0005	5.0000			
13	Queued	0.0%		1.0000e-6	6.0000			
14	Queued	0.0%		1.0000e-5	6.0000			
15	Queued	0.0%		0.0001	6.0000			
16	Queued	0.0%		0.0005	6.0000			

Play the video !

Techniques Covered so Far

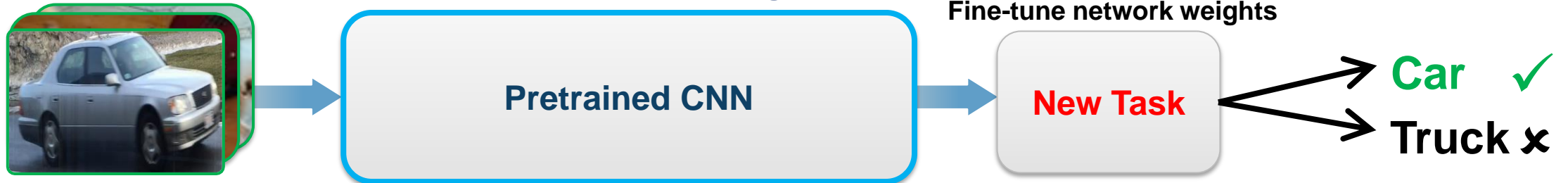
1. Train a Deep Neural Network from Scratch

Convolutional Neural Network (CNN)



2. Fine-tune a pretrained model (transfer learning)

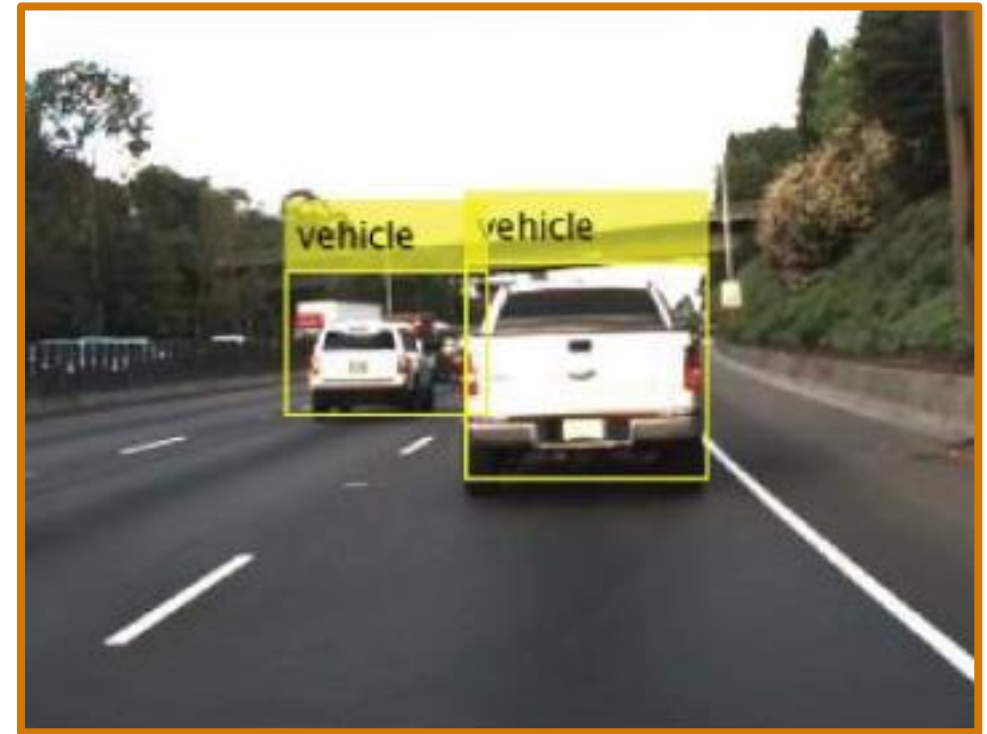
Fine-tune network weights



Classification vs. Object Detection



Classification predicts a label for an entire image.



Object detection predicts the location and label for objects in an image

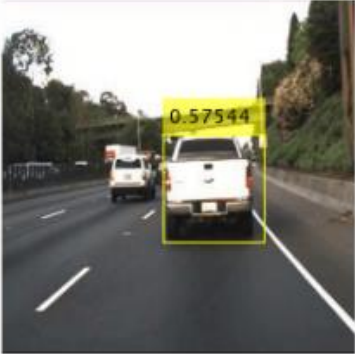
Deep Learning Object Detection Examples in MATLAB




A photograph of a white SUV parked in a lot, with a yellow bounding box around it and a confidence score of 0.99572 displayed above the box.

 **Object Detection Using SSD Deep Learning**

Train a Single Shot Detector (SSD).



A photograph of a white SUV driving on a road, with a yellow bounding box around it and a confidence score of 0.57544 displayed above the box.

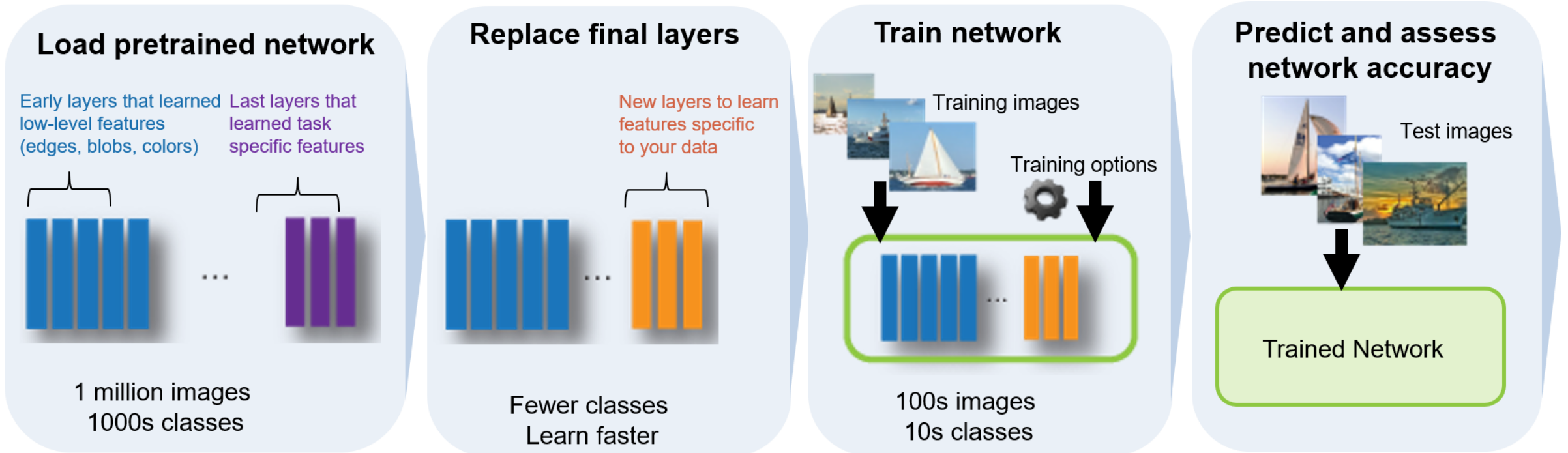
 **Object Detection Using YOLO v2 Deep Learning**

Train a you only look once (YOLO) v2 object detector.

Documentation examples:

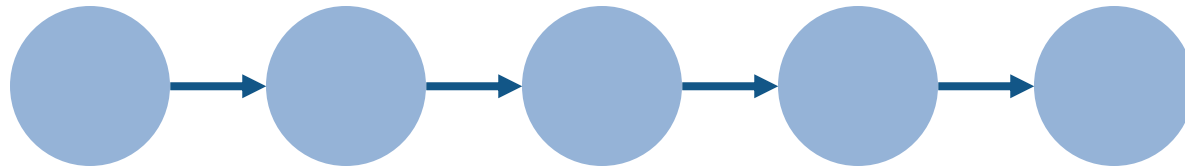
- [Faster R-CNN](#)
- [YOLO v2](#)
- [YOLO v3](#)
- [Single Shot Detector](#)

Transfer Learning is Commonly Used for Object Detection

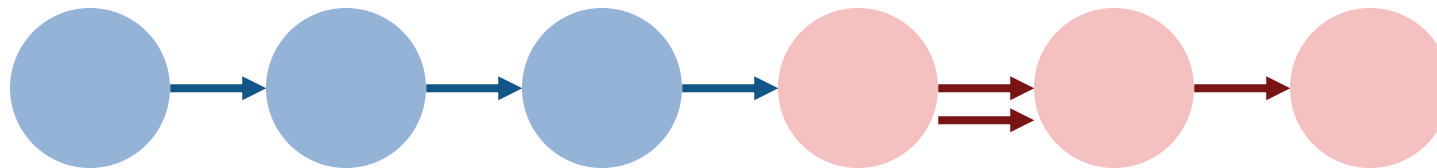


Transfer Learning Applied to YOLO v2 Object Detection

1. Import Pretrained model (ResNet-50)



2. Replace last layers with YOLO detection layers ([yolov2Layers](#))



3. Train network with training data

Exercise 5 – Vehicles

Purpose:

- Use transfer learning to create YOLO v2 network
- Train network to detect vehicles in image

To Do:

1. Open `Work_Vehicles.mlx`.



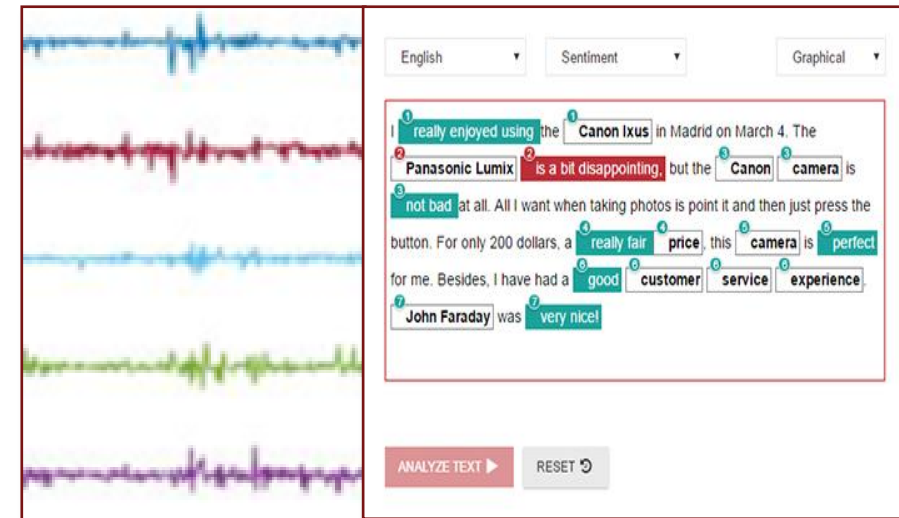
Selecting a Network Architecture

Image
Data

Signal or
Text Data



CNN

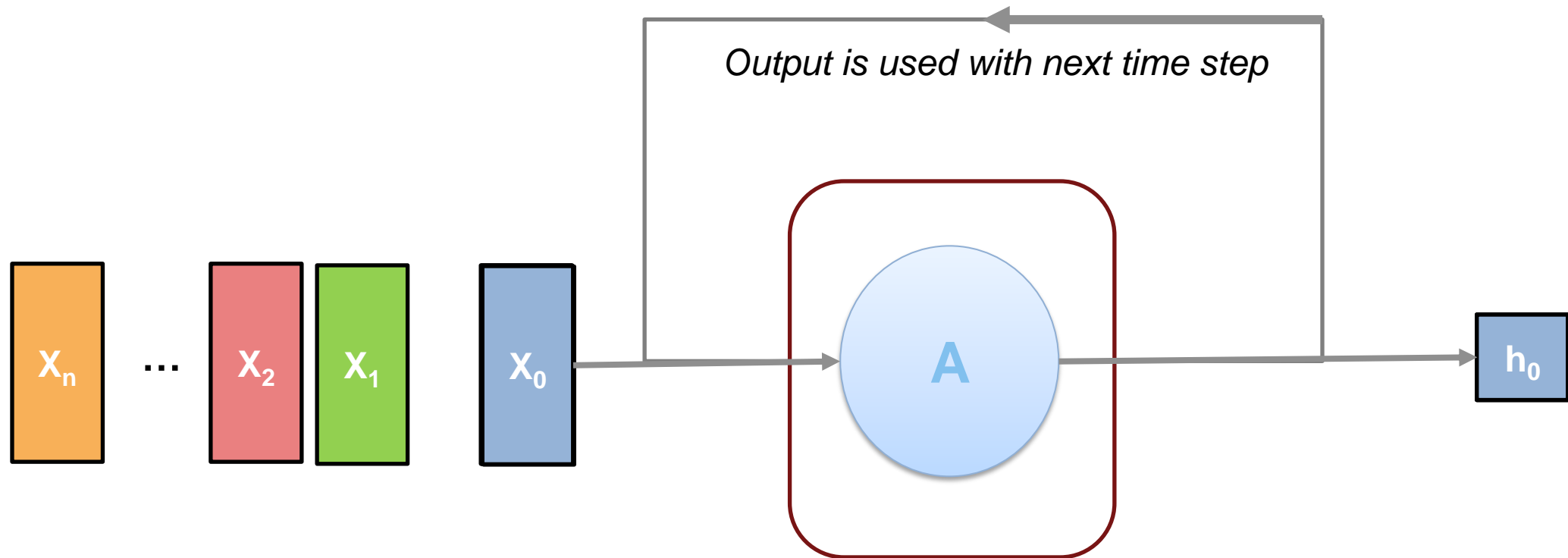


LSTM or CNN

LSTM = Long Short Term Series Network (more detail in later slides)

Recurrent Neural Networks

Take into account previous data when making new predictions



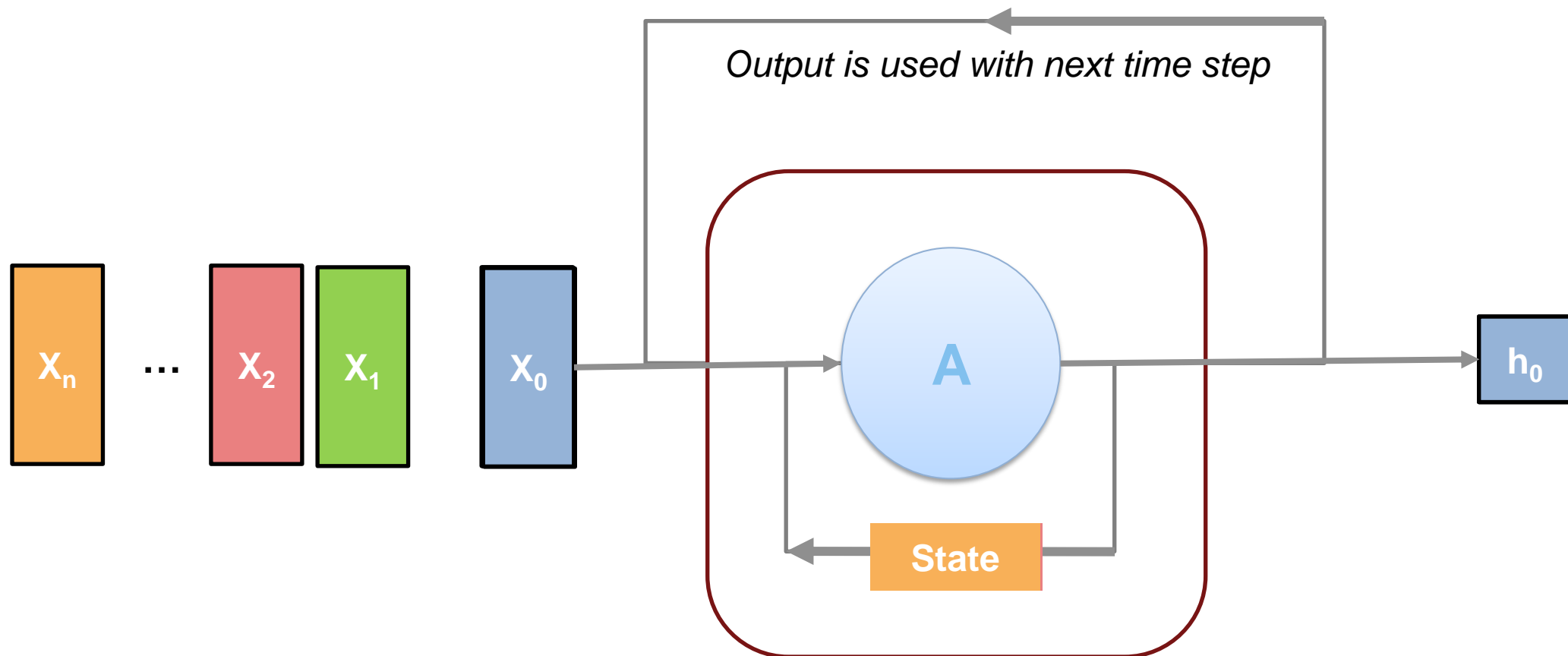
I was born in France...

[2000 words]

... I speak _____ ?

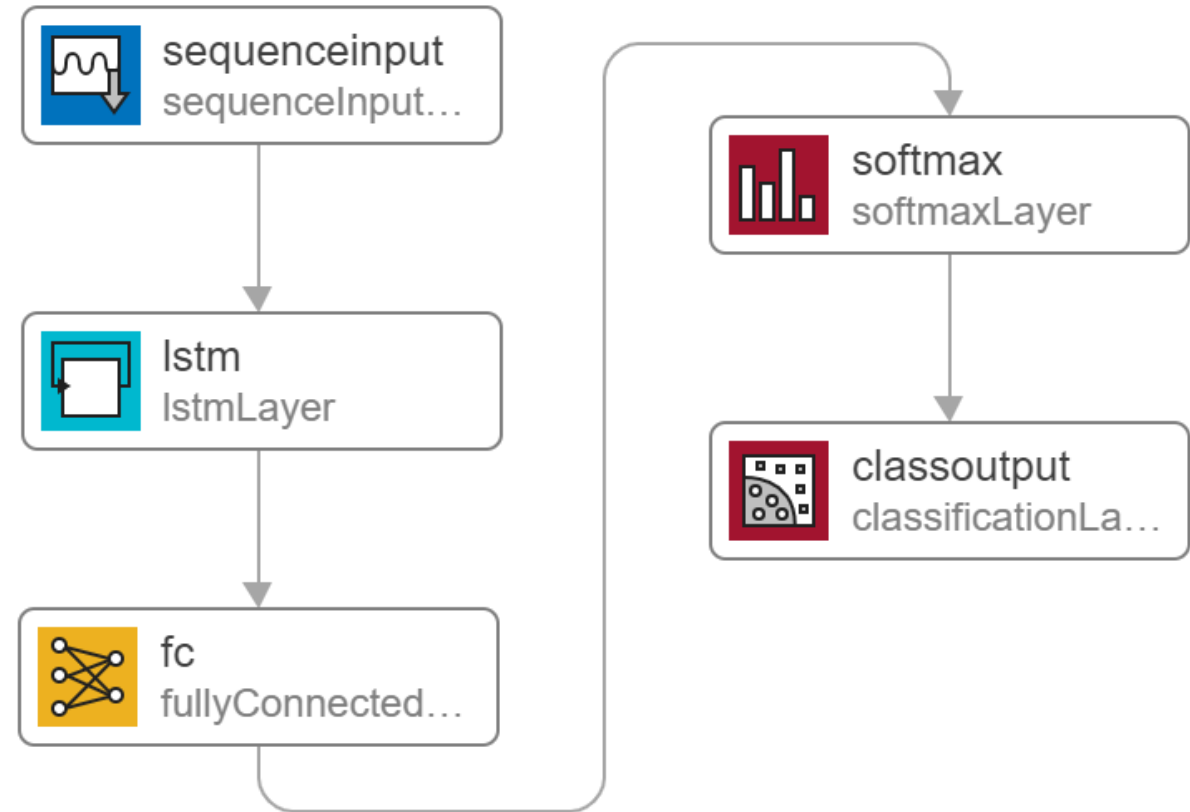
Long Short-Term Memory Network

Recurrent Neural Network that carries a memory cell (state) throughout the process



Simple LSTM Network Architecture

- Layers:
 - Input
 - LSTM
 - Fully Connected
 - Softmax
 - Classification
- LSTMs can be used for classification or regression



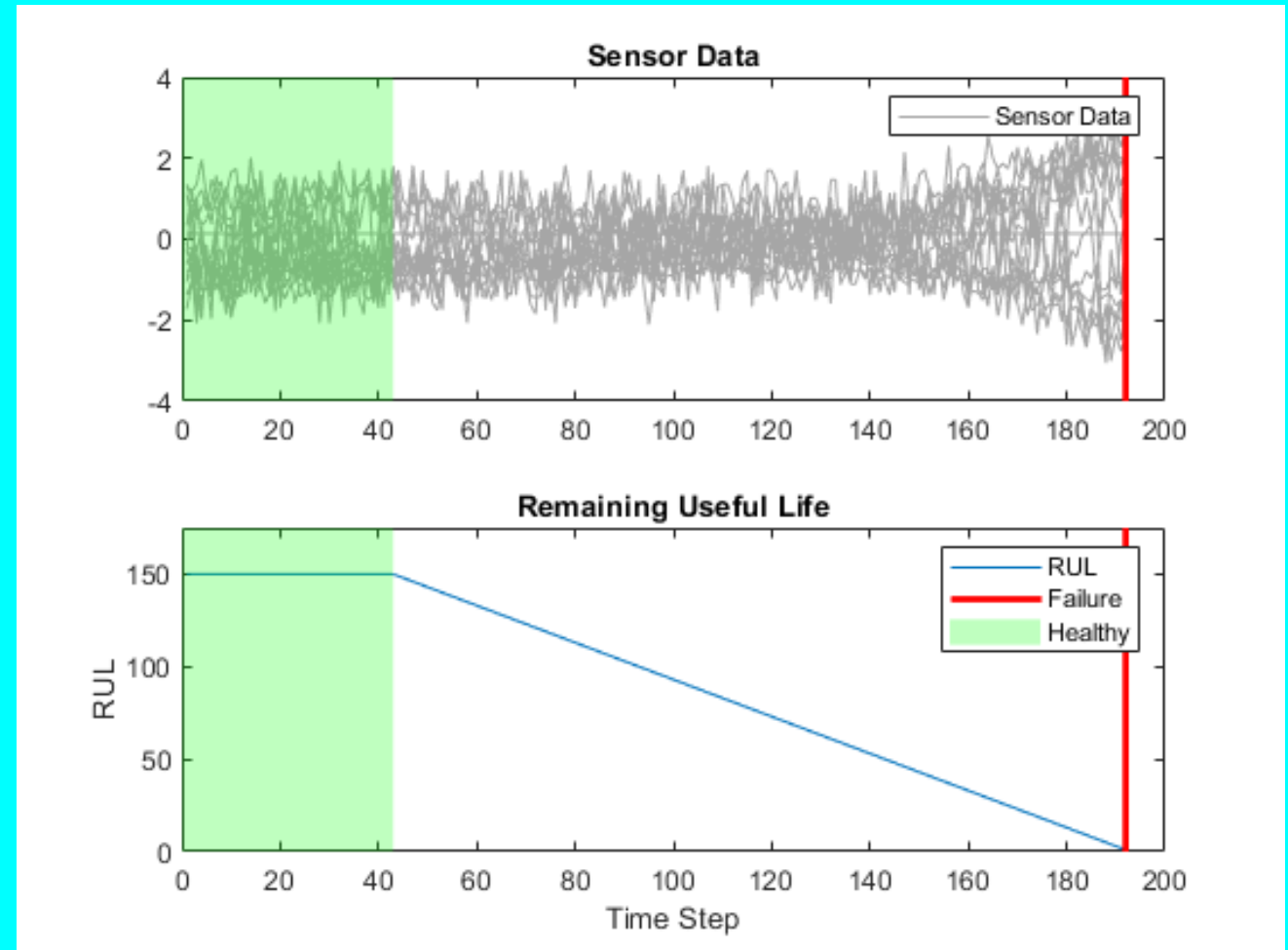
Exercise 6 – Engines

Purpose:

- Use an LSTM network to predict the remaining useful life of engines based on sensor data (regression)

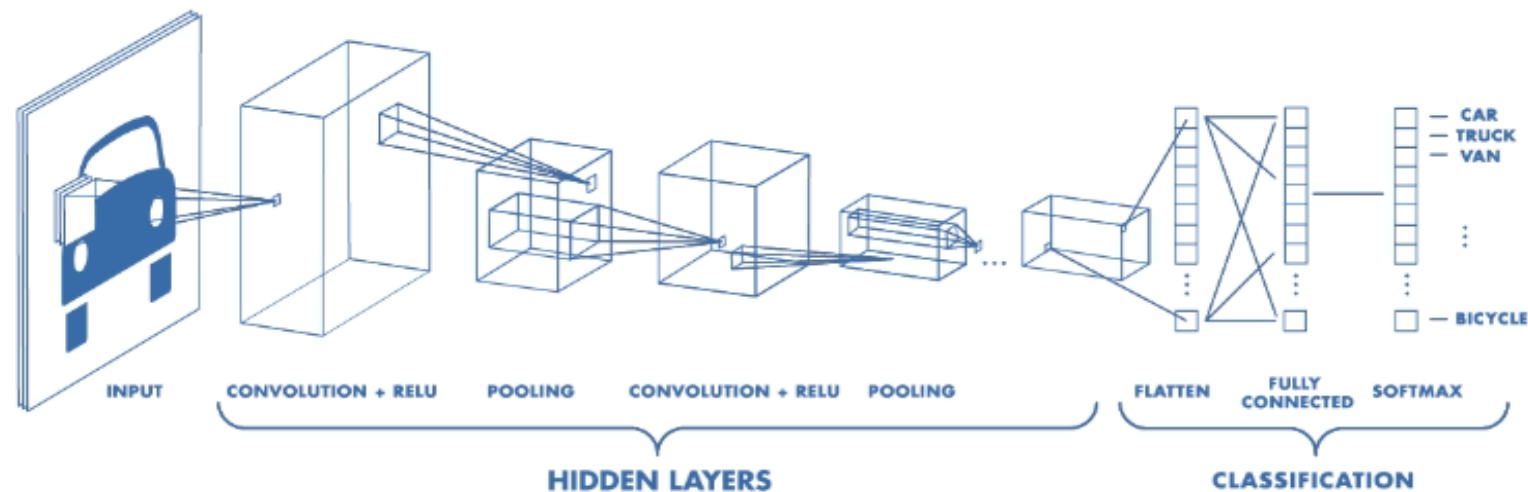
To Do:

- Open `Work_Engines.mlx`.

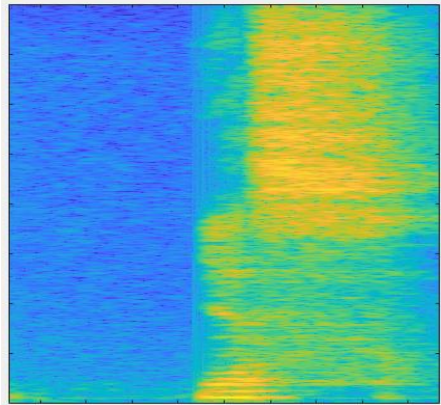


Using CNNs on Signal Data (Time-Freq Transforms)

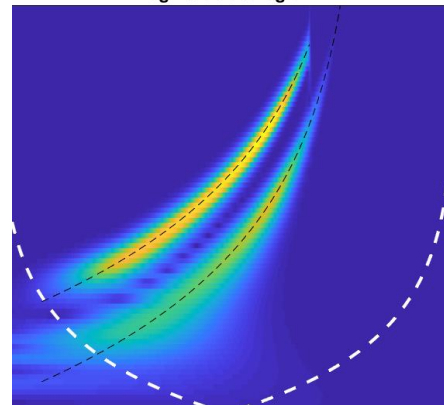
- CNNs are typically used to classify images
- Time-Frequency representations of signals can be used as images
- This approach can serve as a good starting point for signal classifications



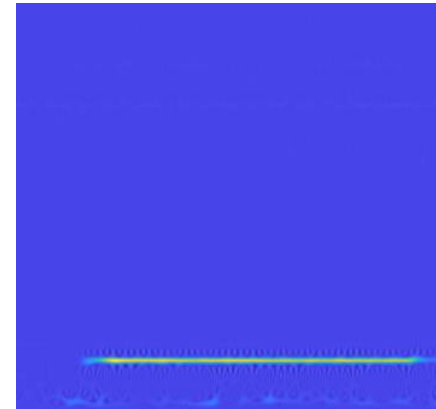
Different Types of Time-Frequency Transforms



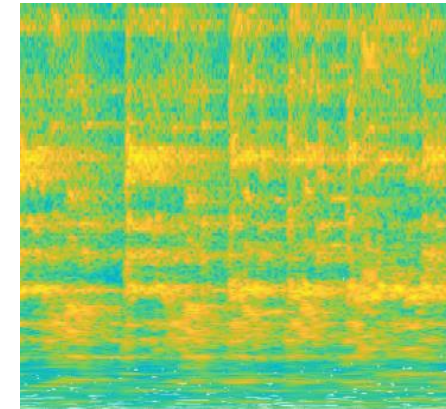
Basic spectrogram



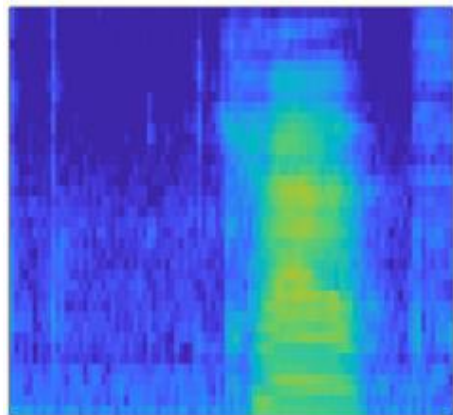
Wavelet scalogram



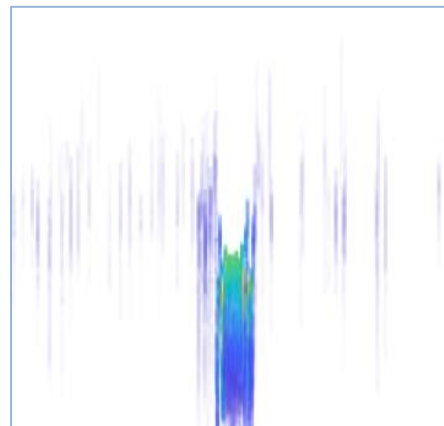
Wigner-Ville
Transform



Hilbert-Huang
Transform



Constant Q transform

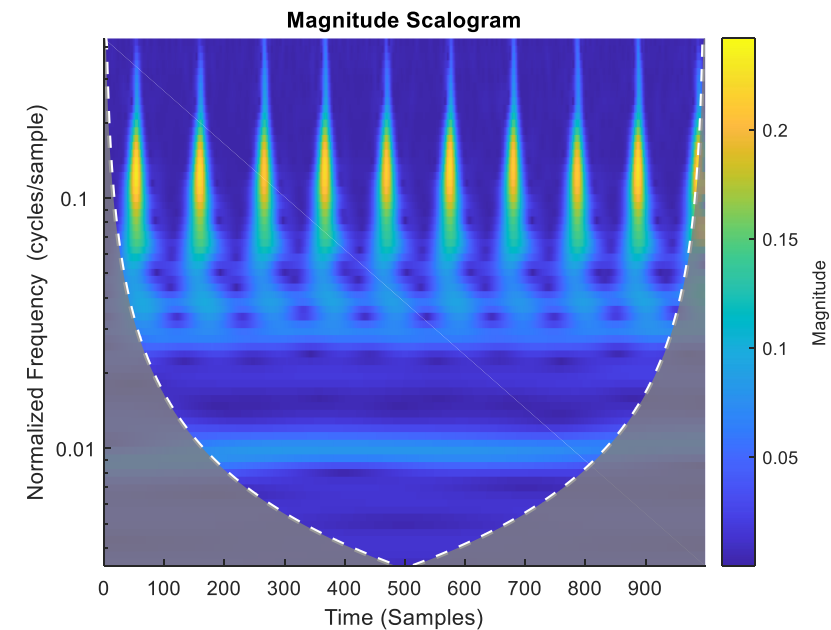
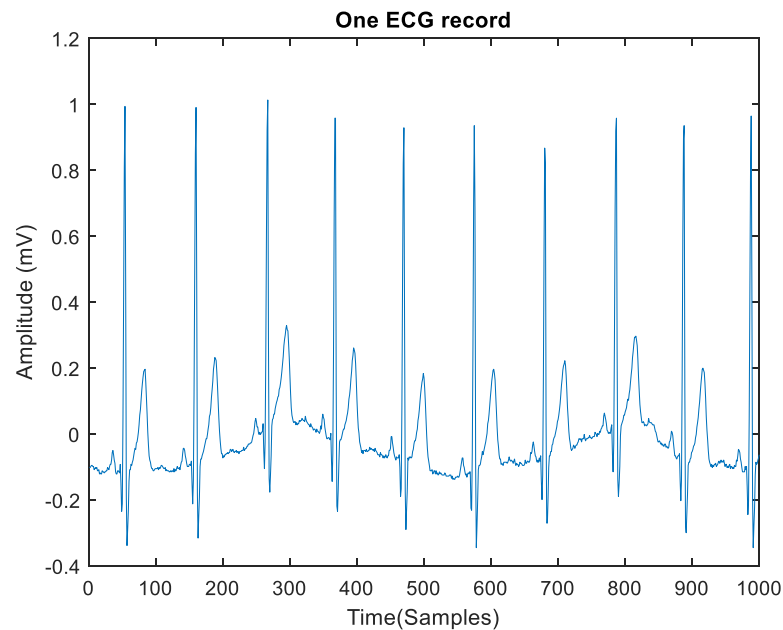


Perceptually-spaced
Spectrogram

[MATLAB Time –Frequency Gallery](#)

Continuous Wavelet Transform

- We will use this time-frequency transform in our exercise (Work_ECG_1).
- Differentiates signals from different classes well compared to basic spectrogram.



Exercise 7 – ECG

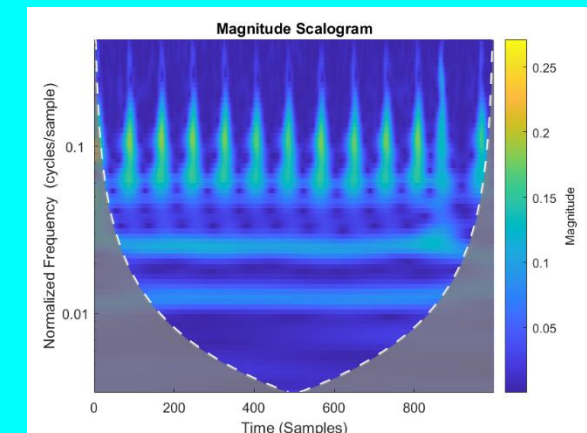
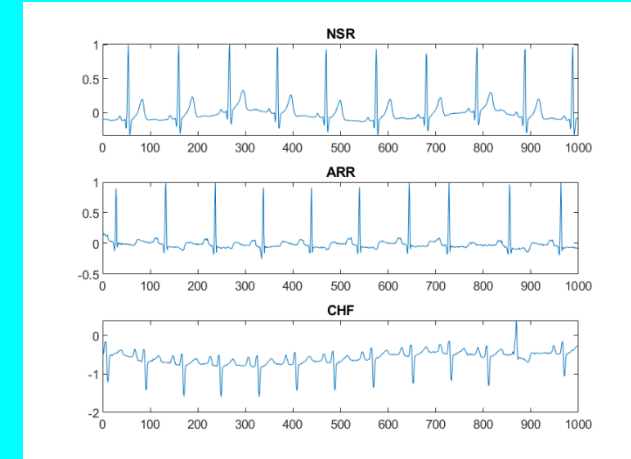
Purpose:

- Part 1: Classify different types of ECG signals using time-frequency transform + CNNs
- Part 2: Classify these same signals using feature extraction + LSTM

To Do:

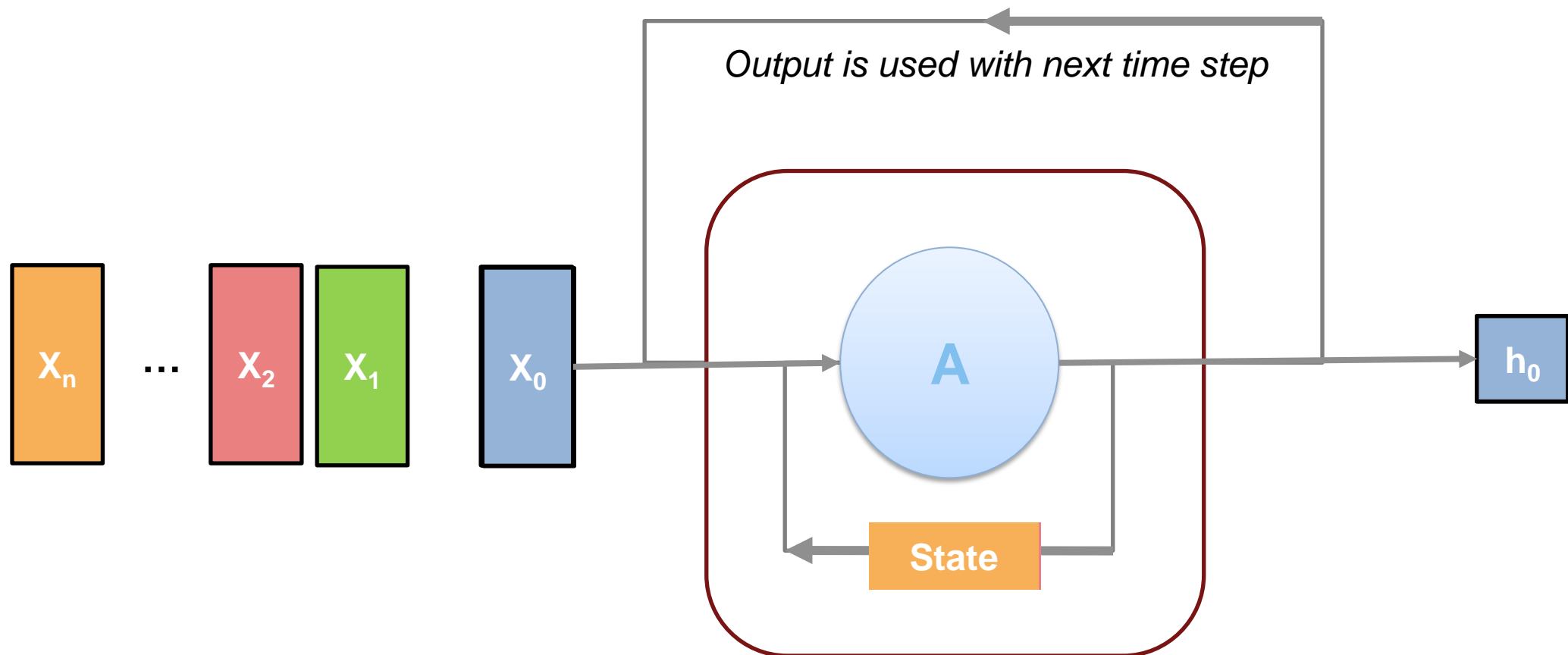
1. Open Work_ECG_1.mlx.
2. Open Work_ECG_2.mlx

Part 1 is required for part 2 to run



Long Short-Term Memory Network

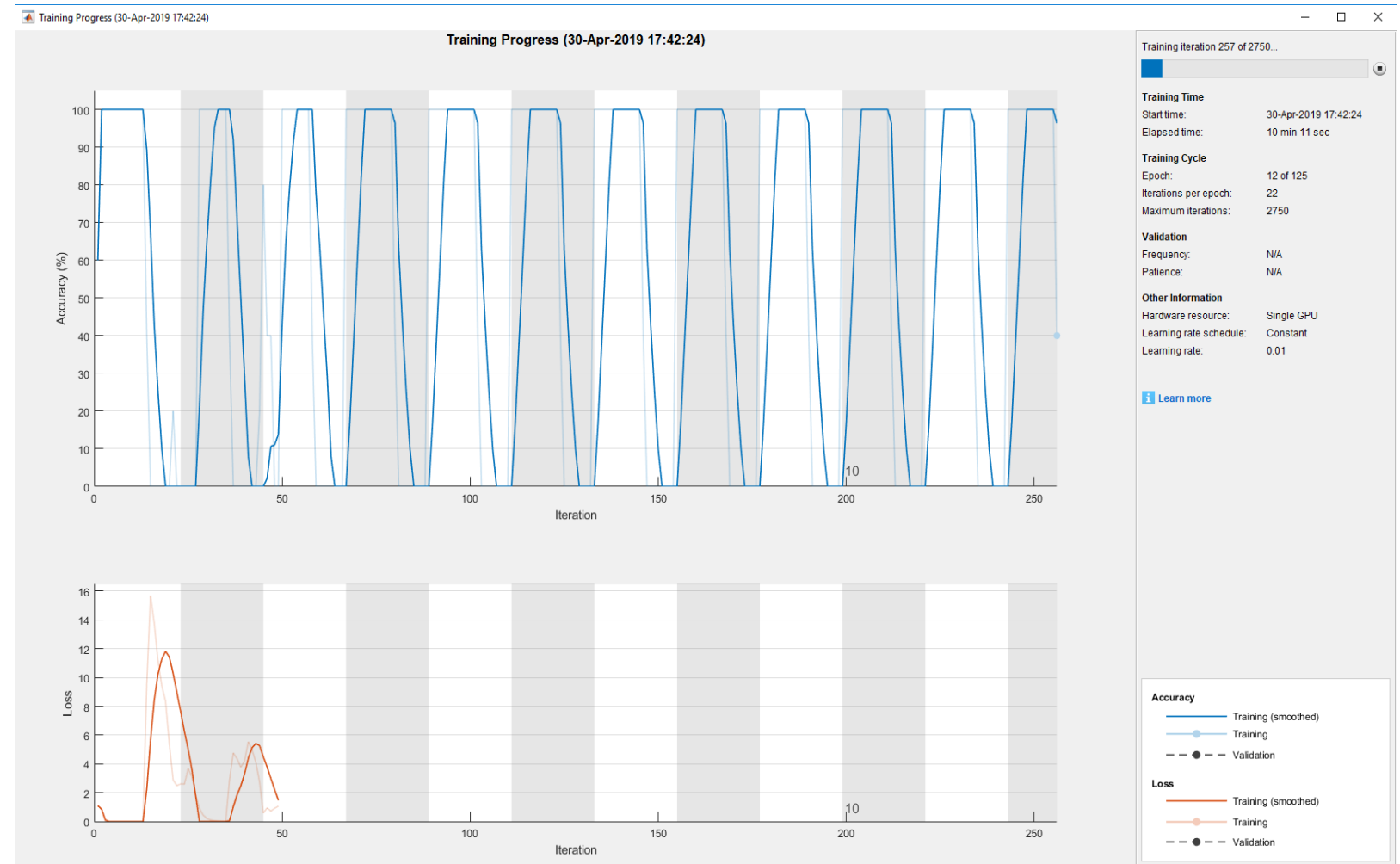
Recurrent Neural Network that carries a memory cell (state) throughout the process



Training LSTMs Directly on Signals?

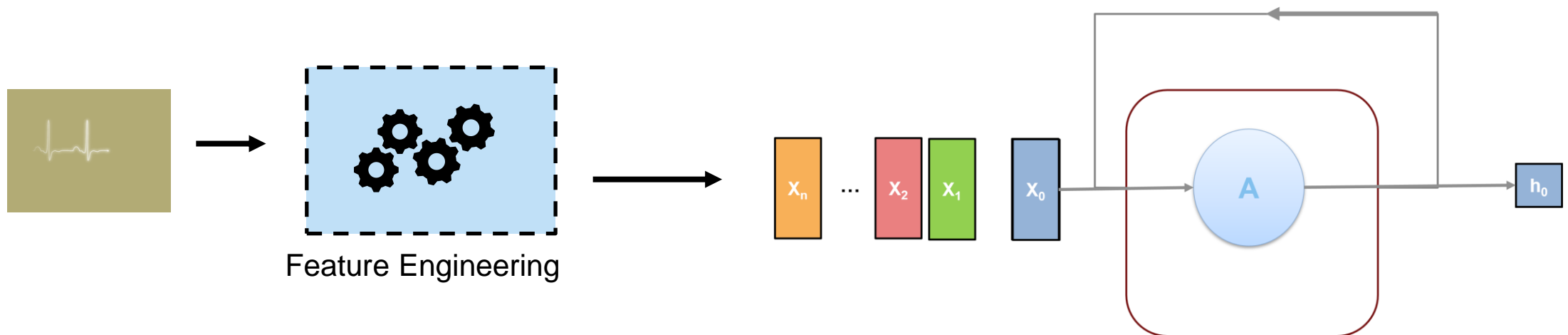
Common Problem

- Long signals subject to signal drift
- Training directly leads to poor network accuracy



Solution: Feature Extraction

- Extracting features from signals will:
 - Preserve important information from signal
 - Have smaller length compared to original signal
- Use extracted feature vectors as input to LSTM



Automatic Feature Extraction with Wavelet Scattering

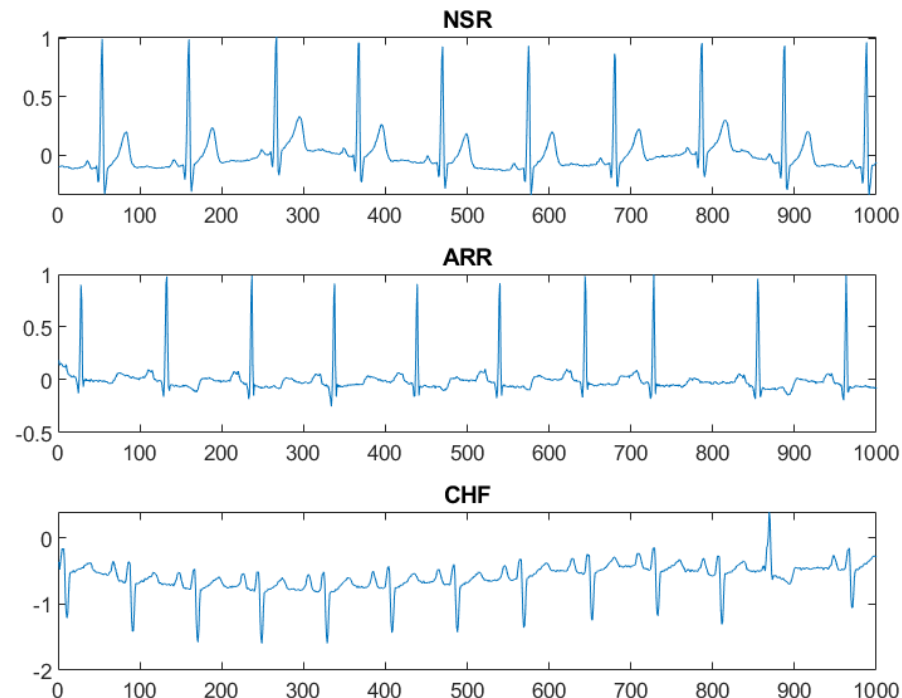
- We will use a technique called wavelet scattering to extract features from our signal

Original Signals

- 65,000+ samples long


Feature extraction

- Dimensions: 499x8
- 499 features with 8 different channels




Deep Learning Workflow – Prepare Data

Data Preparation

 Data cleansing and preparation

 Human insight

 Simulation-generated data

AI Modeling

 Model design and tuning


 Hardware accelerated training

 Interoperability

Simulation & Test

 Integration with complex systems

 System simulation

 System verification and validation

Deployment

 Embedded devices

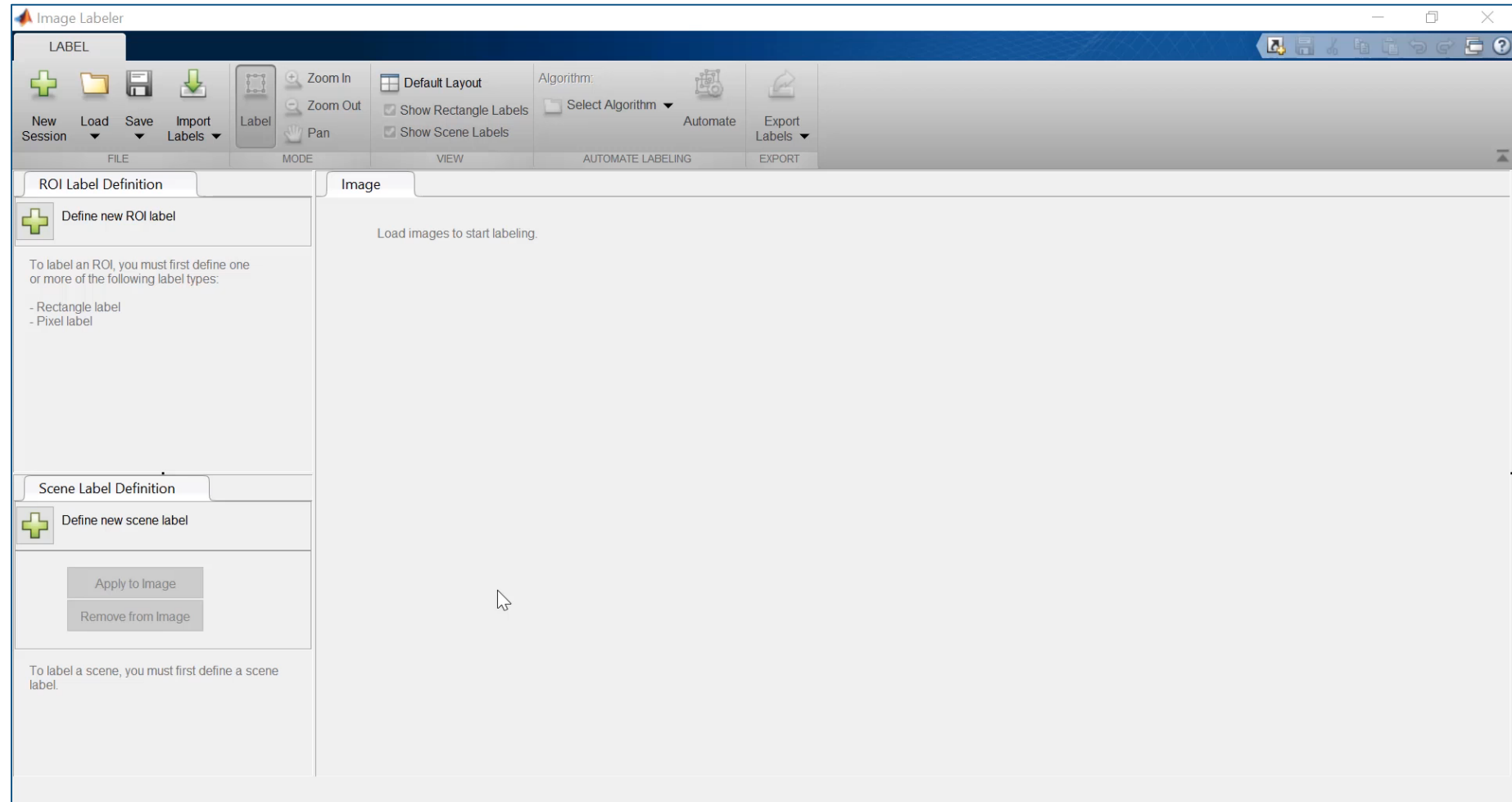
 Enterprise systems

 Edge, cloud, desktop

How do I label my data?

[Image Labeler](#)
[+ Video labeler](#)

Signal Labeler
+ Audio Labeler

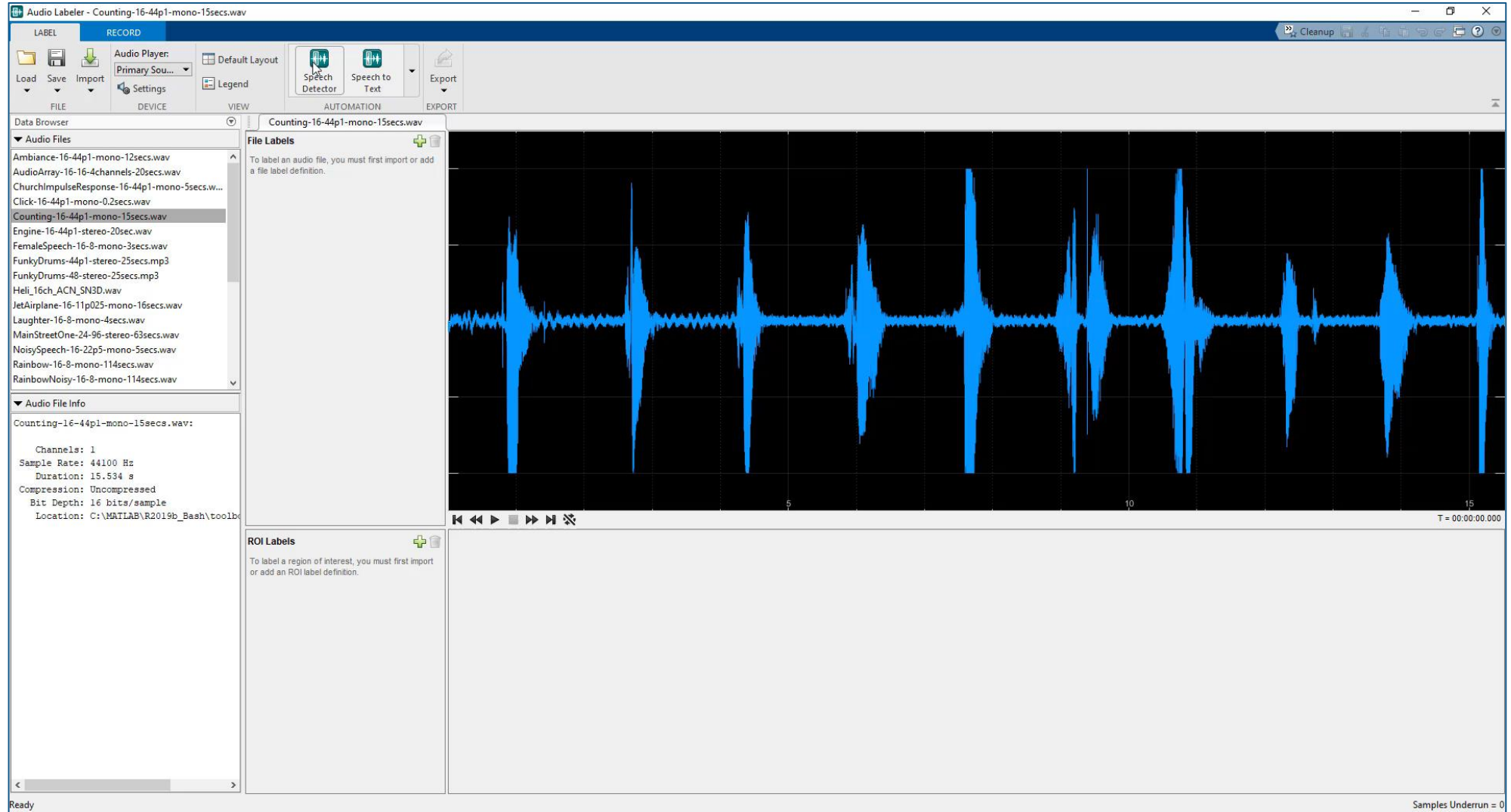


Play the video !

How do I label my data?

Image Labeler
+ Video labeler

[Signal Labeler](#)
[+ Audio Labeler](#)




The screenshot displays the 'Audio Labeler' application window. The title bar reads 'Audio Labeler - Counting-16-44p1-mono-15secs.wav'. The interface is divided into several sections:

- Top Bar:** Includes 'LABEL' and 'RECORD' tabs, a 'Cleanup' button, and a 'Primary Sou...' dropdown.
- Toolbars:** Features icons for 'Load', 'Save', 'Import', 'Settings', 'Default Layout', 'Legend', 'Speech Detector', 'Speech to Text', and 'Export'.
- Data Browser:** A list of audio files with 'Counting-16-44p1-mono-15secs.wav' selected.
- File Labels:** A panel with a plus icon and the text: 'To label an audio file, you must first import or add a file label definition.'
- Audio File Info:** A panel showing metadata for 'Counting-16-44p1-mono-15secs.wav':
 - Channels: 1
 - Sample Rate: 44100 Hz
 - Duration: 15.534 s
 - Compression: Uncompressed
 - Bit Depth: 16 bits/sample
 - Location: C:\MATLAB\R2019b_Bash\toolb...
- ROI Labels:** A panel with a plus icon and the text: 'To label a region of interest, you must first import or add an ROI label definition.'
- Main View:** A large black area displaying a blue audio waveform. The x-axis is marked with '5', '10', and '15'. Playback controls are visible at the bottom of this area.
- Status Bar:** Shows 'Ready' on the left and 'Samples Underrun = 0' on the right.


Play the video !

Deep Learning Workflow – Deploy System

Data Preparation

 Data cleansing and preparation

 Human insight

 Simulation-generated data

AI Modeling

 Model design and tuning


 Hardware accelerated training

 Interoperability

Simulation & Test

 Integration with complex systems

 System simulation

—  System verification and validation
— 

Deployment

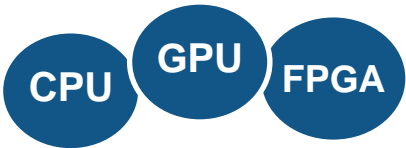
 Embedded devices

 Enterprise systems

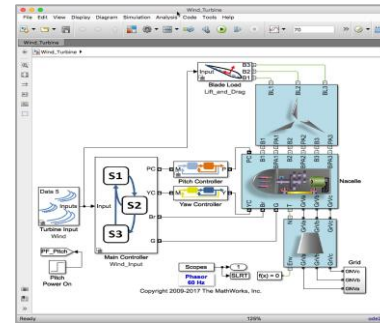
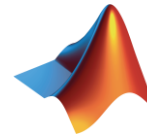
 Edge, cloud, desktop

Deployment and Scaling for A.I.

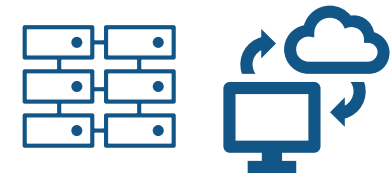
Embedded Systems



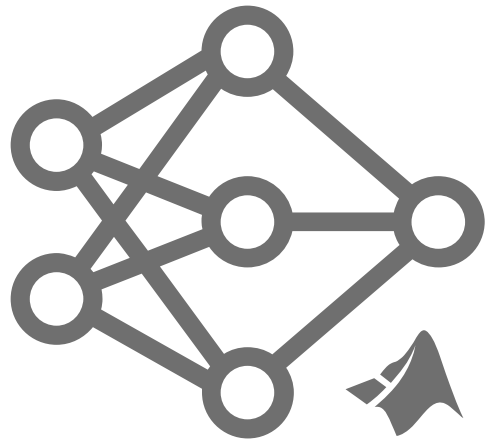
MATLAB



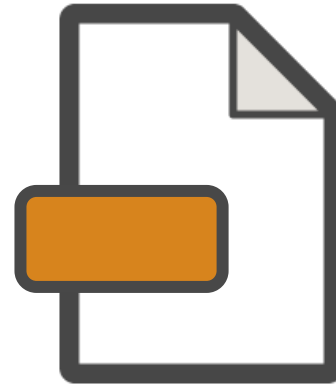
Enterprise Systems



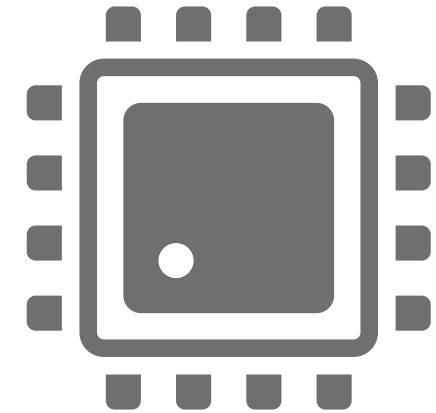
Embedded Deployment – Automatic Code Generation



MATLAB Code

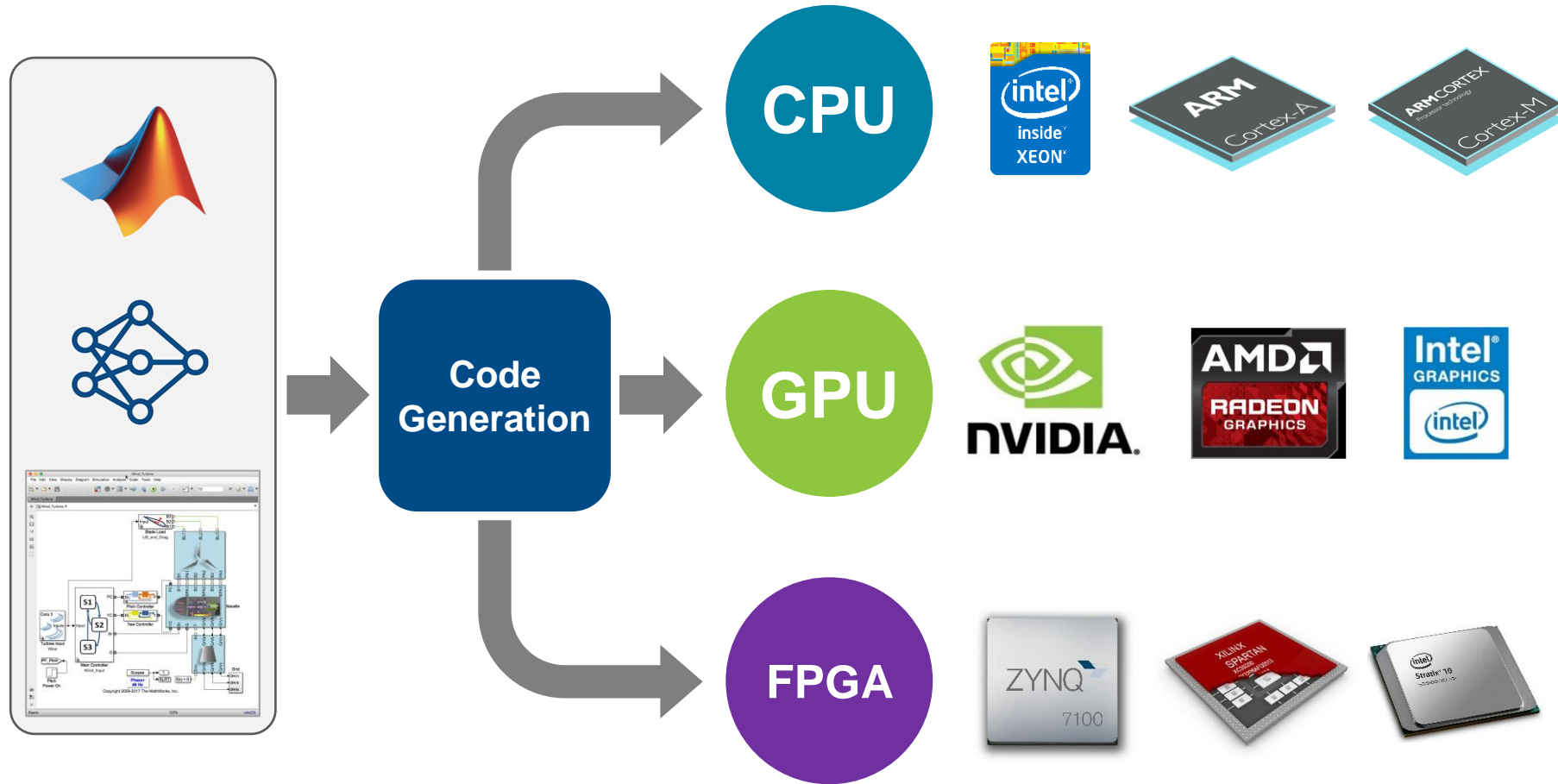


**Auto-generated Code
(C/C++/CUDA)**



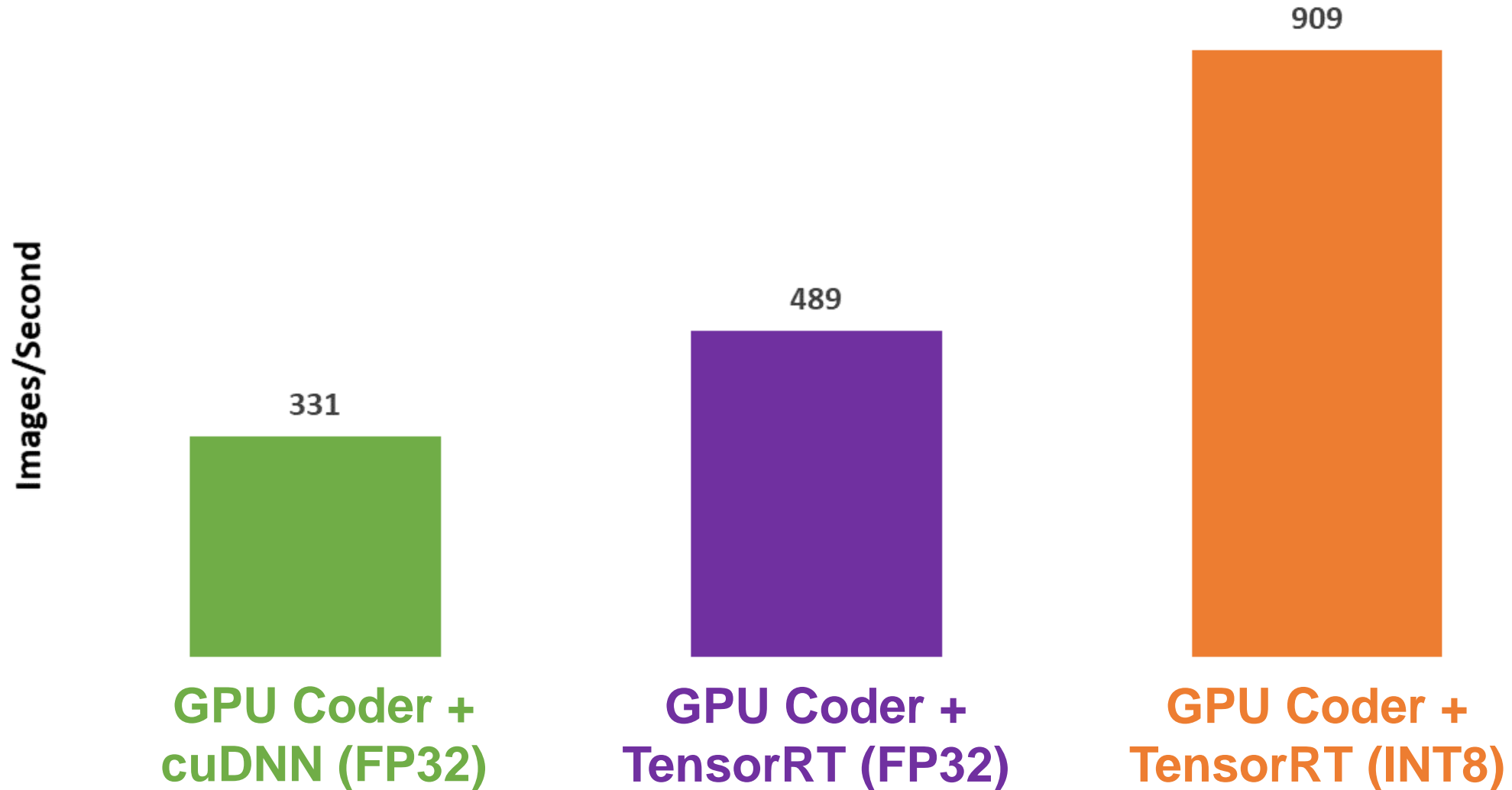
**Deployment
Target**

Deploying Models for Inference

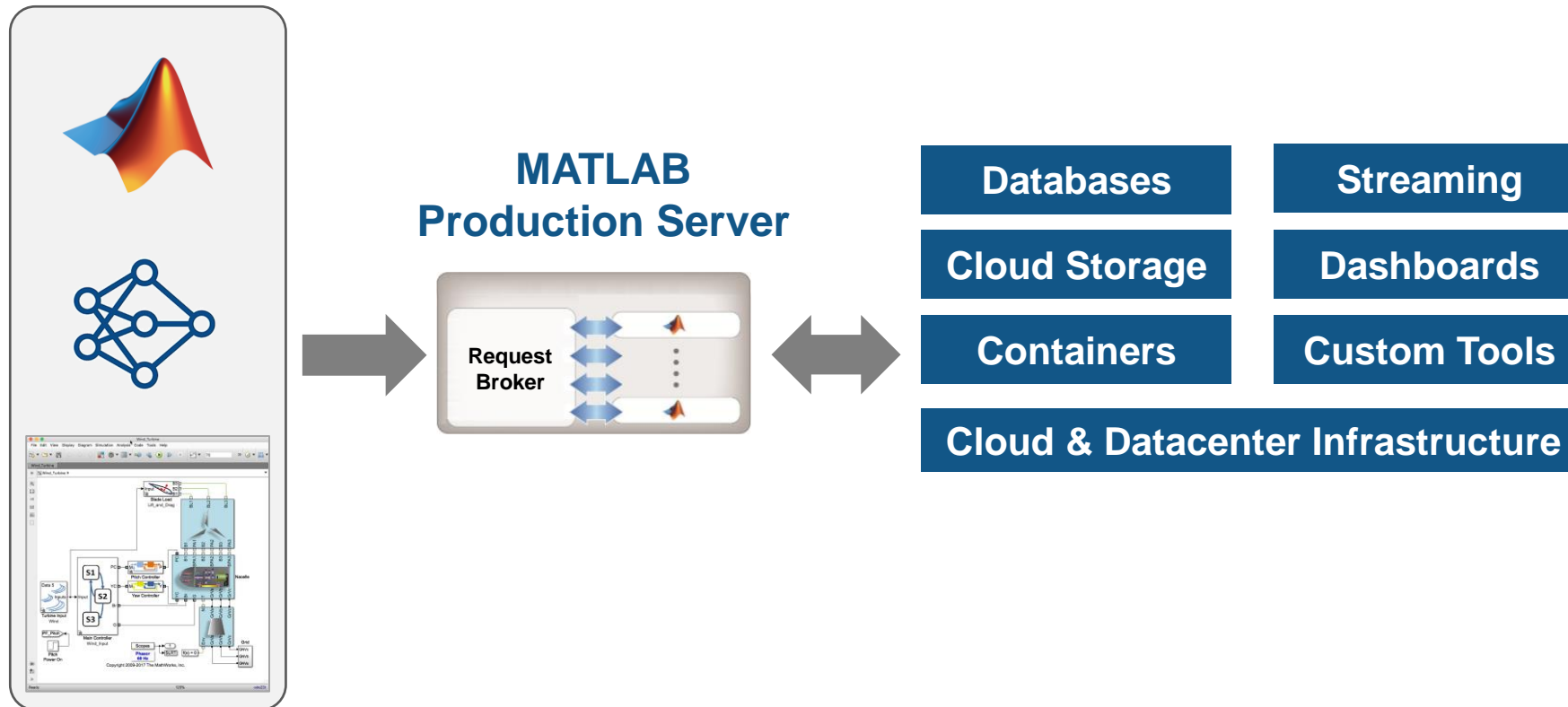


GPU Coder Inference Performance with ResNet-50 on Titan V

Batch 1



Deploy to Enterprise IT Infrastructure



Generate GPU Code for Deep Networks

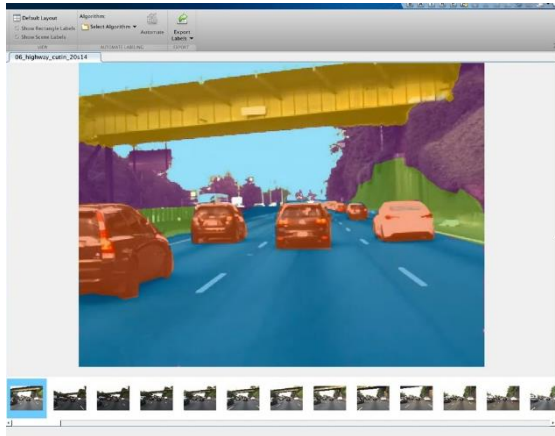
GPU Coder

Generate Code for Deploying Deep Networks



Play the video !

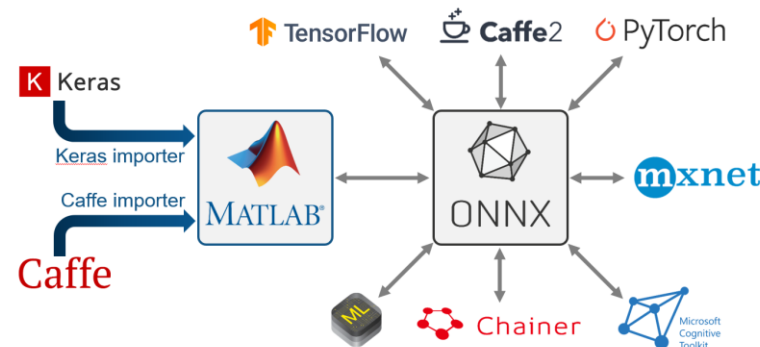
Why Use MATLAB?



MATLAB supports the **data preparation, training, and deployment** workflow



MATLAB has specialized DL tools designed for **scientists and engineers**



MATLAB **interoperates and enhances** Open Source frameworks

MathWorks Engineering Support



Training



Consulting



Onsite Workshops and Seminars



Guided Evaluations



Technical Support

Self-Paced Online Courses

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

Every MATLAB Campus License in AU/NZ has access to these courses, eg: ANU, UNSW, Usyd, UTS, UQ, QUT, etc

Get Started

MATLAB Onramp
2 hours

Deep Learning Onramp
2 hours

Simulink Onramp
3 hours

Stateflow Onramp
Learn the basics of creating, editing, and simulating state machines in Stateflow.
2 hours

Machine Learning Onramp
Learn the basics of practical machine learning methods for classification problems.
2 hours

Reinforcement Learning Onramp
Master the basics of creating intelligent controllers that learn from experience.
Launch Details

Image Processing Onramp
Learn the basics of practical image processing techniques in MATLAB.
Launch Details

Computational Mathematics

*Available only to users at universities that offer campus-wide online training access.

Solving Nonlinear Equations with MATLAB
1.5 hours

Solving Ordinary Differential Equations with MATLAB
2 hours

Introduction to Linear Algebra with MATLAB
1.5 hours

Introduction to Statistical Methods with MATLAB
2 hours

Introduction to Symbolic Math with MATLAB
2 hours

Core MATLAB Functionality

MATLAB Fundamentals
20 hours

MATLAB Programming Techniques
14 hours

MATLAB for Financial Applications
20 hours

Data Analytics

MATLAB for Data Processing and Visualization
7 hours

Machine Learning with MATLAB
14 hours

Deep Learning with MATLAB
14 hours

6 in-depth courses for enhancing MATLAB skills