

MATLAB EXPO

 INDIA

11 July 2024, Bengaluru

The Industrial AI Lifecycle

Dreaming, Designing, and Delivering in the Digital Age

Koustubh Shirke



Sammit Jain



Monalisa Pal



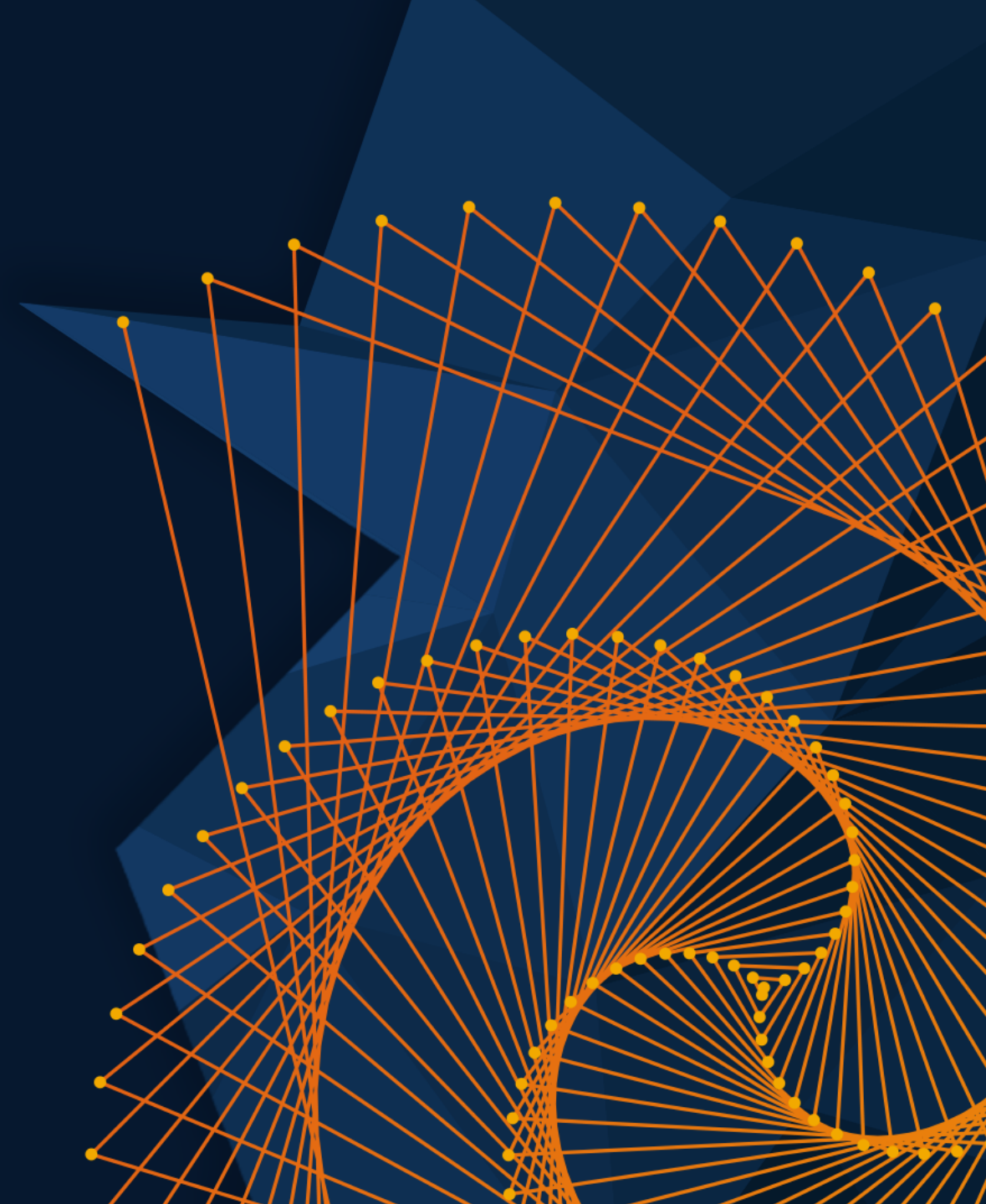
Nikita Pinto



Jayanth Balaji



Peeyush Pankaj (Moderator)



Approaching (Almost) Any Data Science Project

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

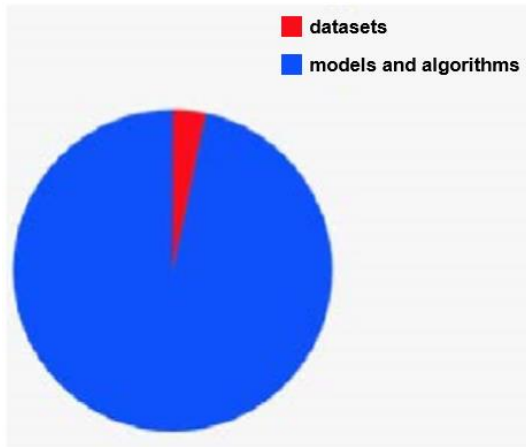


Edge, cloud, desktop

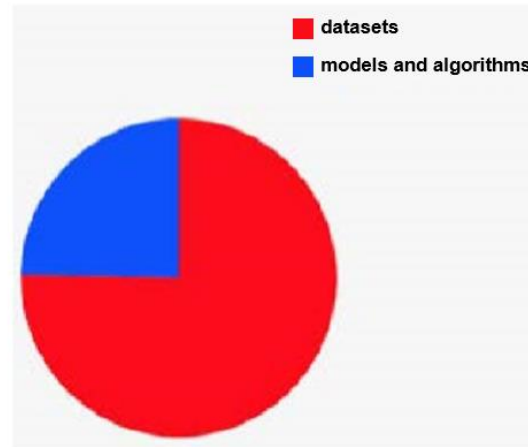
Data preparation represents most of your AI effort

Amount of lost sleep over...

PhD



Tesla



Source: Andrej Karpathy's slide from TrainAI 2018

Challenges in data preparation

- Variety, velocity and volume of data
- Quality and quantity of data annotation
- Leveraging domain expertise
- Lack of data

MATLAB makes it easy to handle different data formats across industries



Tabular



Signal



Text



Image



Video



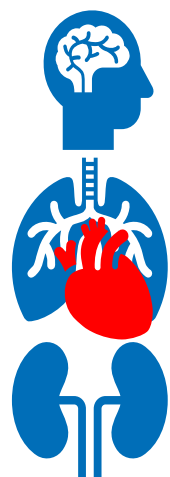
Audio



LIDAR



RADAR



Clinical



Microscopic



Preclinical



Genetic Data



PK/PD Data

Biomedical Data

Geoscience Data



Seismic Data



Well logs



Energy Production



Satellite Imaging



Downstream processing



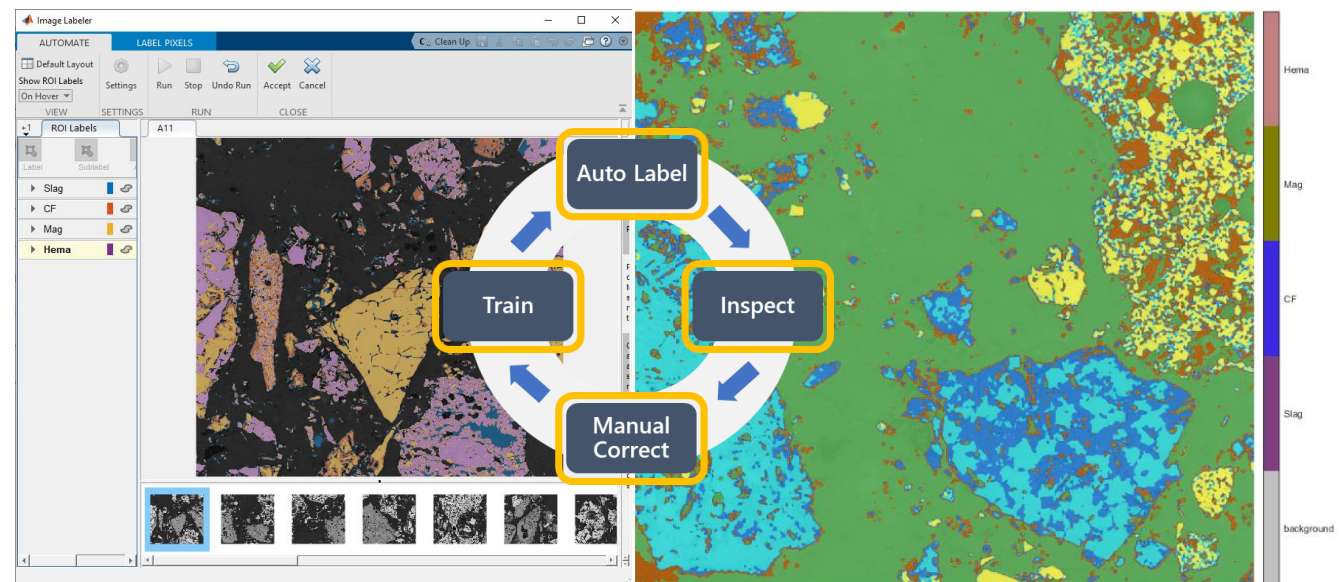
Ocean acoustics

Reduce human supervision and development time



“Even though I had limited knowledge on Image processing and Deep Learning, I could successfully adopt deep learning for my project. **With evaluation support from MathWorks, we could prototype our approach easily with limited time bound.**”

- Created a custom labeling algorithm for automatic labelling material
- Improved prediction accuracy using deep learning
- Partnered with MathWorks to leverage the full benefits of MATLAB



Automated labeling apps save you weeks to months

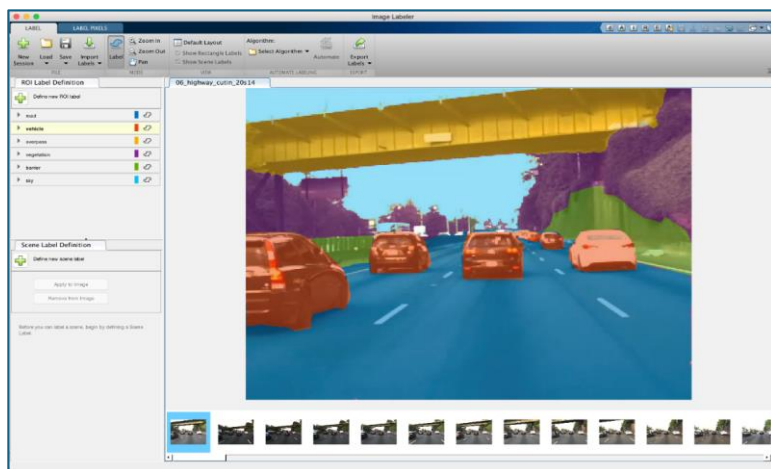
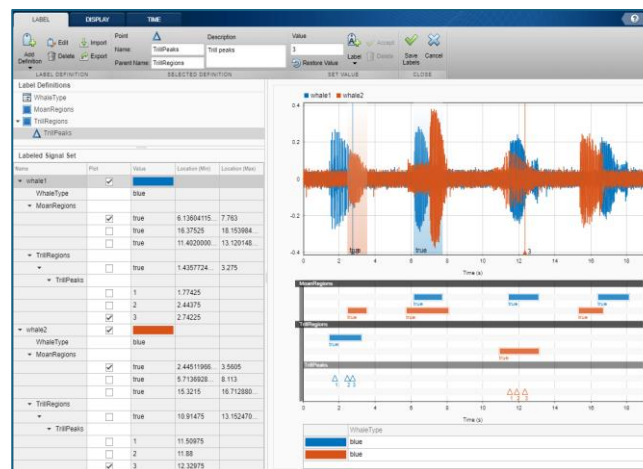
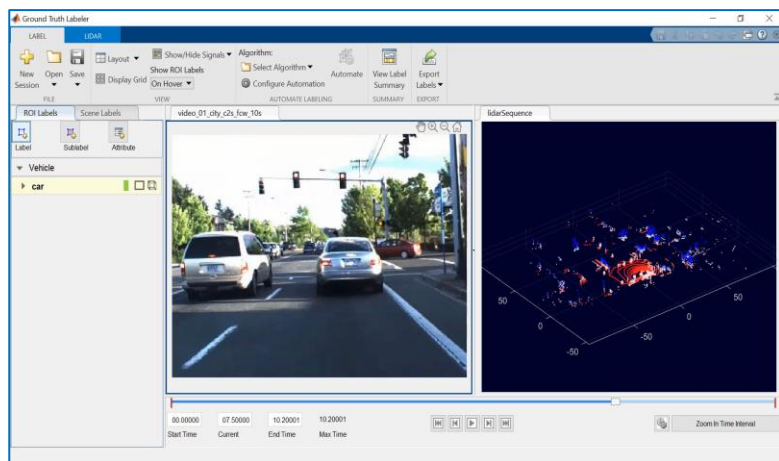


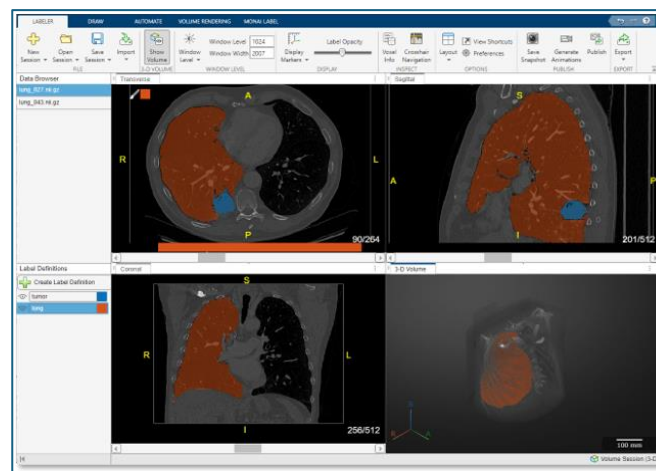
Image Labeler



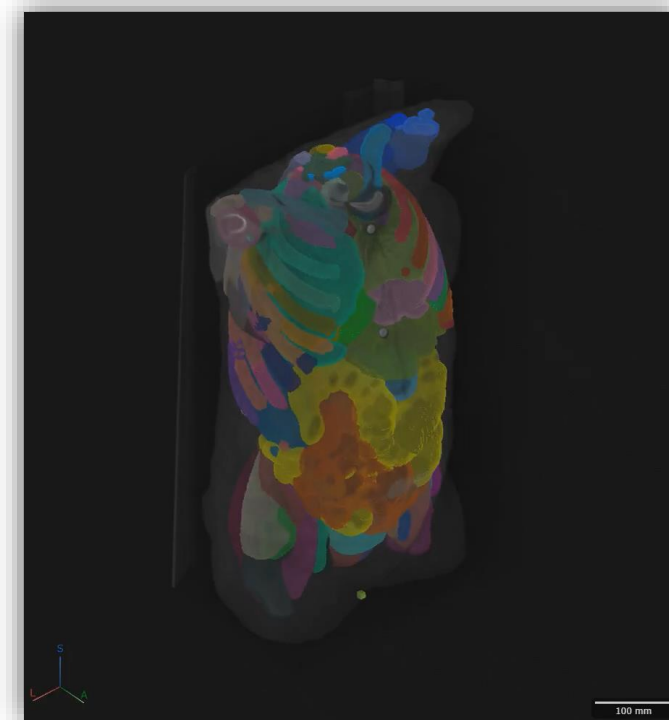
Signal Labeler



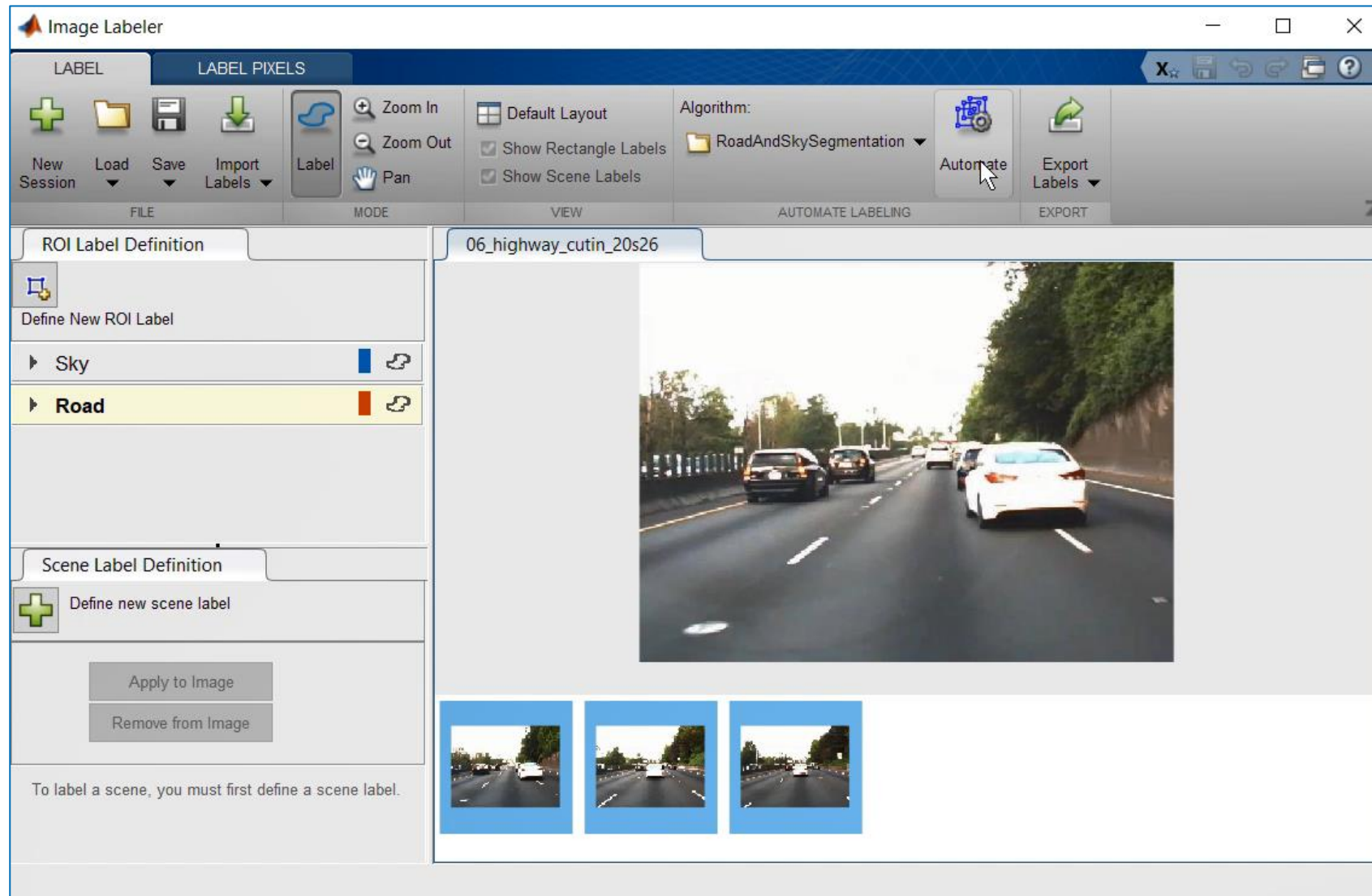
Ground Truth Labeler



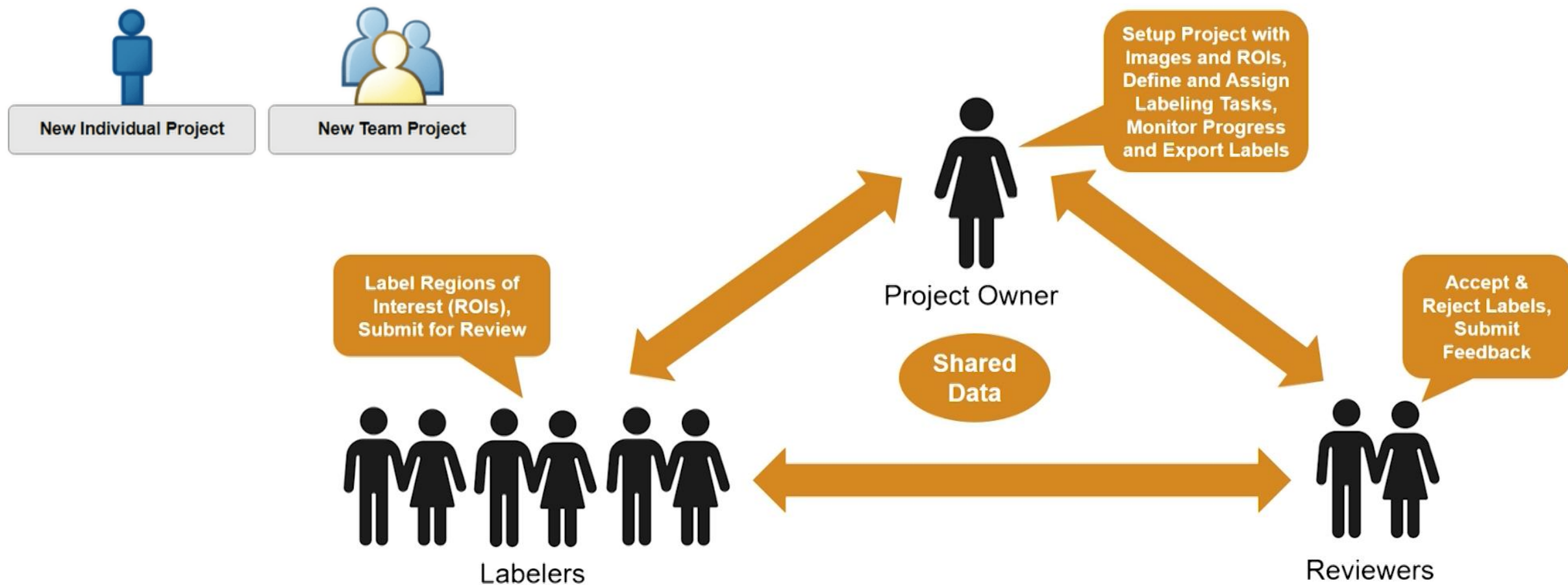
Medical Image Labeler



Label data faster with automated workflows

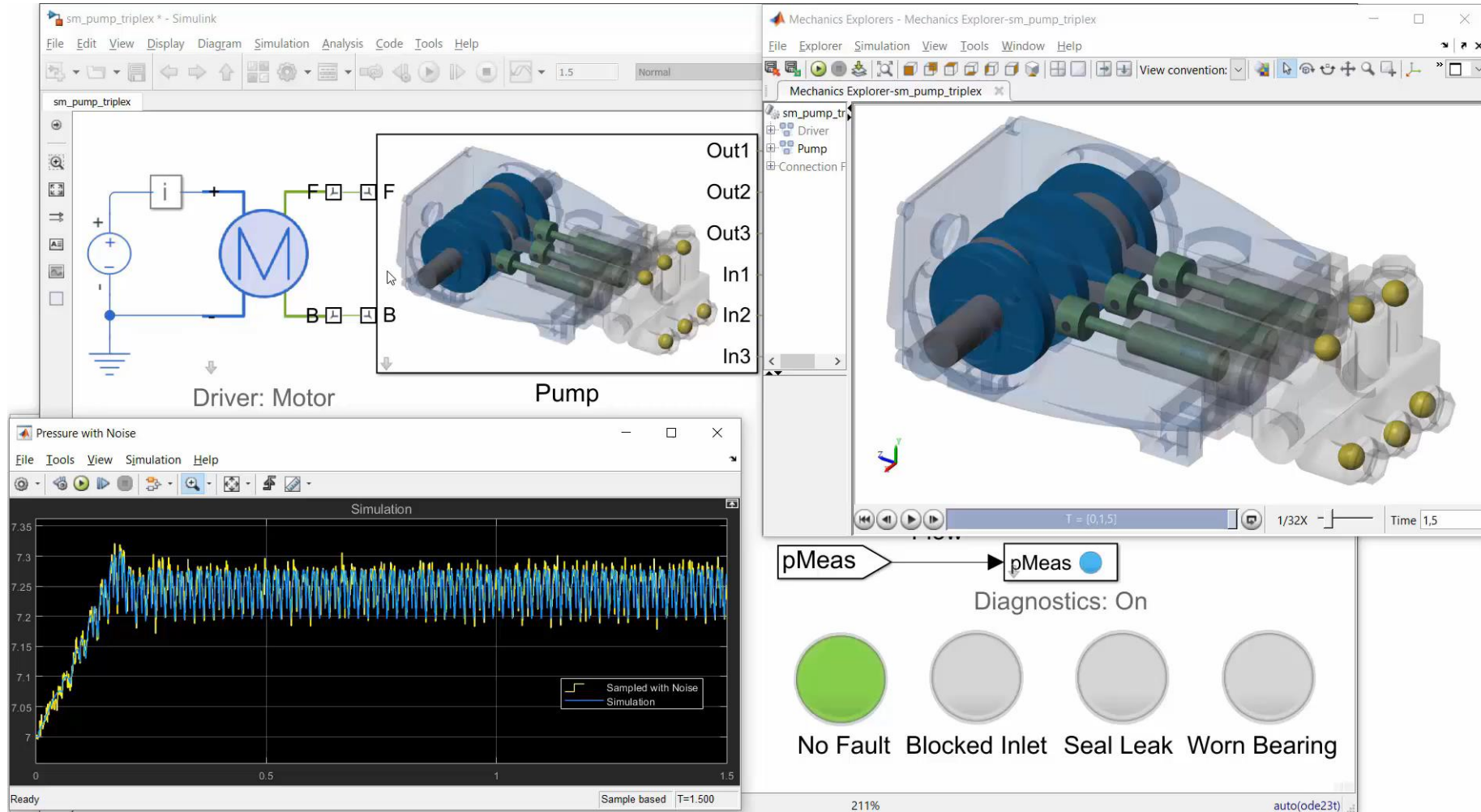


Label data faster with Team-based labeling



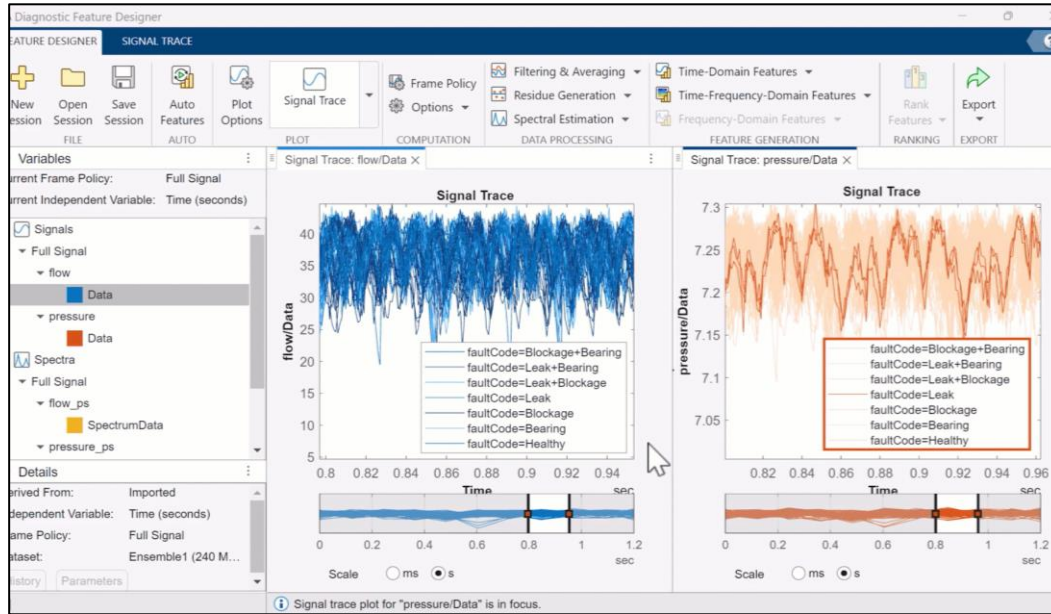
Big Data with No Data?




Simulate rare system failures to avoid them in the real world



App-based feature extraction and selection

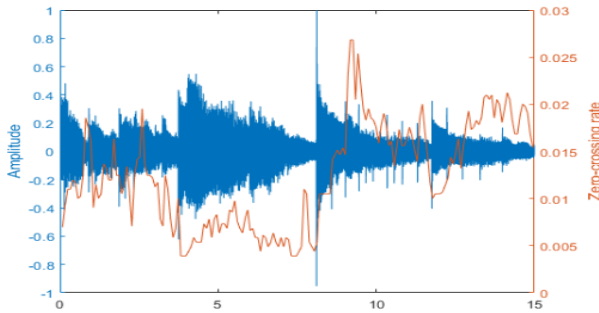
Diagnostic Feature Designer App



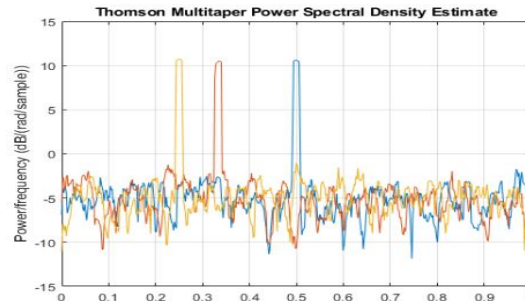
- 
Rotating Machinery Features
 Compute features from rotating machinery signals
- 
Bearing Faults Features
 Compute spectral features for bearing faults
- 
Gear Mesh Faults Features
 Compute spectral features for gear mesh faults

- Extract, visualize, and rank features
- Explore techniques without MATLAB coding
- Handle out-of-memory data
- Generate MATLAB code to automate tasks
- Physics-based features for rotating machines

Time Domain



Frequency-Domain



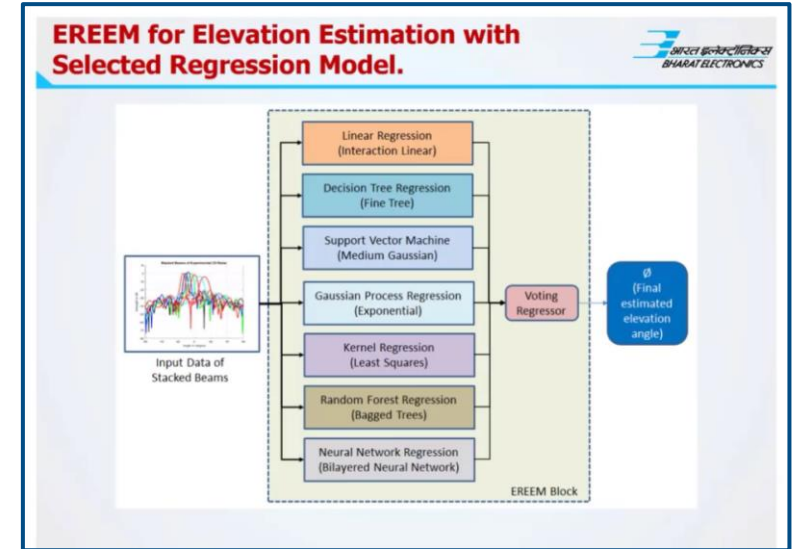
Bharat Electronics Applies AI to Elevation Estimations from 3D Radar

Using MATLAB, the Regression Learner app, and the Sensor Array Analyzer app, the Bharat Electronics team was able to model more accurate and robust predictions of target elevation angles from radar data.

Key Outcomes/Advantages:

- The Sensor Array Analyzer app enabled custom sensor array design and visualization without requiring extra time to code complex simulations for generating data sets to train AI models
- The curve-fitting tool in MATLAB simplified the process of calculating elevation angle estimates and delivered a more accurate result
- The Regression Learner app evaluated data with multiple regression methods in order to find the best fit for accurate predictions

[Link to user story](#)



Complete workflow for the elevation estimation with a selected regression model.

“With the help of AI, a lot more can be done. We have found that if more data is not available, then simulated data can also be generated with the help of MATLAB.”

- Ram Pravesh, Bharat Electronics Limited

Interactive Apps for AI Workflow in MATLAB

- Split Data
- Visualize Data
- Train and Compare Models
- Tune hyperparameters
- Export Trained Model or Generate a Function

Validation

Validation Scheme

Cross-Validation

Protects against overfitting. For data not set aside for testing, the app partitions the data into folds and estimates the accuracy on each fold.

Cross-validation folds: 5

[Read about validation](#)

Test

Set aside a test data set

Percent set aside: 20

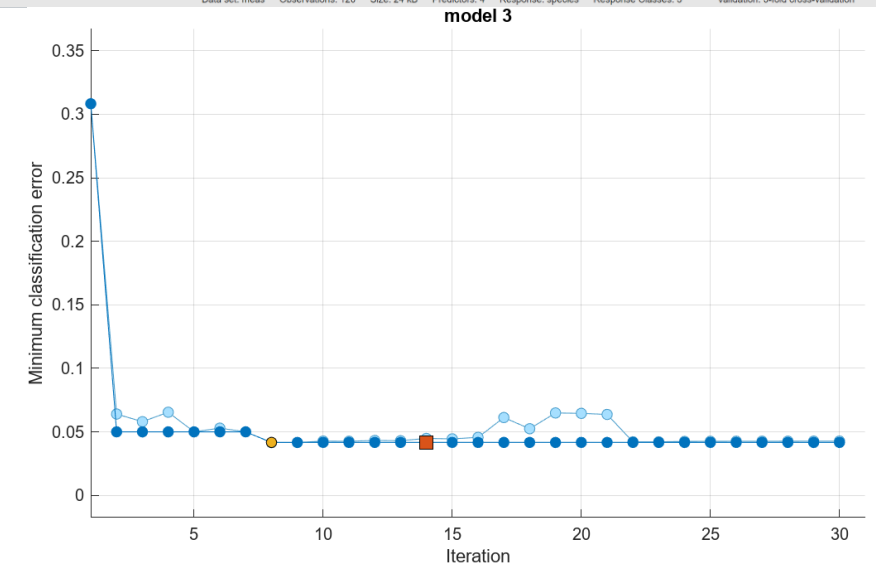
Use a test set to evaluate model performance after tuning and training models. To import a separate test set instead of partitioning the current data set, use the Test Data button after starting an app session.

[Read about test data](#)

Export Plot to Figure
Generate Function
Export Model

EXPORT

True Class \ Predicted Class	setosa	versicolor	virginica
setosa	40		
versicolor		37	3
virginica		2	38



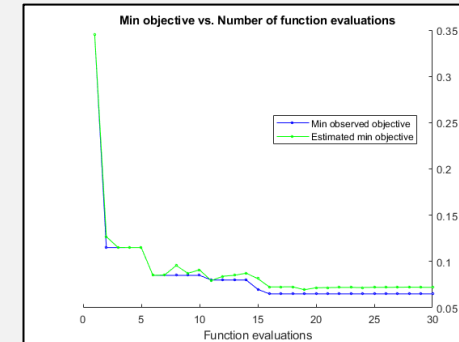
Improve Productivity with AutoML

- Productivity: build accurate models faster
- Narrow skill gap between engineers and data scientists
- Validate success of your “traditional” model building

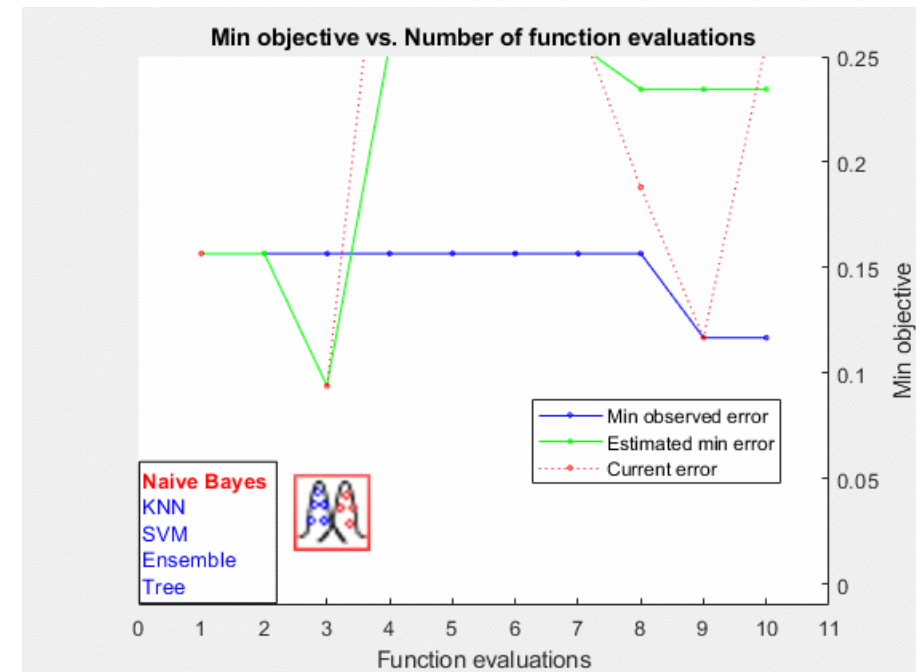
Model Selection

fitcauto/fitrauto

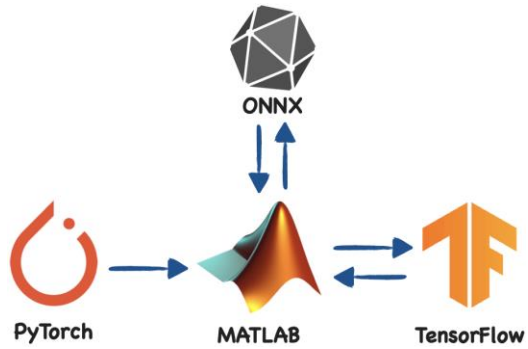
Hyper-parameter Optimization



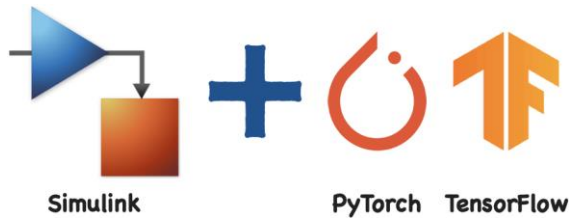
Decision Tree?
SVM?
KNN?
Ensemble?
...?



Analyzing the Deep Neural Networks



Interactively design networks



Analyze the network


Analysis for dlnetwork usage

Name: Network from Deep Network Designer

Analysis date: 21-Jun-2024 14:10:15

1.2M total learnables, 68 layers, 0 warnings, 0 errors

LAYER INFORMATION			
	Name	Type	Activation
1	data	Image Input	227(S) × 227(S) × 3
2	conv1	2-D Convolution	113(S) × 64 × 3 × 3 convolutions with stride [2 2] a...
3	relu_conv1	ReLU	113(S) ×
4	pool1	2-D Max Pooling	56(S) × 3 × 3 max pooling with stride [2 2] and pa...
5	fire2-squeeze1x1	2-D Convolution	56(S) × 16 × 1 × 1 × 64 convolutions with stride [1 1] ...
6	fire2-relu_squeeze1x1	ReLU	56(S) ×
7	fire2-expand1x1	2-D Convolution	56(S) × 64 × 1 × 1 × 16 convolutions with stride [1 1] ...
8	fire2-relu_expand1x1	ReLU	56(S) ×
9	fire2-expand3x3	2-D Convolution	56(S) × 64 × 3 × 3 × 16 convolutions with stride [1 1] ...

 jianghaw Merge pull request #17 from sureshmaddina/master	54ee7d2 · 2 months ago	52 Commits
Images	Adding a section called "Path planning with Motion Planni...	2 months ago
LICENSE	Initial Commit	3 years ago
MATLABDeepLearningModelHub.mlx	updated MATLABDeepLearningModelHub.mlx	last year
README.md	Adding a section called "Path planning with Motion Planni...	2 months ago
SECURITY.md	Initial Commit	3 years ago
viewDeepLearningModelHubGitHub.m	Script for launching the Deep Learning Model hub in a bro...	2 years ago

MATLAB Deep Learning Model Hub

Discover pretrained models for deep learning in MATLAB.

Models

About

Discover pretrained models for deep learning in MATLAB

www.mathworks.com/solutions/deep-lea...

- awesome
- deep-learning
- model-zoo
- matlab
- awesome-list
- pretrained-models
- matlab-deep-learning

- Readme
- View license
- Security policy
- Activity
- Custom properties
- 418 stars
- 27 watching
- 107 forks
- Report repository

Releases 4

R2024a **Latest**

Compare models and tune hyperparameters with Experiment Manager

Experiment Manager - Experiment1 Configuration

Description: Classify digits, different learning rates and momentum, more epochs

Hyperparameter Table:

Name	Values
myInitialLearnRate	[0.005 0.006 0.007 0.008 0.009 0.01]
myMomentum	0.1*(5:9)

Setup Function: Experiment1_setup1

Metrics: oneAsSeven

Experiment Manager - Result Details

Result Details: Experiment1 1/10/2020, 1:46:56 AM 30/30

Summary: Complete 30, Stopped 0, Error 0, Running 0, Queued 0, Canceled 0

Trial	Status	Progress	myInitial...	myMomen...	Validation Acc...	Validation Loss
1	Complete	100.0%	0.0050	0.5000	57.8000	1.2512
2	Complete	100.0%	0.0060	0.5000	59.0000	1.2348
3	Complete	100.0%	0.0070	0.5000	60.2400	1.2219
4	Complete	100.0%	0.0080	0.5000	61.2800	1.2123

Visualizations:

Training Plot (Trial 1, Result1, Experiment1)

Accuracy (%) vs Iteration (0 to 450). Final accuracy is approximately 66.04%.

Loss vs Iteration (0 to 450). Final loss is approximately 1.3692.

Filters:

- Validation Accuracy (%): Histogram showing distribution from 57.8000 to 66.0400.
- Validation Loss: Histogram showing distribution from 1.1780 to 1.3692.

Mercedes-Benz Simulates Hardware Sensors with Deep Neural Networks

Challenge

Simulate automotive hardware sensors with deep neural networks

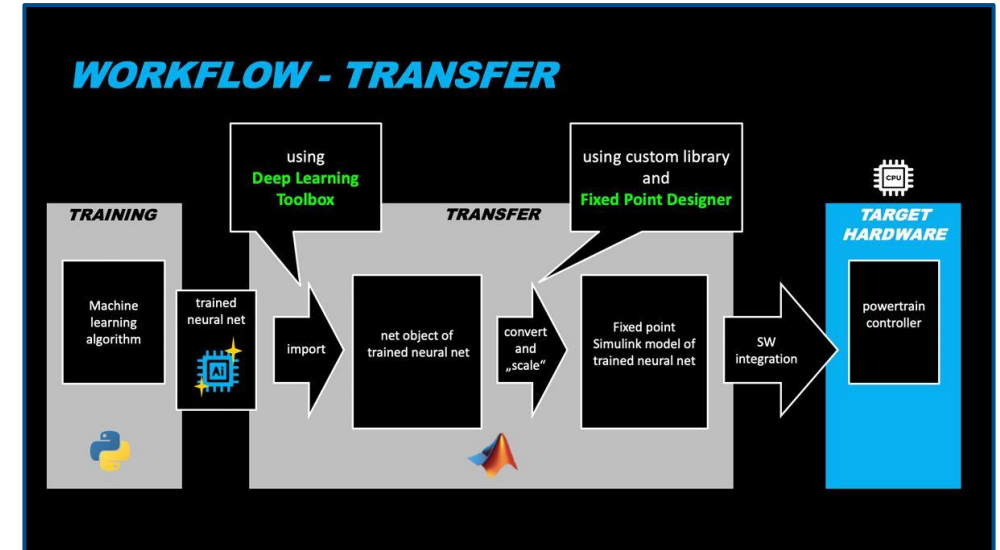
Solution

Use MATLAB, Simulink, Deep Learning Toolbox, and Fixed-Point Designer to convert Qkeras deep learning models into code that can be deployed to an automotive ECU

Results

- CPU, memory, and performance requirements met
- Flexible process established
- Development speed increased 600%

[Link to user story](#)



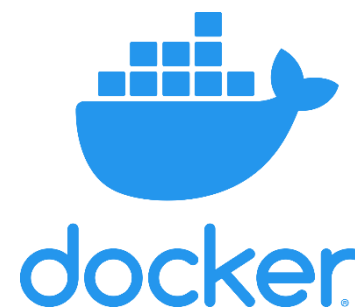
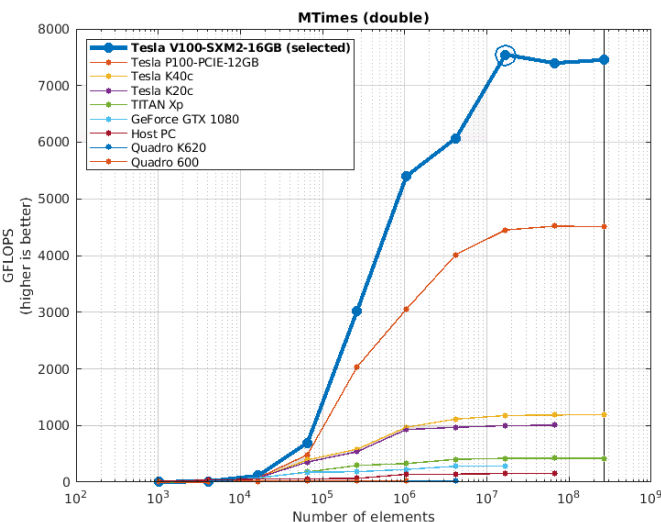
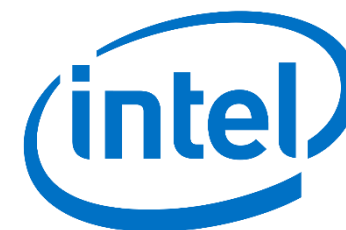
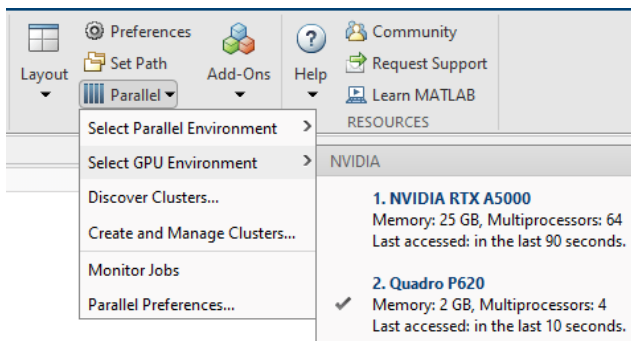
Automated workflow for deploying virtual sensors to powertrain ECU.

“This was the first time we were simulating sensors with neural networks on one of our powertrain ECUs. Without MATLAB and Simulink, we would have to use a tedious manual coding process that was very slow and error-prone.”

- Katja Deuschl, AI developer at Mercedes-Benz

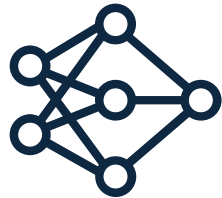
Hardware acceleration and scaling are critical for training

MATLAB accelerates AI training on GPUs, cloud, and datacenter resources without specialized programming.

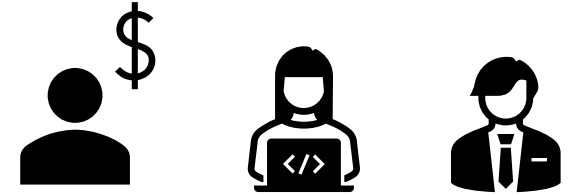


There is a desire to explain, verify and validate AI in production

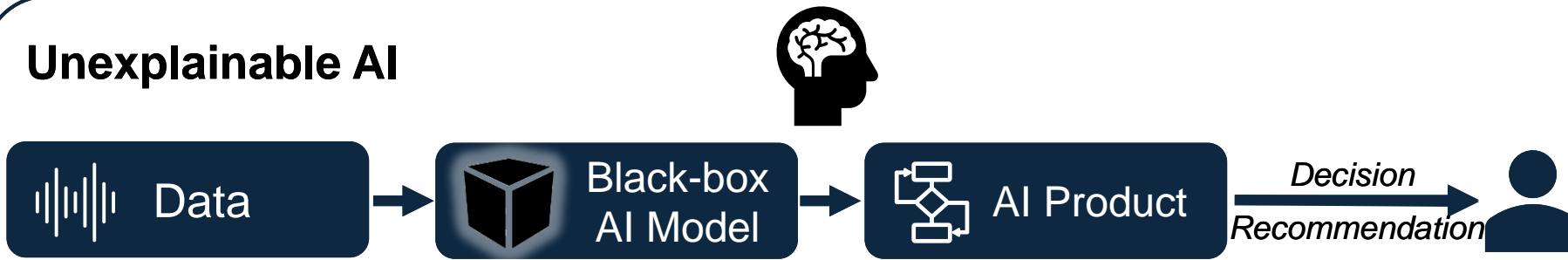
**AI provides the best results
for many tasks**



Explainable AI facilitates informed decisions

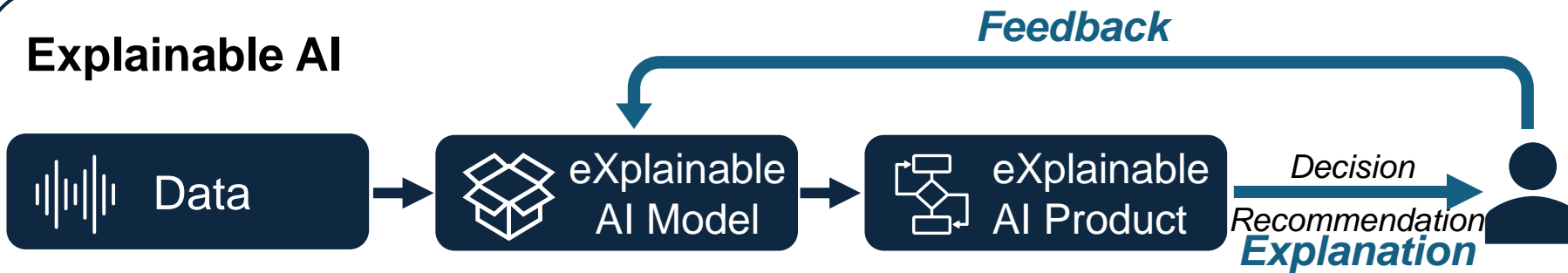


Unexplainable AI



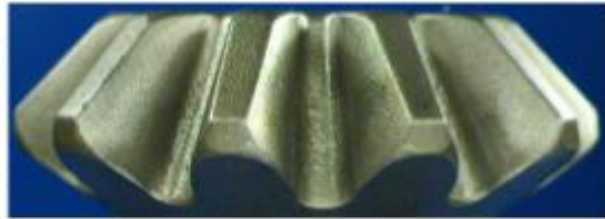
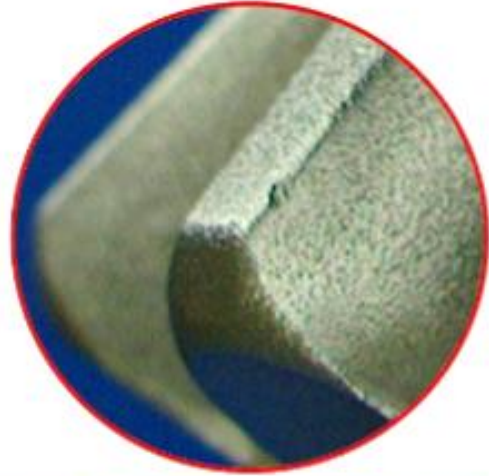
- Why did you do that?
- Why did you not do that?
- When do you succeed or fail?
- When can I trust you?
- How do I correct an error?

Explainable AI



- I understand why
- I understand why not
- I know why you succeed or fail
- I know when to trust you
- I know why you erred

Musashi Seimitsu Industry Uses Deep Learning for Visual Inspection of Automotive Parts



“

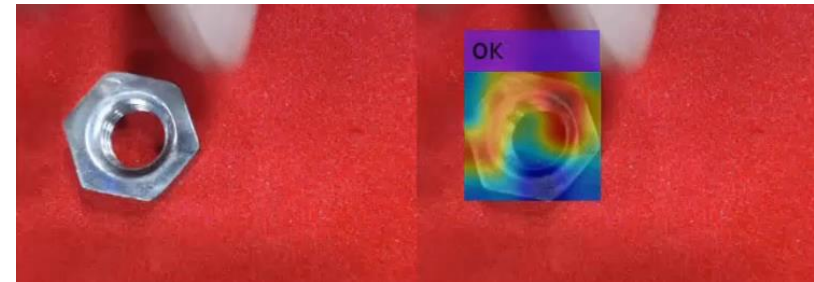
Using camera connection, preprocessing, and various pretrained models in MATLAB enabled us to work on the entire workflow. Through discussions with consultants, our team gained many tips for solving problems, growing the skills of our engineers.

”

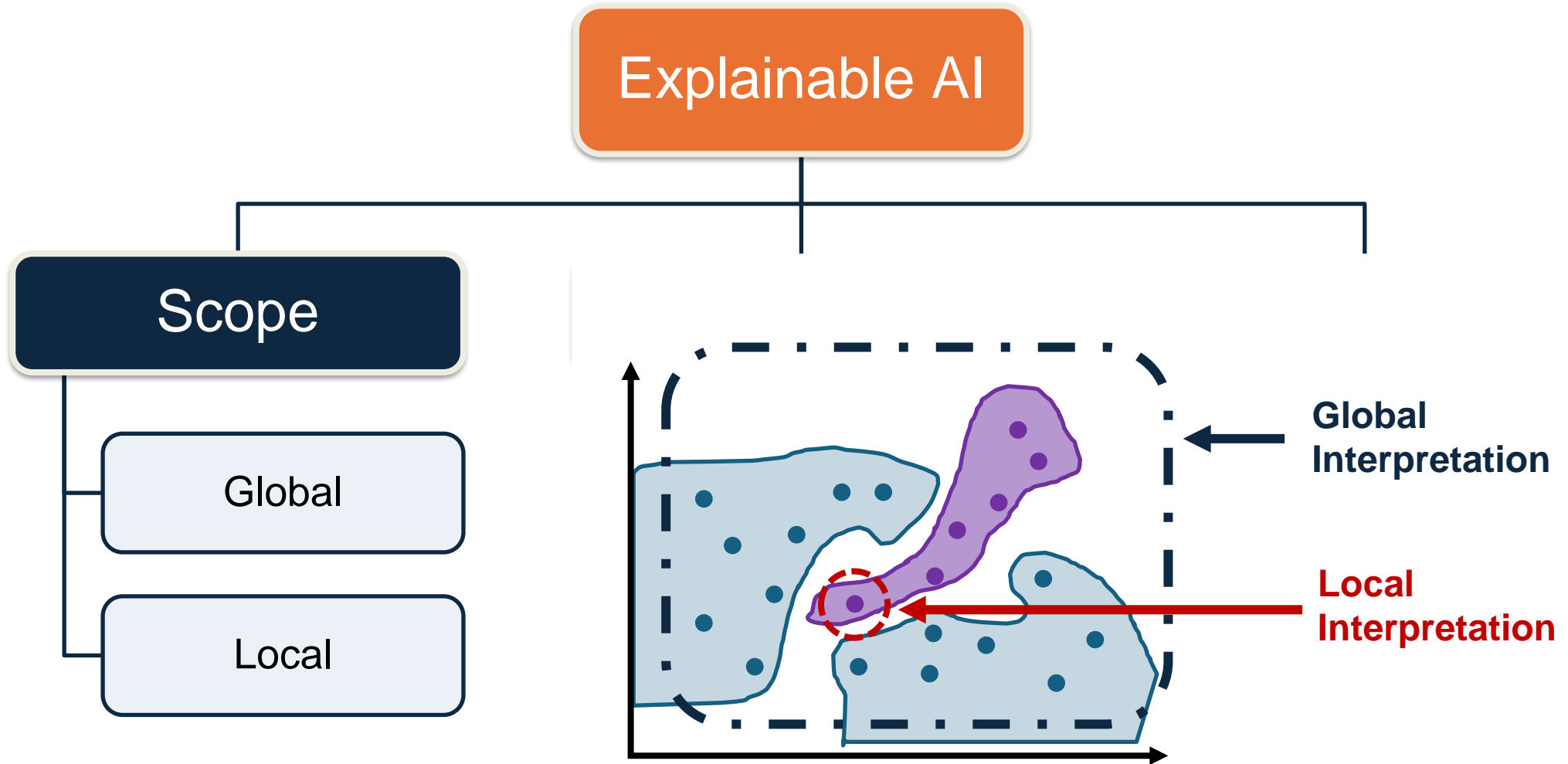
How XAI was used:

Estimate and visualize the defect area
using Class Activation Mapping

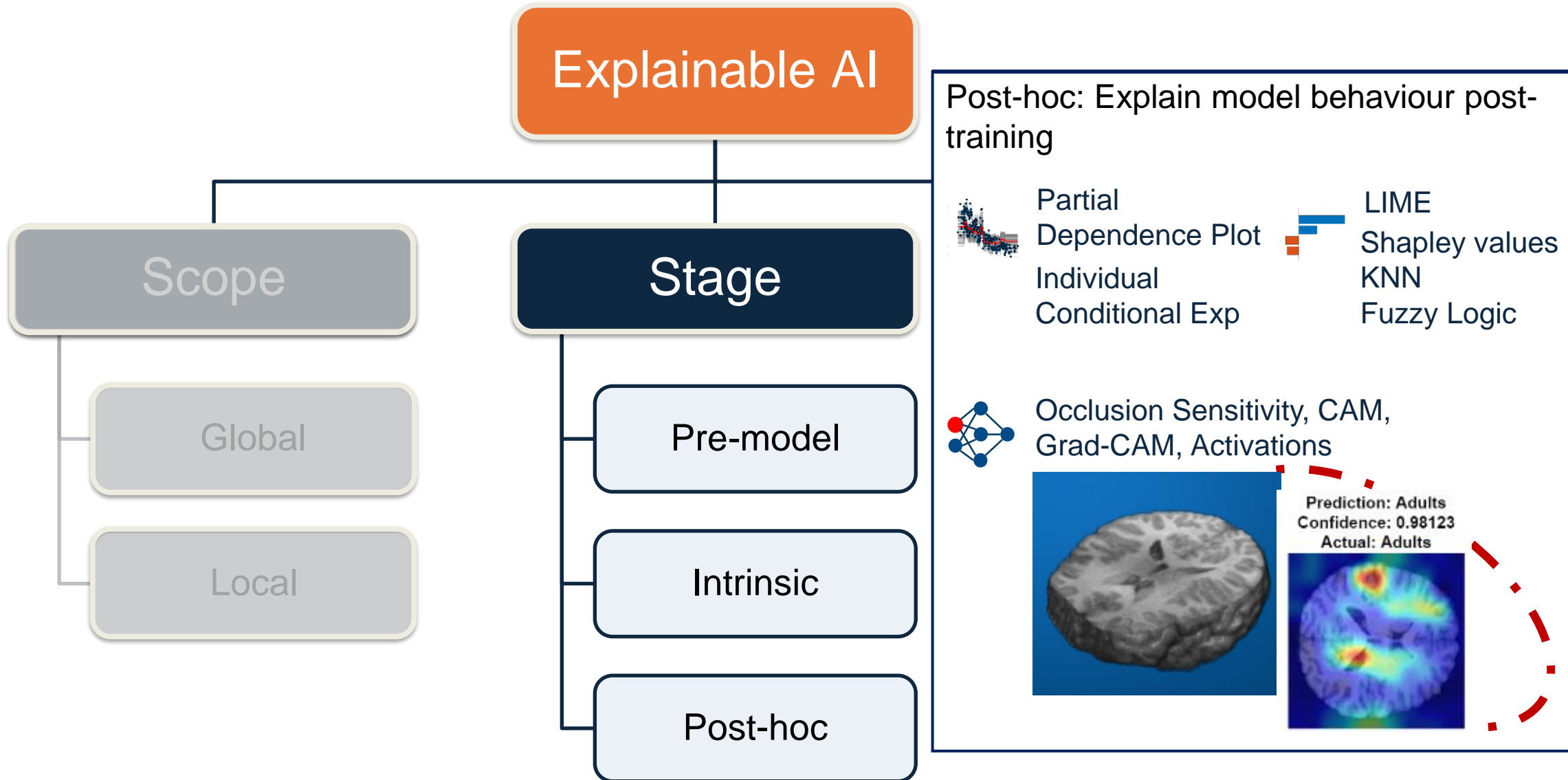
Class Activation Mapping



A taxonomy of Explainable AI

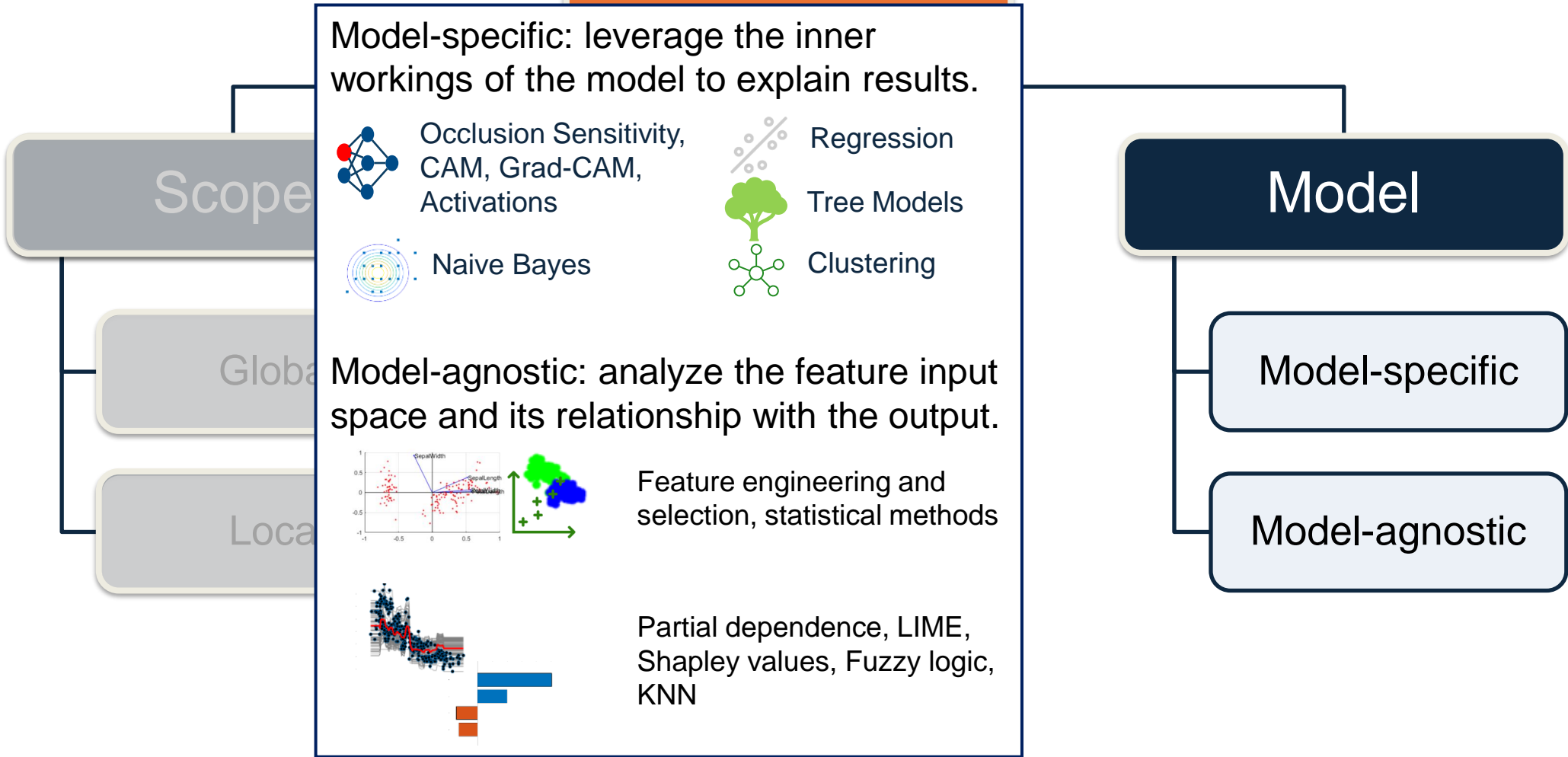


A taxonomy of Explainable AI

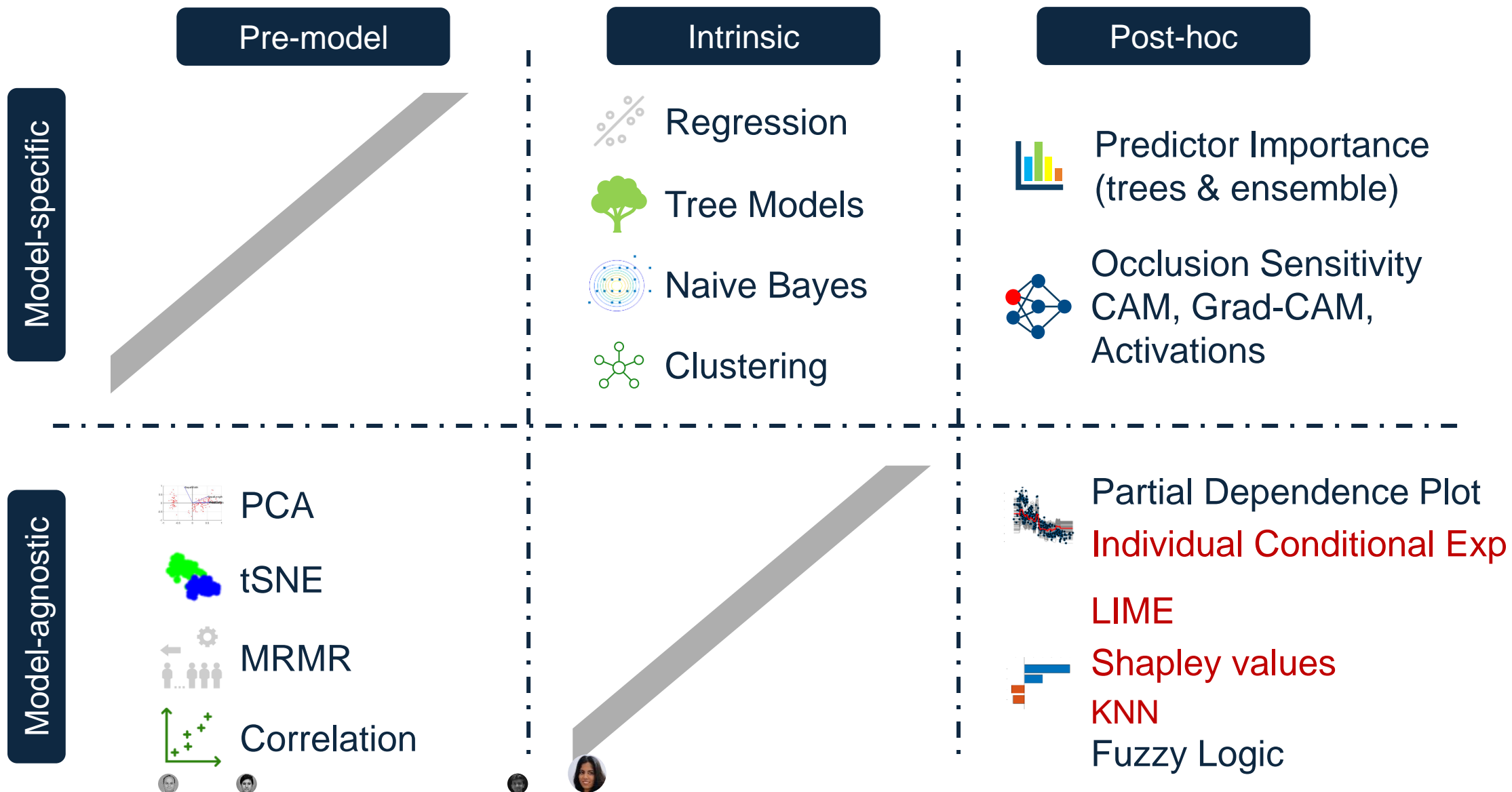


A taxonomy of Explainable AI

Explainable AI



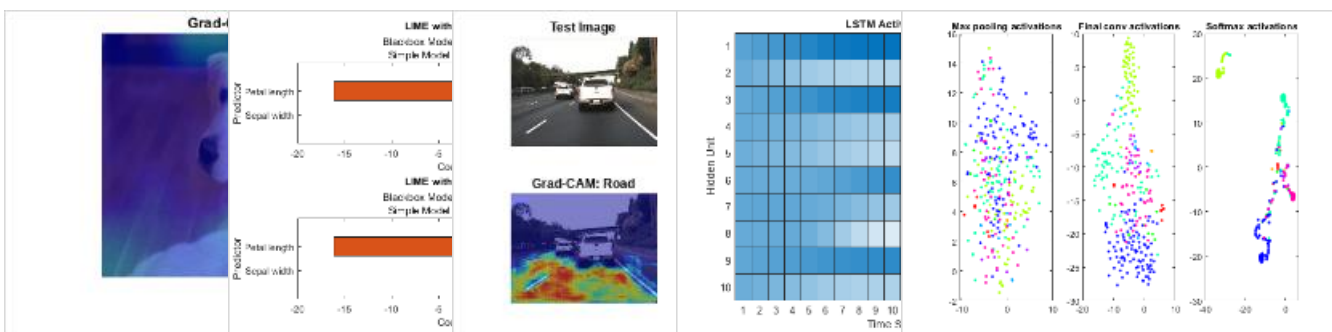
Different Explainability methods



Start explaining your AI model with Golden References Today



MATLAB Deep Learning
mathworks.github.io
<https://www.mathworks.com/solutions/deep-le...>



Grad Bel Dev

Use active tech learn

Int Pre Us

Use agr tech pre

Ex Se Us

Exp pre net

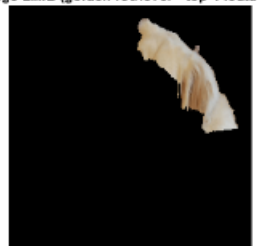
Vi LS

Inv fea by

Vi U

Use ac vie ne

Image LIME (golden retriever - top 4 features)



Understand Network Predictions Using Occlusion Sensitivity

Use occlusion sensitivity to understand why a deep neural network makes a classification decision. Occlusion sensitivity is used to understand why a deep neural network makes a classification decision.

Visualize Image Classifications Using Maximal and Minimal Activation

Use a data set to find which parts of an image activate the channels of a neural network. This helps to understand how a deep neural network makes a classification decision.

Investigate Sparsely Activated Neurons in Image Classifications

Use locally interpretable model-agnostic explanations to investigate the robustness of a convolutional neural network.

Deep Dream Images Using GoogLeNet

Generate images using deepDreamImage on a pretrained convolutional neural network GoogLeNet.

Understand Network Predictions Using LIME

Use locally interpretable model-agnostic explanations (LIME) to understand why a deep neural network makes a classification decision.

and more...

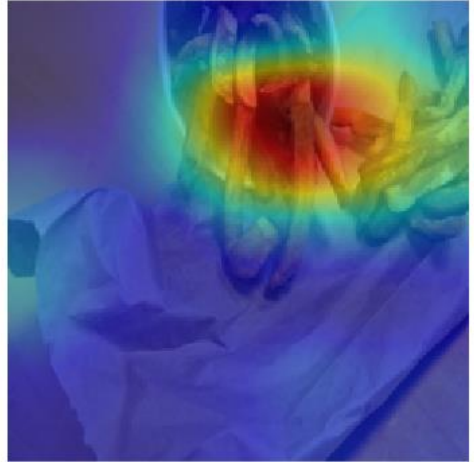
Understanding Network Predictions for Image Classification (UNPIC)

Image Data Accuracy Predict Prediction Explorer Features t-SNE

Visualize which parts of an image are most important for classification. Visualize for a chosen image file, or a random image from the test data.

Choose image file:

Random image from class:



True class: french fries

Grad-CAM uses the gradient of the classification score with respect to the convolutional features determined by the network to understand which parts of the image are most important for classification.

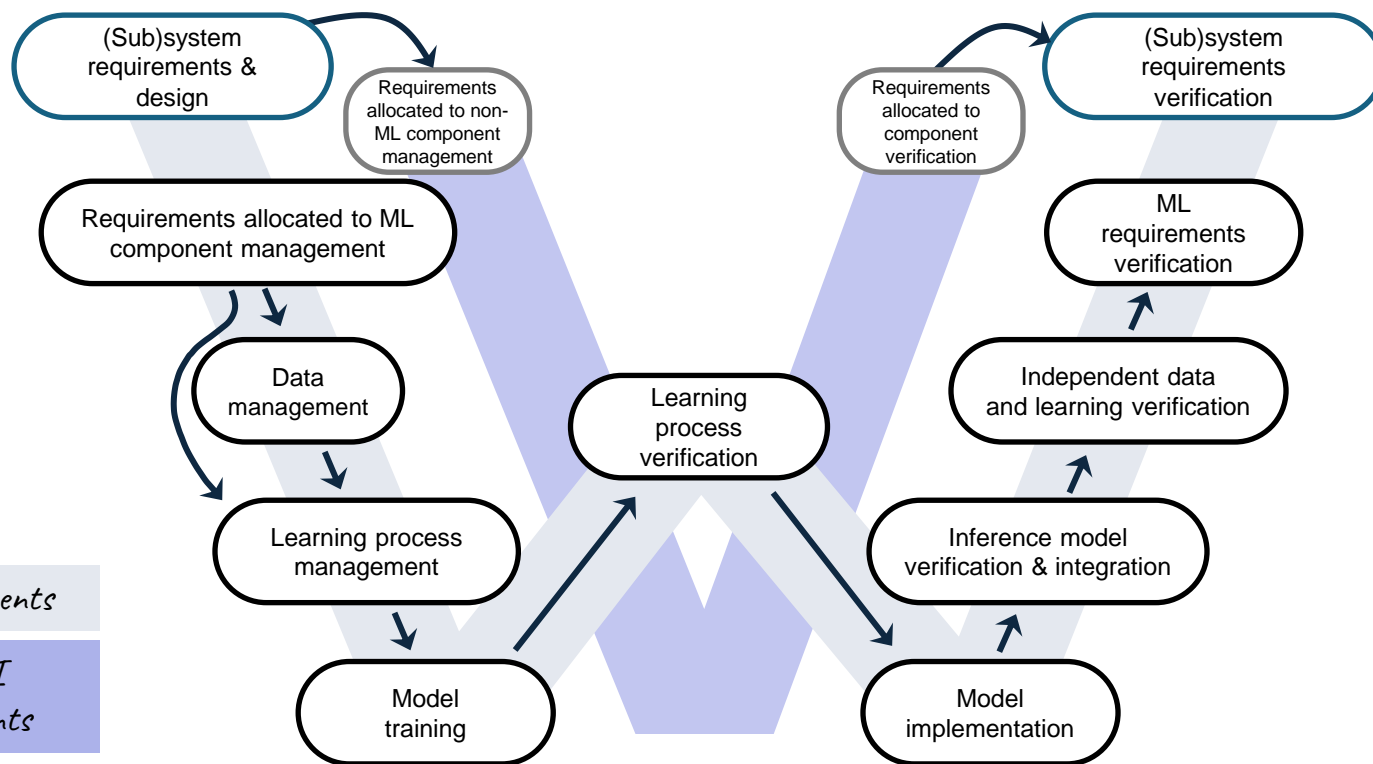
Select a target class to see which parts of the image are most important for that class.

Grad-CAM Settings

Target class:

Feature map:

Industries are making progress on verifying AI in systems and its inevitable



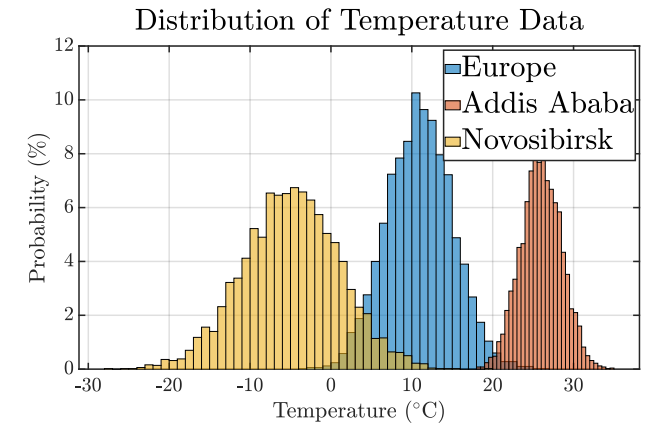
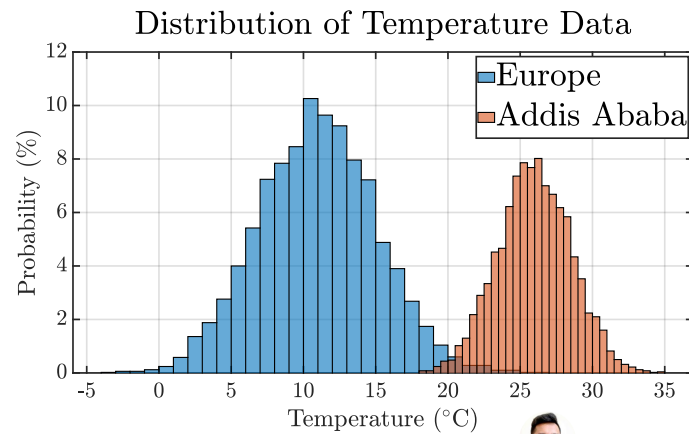
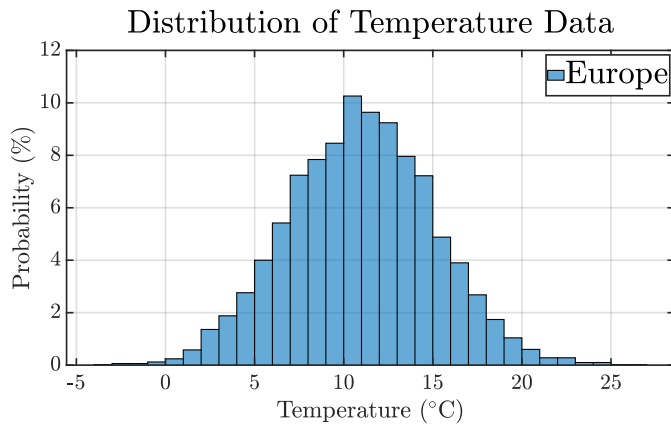
Credit: EASA

MathWorks has capabilities addressing each area of the W-diagram

MathWorks is actively engaging with research groups and certification bodies

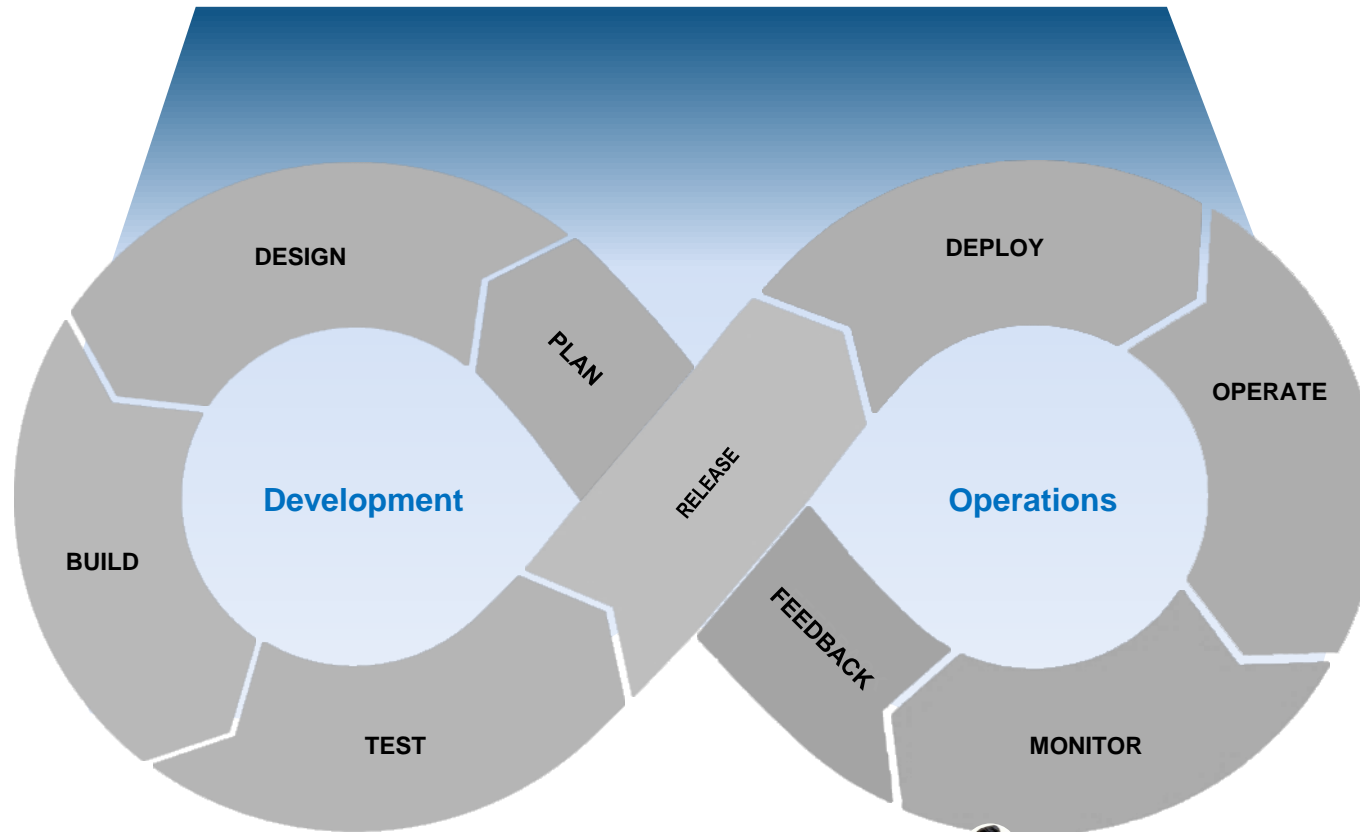


Statistical changes in the input data may decrease a model's predictive or classification accuracy.



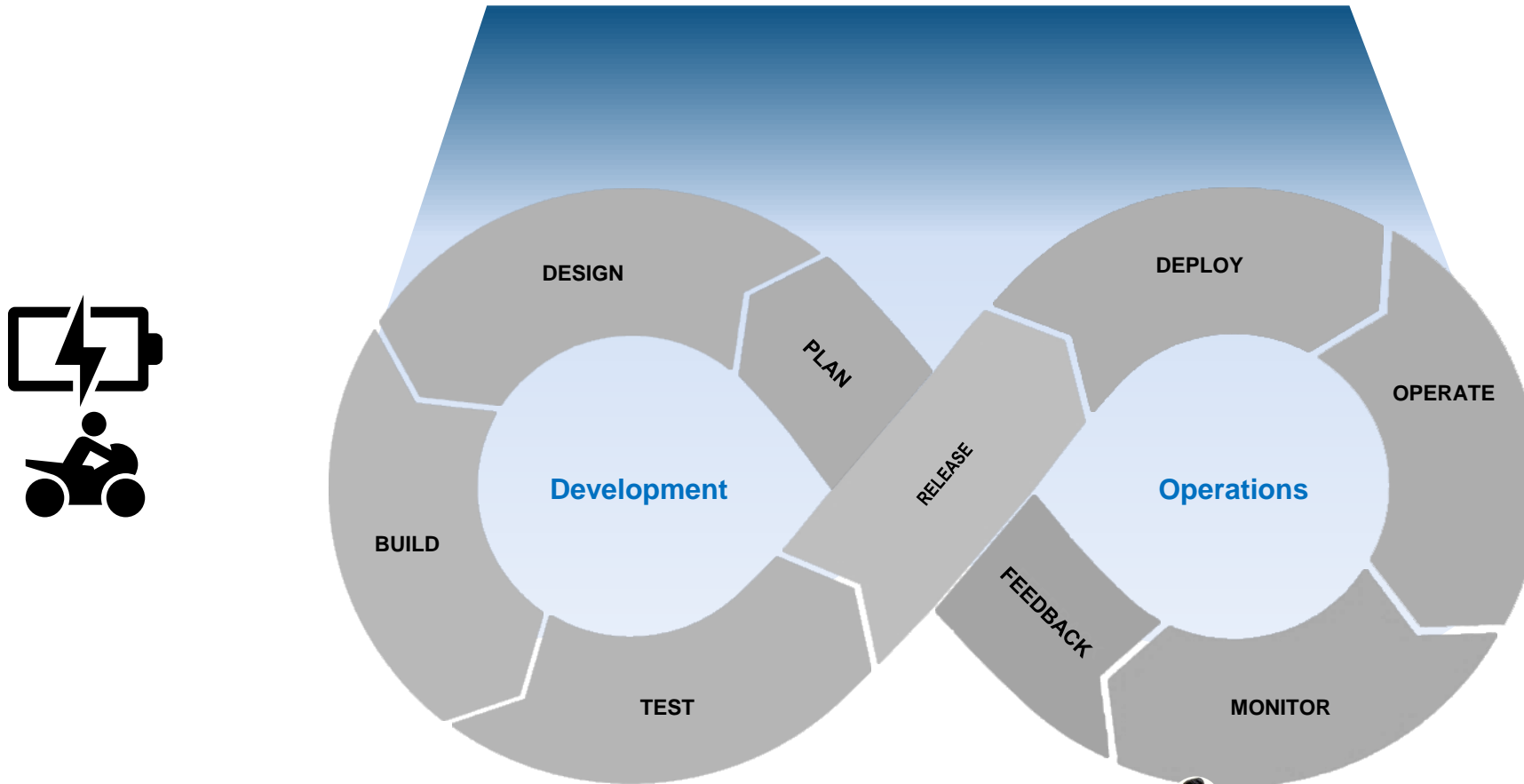
Prompt calibration and model evaluation produces better models and more accurate decisions.

ML Ops

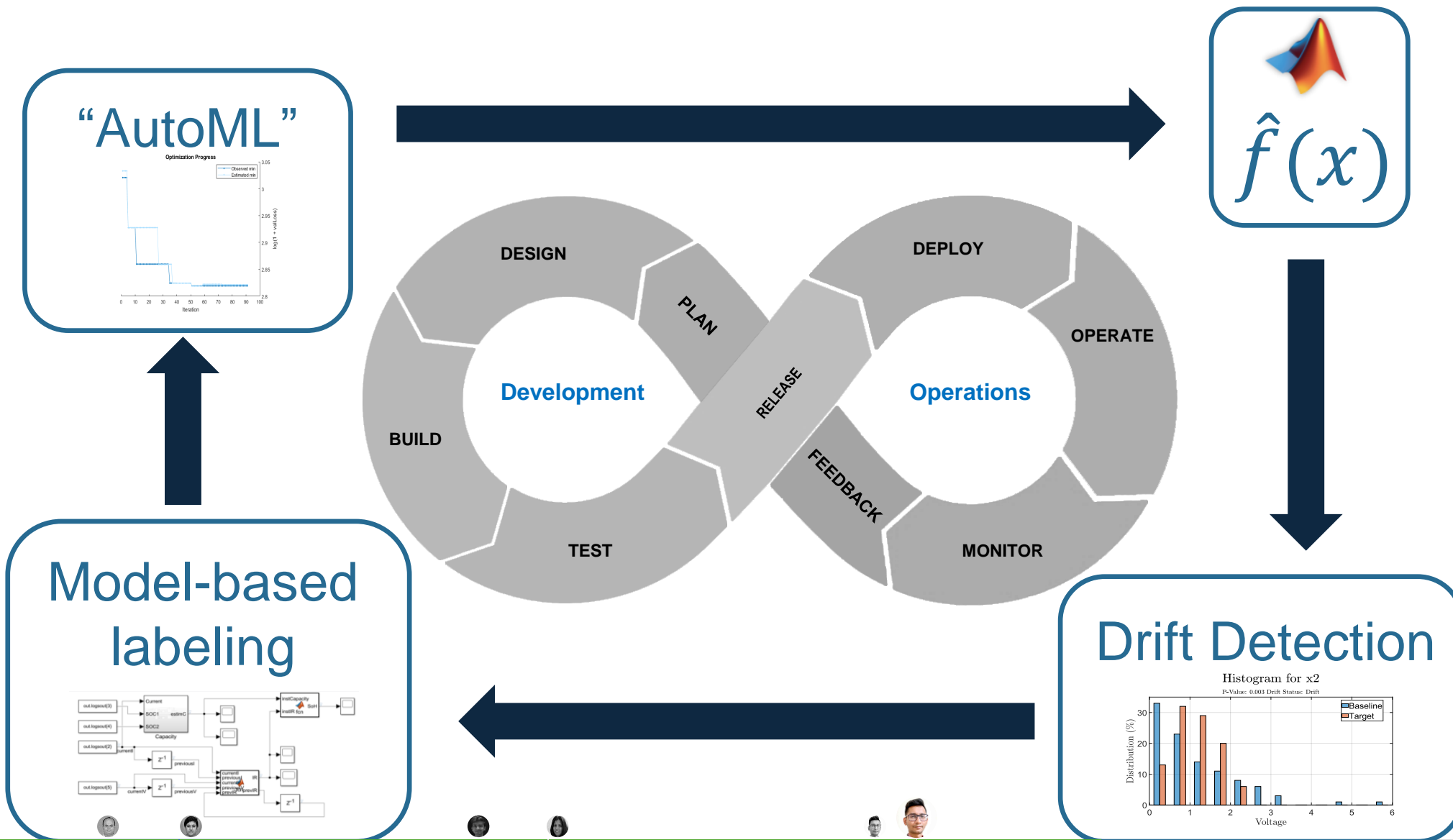


Understanding the lifecycle of a machine learning solution lets you know if you've automated all of it.

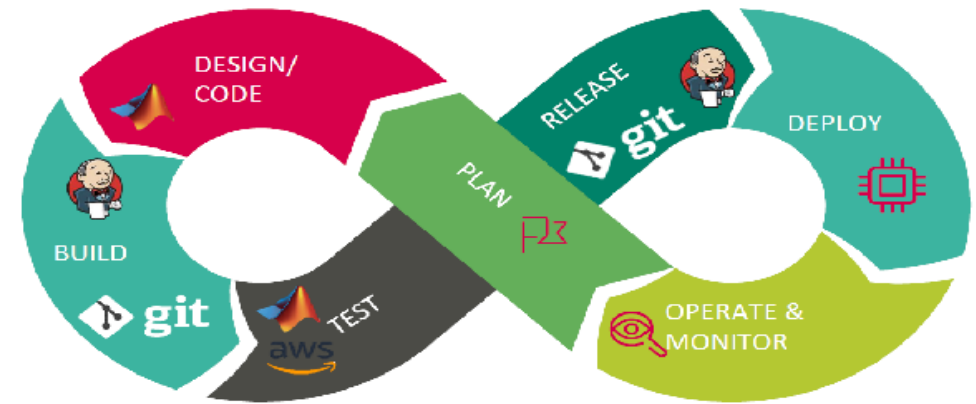
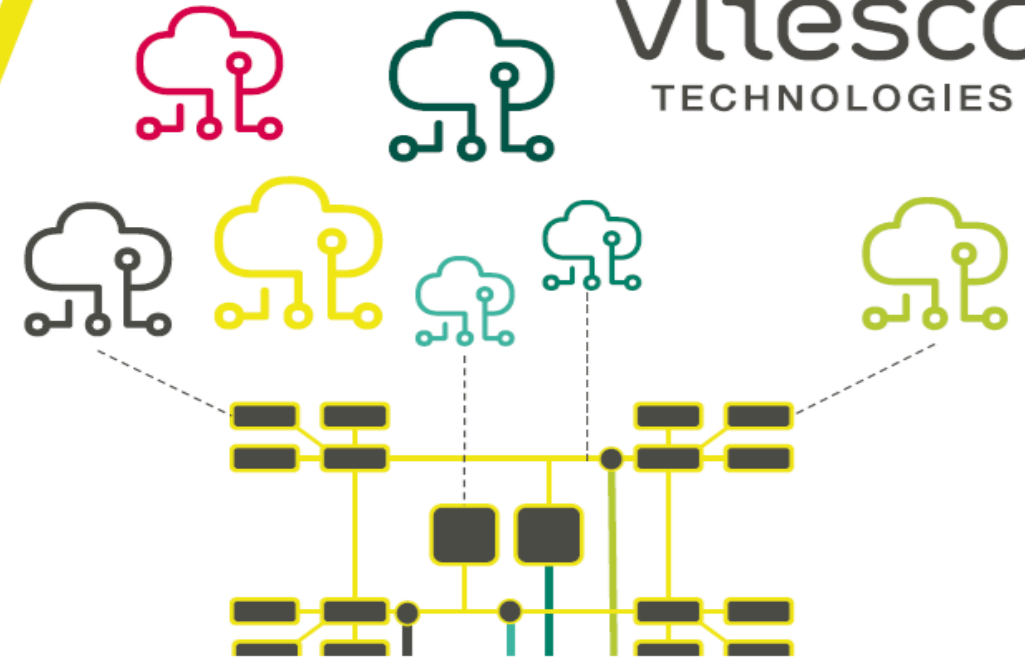
ML Ops



Lots of Moving components and need a strong template to get started

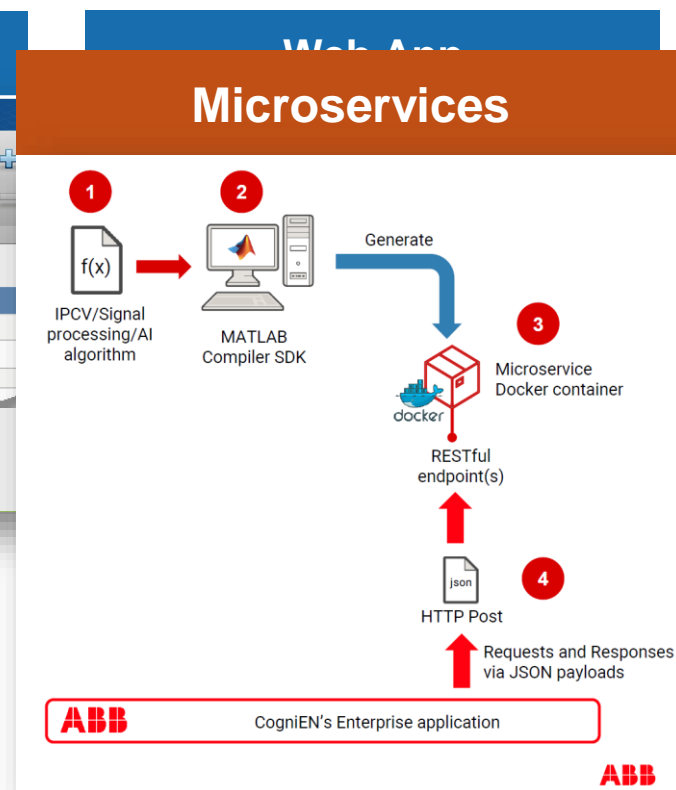
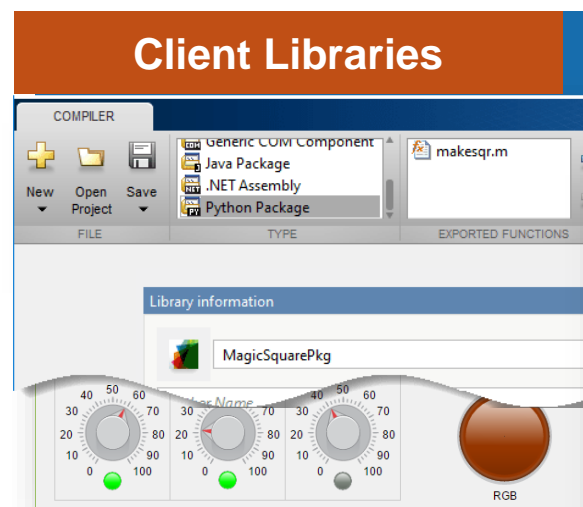
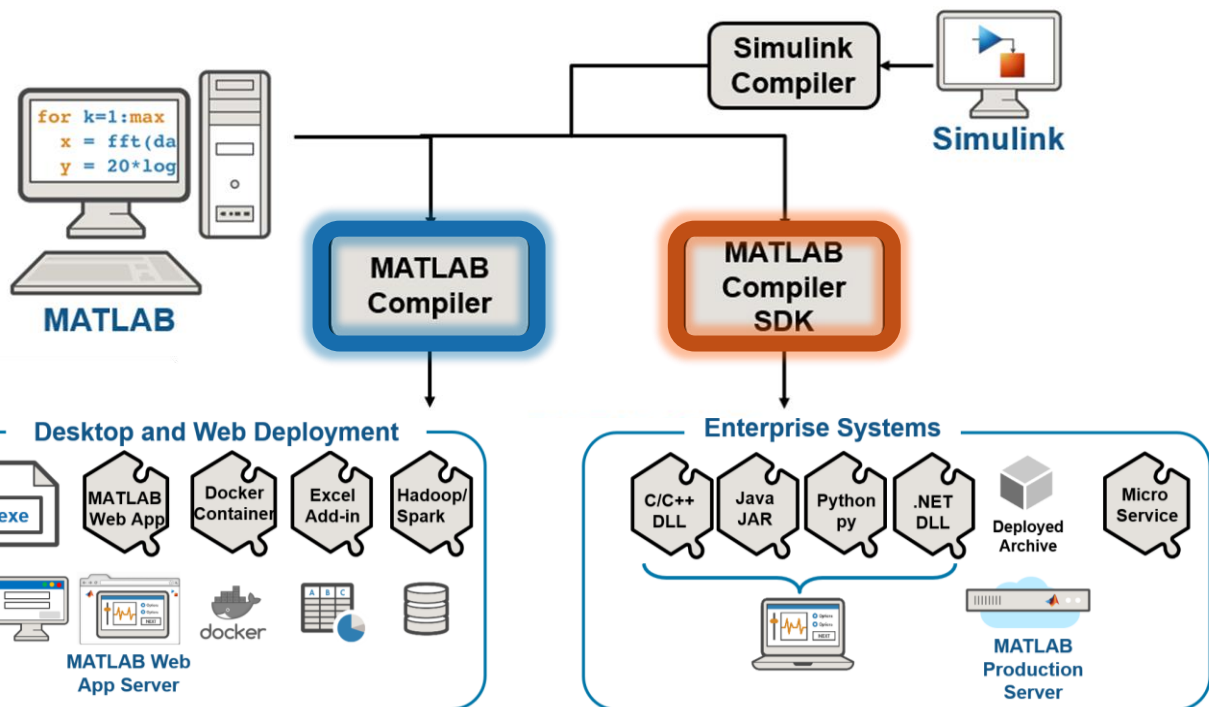


MACHINE LEARNING AND CLOUD FOR EV SYSTEM DEVELOPMENT

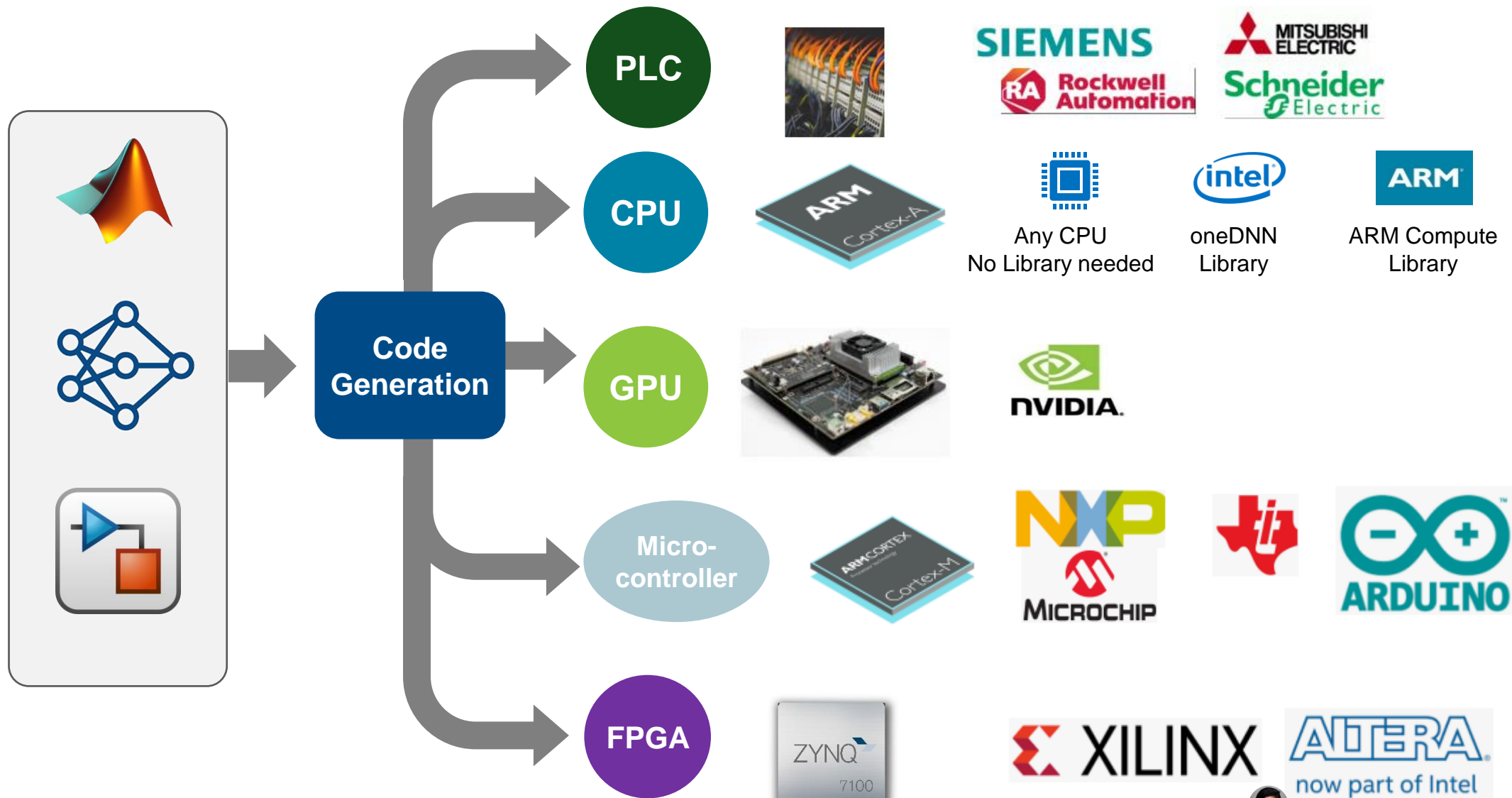


V.Venkobarao,A.Konstantin,M.Wutz, M.Khan,P.Patil,S.Pittan /
Vitesco Technologies /External

AI Deployment on Enterprise Systems



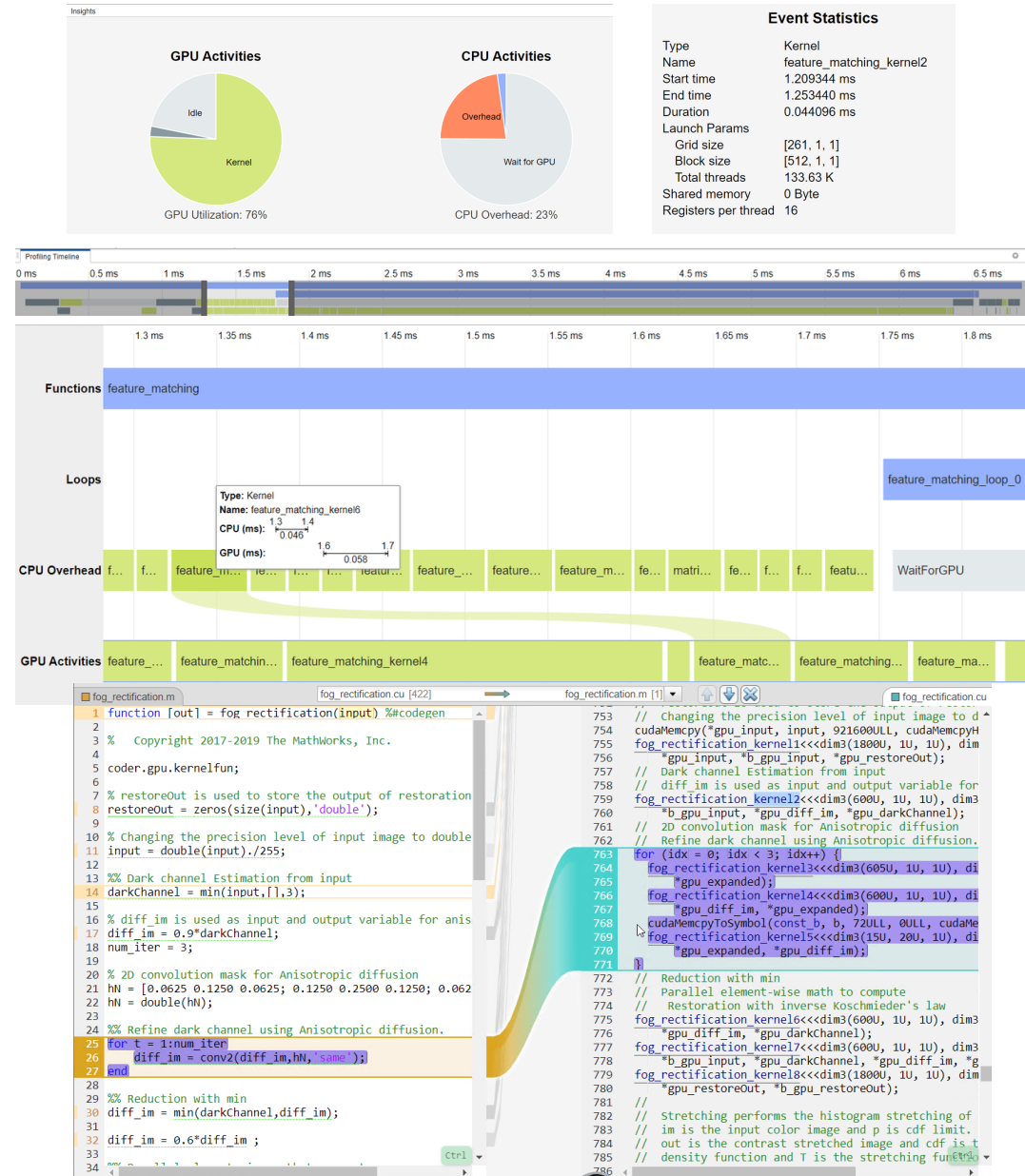
One Codebase – Many Embedded Deployment targets



How to Optimize Further?

Profiling and Bidirectional Traceability Tools

- Use the GPU Coder Performance Analyzer to profile the generated CUDA code
 - Identify bottlenecks & opportunities to optimize performance
- Use bidirectional traceability to map to/from CUDA code back to MATLAB code
 - Helps you to understand how CUDA kernels are created from your MATLAB algorithms



Analyze and Find Issues in Deep Learning Network for Code Generation

- Use [analyzeNetworkForCodegen](#) function
- Detects issues
 - including unsupported layers for code generation, network issues, built-in layer-specific issues, and issues with custom layers
- Requires [MATLAB Coder Interface for Deep Learning Libraries](#) and [GPU Coder Interface for Deep Learning Libraries](#) Support Packages
- Analyze Network for Code Generation



```
targetLibraries = {'none','arm-compute','arm-compute-mali',...
    'mklldnn','cmsis-nn','cudnn','tensorrt'};
S = analyzeNetworkForCodegen(dlnet,TargetLibrary = targetLibraries);
```

	Supported	NetworkDiagnostics	
none	"Yes"	" "	" "
arm-compute	"Yes"	" "	" "
arm-compute-mali	"No"	"Found 1 issue(s). View network diagnostics."	"Found 2 unsupported layer type(s). View network diagnostics."
mklldnn	"Yes"	" "	" "
cmsis-nn	"No"	"Found 1 issue(s). View network diagnostics."	"Found 2 unsupported layer type(s). View network diagnostics."
cudnn	"Yes"	" "	" "
tensorrt	"Yes"	" "	" "

S(5).LayerDiagnostics

```
ans=4x3 table
```

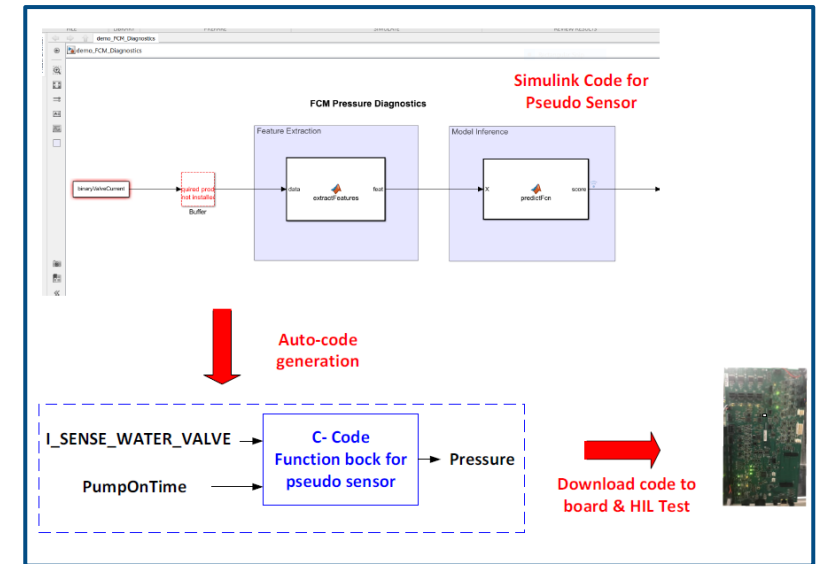
LayerName	LayerType	Diagnostics
"add"	"AdditionLayer"	"Unsupported layer type."
"relu_body1"	"ReLULayer"	"Unsupported layer type."
"relu_body3"	"ReLULayer"	"Unsupported layer type."
"output2"	"ReLULayer"	"Unsupported layer type."

Coca-Cola Develops Virtual Pressure Sensor with Machine Learning to Improve Beverage Dispenser Diagnostics

Using MATLAB and Simulink, Coca-Cola designed and deployed a machine learning algorithm that serves as a virtual pressure sensor, improving field diagnostics and eliminating the need to retrofit 10,000 Freestyle beverage dispensers with costly sensors.

Key Outcomes/Advantages:

- Transformed a standard flow control module into a diagnostics-capable smart component
- Eliminated the need to retrofit thousands of existing dispensers with costly sensors
- Achieved up to 91% accuracy in pressure predictions



Modeling, deployment, and testing of a virtual (“pseudo”) pressure sensor.

With the help of MathWorks, the team was able to reduce the footprint of this code so that it will fit nicely in the ARM-Cortex M microprocessor. It has transformed the flow control module into a smart component.

[Link to user story](#)

Deploying AI is difficult

Four specific challenges

- **Integrate AI** model with an Embedded systems
- Fit large AI models on **limited hardware memory**
- **Error-free Code** Generation
- Achieving **Real-Time Performance** post-deployment

AI Compression Techniques

Minimize model size and speed up inference for deploying AI models

Reduce memory and power needs of deployed models

Quantization

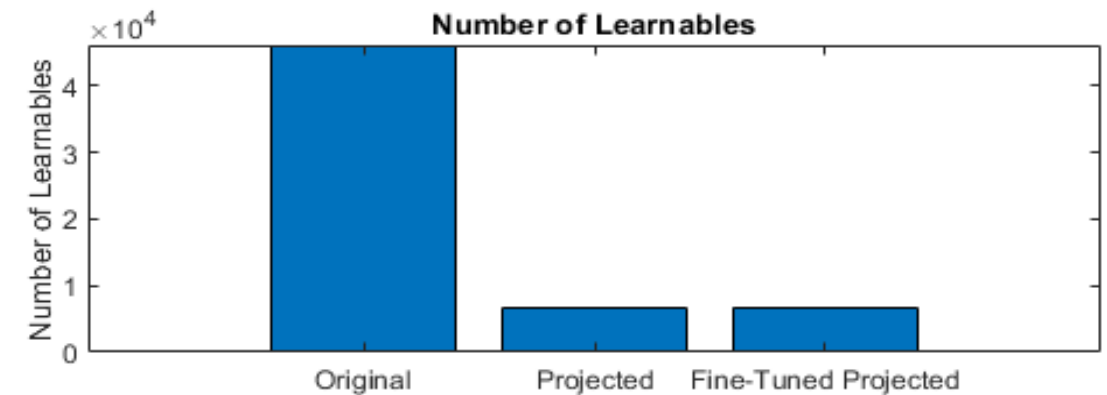
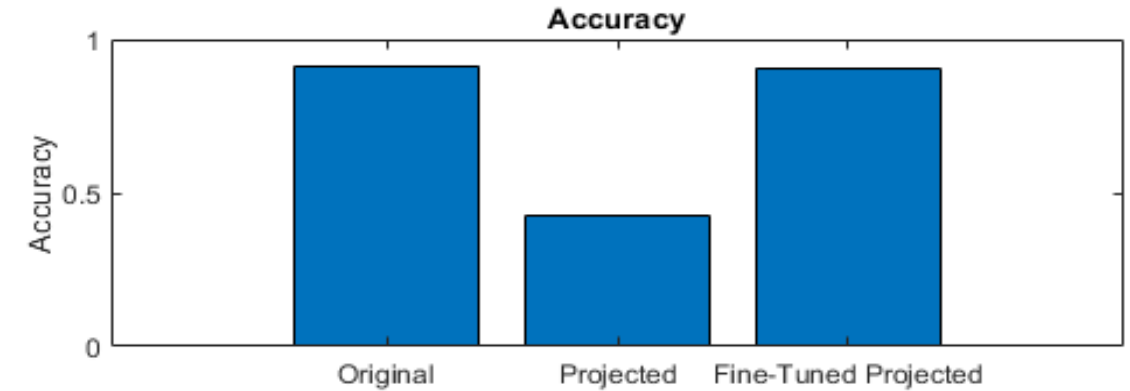
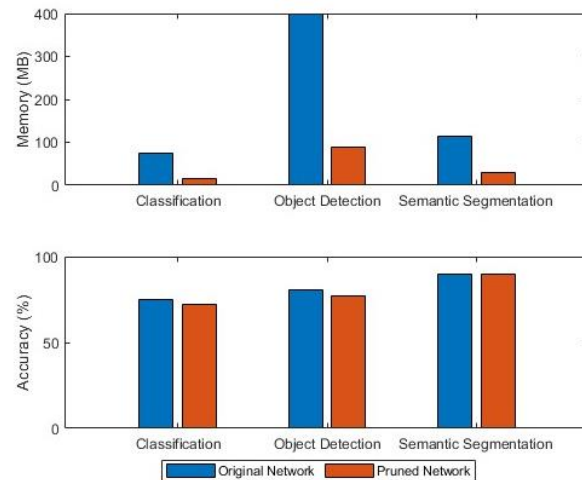
- Convert learnable from floating point to fixed point

Pruning

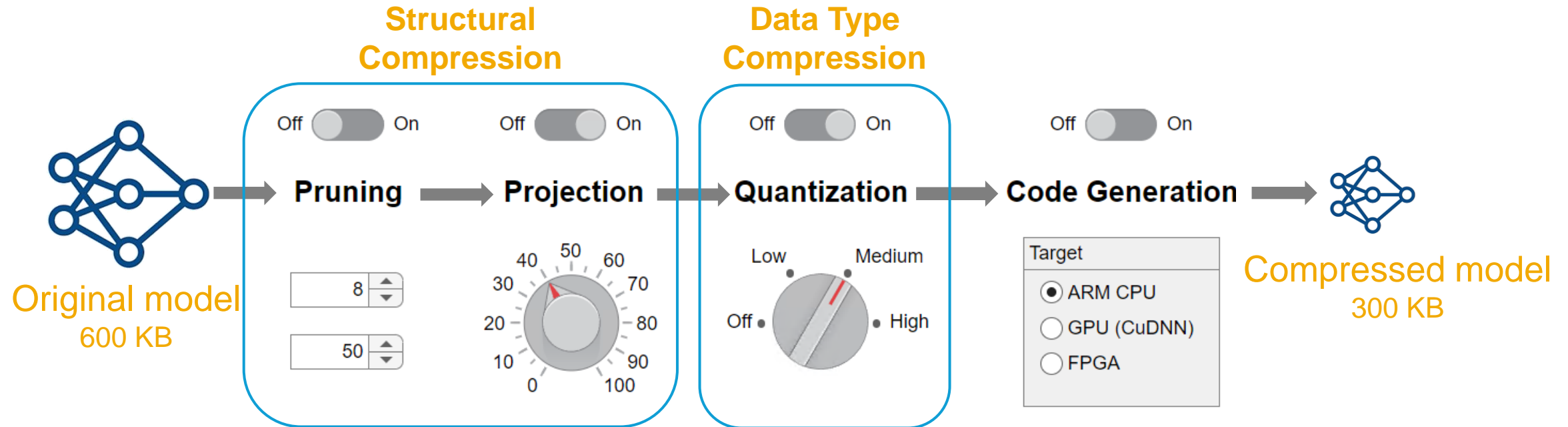
- Remove unimportant parts of the network

Projection

- Perform principal component analysis to identify redundancies



Reduce model footprint and accelerate inference of DL models for deployment to the edge



*“original [network] was 40MB, was told needed to be **less than 10MB to fit.**”*



*“model is 600kb and want to **reduce it to 300kb.** if I'm not fitting it in, I don't have a working solution”*



MathWorks ✓

@MathWorks

Share the EXPO experience
#MATLABEXPO



MATLAB EXPO



INDIA



© 2024 The MathWorks, Inc. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See [mathworks.com/trademarks](https://www.mathworks.com/trademarks) for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

