



Doğukan M. KILIÇ
Yapay Zeka ve Makine Öğrenmesi
Mühendisi

Pamukkale Üniversitesi

29.09.2025

MATLAB ile Yapay Zeka





Agenda

01 Introduction

- What is MATLAB?
- MATLAB for Artificial Intelligence

02 What is Machine Learning?

- Machine Learning Workflow
- Machine Learning vs Deep Learning
- Types of Machine Learning

03 Statistics and Machine Learning Introduction & Toolbox

- Classification Learner App
- Regression Learner App
- Clustering

04 What is Deep Learning?

- Deep Learning Workflow

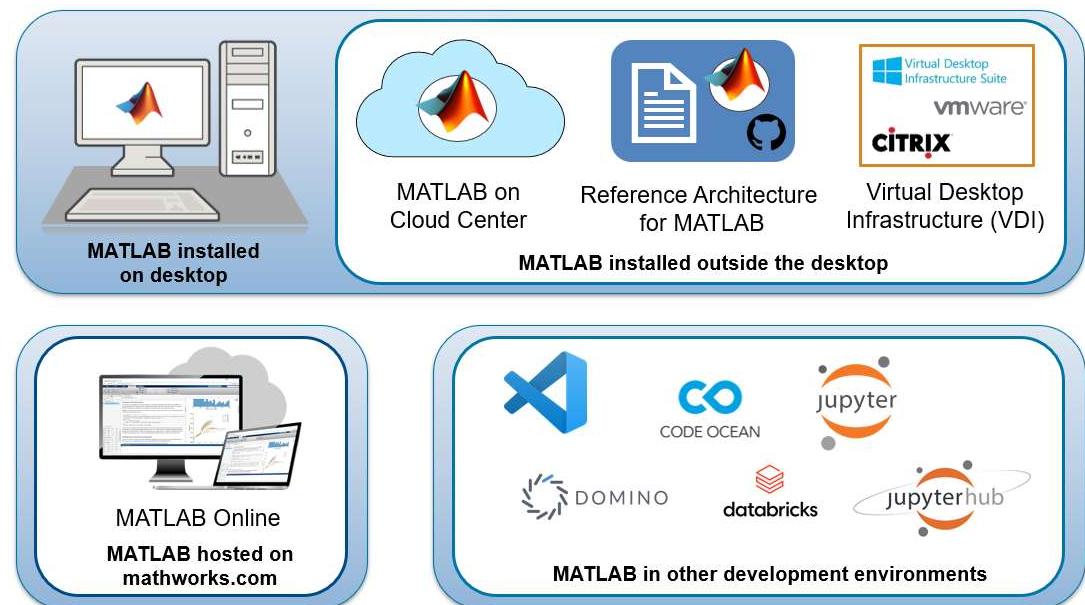
Example: Deployment Algorithm to Microprocessor

Example for PAÜ: Deep Learning for Signals

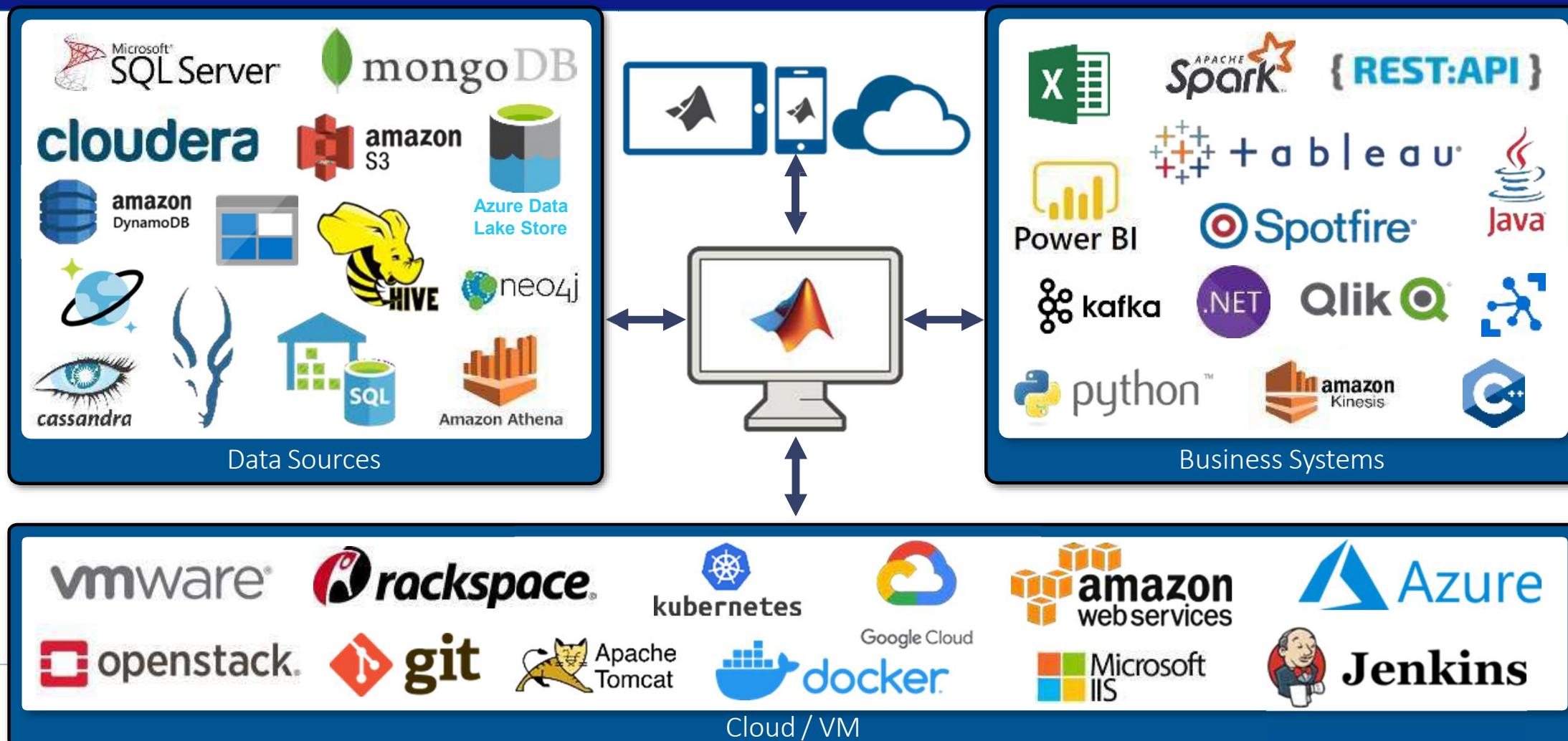
Example for PAÜ: Code Generation for ECG Data Project

What is MATLAB?

- Interactive Development Environment

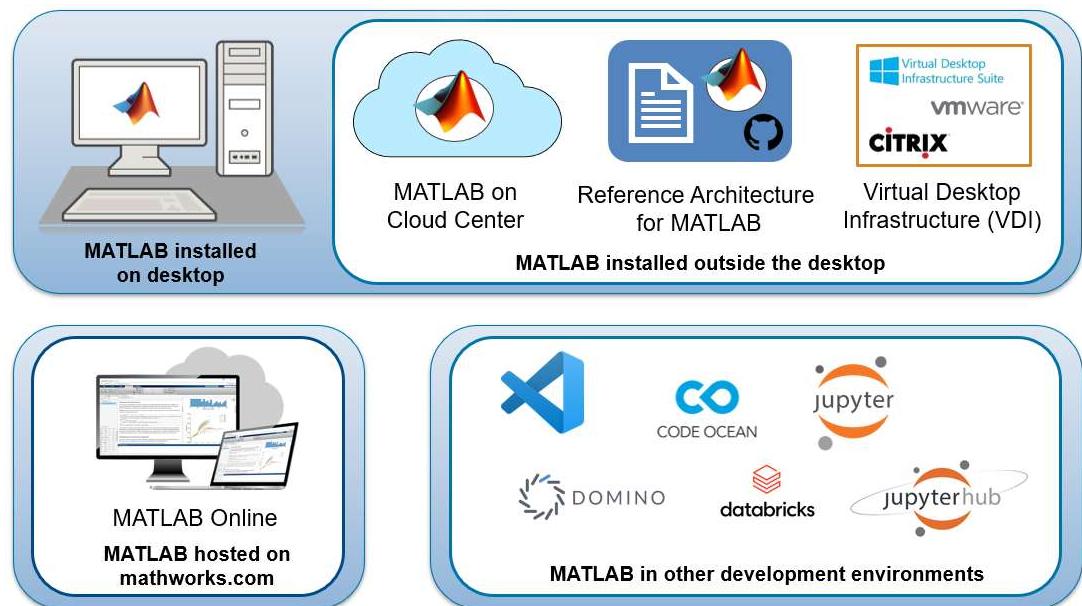


MATLAB and the Analytics Ecosystem



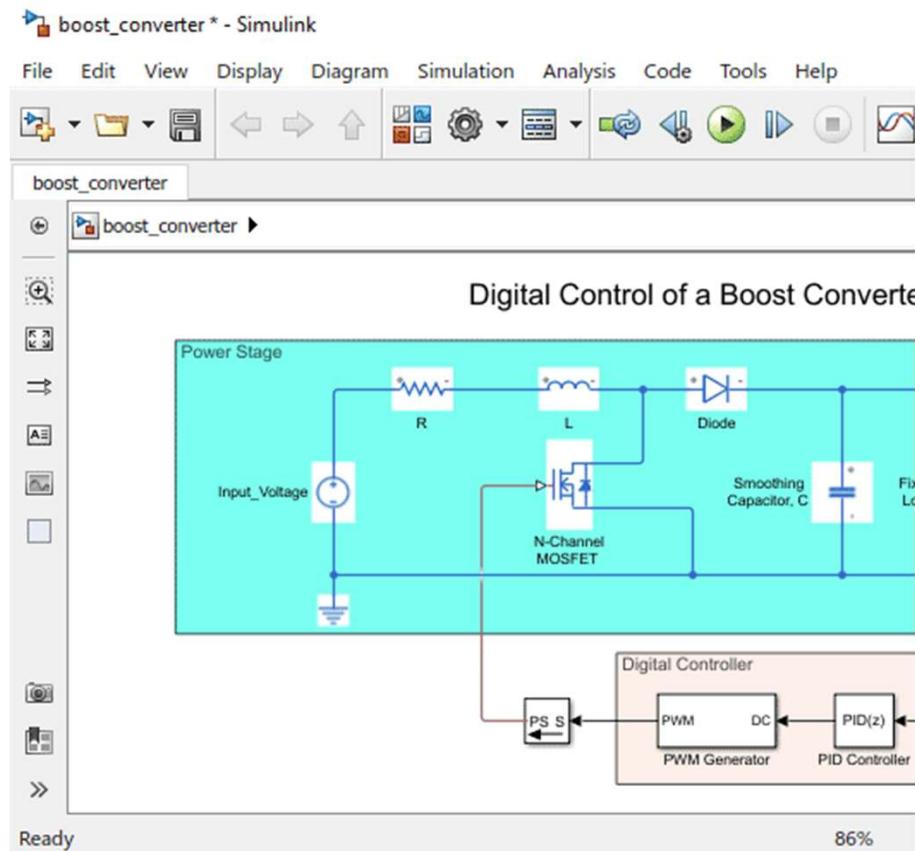
What is MATLAB?

- Interactive Development Environment
- Technical Computing Language
- Data Analysis and Visualization
- Algorithm Development and Application Development

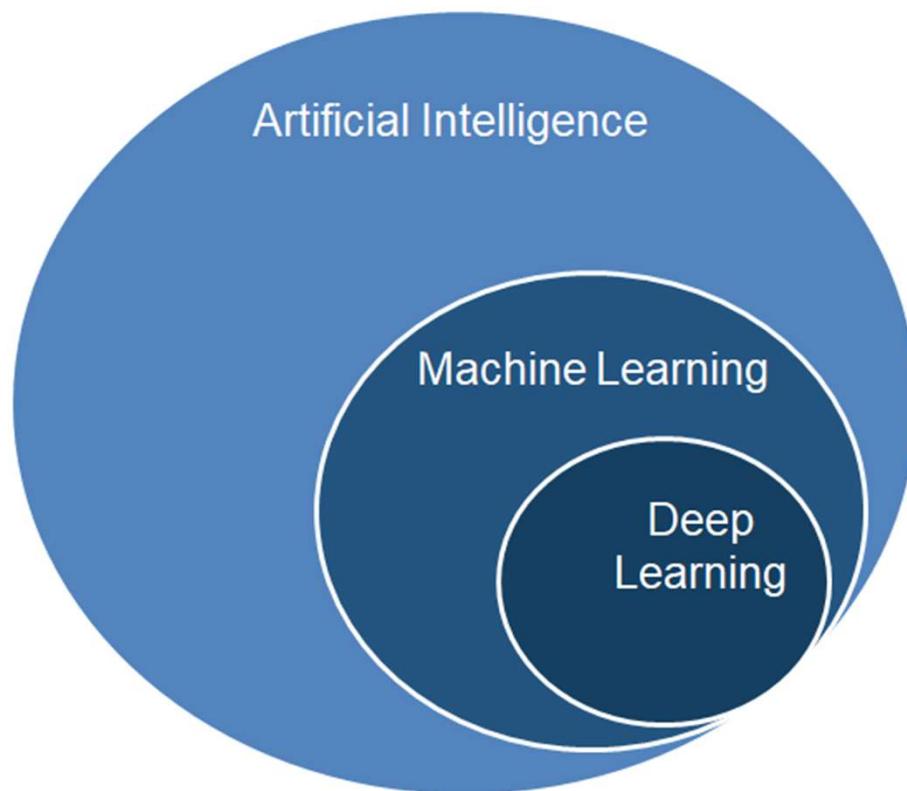


What is Simulink?

- Graphical environment
- Integration with MATLAB
- Multi-domain modelling
- Simulation and validation
- Model-Based Design



MATLAB for Artificial Intelligence



- Machine Learning
- Deep Learning
- Image Processing
- Reinforcement Learning
- Predictive Maintenance
- Data Science / Data Analytics
- Signal Processing
- ...and more



Machine Learning is Everywhere

Solution is too complex for hand written rules or equations



Speech
Recognition



Object
Recognition



Engine Health
Monitoring

learn complex non-linear relationships

Solution needs to adapt with changing data



Weather
Forecasting



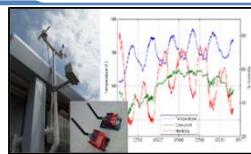
Energy Load
Forecasting



Stock Market
Prediction

update as more data becomes available

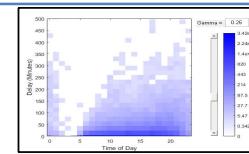
Solution needs to scale



IoT
Analytics



Taxi
Availability

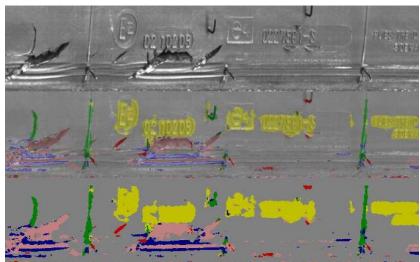


Airline Flight
Delays

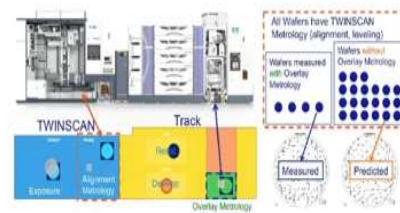
learn efficiently from very large data sets



Machine Learning is Everywhere



Tire Wear



[Overlay metrology improvement](#)



[Telecom customer churn prediction](#)



[Forecasting & Risk Analysis](#)



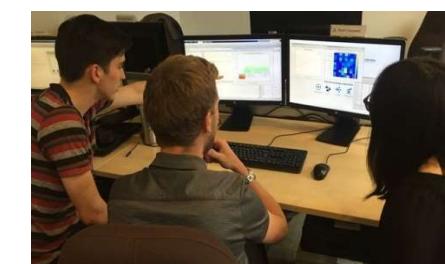
[Detect Oversteer](#)



[Monitor Deployed Compressors using Digital Twin](#)



[Building energy use optimization](#)



[Portfolio Allocation](#)



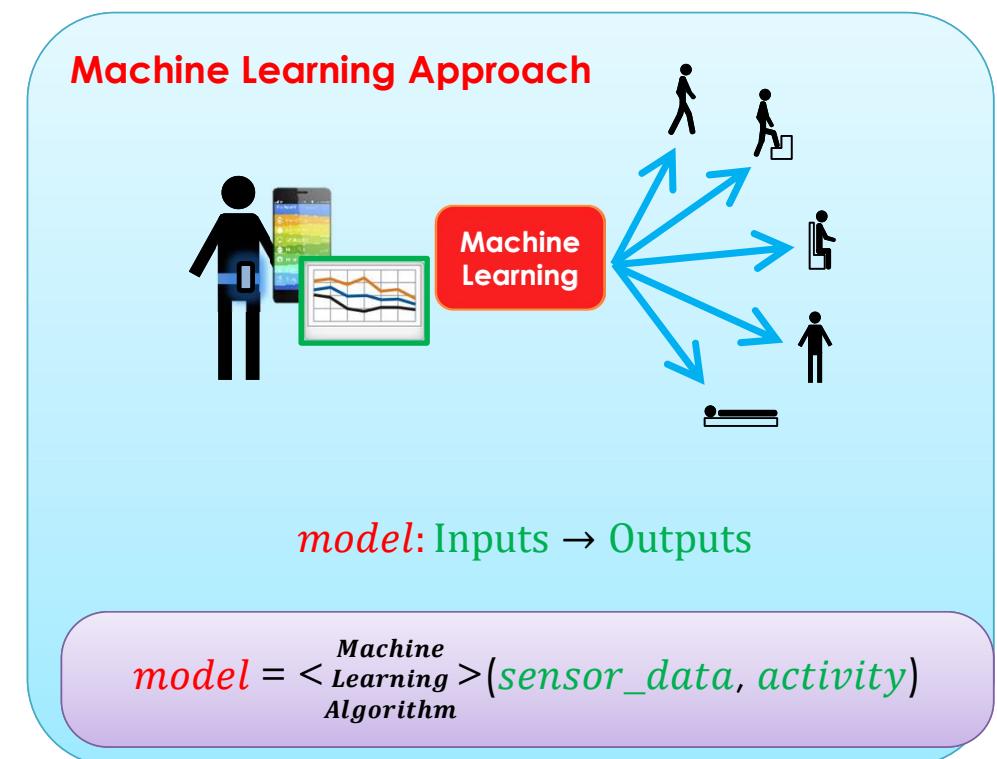
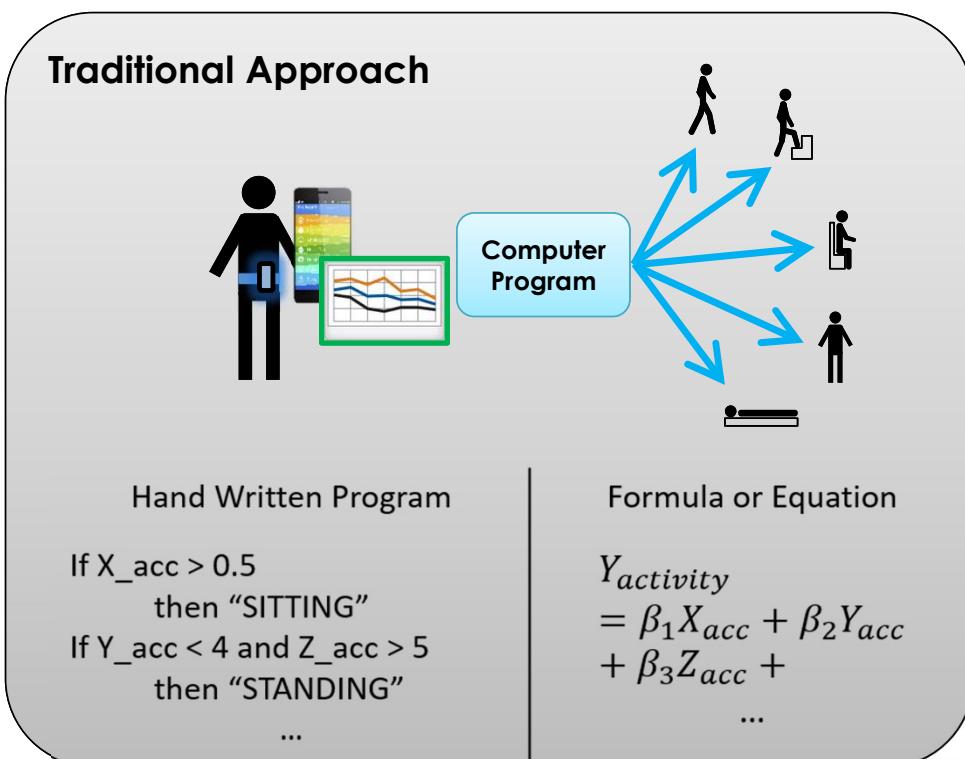
BİZİ TAKİP EDİN!



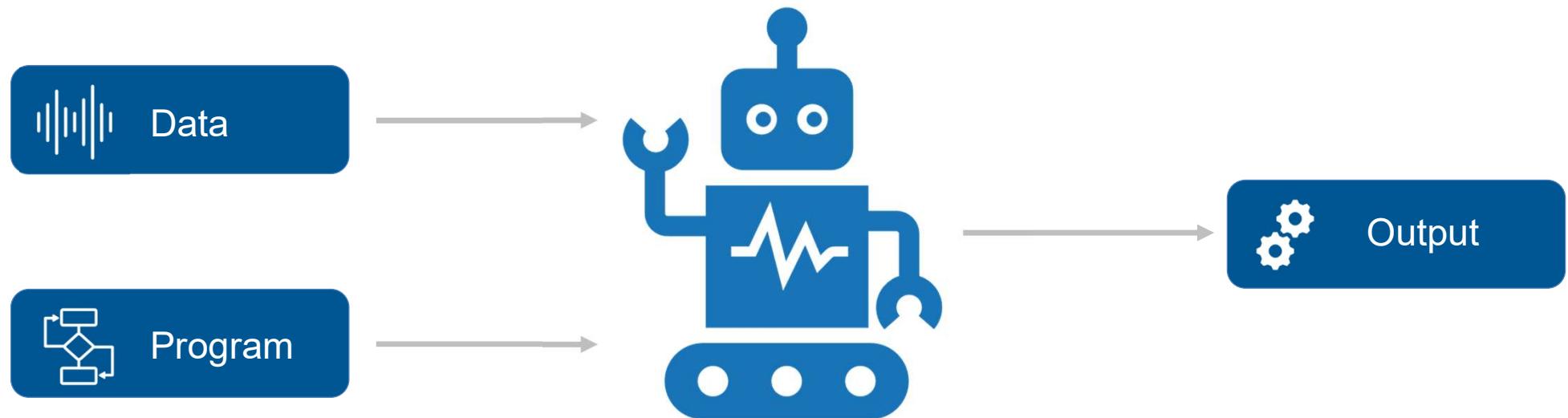
What is Machine Learning?

Machine learning uses **data** and produces a **program** to perform a **task**

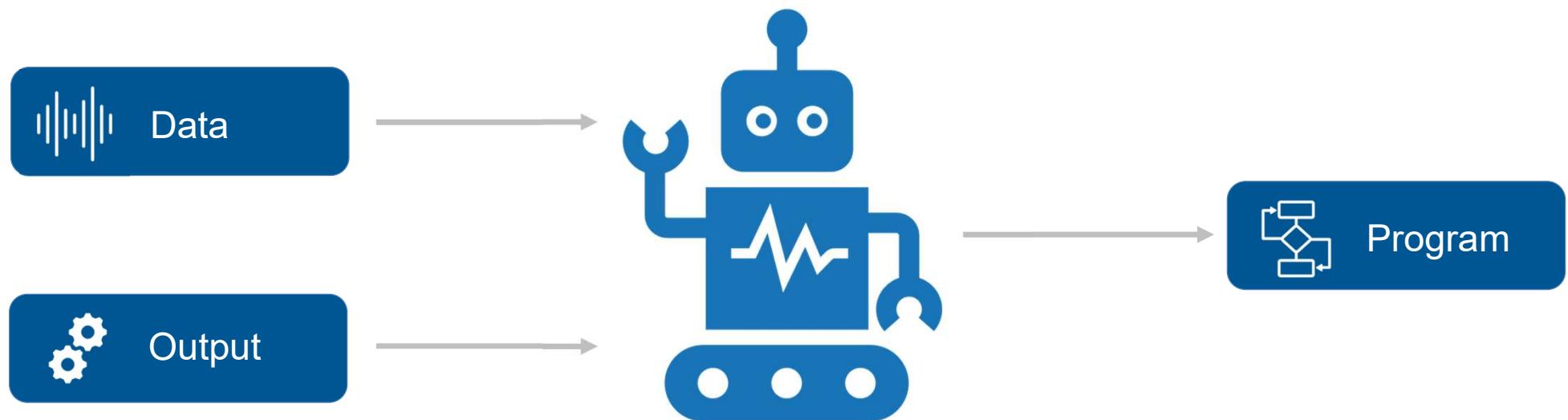
Task: Human Activity Detection



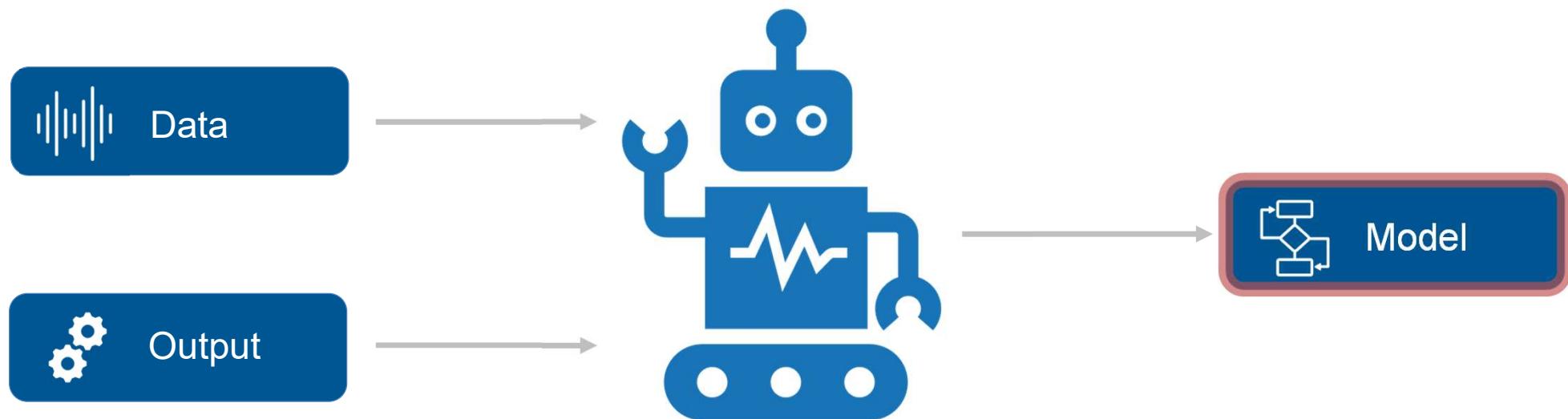
There are two ways to get a robot to do what you want



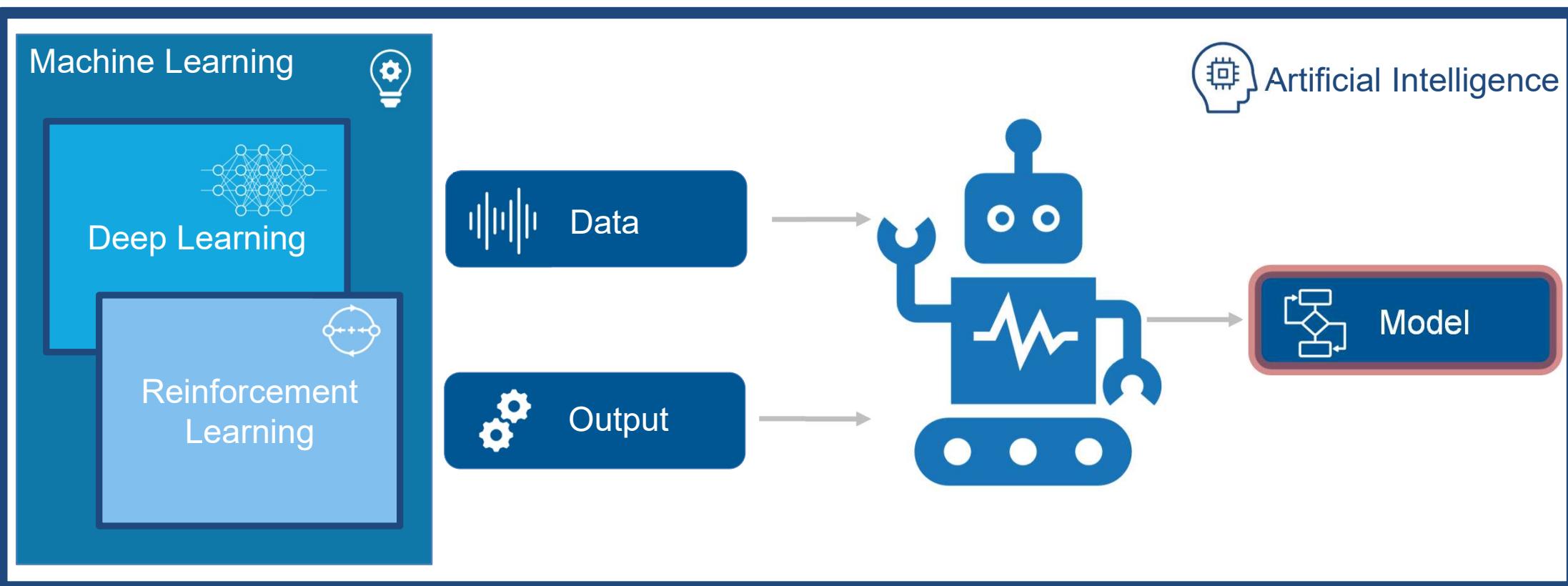
There are two ways to get a robot to do what you want



There are two ways to get a robot to do what you want



There are two ways to get a robot to do what you want



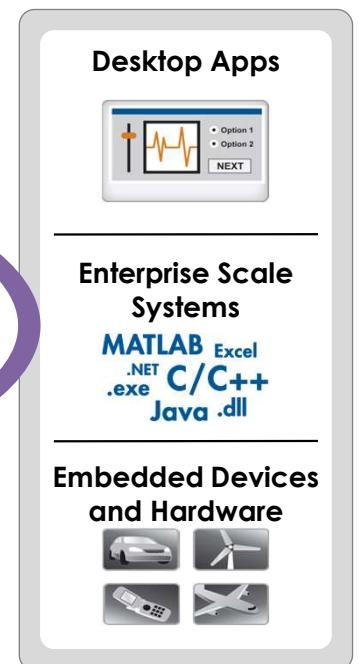
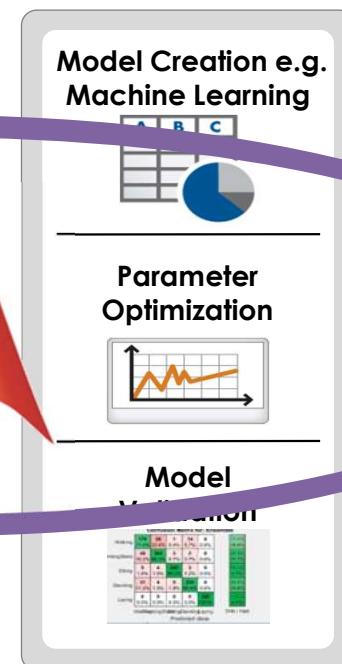
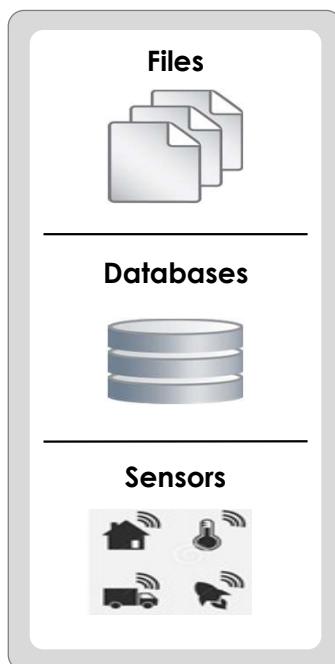
Machine Learning Workflow

Access and Explore Data

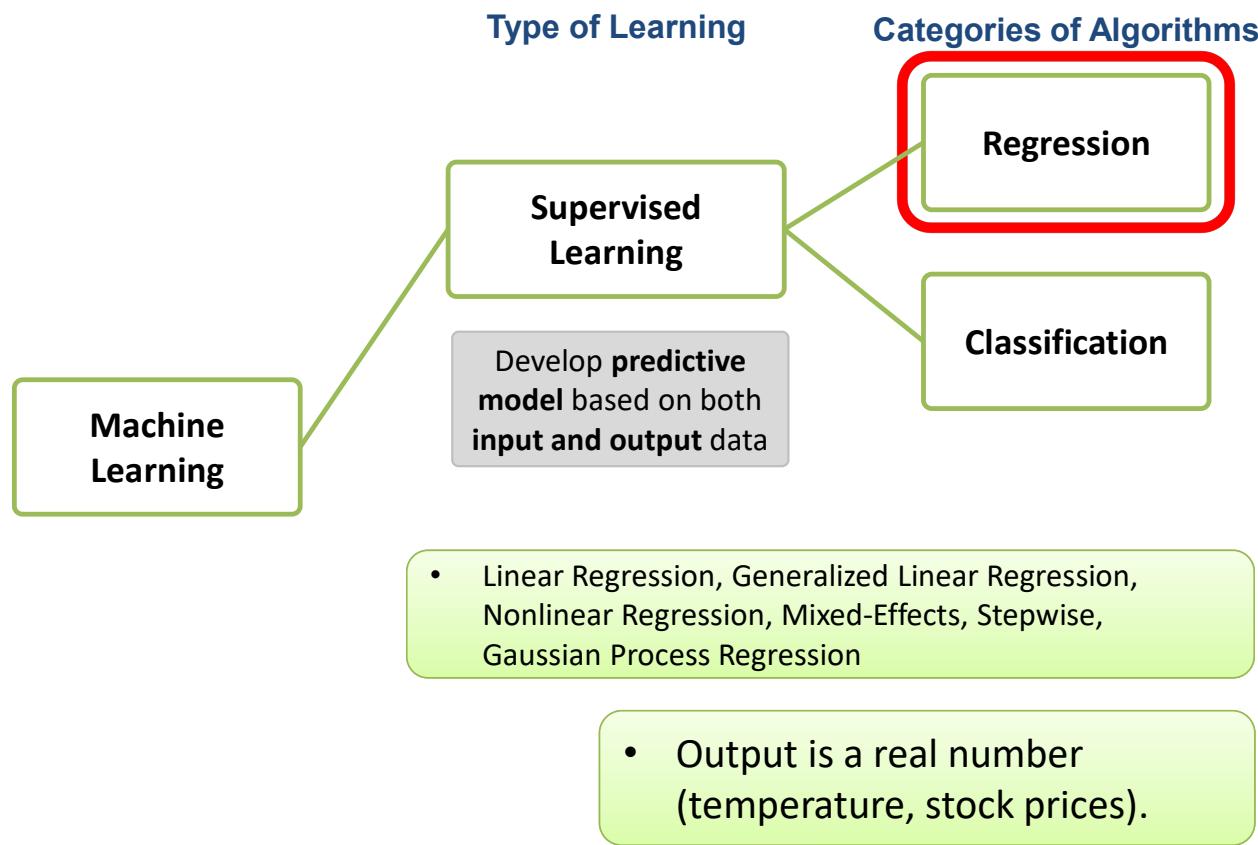
Preprocess Data

Develop Predictive Models

Integrate Analytics with Systems

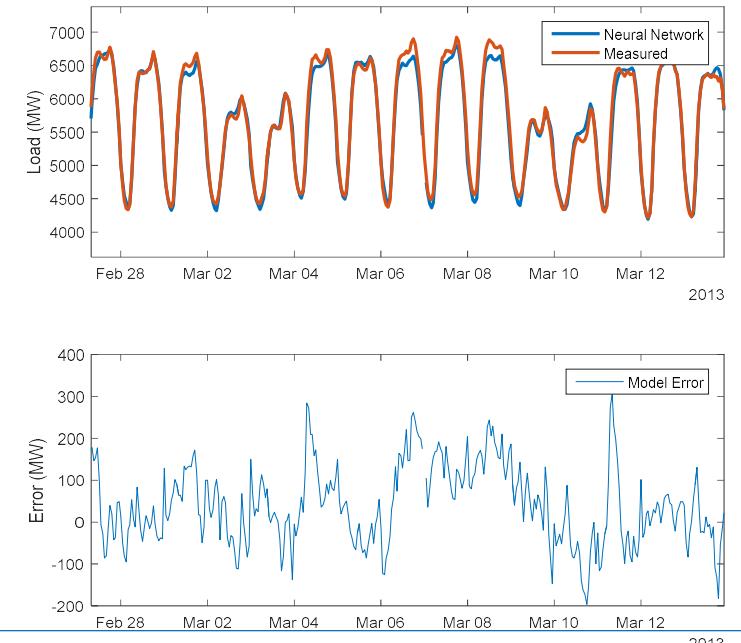


Types of Machine Learning

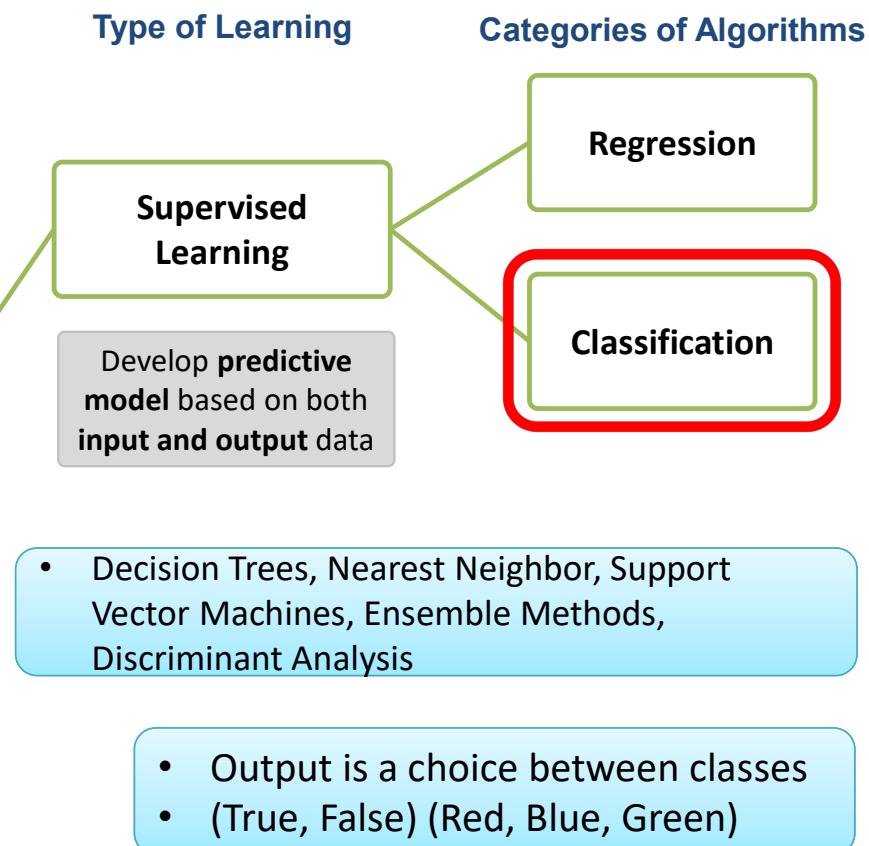


Objective:

Easy and accurate computation of day-ahead system load forecast



Types of Machine Learning



Objective:

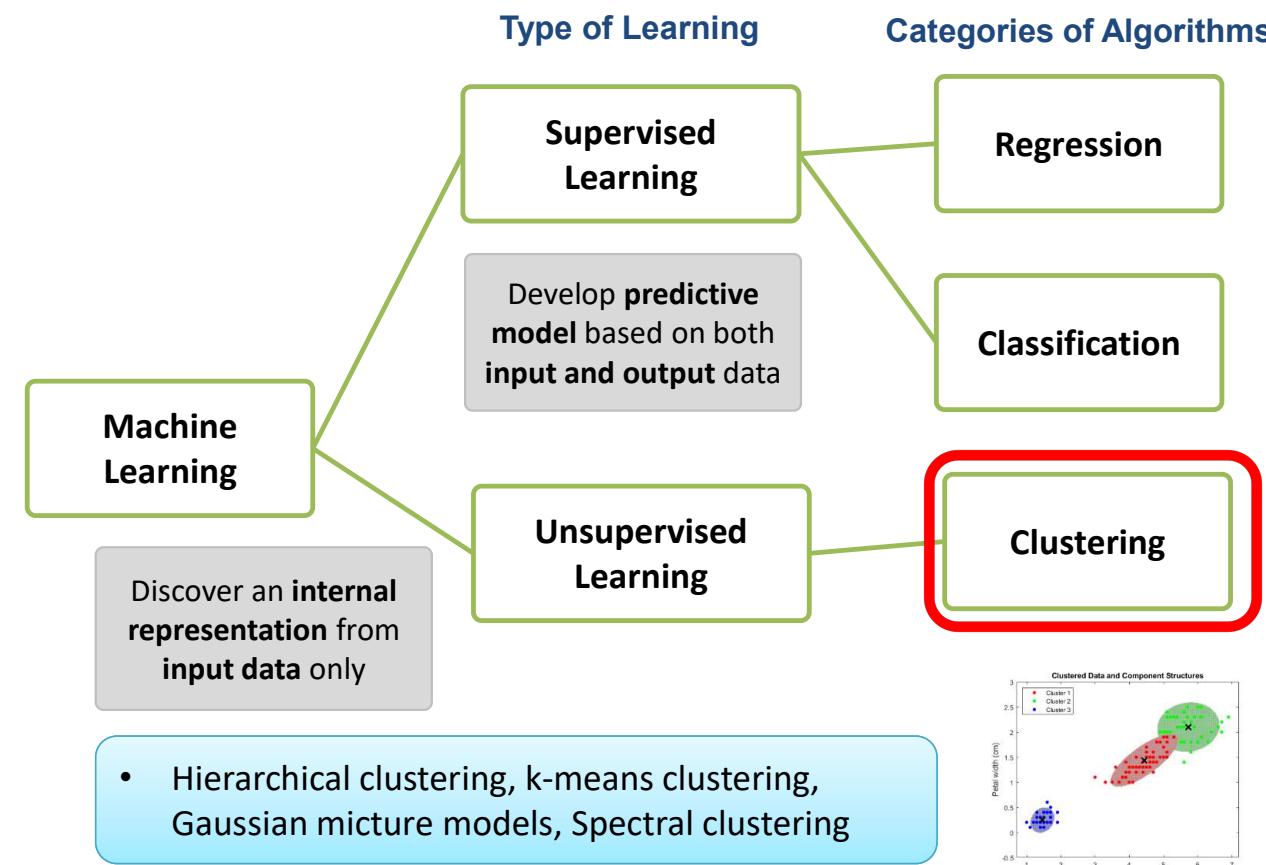
Train a classifier to classify human activity from sensor data

Data:

Inputs	3-axial Accelerometer 3-axial Gyroscope	
Outputs		



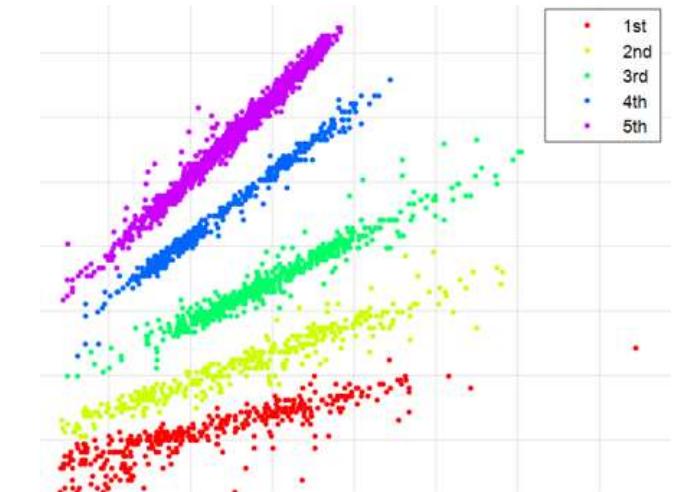
Types of Machine Learning



Objective:

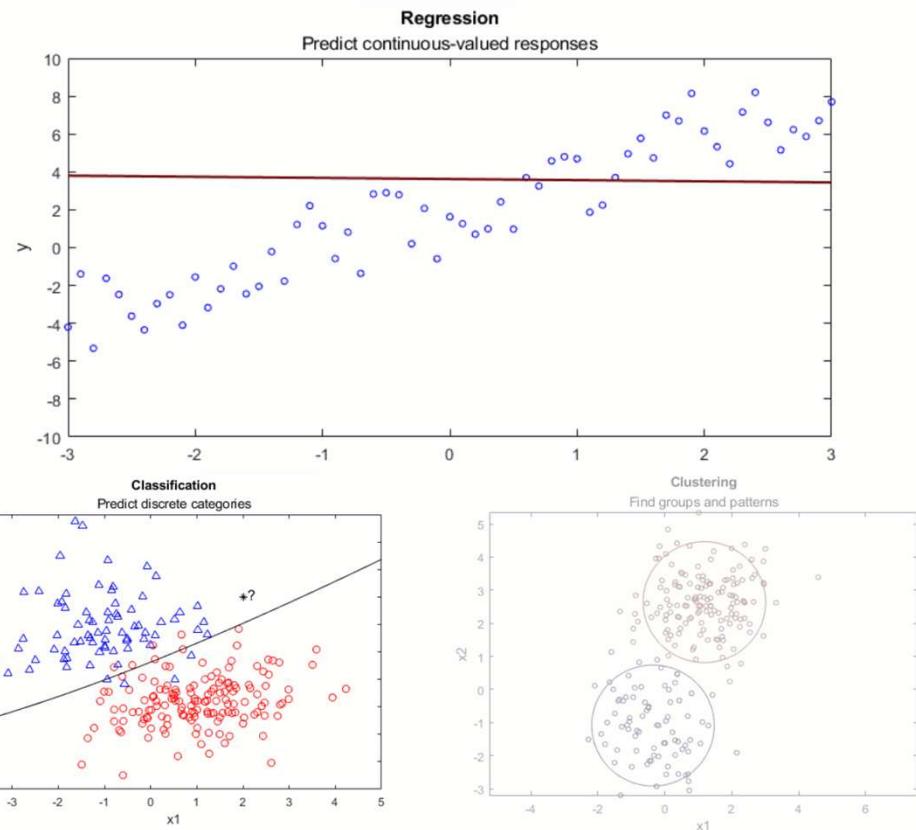
Given data for engine speed and vehicle speed, identify clusters

- Discover a good internal representation
- Learn a low dimensional representation

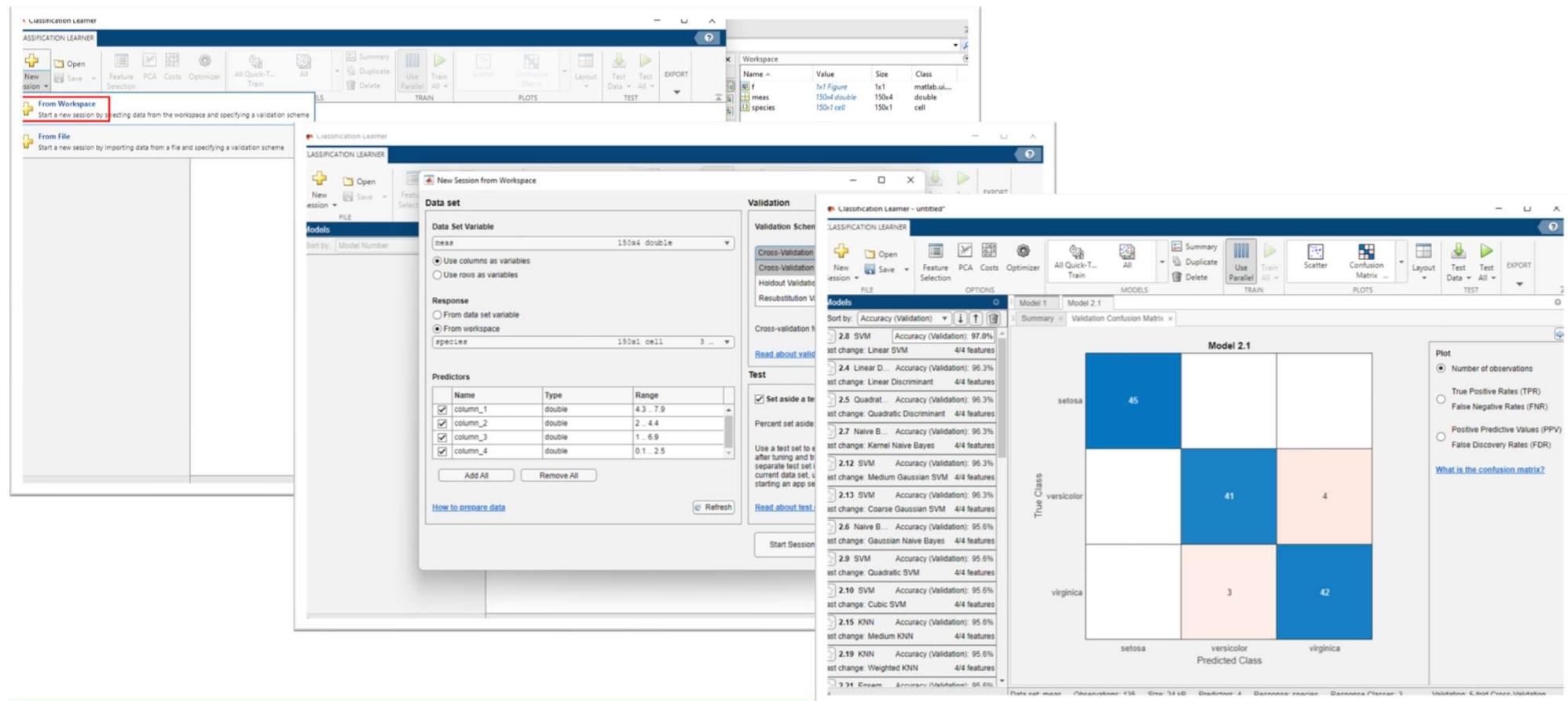


Statistic and Machine Learning Toolbox

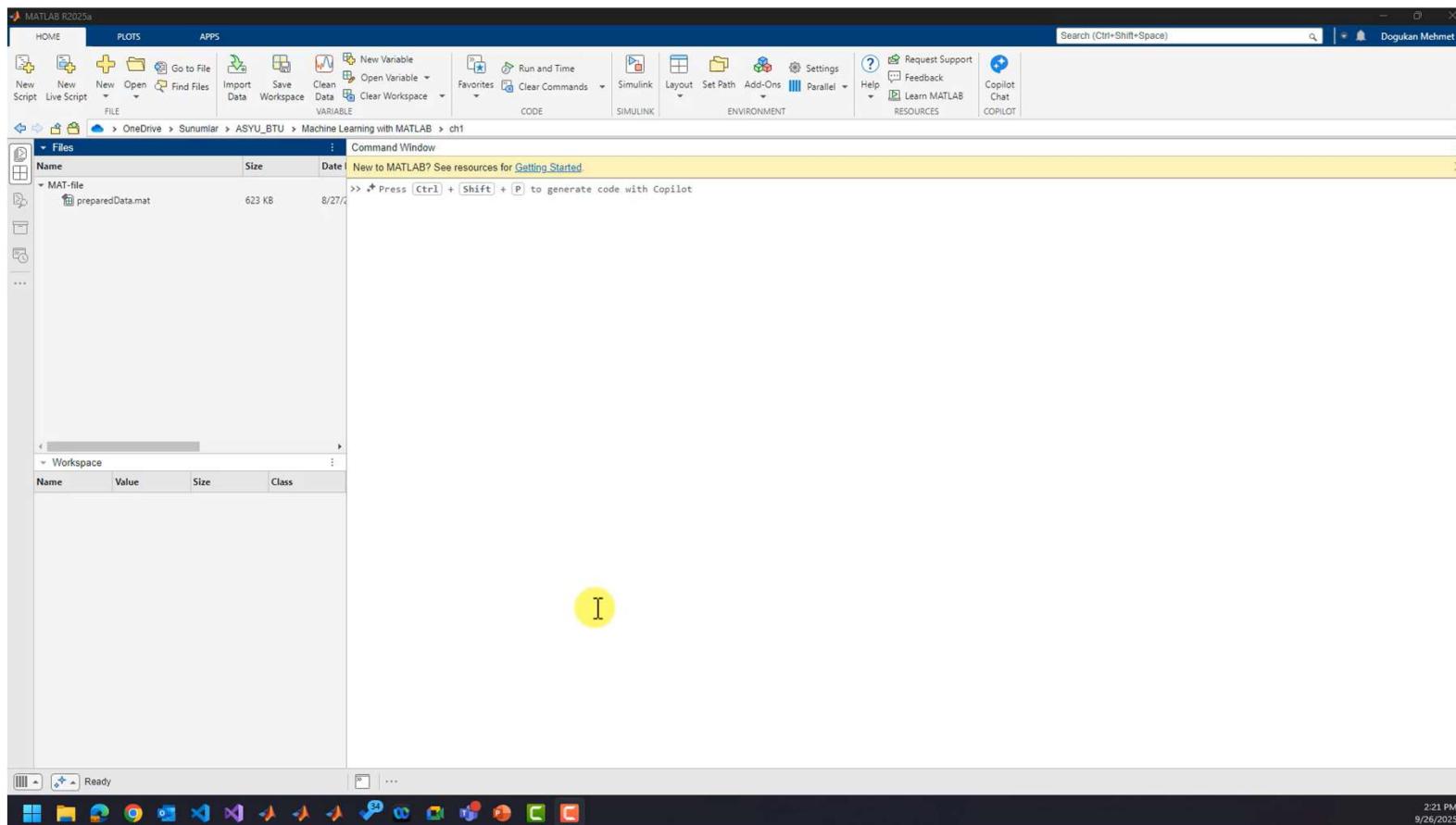
- ✓ Provides functions and apps to describe, analyze, and model data. Use descriptive statistics, visualizations, and clustering for exploratory data analysis; fit probability distributions to data.
- ✓ Regression and classification algorithms let you draw inferences from data and build predictive models either interactively, using the Classification and Regression Learner apps, or programmatically, using AutoML.



Classification Learner App

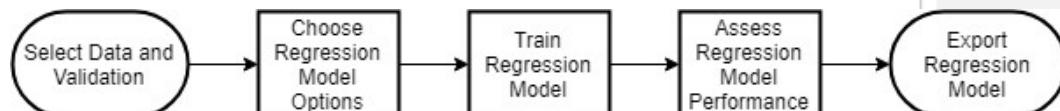
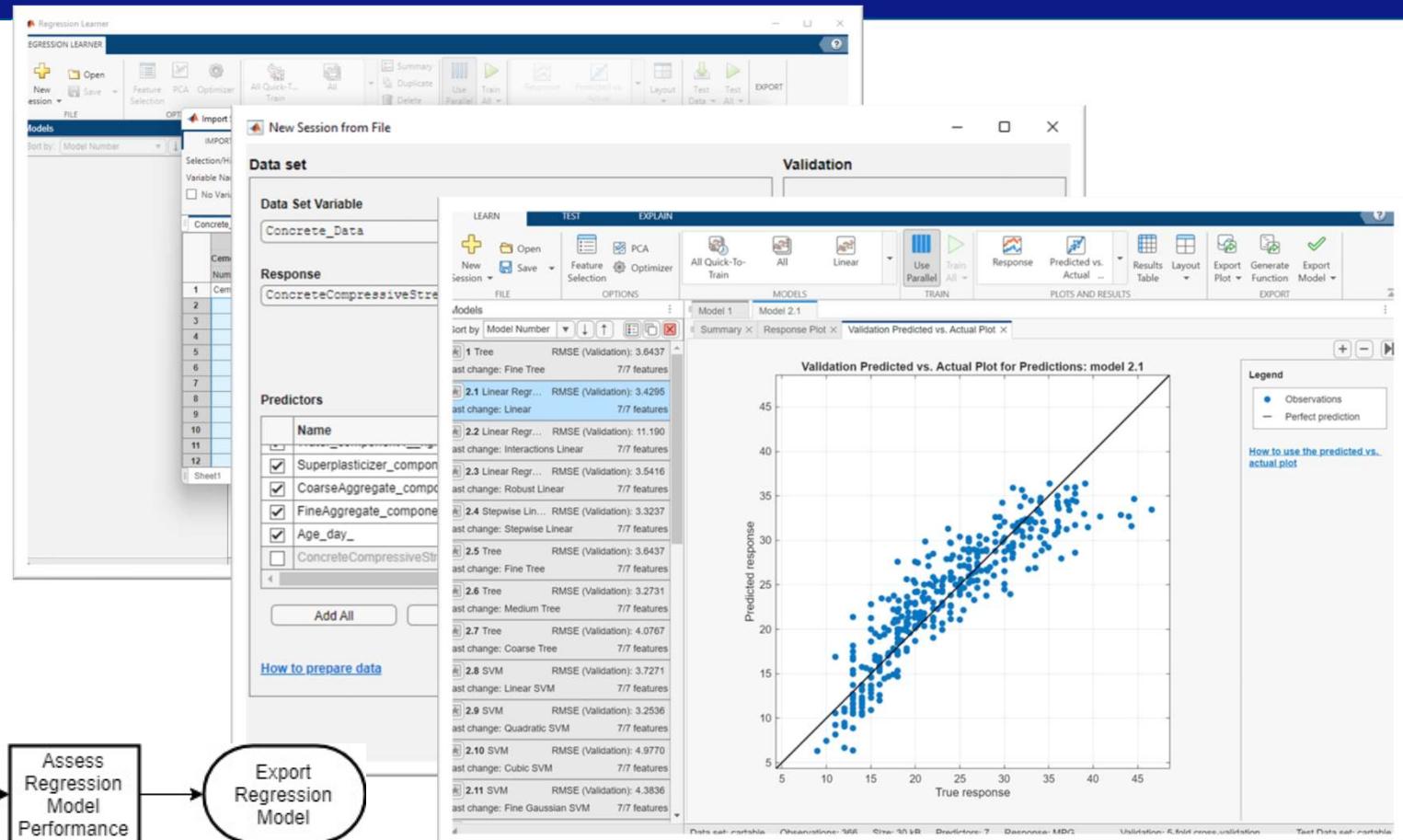


Classification Learner App

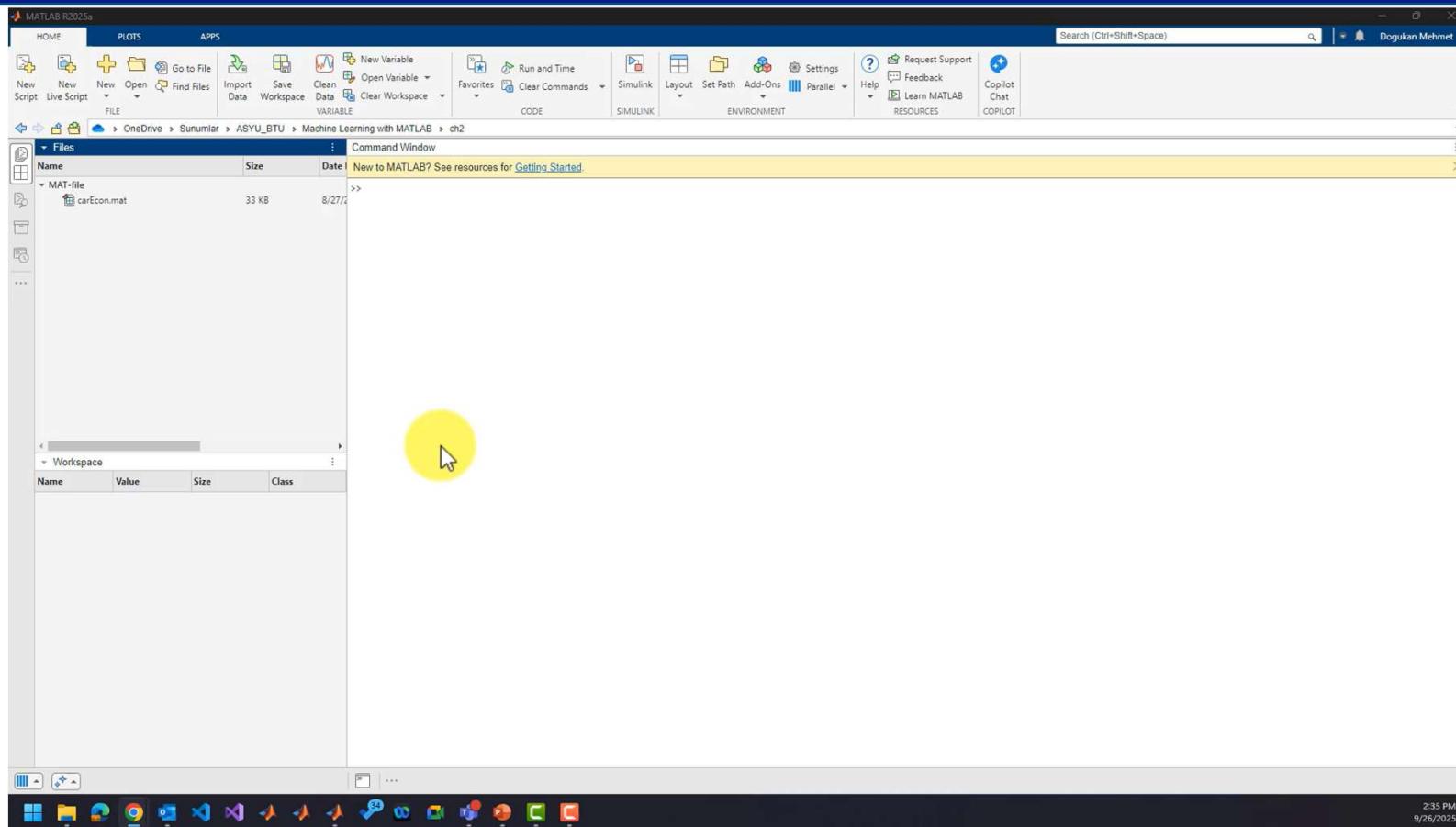


Regression Learner App

Regression models describe the relationship between a response (output) variable, and one or more predictor (input) variables.



Regression Learner App

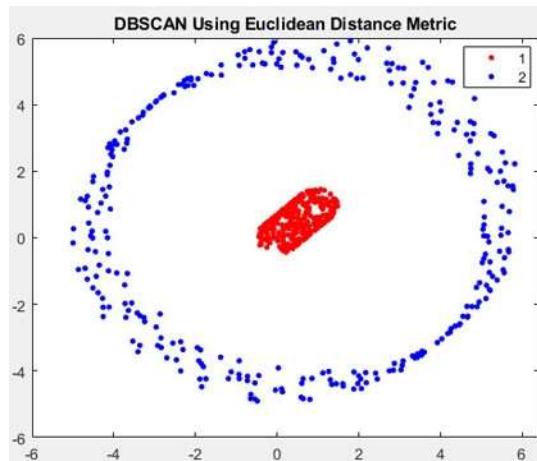


Unsupervised Learning

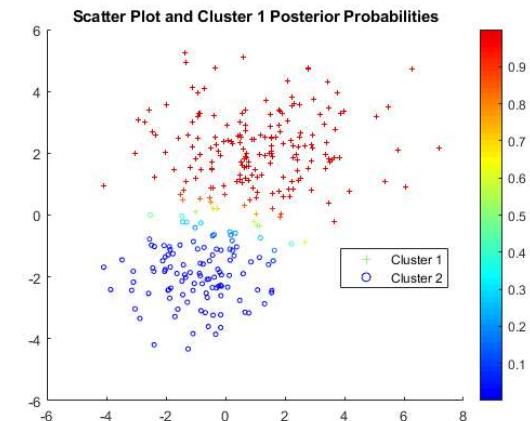
Clustering

Data Clustering / Unsupervised Learning

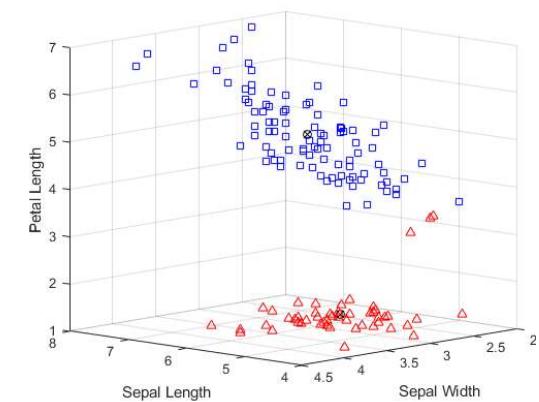
Discover patterns by grouping data using k-means, k-medoids, DBSCAN, hierarchical clustering, and Gaussian mixture and hidden Markov models.



Applying DBSCAN to two concentric groups.



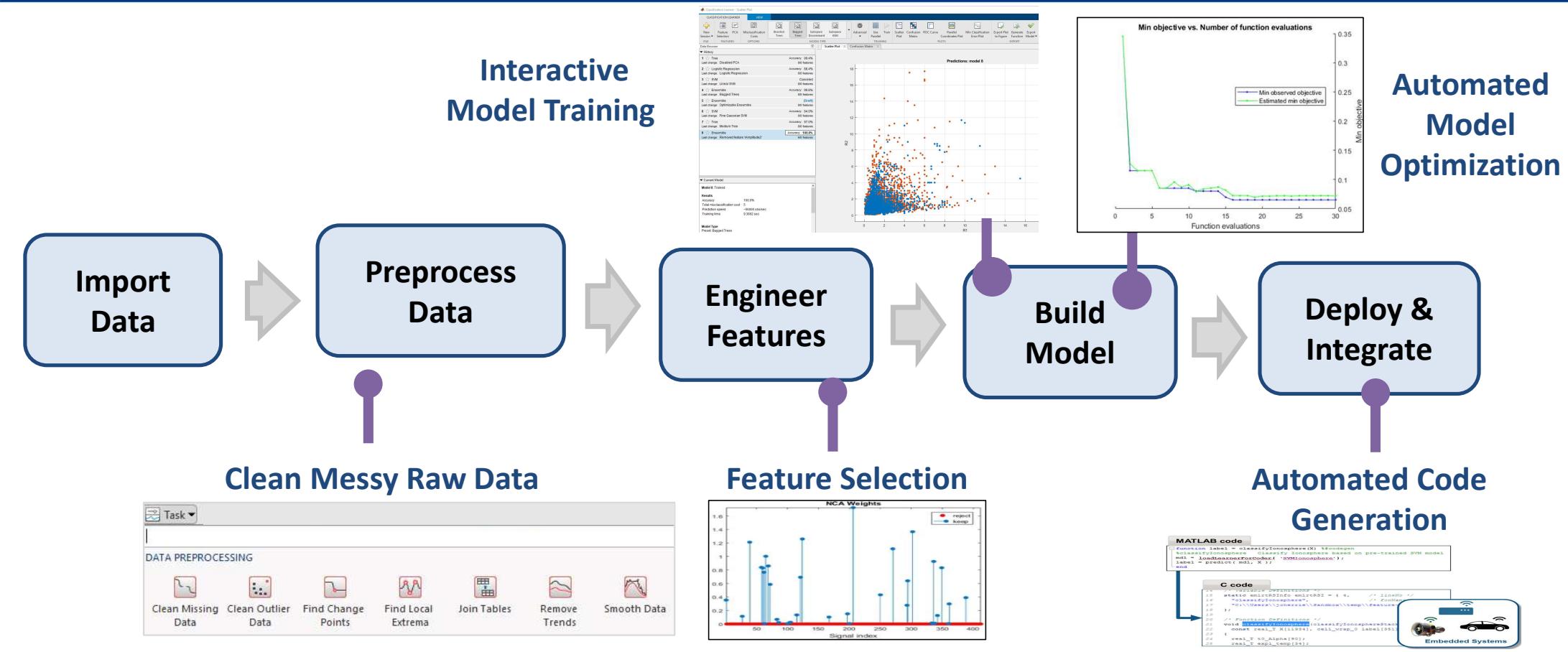
Scatter Plot and Clusters



K-means Clustering Results



Machine Learning Wrap-up



Machine Learning Summary

[Machine Learning Onramp](#) (2 hr online introduction)

[Practical Data Science with MATLAB](#) (4 course Specialization)

Machine Learning with MATLAB:

- [Overview, Cheat sheet](#)
- [Machine Learning Intro](#) (Tech talks)
- [Machine Learning with MATLAB Introduction](#) (eBook)
- [Mastering Machine Learning](#) (eBook)
- [Applied Machine Learning](#) (Tech Talk videos)

Machine and Deep Learning

- [Deep vs. Machine Learning: Choosing the Best Approach](#) (eBook)
- [Deep learning Onramp](#) (2hr online introduction)

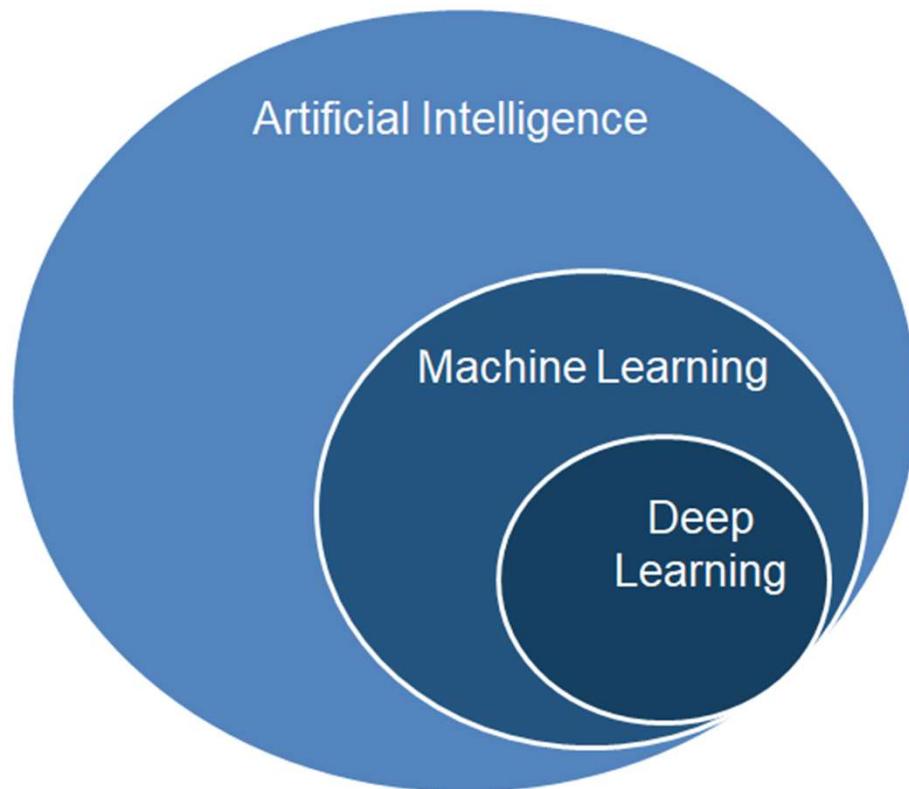
Five Interactive Apps for Machine Learning

No matter what type of problem you're trying to solve, MATLAB® is here to help. Discover apps to interactively model, fit, and label data for machine learning.

[Classification Learner](#) [Regression Learner](#) [Curve Fitting](#) [Image Labeler](#) [Signal Labeler](#)



MATLAB for Artificial Intelligence



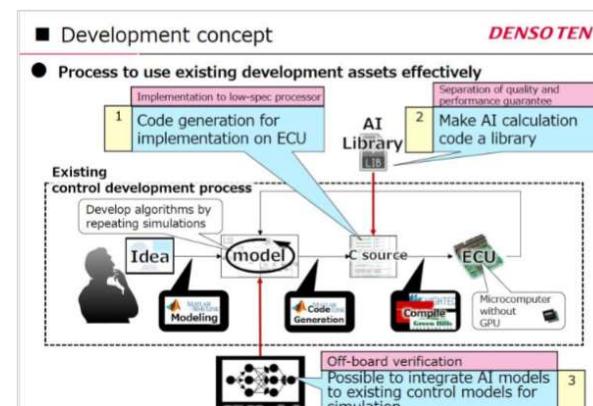
- Machine Learning
- Deep Learning
- Image Processing
- Reinforcement Learning
- Predictive Maintenance
- Data Science / Data Analytics
- Signal Processing
- ...and more



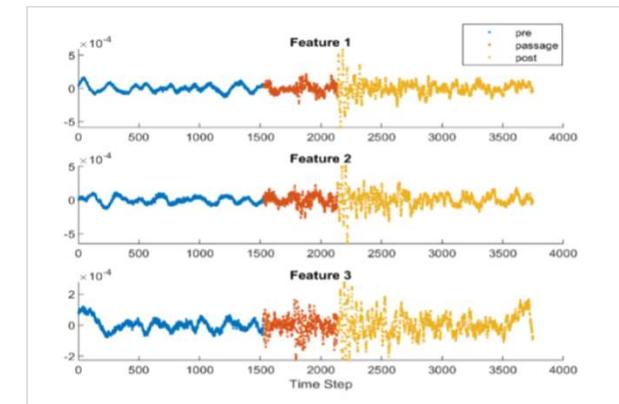
MATLAB Deep Learning used in Industry



Automatic Defect Detection
Airbus



ECU Vehicle Control
Denso



Seismic Event Detection
Shell



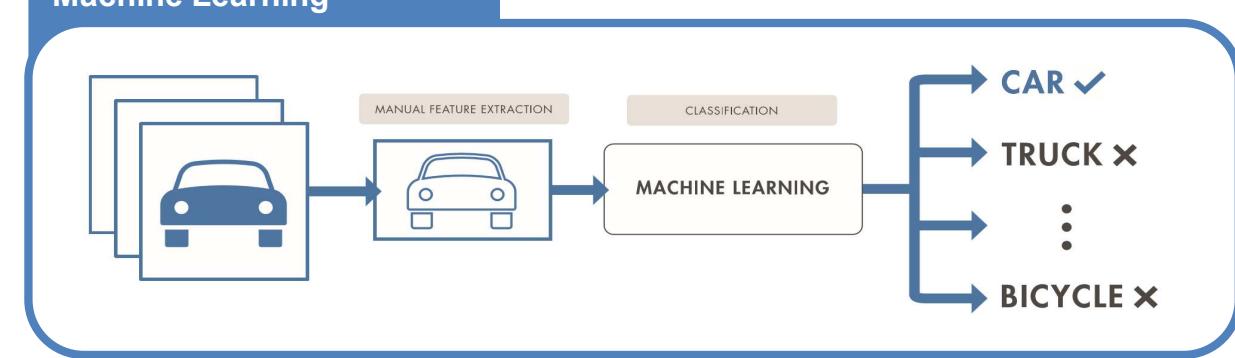
What is Deep Learning?

Machine Learning

Deep Learning

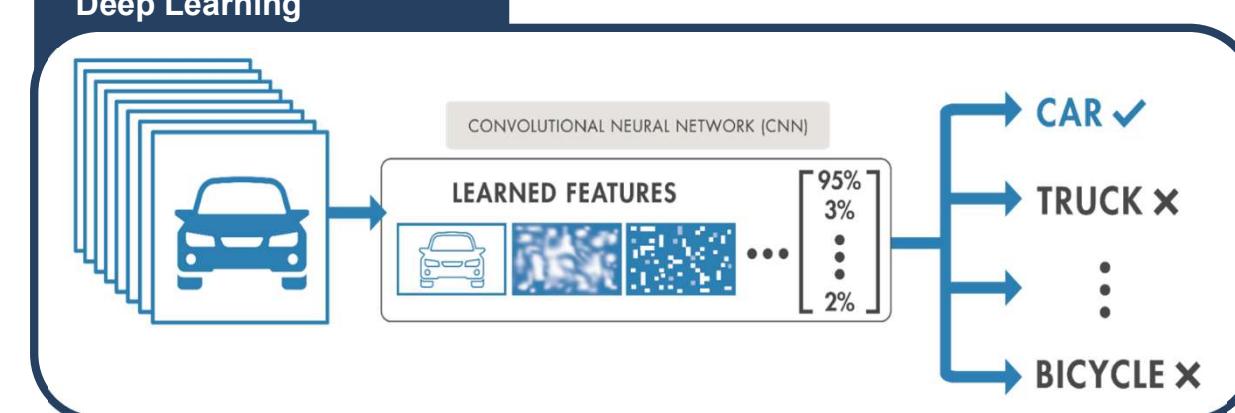
Neural Networks
with many Hidden
Layers

Machine Learning

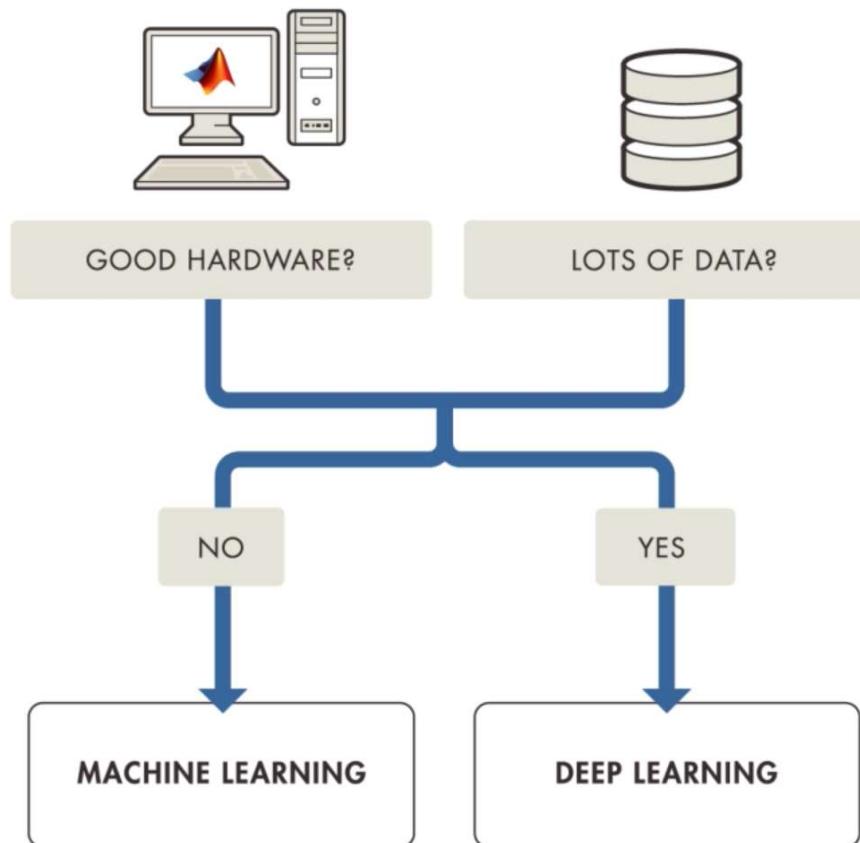


Deep Learning

- Learns directly from data
- More Data = better model
- Computationally Intensive
- **Not interpretable**



Machine Learning vs Deep Learning



	Machine Learning	Deep Learning
Training dataset	Small	Large
Choose your own features	Yes	No
# of classifiers available	Many	Few
Training time	Short	Long



Applications of deep learning for images and video



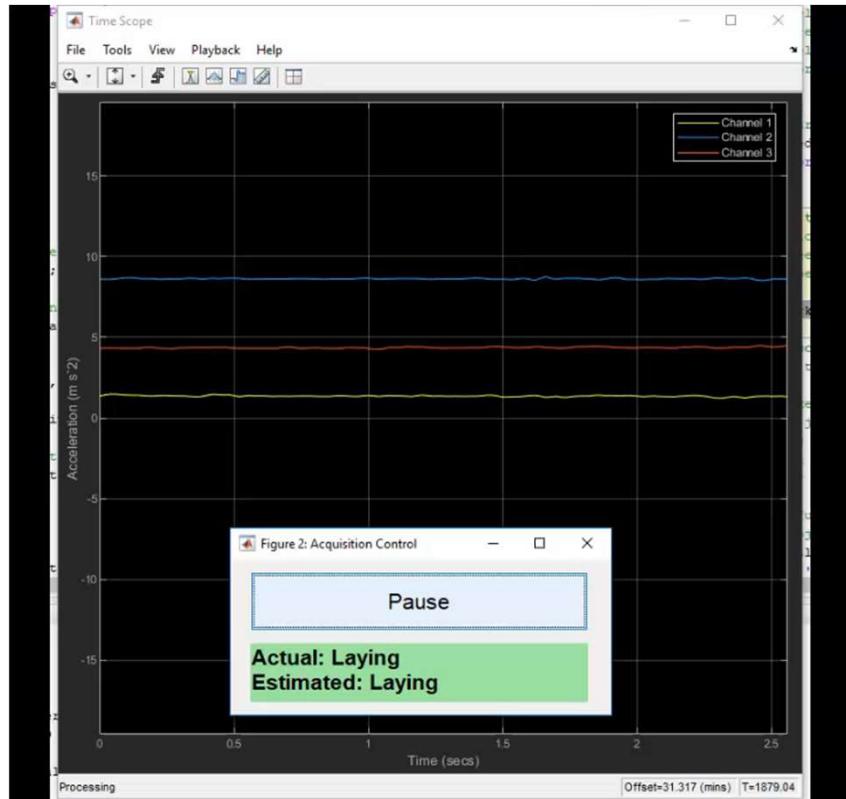
YOLO v2 (You Only Look Once)



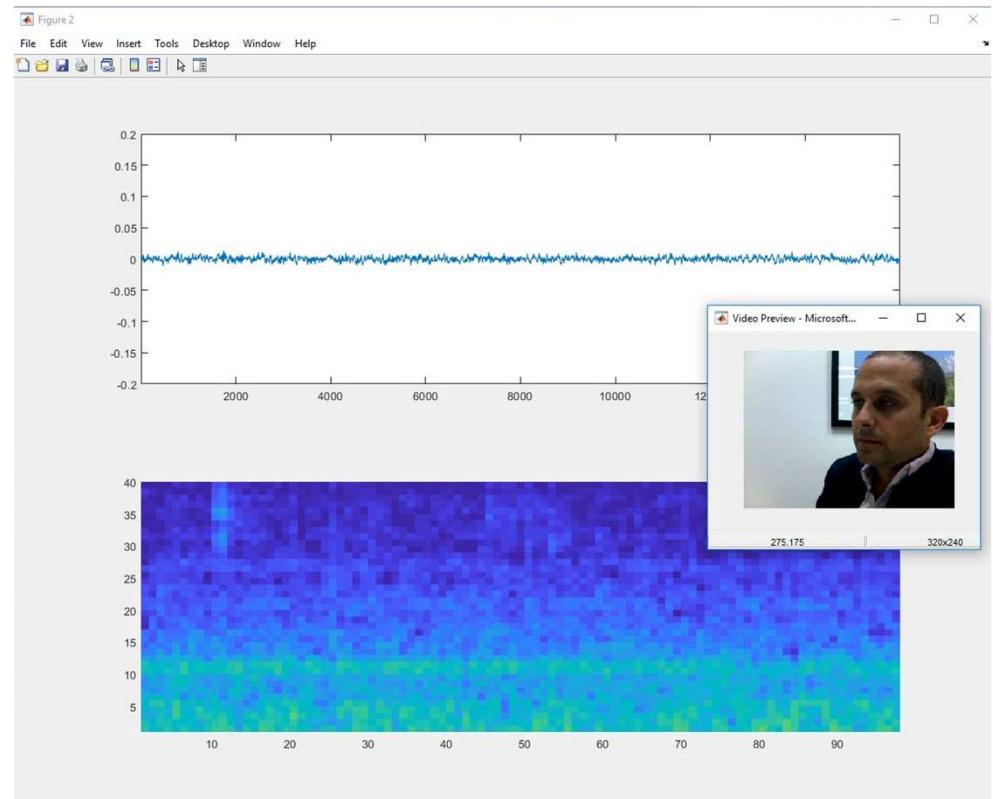
Semantic Segmentation using SegNet



Applications of deep learning for signal processing



Signal Classification using LSTMs

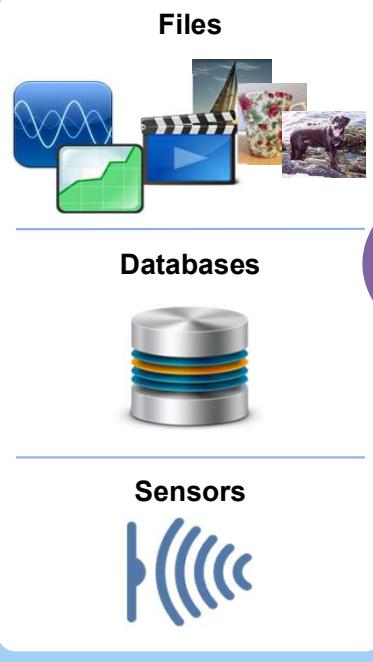


Speech Recognition using CNNs



Deep Learning Workflow

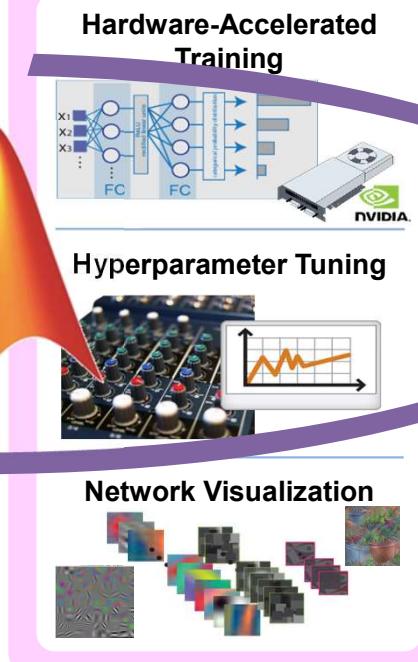
ACCESS AND EXPLORE DATA



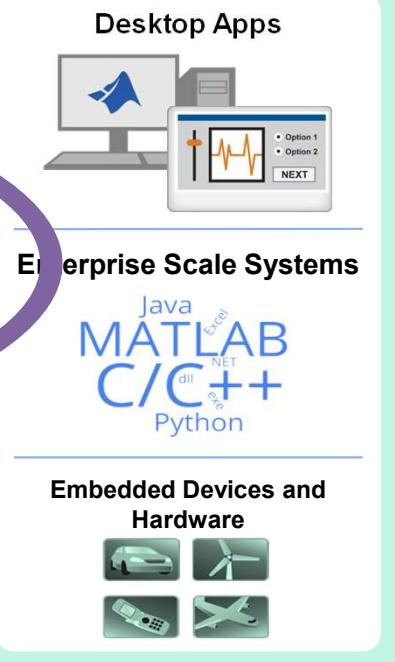
LABEL AND PREPROCESS DATA



DEVELOP PREDICTIVE MODELS



INTEGRATE MODELS WITH SYSTEMS



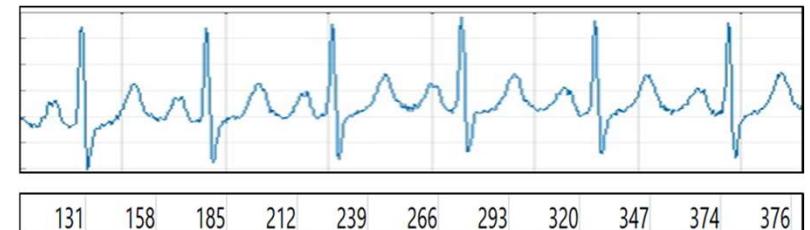
Import Data for Deep Learning Networks

ACCESS AND EXPLORE
DATA



Images are a numeric matrix

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



Signals are numeric vectors

The Bird Flies = [0 13 5 6]

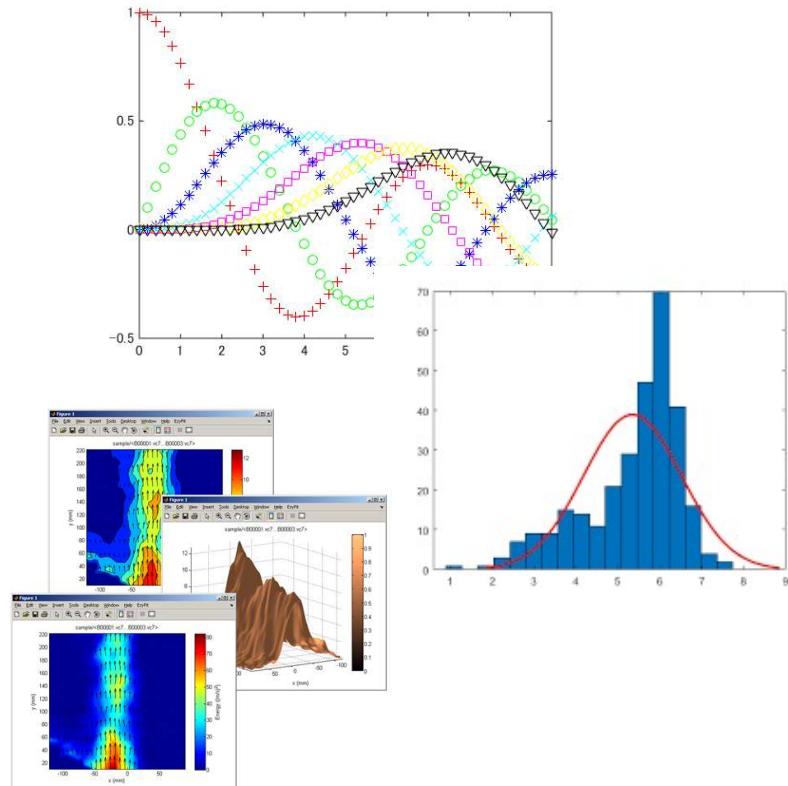
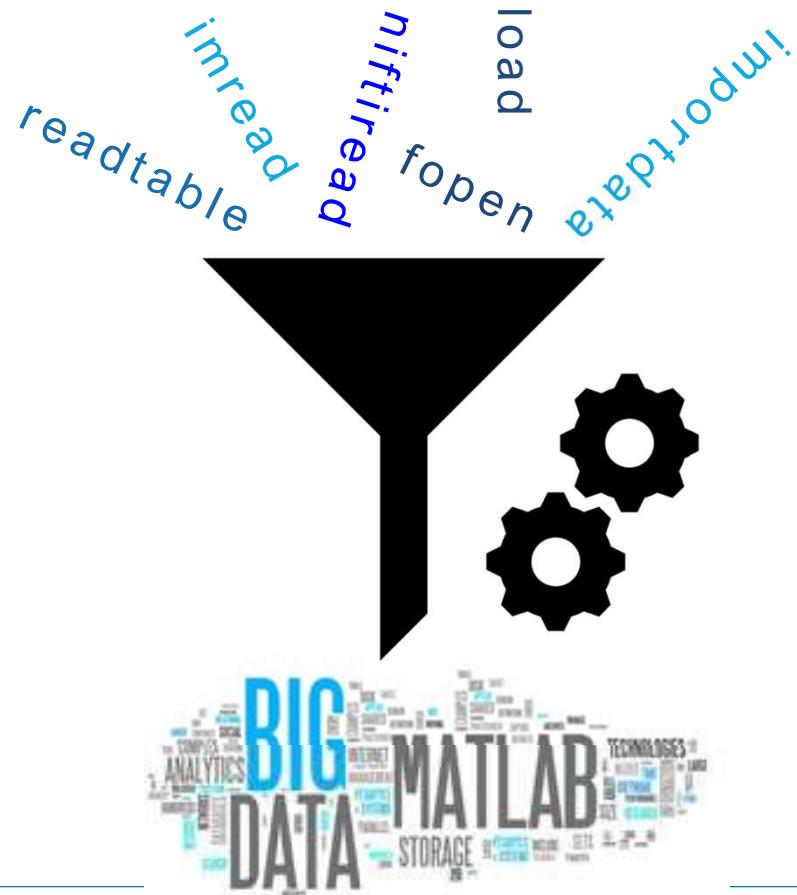
The Leaf Is Brown = [13 3 11 2]

Text is processed as numeric vectors



Deep Learning requires a lot of data

ACCESS AND EXPLORE
DATA



Automated Labeling Apps

LABEL AND PREPROCESS DATA

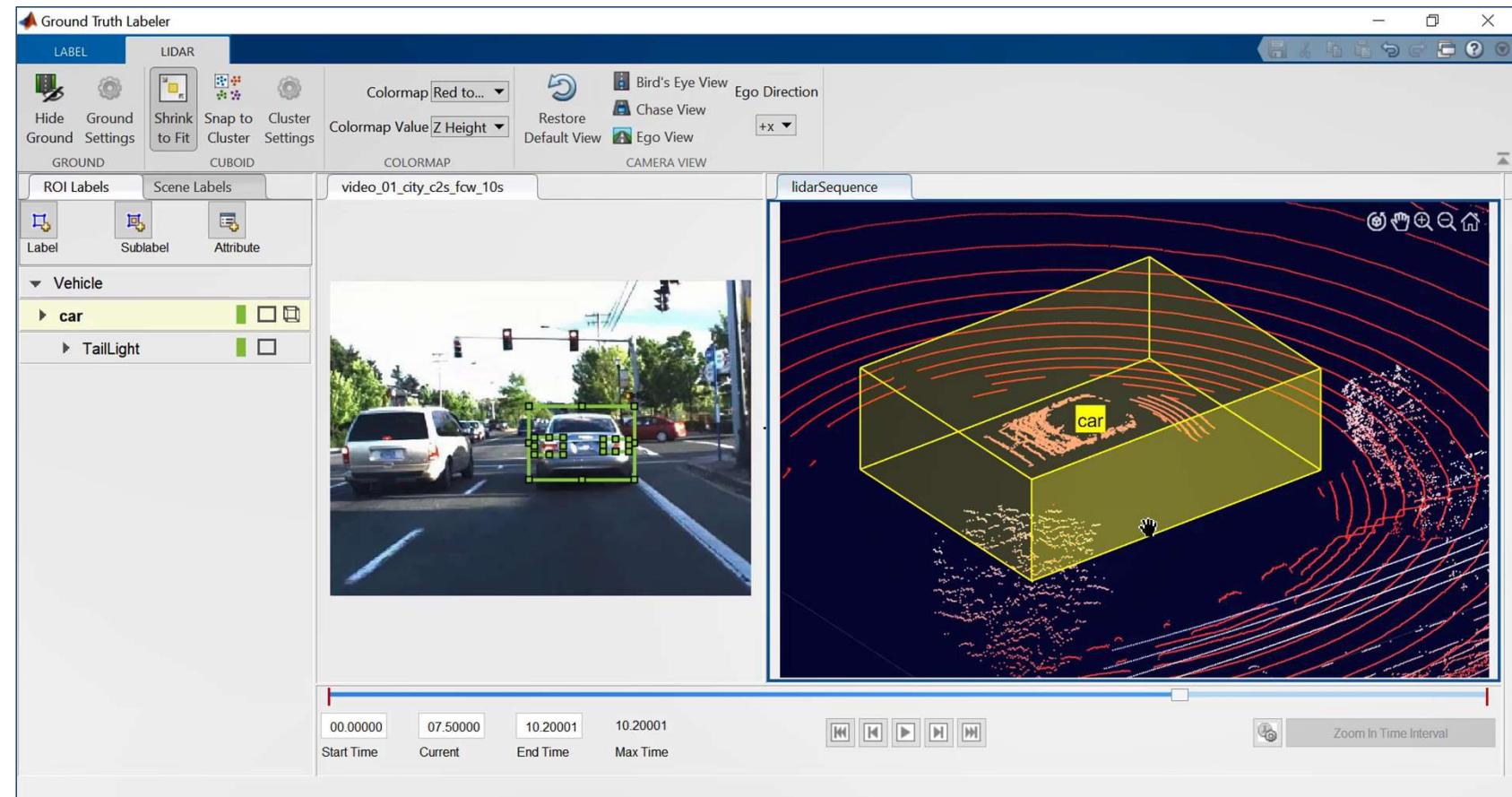
Data Augmentation/
Transformation



Labeling Automation



Import Reference
Models



Synthetic Data Generation

LABEL AND PREPROCESS DATA

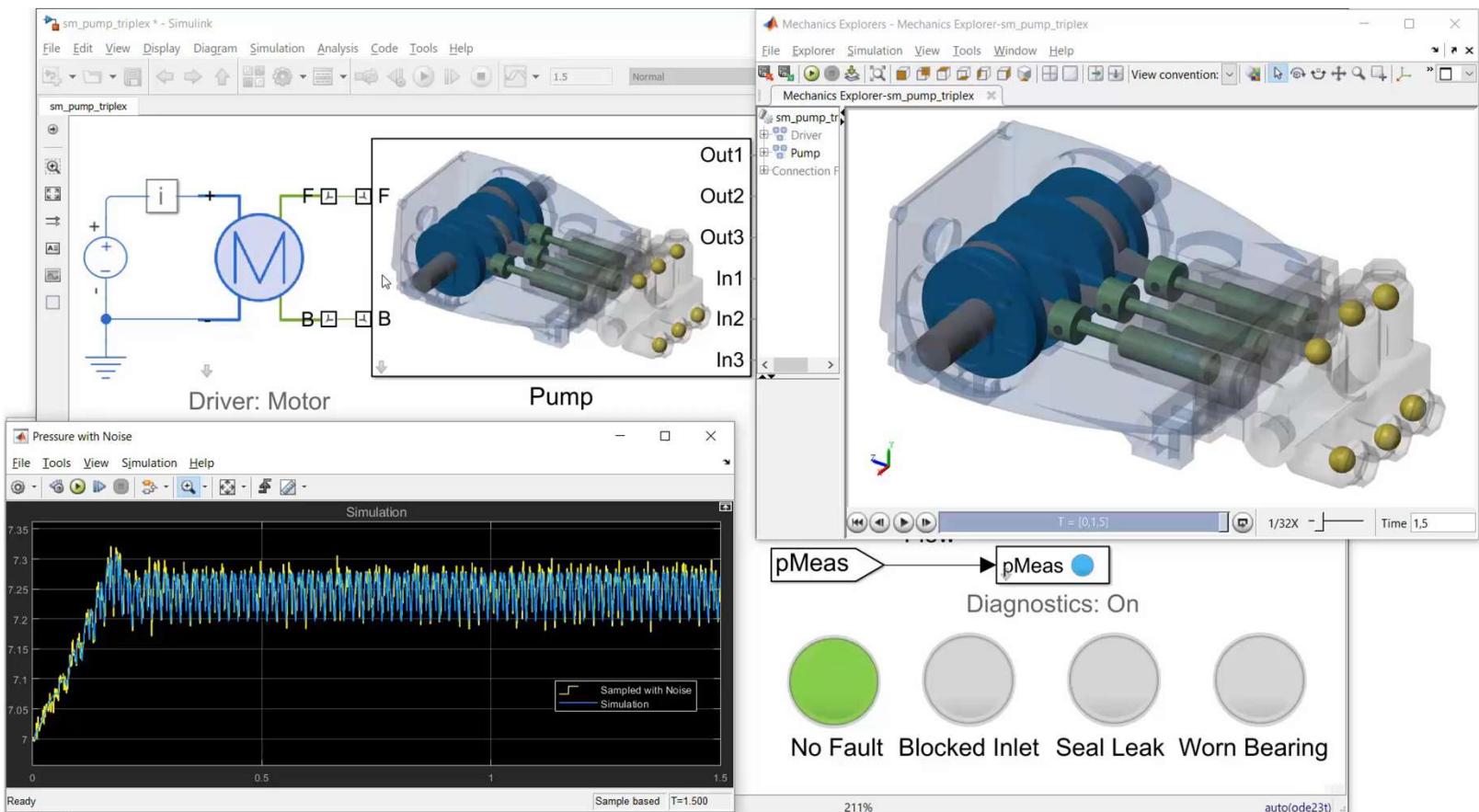
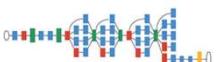
Data Augmentation/
Transformation



Labeling Automation



Import Reference
Models



Automated Labeling Apps Save You Weeks to Months

LABEL AND PREPROCESS DATA

Data Augmentation/ Transformation



Labeling Automation



Import Reference Models

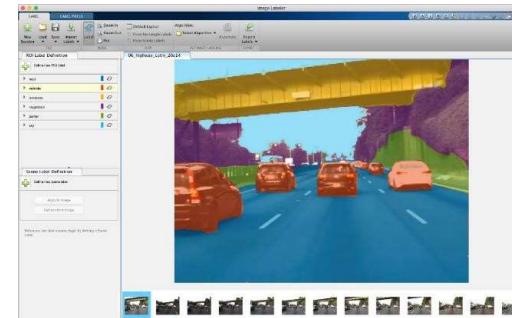
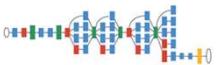
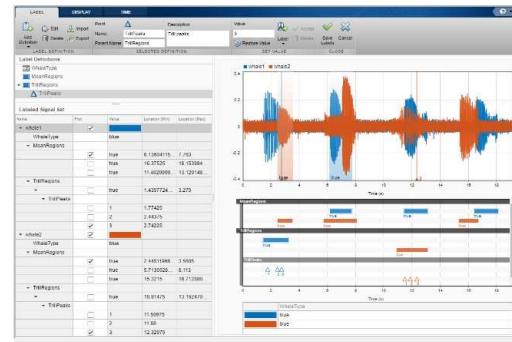


Image Labeler app to label images manually and semi-automatically.



Signal Labeler app to label signals manually and semi-automatically.

AUTOMOTIVE



Ground Truth Labeler

SIGNAL PROCESSING AND COMMUNICATIONS



Audio Labeler



AUTOMOTIVE



Ground Truth Labeler

SIGNAL PROCESSING AND COMMUNICATIONS



Audio Labeler



IMAGE PROCESSING AND COMPUTER VISION



Image Labeler



Lidar Labeler



Video Labeler



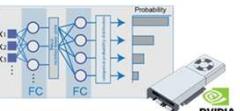
Diagnostic Feature Designer to extract time and frequency domain features from signals.



There are 2 training approaches for Deep Learning with Images

DEVELOP PREDICTIVE MODELS

Hardware-Accelerated Training



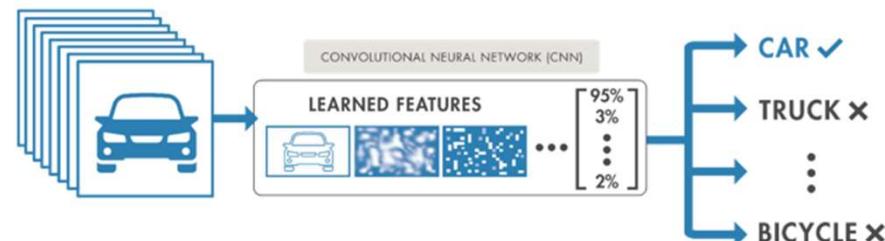
Hyperparameter Tuning



Network Visualization



Train a deep neural network from scratch



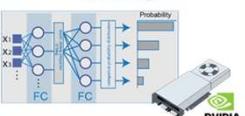
Use a pretrained model



Designing a network from scratch requires some knowledge

DEVELOP PREDICTIVE MODELS

Hardware-Accelerated Training



Hyperparameter Tuning



Network Visualization



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
- (...)

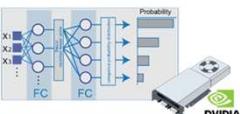
Research papers and [doc examples](#) can provide guidelines for creating architecture.



Start with pre-built models and existing examples

DEVELOP PREDICTIVE MODELS

Hardware-Accelerated Training



Hyperparameter Tuning



Network Visualization



Use pre-built models

Image classification models

AlexNet, GoogLeNet, VGG, SqueezeNet, ShuffleNet, ResNet, DenseNet, Inception...

Reference examples

Object detection

Vehicles, pedestrians, faces...

Semantic segmentation

Roadway detection, land cover classification, tumor detection...

Signal and speech processing

Denoising, music genre recognition, keyword spotting, radar waveform classification...

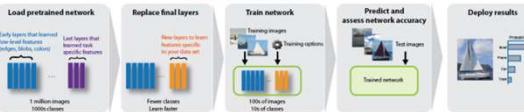
...and more...

Transfer Learning

Modify and reuse pre-built models

[Get started with Transfer Learning](#)

Reuse Pretrained Network



AlexNet

VGG-16

VGG-19

GoogLeNet

ResNet-18

ResNet-101

ResNet-50

Inception-v3

DenseNet-201

Xception

[Get started with these Models](#)

Effective for object detection and semantic segmentation workflows

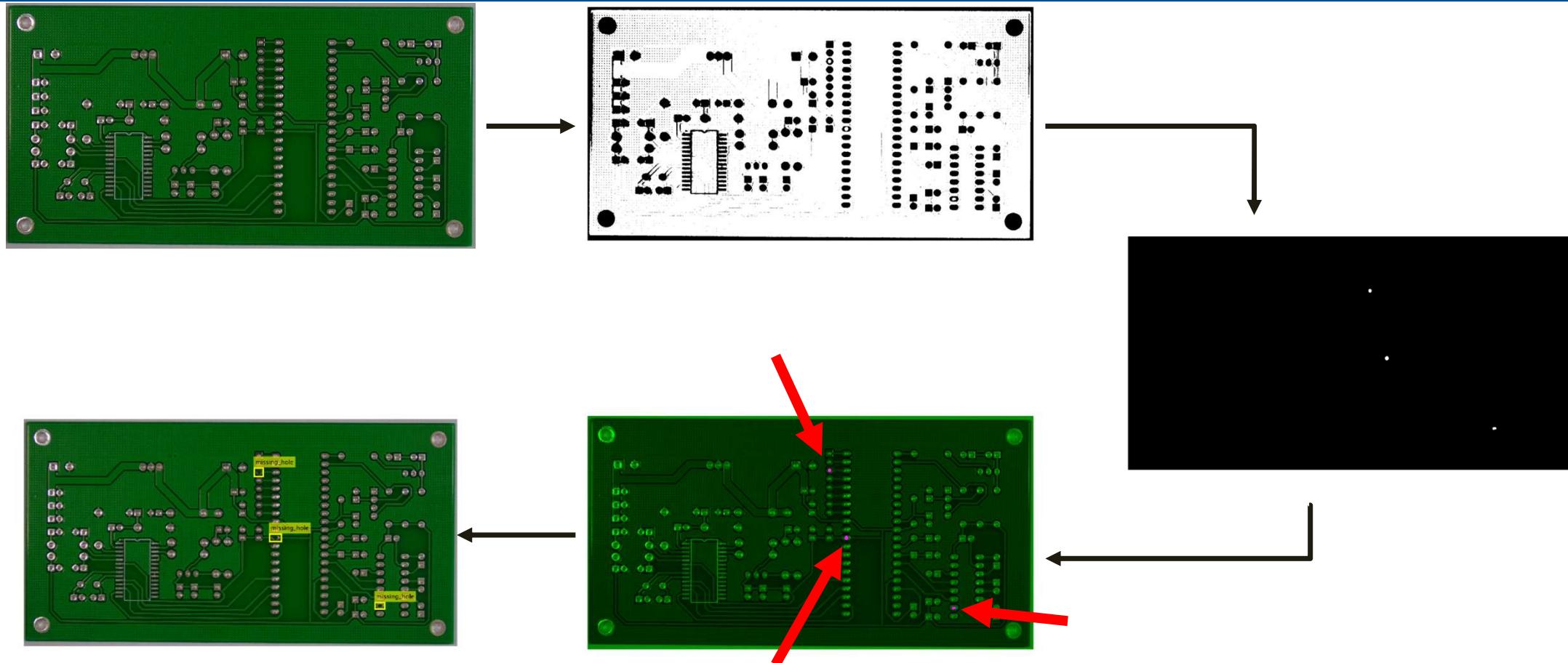


SqueezeNet
MobileNet-v2
ShuffleNet

Lightweight and computationally efficient



Data Preparation Demo



Data Preparation Demo – Pre-recorded Video

Live Editor - C:\Users\kvemulap\Documents\Large\GitLab_Internal\NP_CV_Core\pcb-defection-detection-using-yolov\Part01_DataPreparation.mlx

PCB Defect Detection

Part 1: Data Preparation

Copyright © 2024 MathWorks, Inc.

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

This example shows how to detect, localize, and classify defects in printed circuit boards (PCBs) using a YOLOv object detector. PCBs contain individual electronic devices and their connections. Defects in PCBs can result in poor performance or product failures. By detecting defects in PCBs, production lines can remove faulty PCBs and ensure that electronic devices are of high quality.

Configuration

Make sure we run this as a project.

```
1 try
2     prj = currentProject;
3     catch
4         open("PCBDefectDetection.prj");
5         OpenPart1;
6         prj = currentProject;
7     end
```

Download PCB Defect Data Set

This example uses the PCB defect data set [1] [2]. The data set contains 1,386 images of PCB elements with synthesized defects. The data has six types of defect: missing hole, mouse bite, open circuit, short, spur, and spurious copper. Each image contains multiple defects of the same category in different locations. The data set contains bounding box and coordinate information for every defect in every image. The size of the data set is 1.87 GB.

Missing Hole Short Mouse Bite Spur

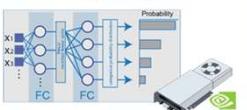
Zoom: 100% UTF-8 LF script



AI modeling Apps automate training, tuning, visualization...

DEVELOP PREDICTIVE MODELS

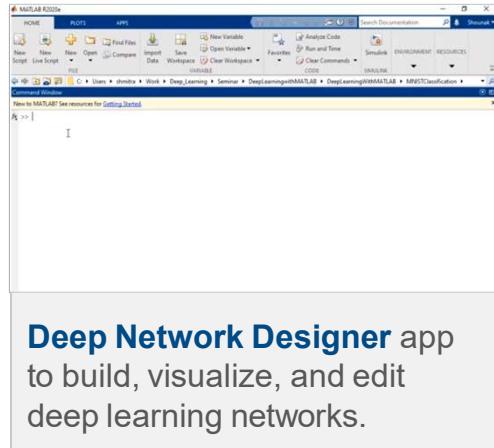
Hardware-Accelerated Training



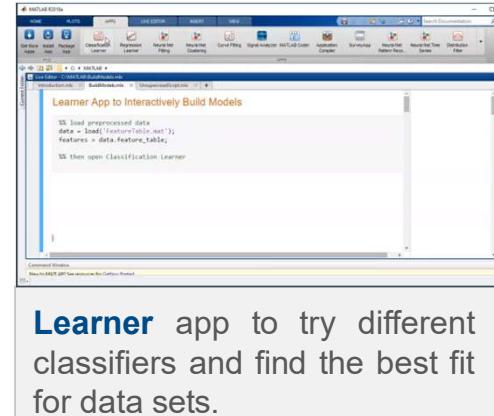
Hyperparameter Tuning



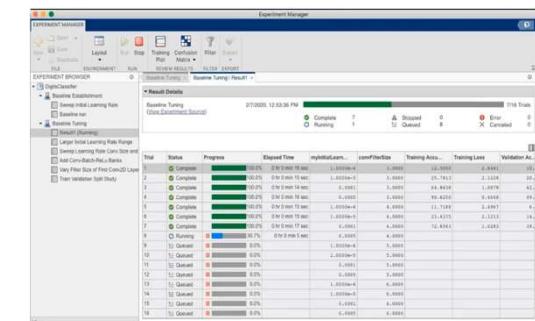
Network Visualization



Deep Network Designer app to build, visualize, and edit deep learning networks.



Learner app to try different classifiers and find the best fit for data sets.



Experiment Manager app to run deep learning experiments to train networks and compare results.

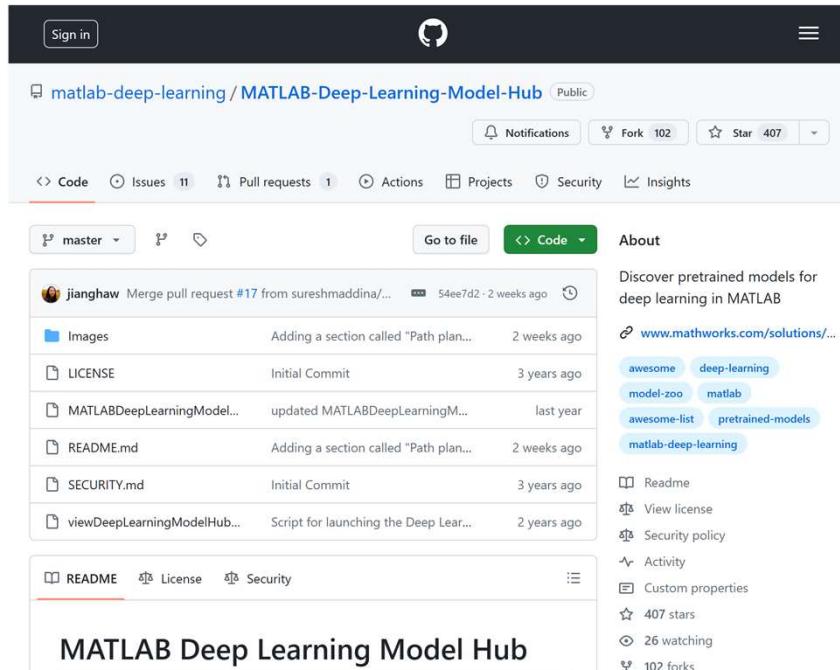


Access pretrained models

Take advantage of the knowledge provided by pretrained networks to learn new patterns in new data

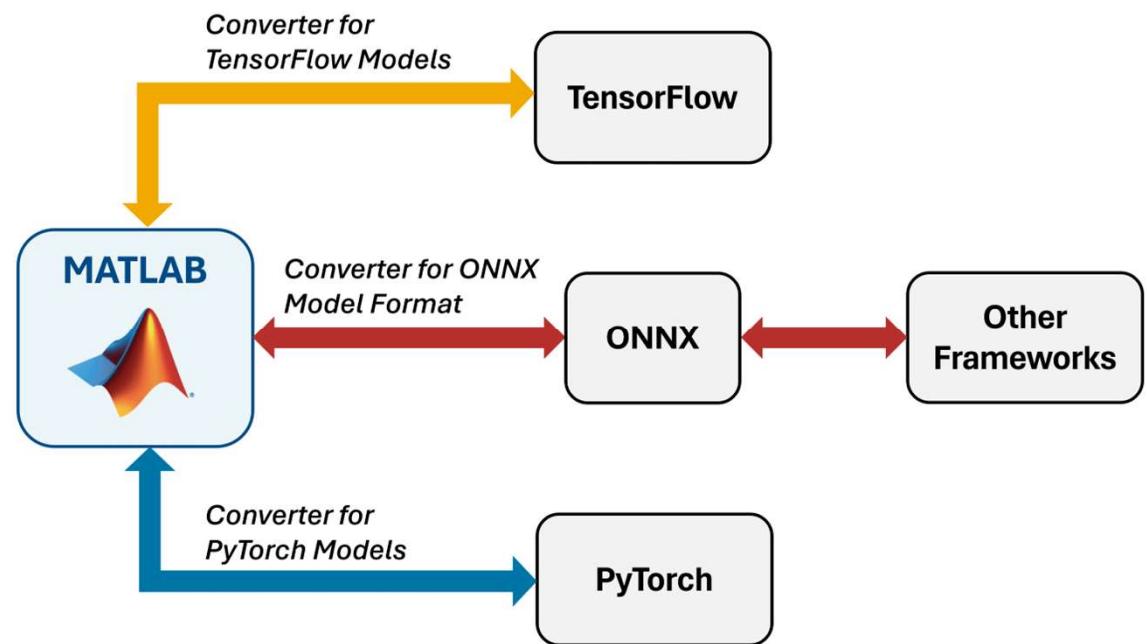
Find one directly in MATLAB

<https://github.com/matlab-deep-learning/MATLAB-Deep-Learning-Model-Hub>



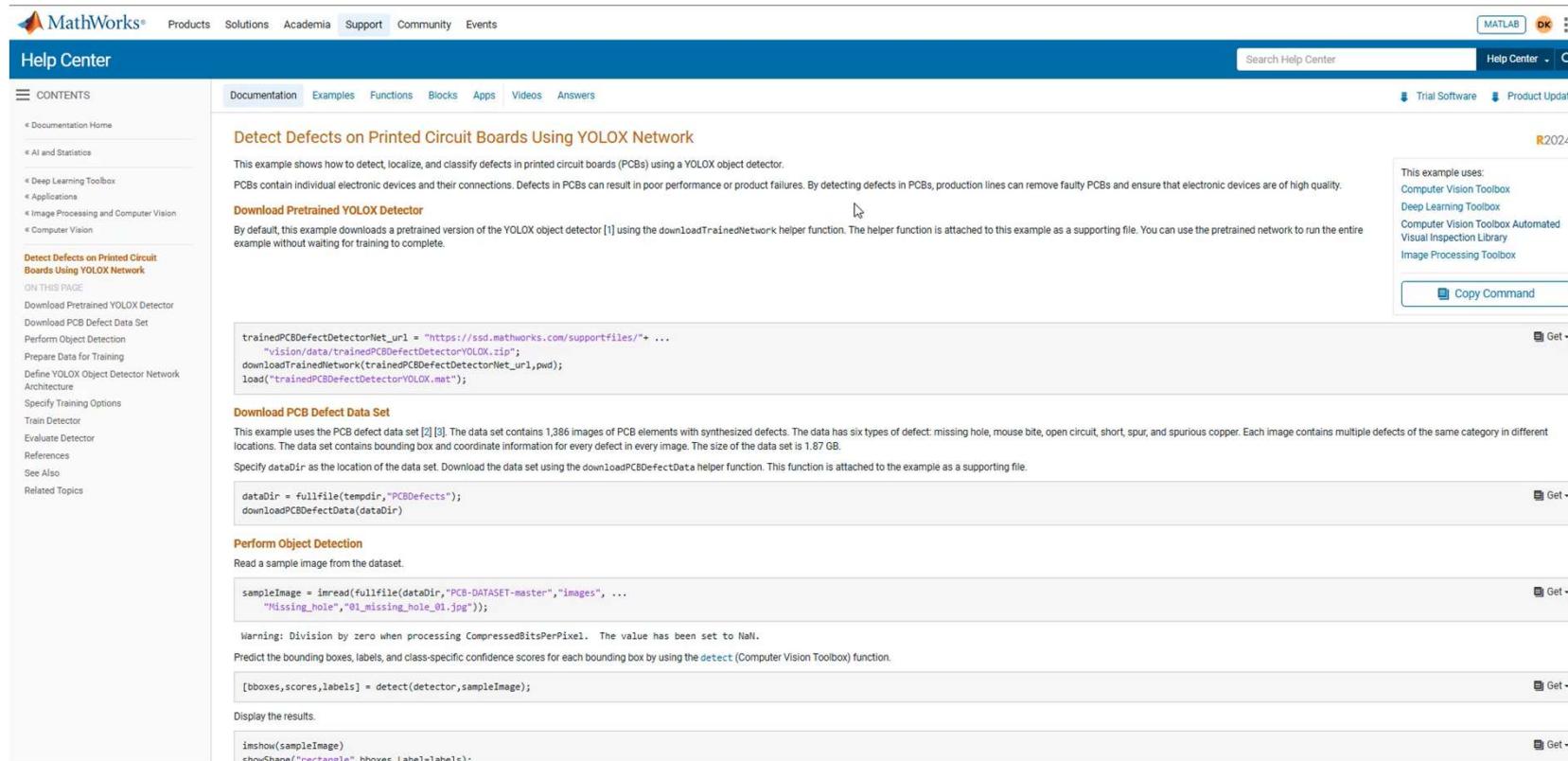
The screenshot shows the GitHub repository page for 'matlab-deep-learning / MATLAB-Deep-Learning-Model-Hub'. The repository is public, has 102 forks, and 407 stars. It contains 11 issues and 1 pull request. The repository was created 3 years ago. The README file is the most recent commit, updated 2 weeks ago. The repository is used for discovering pretrained models for deep learning in MATLAB. It includes links to the MATLAB Model Zoo and the MATLAB Deep Learning Model Hub. The repository has 407 stars and 102 forks.

Import it from other platforms



Modeling Demo

Detect Defects on Printed Circuit Boards Using YOLOX Network



This example shows how to detect, localize, and classify defects in printed circuit boards (PCBs) using a YOLOX object detector. PCBs contain individual electronic devices and their connections. Defects in PCBs can result in poor performance or product failures. By detecting defects in PCBs, production lines can remove faulty PCBs and ensure that electronic devices are of high quality.

Download Pretrained YOLOX Detector

By default, this example downloads a pretrained version of the YOLOX object detector [1] using the `downloadTrainedNetwork` helper function. The helper function is attached to this example as a supporting file. You can use the pretrained network to run the entire example without waiting for training to complete.

```
trainedPCBDefectDetectorNet_url = "https://sd.mathworks.com/supportfiles/+ ...  
"vision/data/trainedPCBDefectDetectorYOLOX.zip";  
downloadTrainedNetwork(trainedPCBDefectDetectorNet_url,pwd);  
load("trainedPCBDefectDetectorYOLOX.mat");
```

Download PCB Defect Data Set

This example uses the PCB defect data set [2] [3]. The data set contains 1,386 images of PCB elements with synthesized defects. The data has six types of defect: missing hole, mouse bite, open circuit, short, spur, and spurious copper. Each image contains multiple defects of the same category in different locations. The data set contains bounding box and coordinate information for every defect in every image. The size of the data set is 1.87 GB.

Specify `dataDir` as the location of the data set. Download the data set using the `downloadPCBDefectData` helper function. This function is attached to the example as a supporting file.

```
dataDir = fullfile(tempdir,"PCBDefects");  
downloadPCBDefectData(dataDir)
```

Perform Object Detection

Read a sample image from the dataset.

```
sampleImage = imread(fullfile(dataDir,"PCB-DATASET-master","images", ...  
"Missing_hole","01_missing_hole_01.jpg"));
```

Warning: Division by zero when processing `CompressedBitsPerPixel`. The value has been set to `NaN`.

Predict the bounding boxes, labels, and class-specific confidence scores for each bounding box by using the `detect` (Computer Vision Toolbox) function.

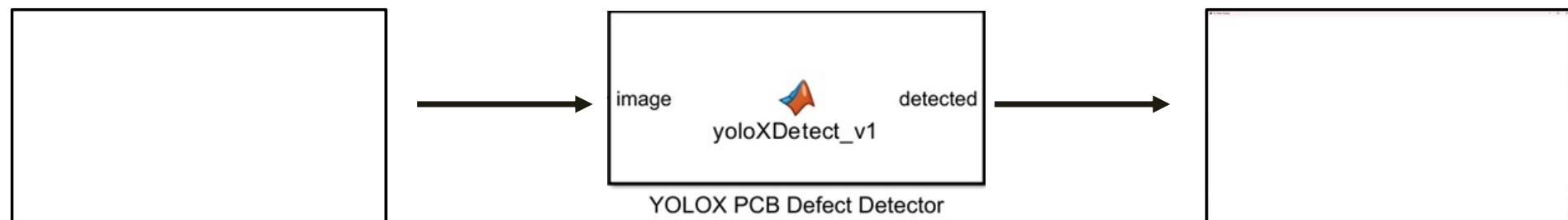
```
[bboxes,scores,labels] = detect(detector,sampleImage);
```

Display the results.

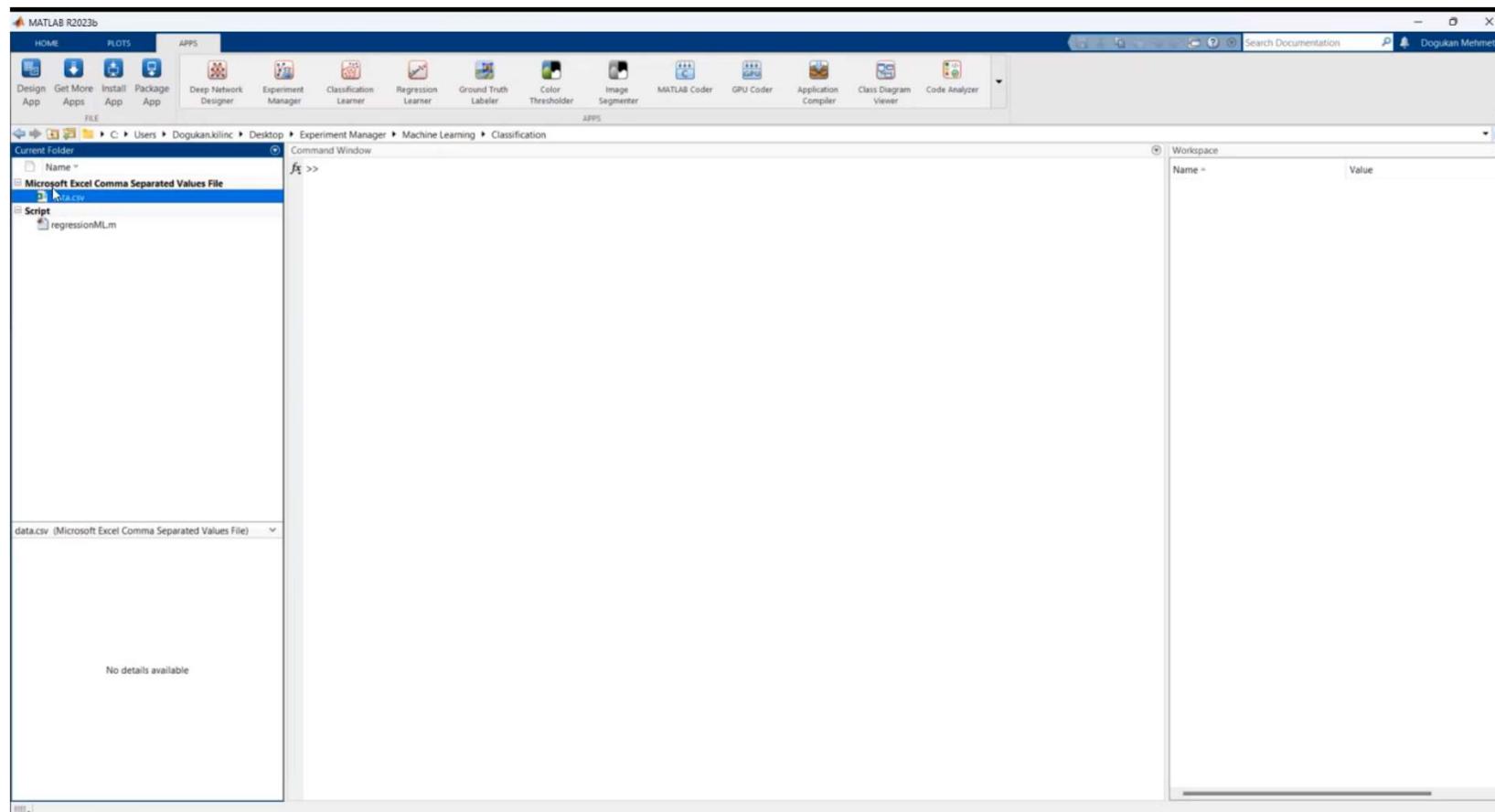
```
imshow(sampleImage)  
showShape("rectangle",bboxes.Label==labels);
```



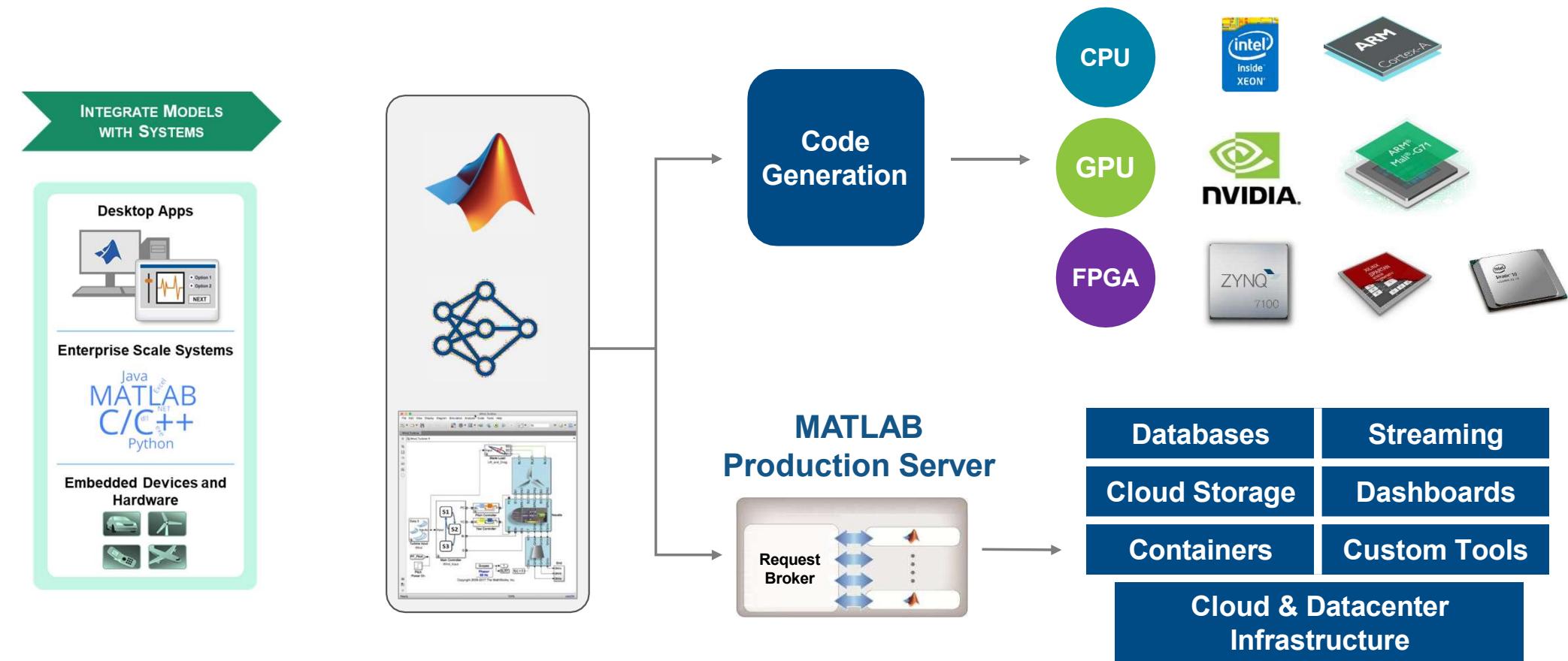
Simulation & Test Demo



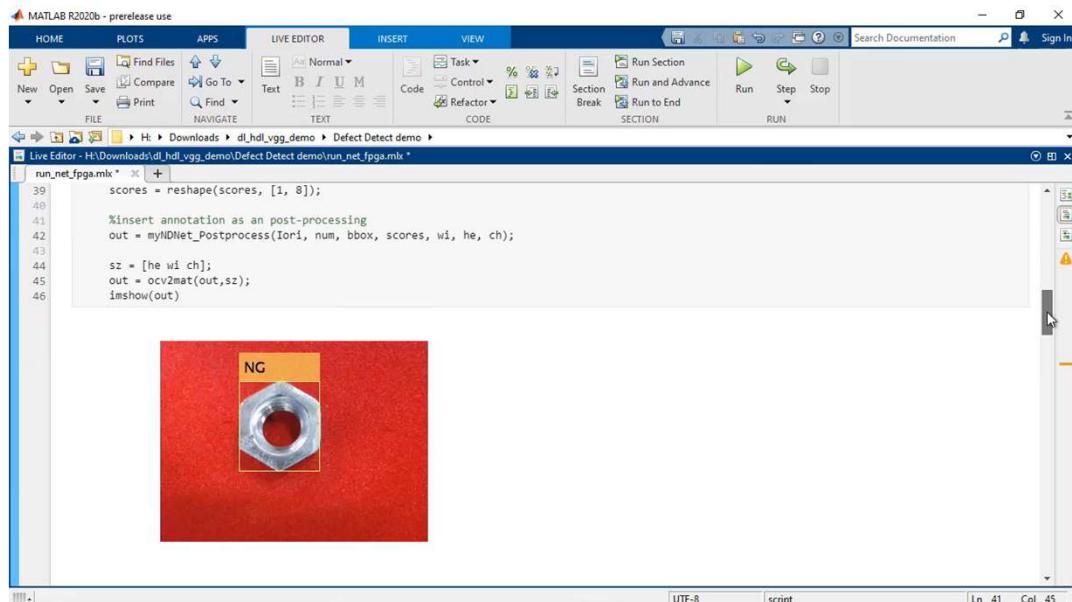
MATLAB Experiment Manager after Deep Learning



Deploy to Any Processor with Best-in-class Performance



Example: Deployment Algorithm to Microprocessor



Deploy defect detection algorithms from MATLAB
to ZCU102 board from Xilinx

Deploy defect detection algorithms from MATLAB
to Jetson AGX Xavier



Deep Learning Summary

Deep Learning Code Generation – Examples

Code Generation for Deep Learning Networks

Perform code generation for an image classification application that uses deep learning. It uses the codegen command to generate a MEX.

Code Generation for Semantic Segmentation Network

Code generation for an image segmentation application that uses deep learning. It uses the codegen command to generate a MEX.

Lane Detection Optimized with GPU Coder

Develop a deep learning lane detection application that runs on NVIDIA® GPUs.

Quantize Residual Network Trained for Image Classification and Generate GPU Code

Quantize the learnable parameters in the convolution layers of a deep learning neural network that has residual connections and has been trained for image classification.

Parameter Pruning and Quantization of Image Classification Network

Use parameter pruning and quantization to reduce network size.

Quantization and Pruning

Parameter Pruning and Quantization of Image Classification Network

Prune Image Classification Network Using Taylor Scores

Prune Filters in a Deep Network Using Taylor Scores

Code Generation For Object Detection Using YOLO v3 Deep Learning

Generate CUDA® MEX for a you only look once (YOLO) v3 object detector. YOLO v3 improves upon YOLO v2 by adding detection at multiple scales.

Code Generation for Deep Learning Networks

Perform code generation for an image classification application that uses deep learning. It uses the codegen command to generate a MEX.

Code Generation for a Sequence-to-Sequence LSTM Network

Demonstrates how to generate CUDA® code for a long short-term memory (LSTM) network. The example generates a MEX.

Deep Learning Prediction on ARM Mali GPU

Use the cncodegen function to generate code for an image classification application that uses deep learning on ARM® Mali GPUs.

Deploy Signal Classifier on NVIDIA Jetson Using Wavelet Analysis and Deep...

Generate and deploy a CUDA® executable that classifies human electrocardiogram (ECG) signals using features extracted by the Wavelet Analysis and Deep...

Code Generation for Object Detection by Using YOLO v2

Generate CUDA® MEX for a you only look once (YOLO) v2 object detector. A YOLO v2 object detection network is composed of two subnetworks. A

Code Generation for LSTM Network on Raspberry Pi

Generate code for a pretrained long short-term memory network to predict Remaining Useful Life (RUL) of a machine.

Cross Compile Deep Learning Code for ARM Neon Targets

Generate library or executable code on host computer for deployment on ARM hardware target.

Deep Learning Code Generation from MATLAB Applications

CPU Code Generation from MATLAB Applications

Code Generation for Deep Learning on ARM Targets

Generate and deploy code for prediction on an ARM®-based device without using a hardware support package.

Deep Learning Prediction with ARM Compute Using codegen

Use codegen to generate code for a logo classification application that uses deep learning on ARM® processors. The logo classification

Deep Learning Code Generation for Intel Targets for Different Batch Sizes

Use the codegen command to generate code for an image classification application that uses deep learning on Intel® processors. The generated code uses the Intel Math Kernel

Generate C++ Code for Object Detection Using YOLO v2 and Intel MKL-DNN

Generate C++ code for the YOLO v2 object detection network on an Intel® processor. The generated code uses the Intel Math Kernel

Deploy Signal Classifier Using Wavelets and Deep Learning on Raspberry Pi

The workflow to classify human electrocardiogram (ECG) signals using the Continuous Wavelet Transform (CWT) and a deep

Classification with ResNet_50

Classification with ResNet_50

Classification with MobileNetV2

Classification with MobileNetV2

Classification with InceptionV3

Classification with InceptionV3

Classification with InceptionV3

Classification with InceptionV3

Classification with InceptionV3

Classification with InceptionV3

Deep Learning Prediction with GPU Coder

Generate CUDA® MEX for a lane detection application that runs on NVIDIA® GPUs.

Deep Learning Prediction with NVIDIA TensorRT Library

Generate CUDA® MEX code for a traffic sign detection and recognition application that uses deep learning. It uses the pretrained network called LogoNet and

Deep Learning Prediction on GPU

Generate CUDA® MEX from MATLAB® code and denoise grayscale images by using the denoising convolutional neural

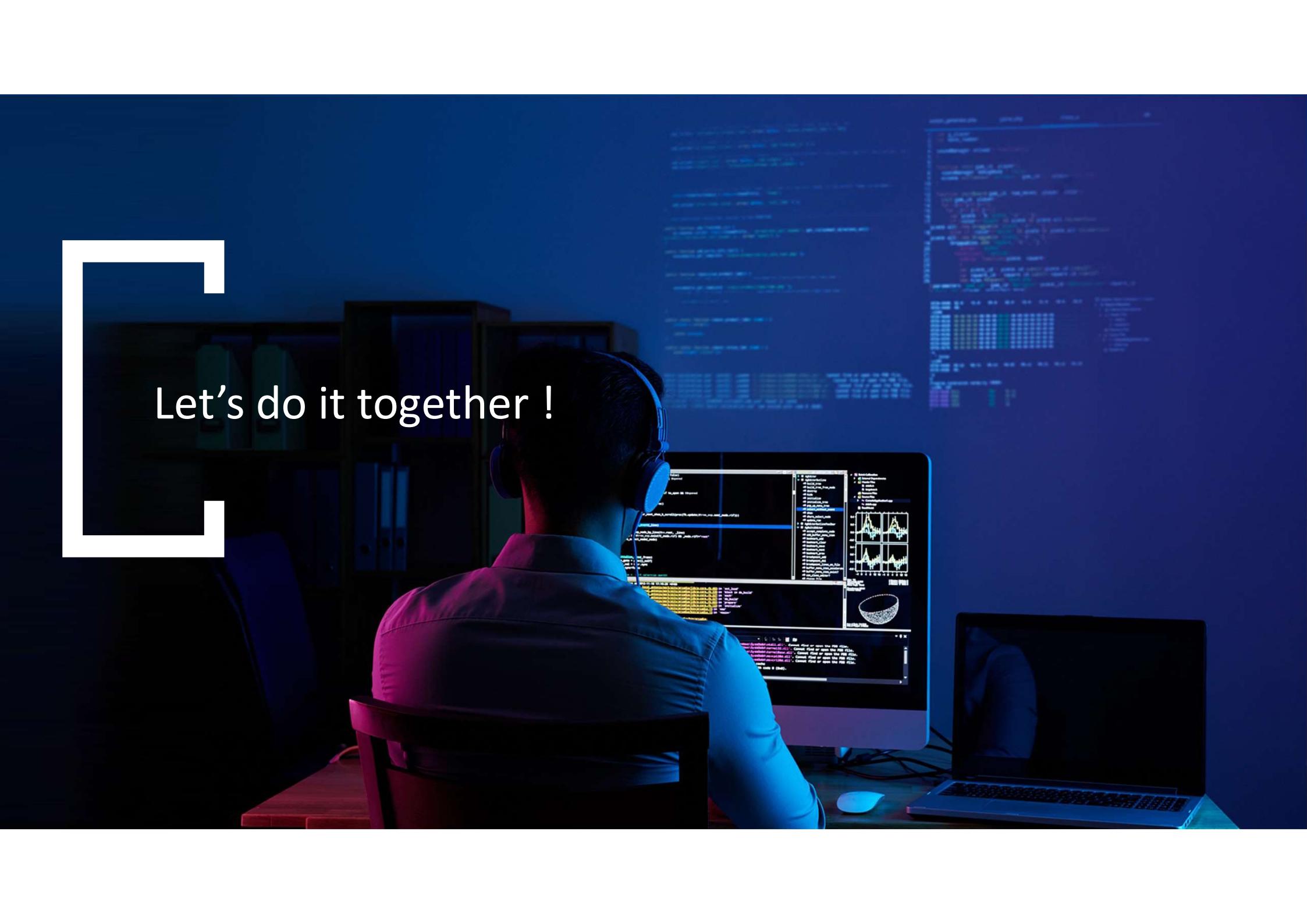
Deep Learning Prediction with GPU Coder

Generate CUDA® MEX from MATLAB® code and denoise grayscale images by using the denoising convolutional neural

Deep Learning Prediction with GPU Coder

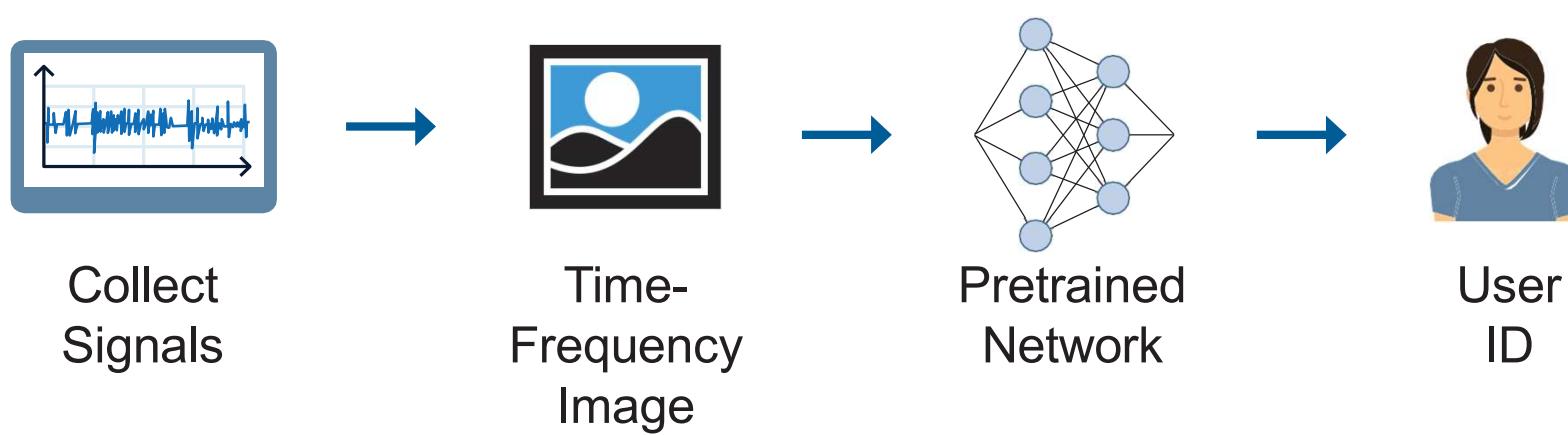
Generate CUDA® MEX from MATLAB® code and denoise grayscale images by using the denoising convolutional neural



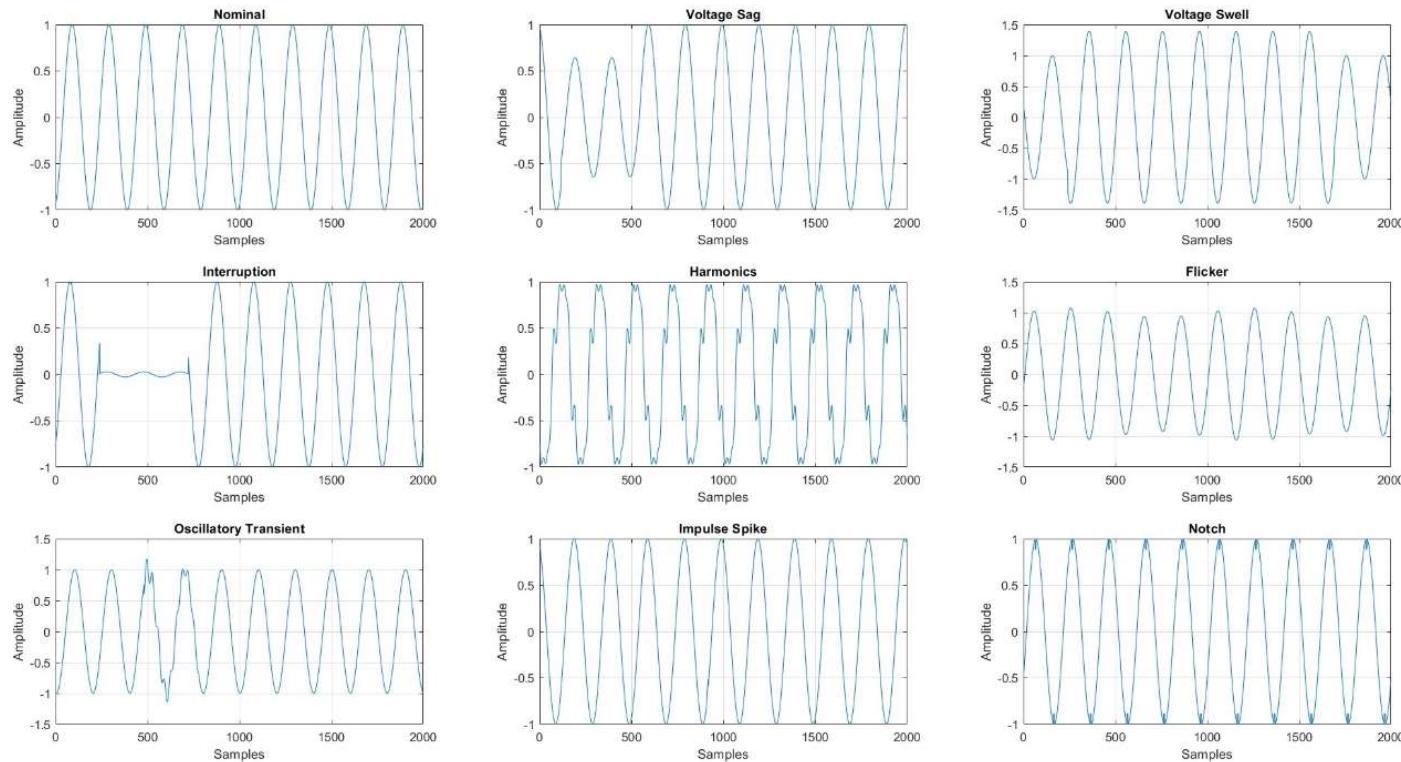


Let's do it together !

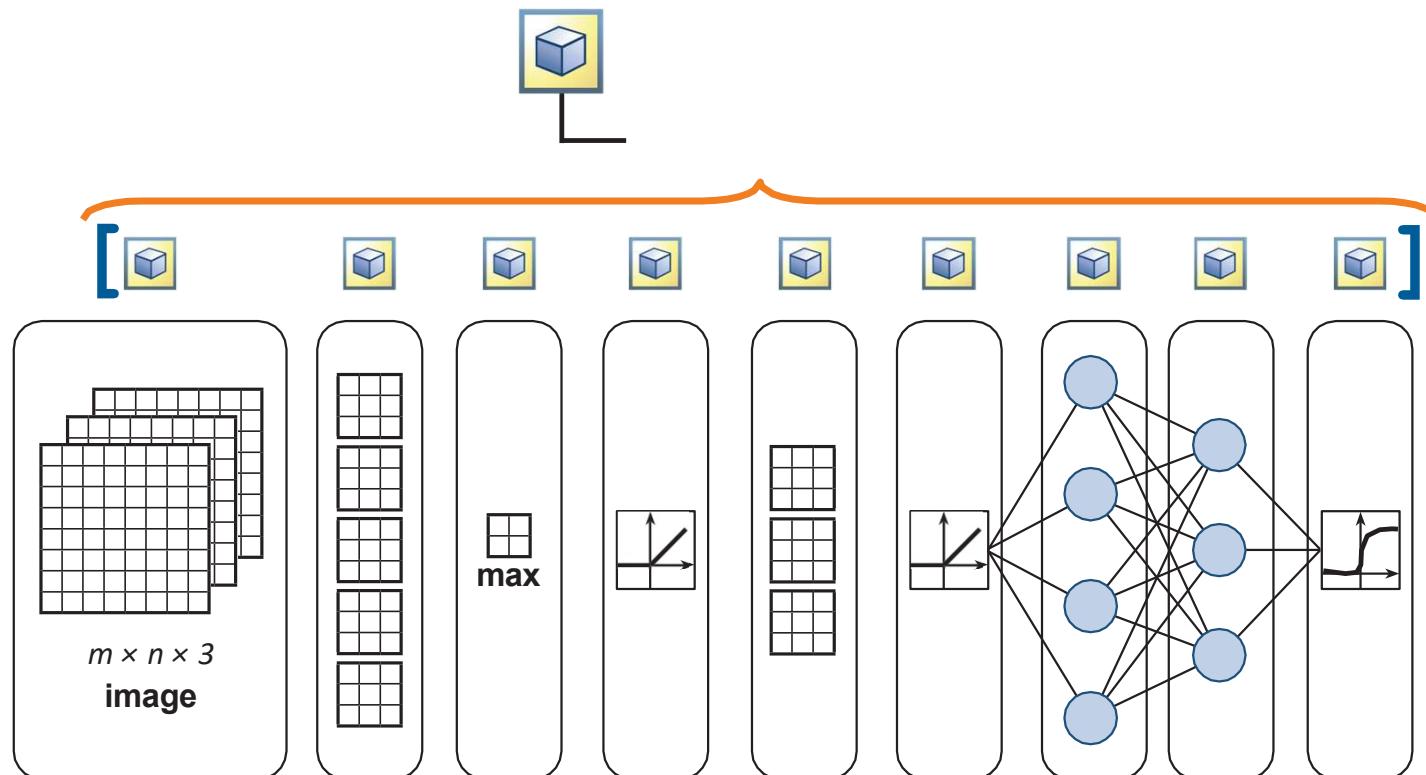
Signal Acquisition



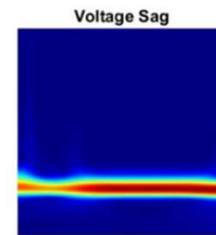
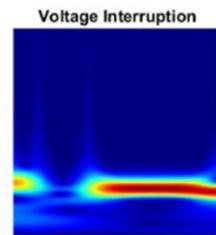
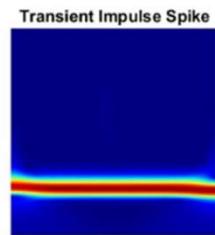
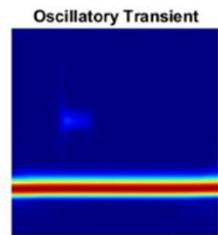
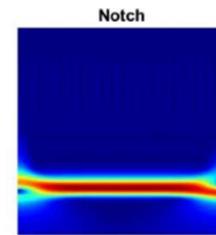
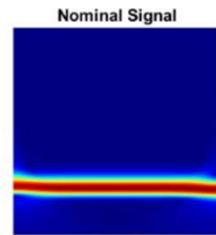
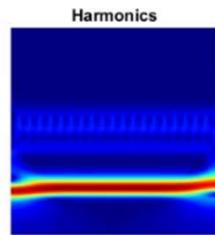
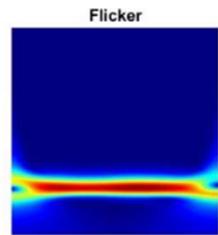
Preprocessing Signals for Classification



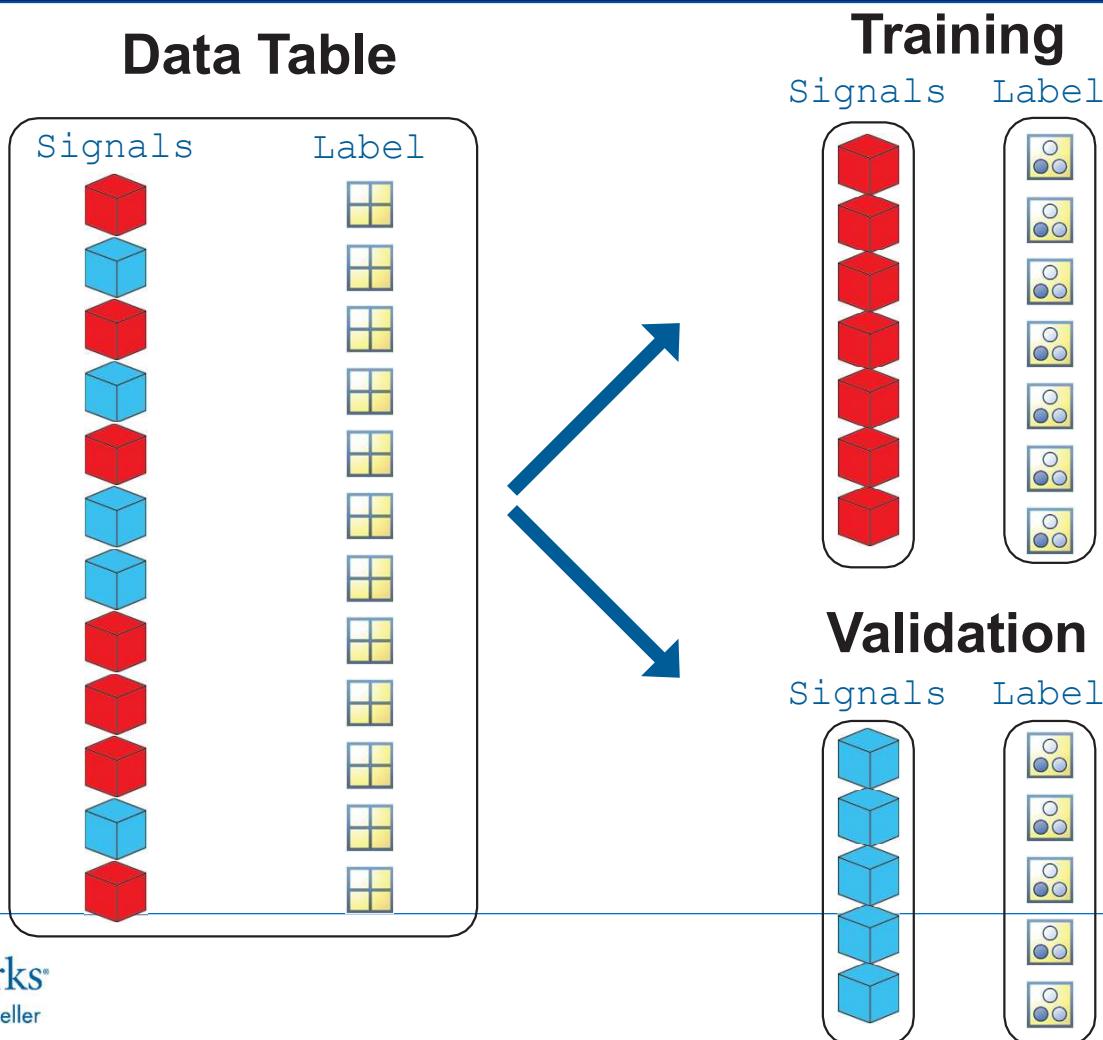
Pretrained Convolutional Neural Networks



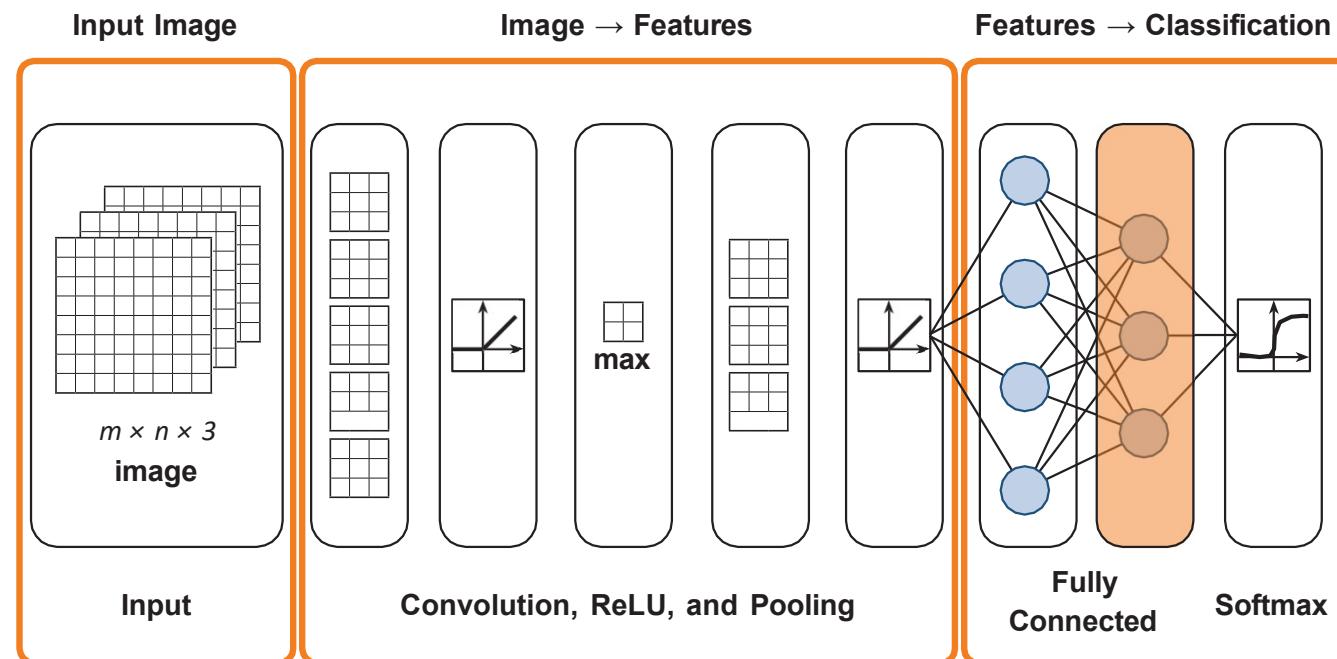
Converting Signals to Images



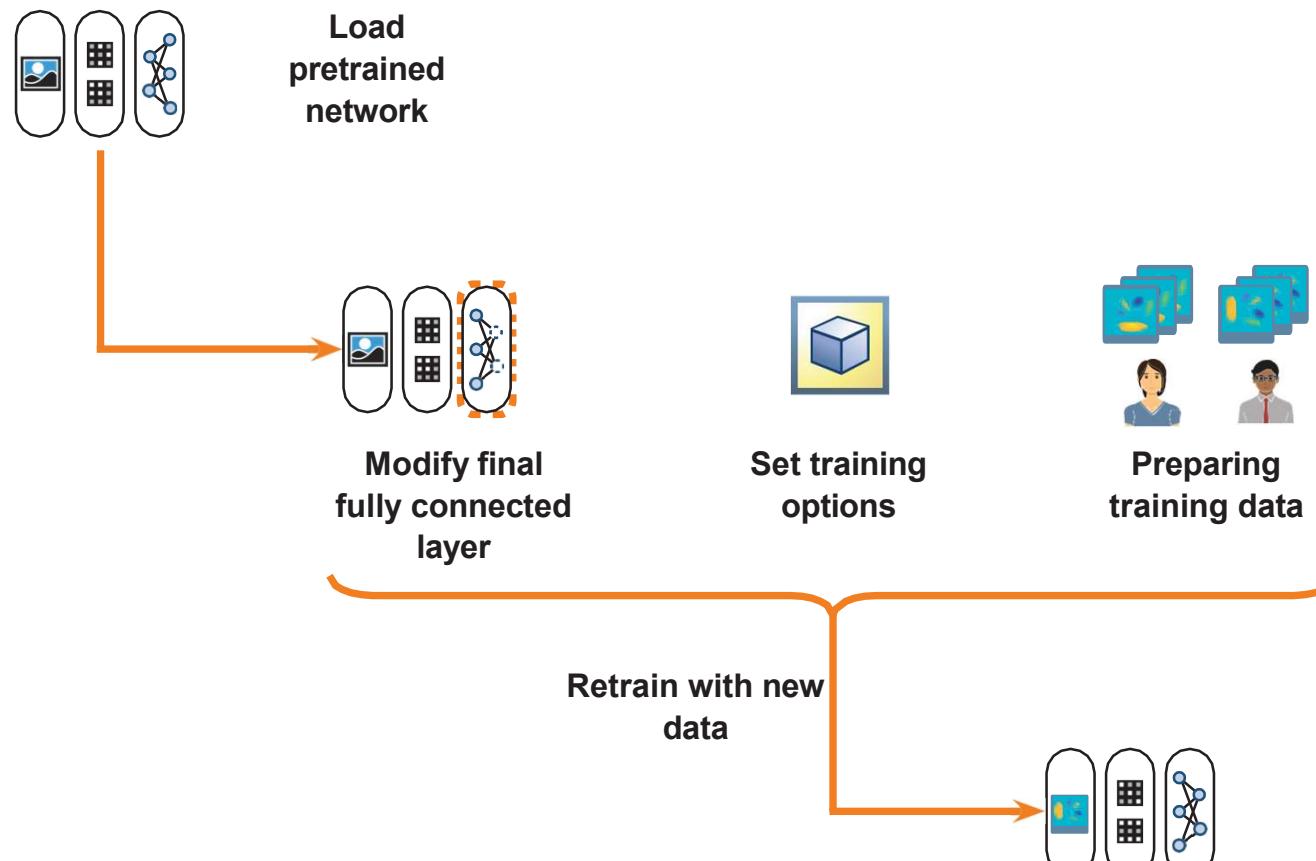
Extracting Training and Validation Data



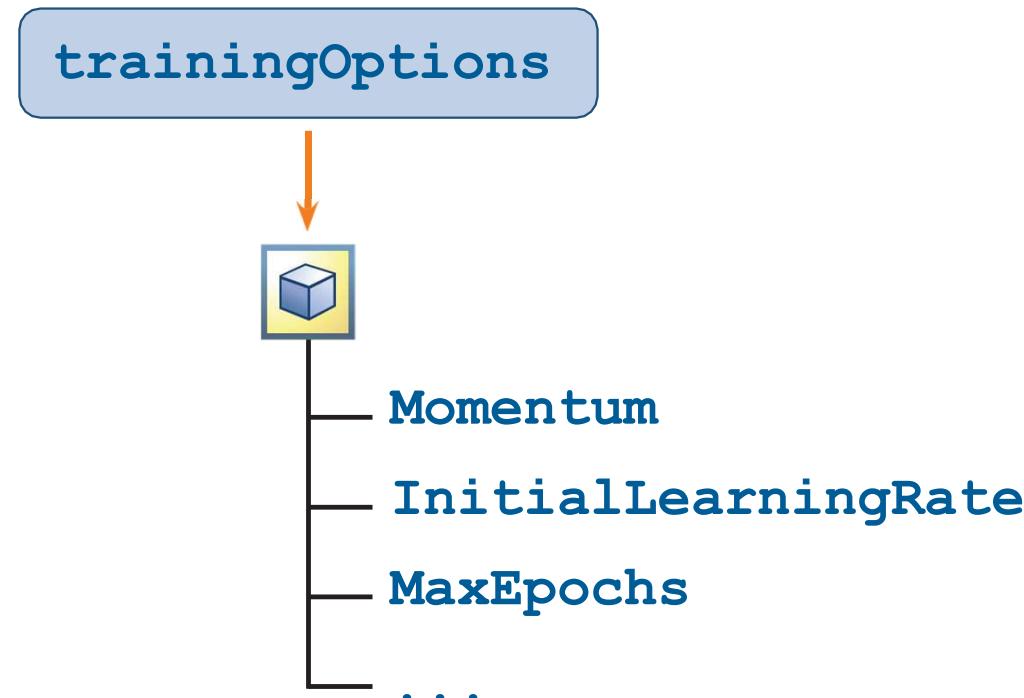
Modifying a Pretrained Network



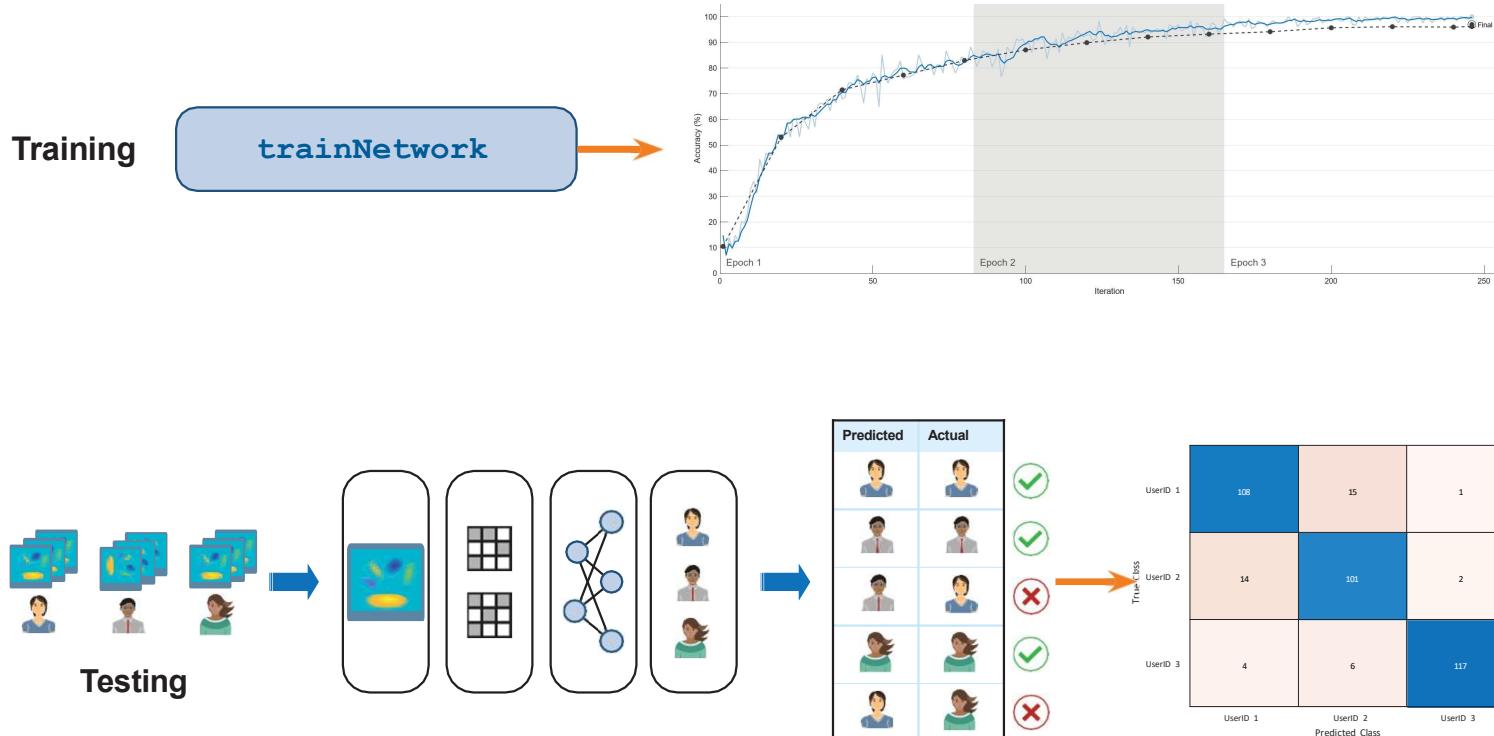
Transfer Learning



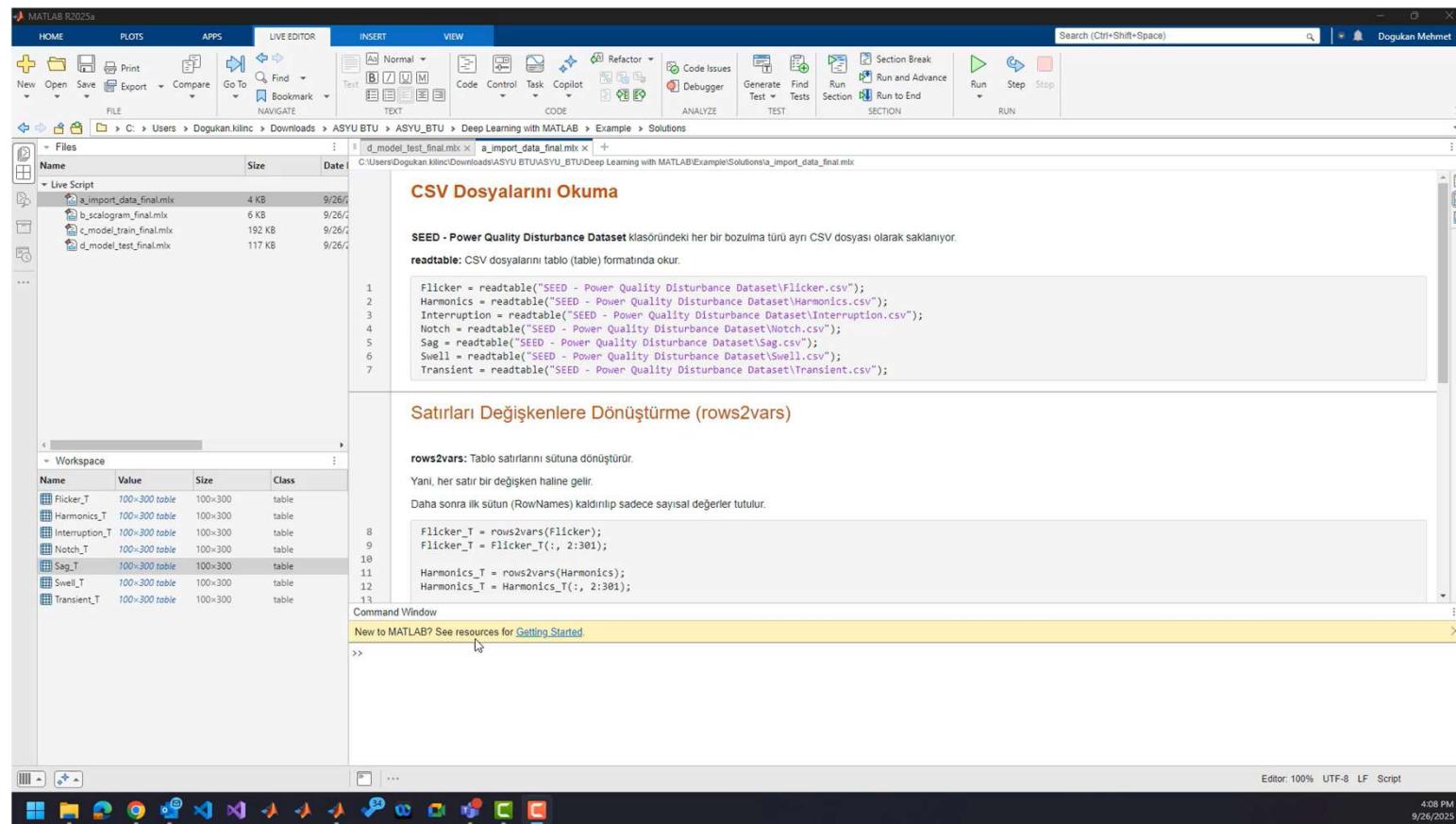
Setting Training Options



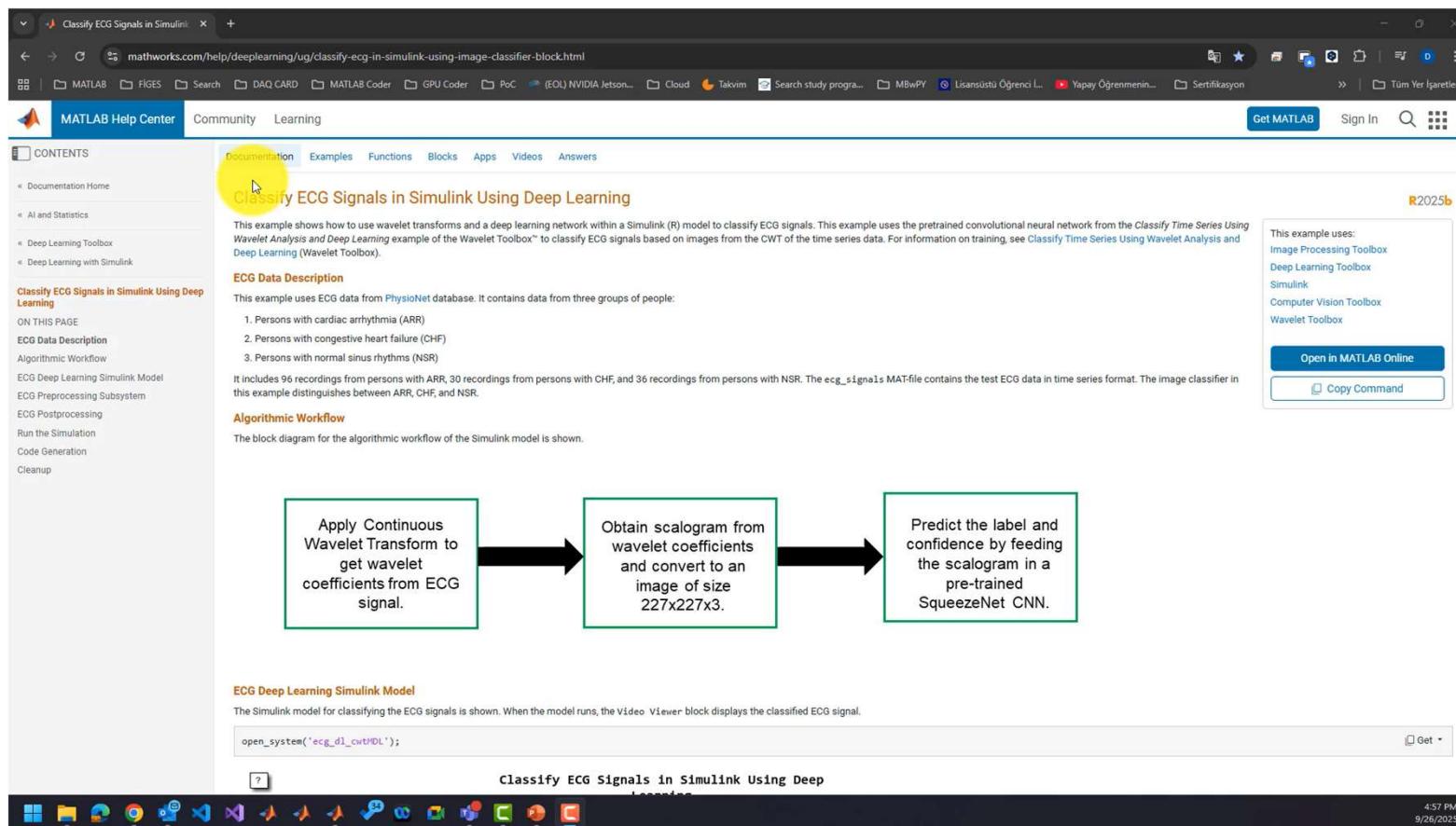
Training and Evaluating the Network



Deep Learning for PQD Signals (← Click with CTRL + Left Click)



Code Generation for ECG Data Project



Classify ECG Signals in Simulink Using Deep Learning

This example shows how to use wavelet transforms and a deep learning network within a Simulink (R) model to classify ECG signals. This example uses the pretrained convolutional neural network from the *Classify Time Series Using Wavelet Analysis and Deep Learning* example of the Wavelet Toolbox™ to classify ECG signals based on images from the CWT of the time series data. For information on training, see *Classify Time Series Using Wavelet Analysis and Deep Learning* (Wavelet Toolbox).

ECG Description

This example uses ECG data from [PhysioNet](#) database. It contains data from three groups of people:

1. Persons with cardiac arrhythmia (ARR)
2. Persons with congestive heart failure (CHF)
3. Persons with normal sinus rhythms (NSR)

It includes 96 recordings from persons with ARR, 30 recordings from persons with CHF, and 36 recordings from persons with NSR. The `ecg_signals` MAT-file contains the test ECG data in time series format. The image classifier in this example distinguishes between ARR, CHF, and NSR.

Algorithmic Workflow

The block diagram for the algorithmic workflow of the Simulink model is shown.

```
graph LR
    A[Apply Continuous Wavelet Transform to get wavelet coefficients from ECG signal.] --> B[Obtain scalogram from wavelet coefficients and convert to an image of size 227x227x3.]
    B --> C[Predict the label and confidence by feeding the scalogram in a pre-trained SqueezeNet CNN.]
    style A fill:#0072bc,color:#fff
    style B fill:#0072bc,color:#fff
    style C fill:#0072bc,color:#fff
```

ECG Deep Learning Simulink Model

The Simulink model for classifying the ECG signals is shown. When the model runs, the Video Viewer block displays the classified ECG signal.

```
open_system('ecg_dl_cwtIDL');
```

4:57 PM
9/26/2025





TEŞEKKÜRLER



dogukan.kilinc@figes.com.tr