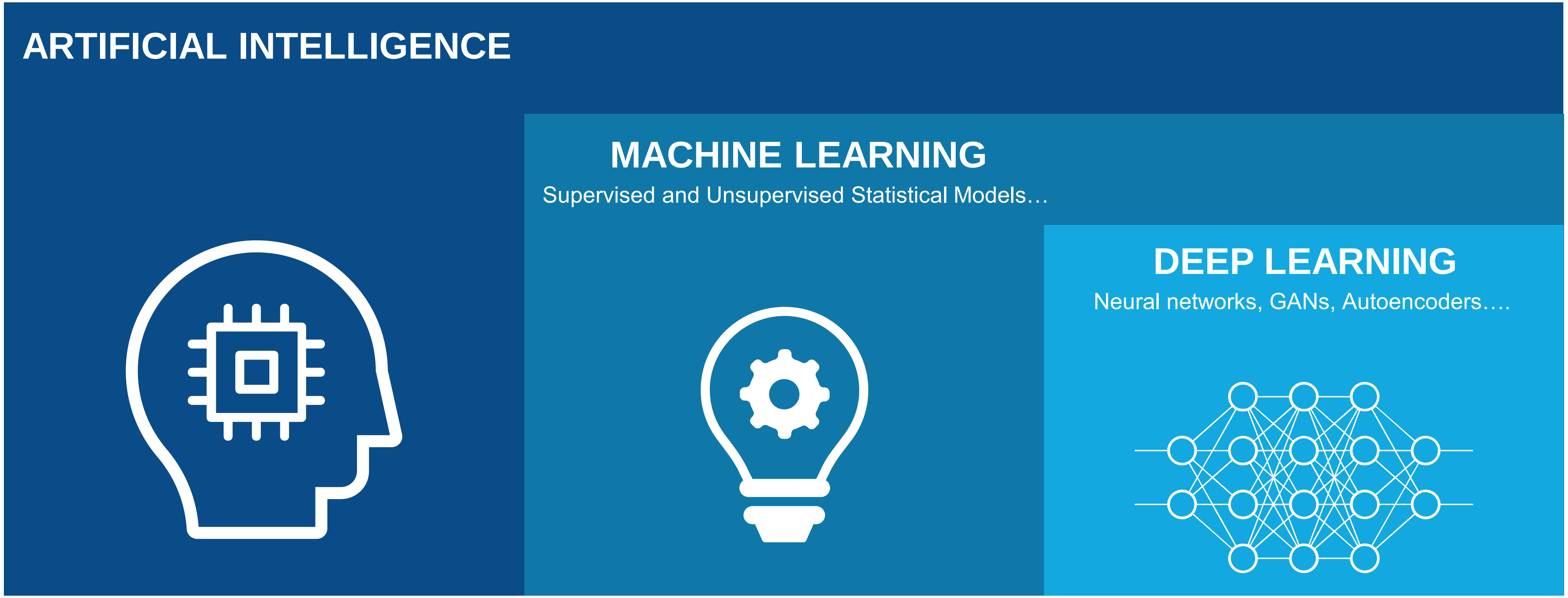
MATLAB EXPO

# Building AI applications for Signals and Time-Series Data

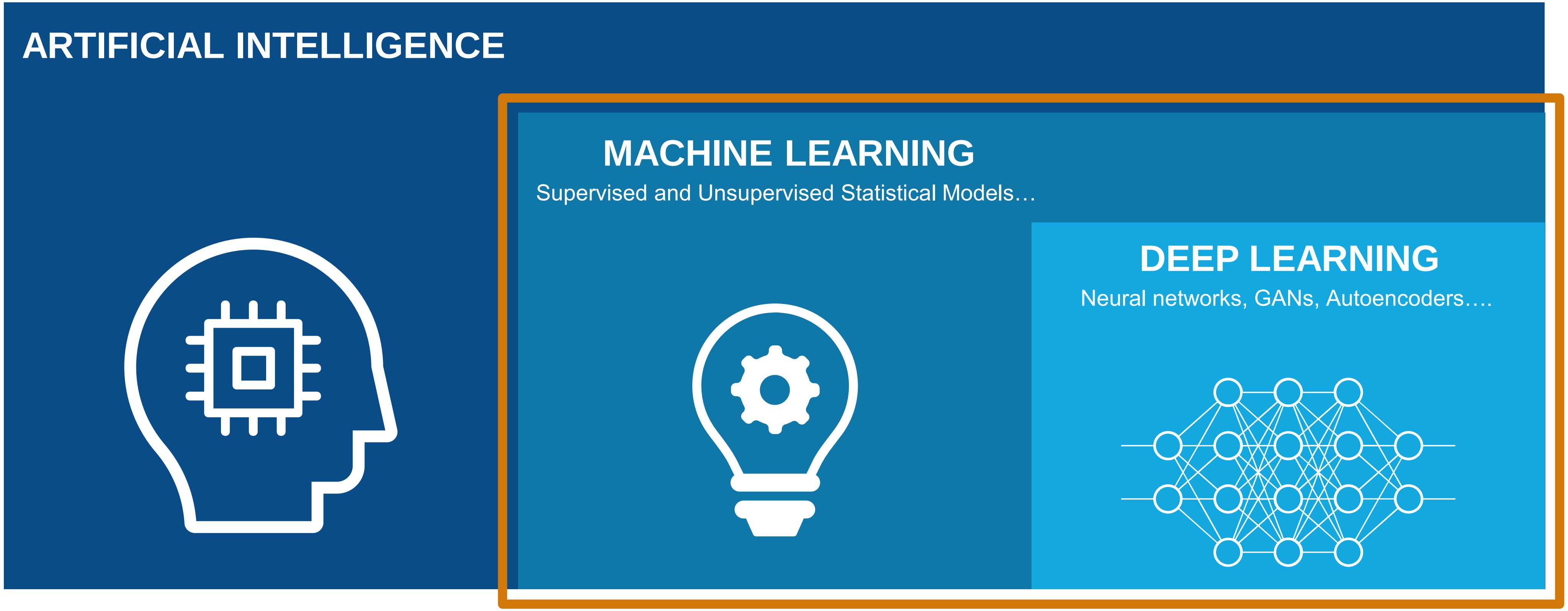
*Esha Shah, MathWorks*
*Francis Tiong, MathWorks*

MathWorks®

# Machine Learning and Deep learning have grown rapidly over the last decade



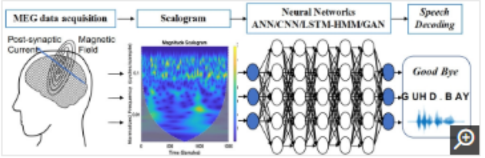
# Machine Learning and Deep learning have grown rapidly over the last decade



# Use of AI in signal processing applications is growing rapidly

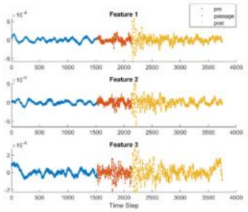
## UT Austin Researchers Convert Brain Signals to Words and Phrases Using Wavelets and Deep Learning

> "MATLAB is an industry-standard tool, and one that you can trust. It is easier to learn than other languages, and its toolboxes help you get started in new areas because you don't have to start from scratch."
>
> — *Dr. Jun Wang, UT Austin*




Classifying the brain signals corresponding to the imagined word "goodbye" using feature extraction and deep neural networks.

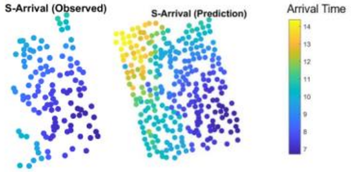
## Shell performs Seismic Event Detection with Deep Learning




**Challenges**
- Terabytes of passive seismic data from geophones
- Traditional methods time/labor intensive (5 months &~ $100K)
- Event detection inconsistent/unreliable in 'low' signal to noise records

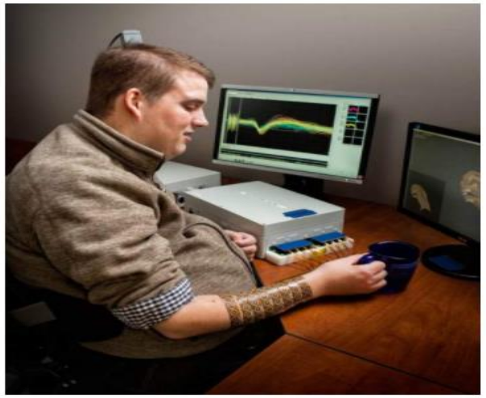
**Solution**
- Train LSTM network to detect P-wave and S-wave arrivals via sequence-to-sequence classification

**Results**
- >98% accuracy for arrival prediction
- Networks generalizes to other data (sites, source mechanisms)

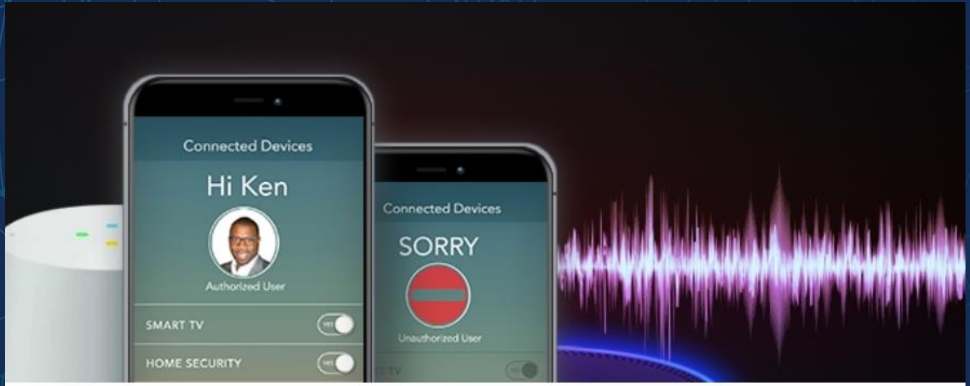



## Battelle Neural Bypass Technology Restores Movement to a Paralyzed Man's Arm and Hand

> "The algorithms we developed using MATLAB gave the participant back basic control of his arm and hand. By the end of the study, he could grip a bottle, pour out its contents, and set it down, as well as pick up a stir stick and execute a stirring motion."
>
> — *David Friedenberg, Battelle*



Patient using the Battelle NeuroLife system.



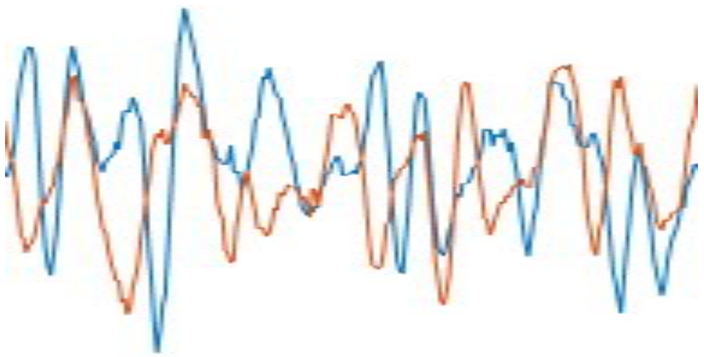

## Voice Interface: The Touchscreen of the Next Century

How AI and Signal Processing Came Together to Track the DNA of Sound

# Modulation Classification of RF waveforms

# Modulation Classification of RF waveforms

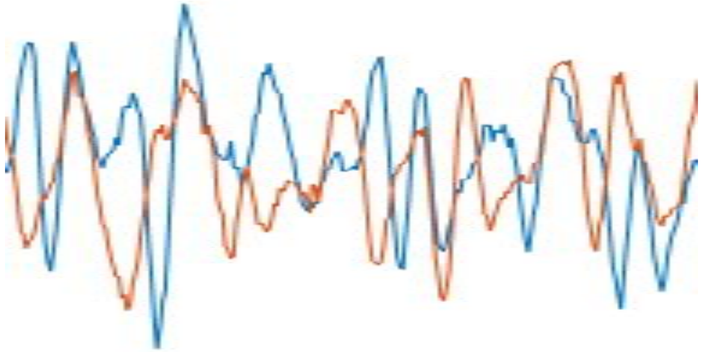
**TRANSMITTER**
(Software Defined Radio)



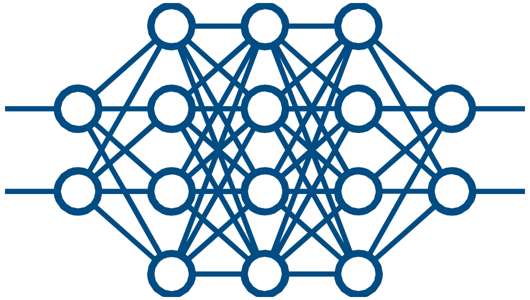
**RECEIVER**
(Software Defined Radio)

# Modulation Classification of RF waveforms

**TRANSMITTER**
(Software Defined Radio)

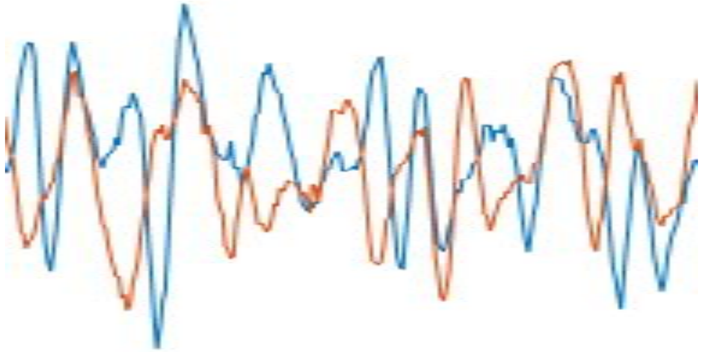


**RECEIVER**
(Software Defined Radio)

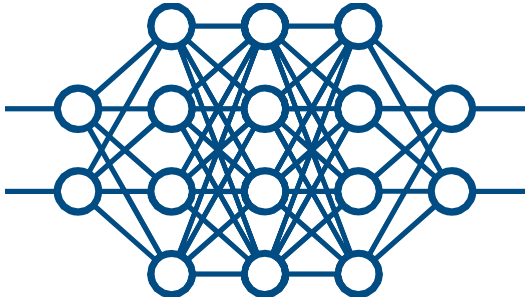


# Modulation Classification of RF waveforms

**TRANSMITTER**
(Software Defined Radio)



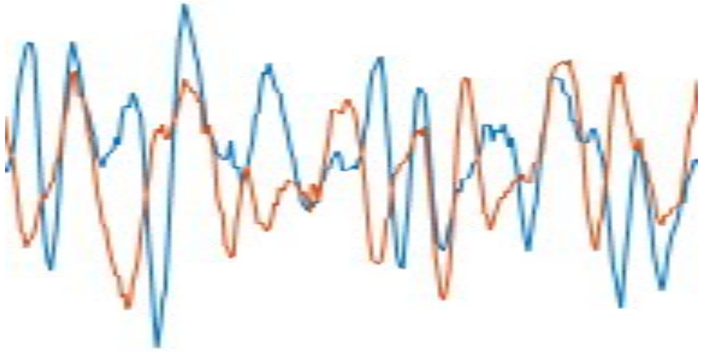
**RECEIVER**
(Software Defined Radio)



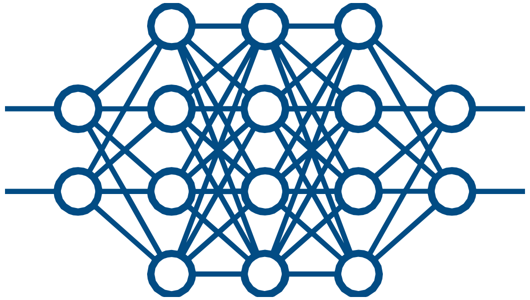
**Modulation Type**

# Modulation Classification of RF waveforms

**TRANSMITTER**
(Software Defined Radio)

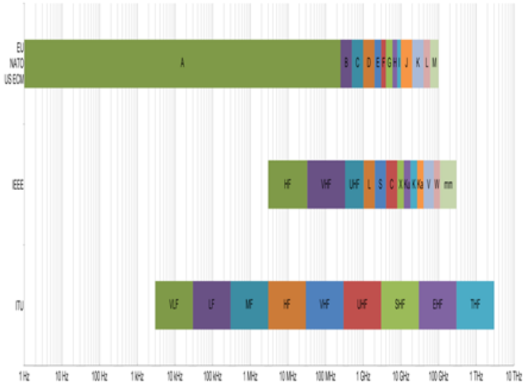


**RECEIVER**
(Software Defined Radio)

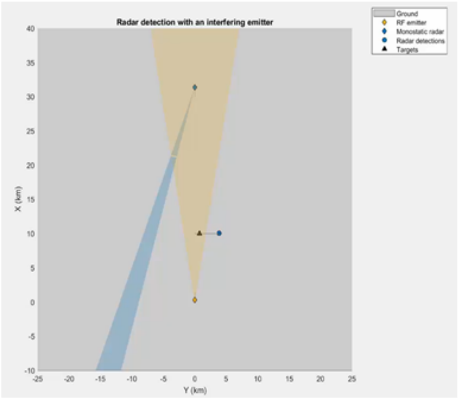


**Modulation Type**

Intelligent Receivers



Spectrum Management




Radar Interference Detection

# AI-driven system design

# AI-driven system design

## Data Preparation

- Data cleansing and preparation
- Human insight
- Simulation-generated data

# AI-driven system design

| Data Preparation | AI Modeling |
|---|---|
| Data cleansing and preparation | Model design and tuning |
| Human insight | Hardware accelerated training |
| Simulation-generated data | Interoperability |

# AI-driven system design

| Data Preparation | AI Modeling | Deployment |
|---|---|---|
| Data cleansing and preparation | Model design and tuning | Embedded devices |
| Human insight | Hardware accelerated training | Enterprise systems |
| Simulation-generated data | Interoperability | Edge, cloud, desktop |

# AI-driven system design

## Data Preparation

- Data cleansing and preparation
- Human insight
- Simulation-generated data

## AI Modeling

- Model design and tuning
- Hardware accelerated training
- Interoperability

## Deployment

- Embedded devices
- Enterprise systems
- Edge, cloud, desktop

**Iteration and Refinement**

# Preparing and labelling data

## Data Preparation

- Data cleansing and preparation
- Human insight
- Simulation-generated data

# Preparing and labelling data

**Data Preparation**

- Data cleansing and preparation
- Human insight
- Simulation-generated data

Q. How to label collected data?

# Preparing and labelling data

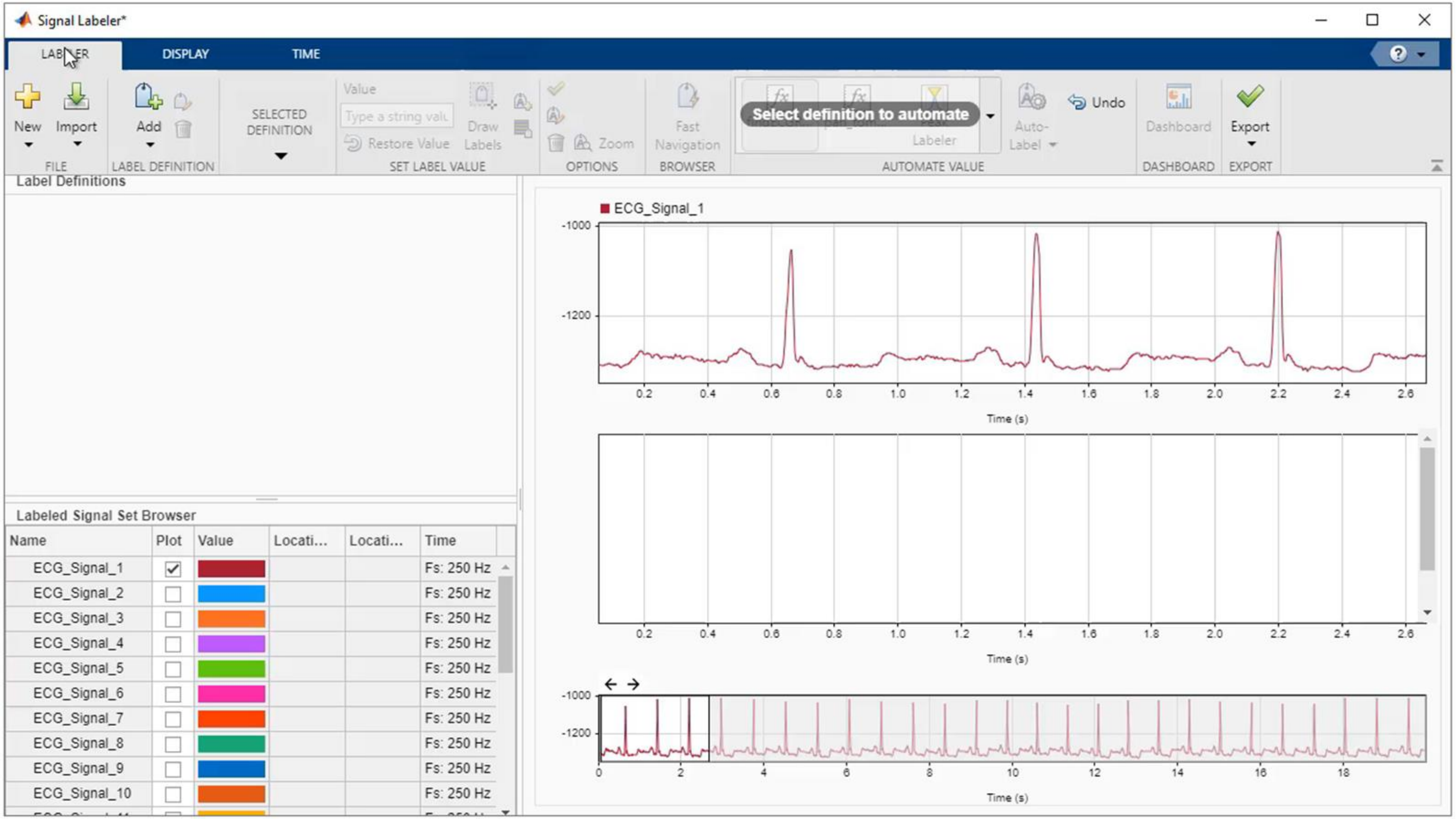
**Data Preparation**

- Data cleansing and preparation
- Human insight
- Simulation-generated data

Q. How to label collected data?

Q. What if it is not possible to collect data?

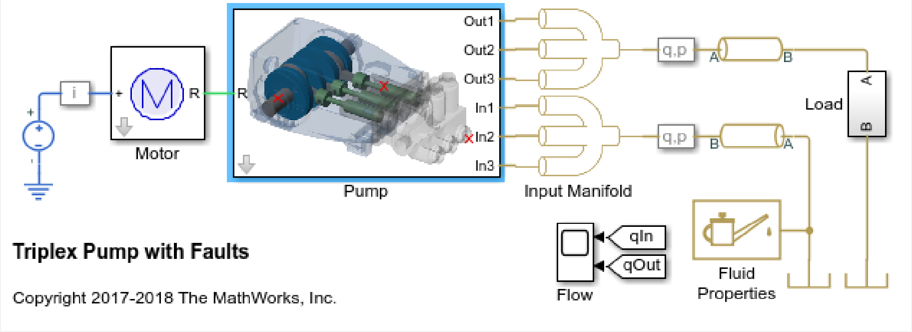
# Labeling Signals with Signal Labeler App



# Generate Synthetic Data for various applications in MATLAB

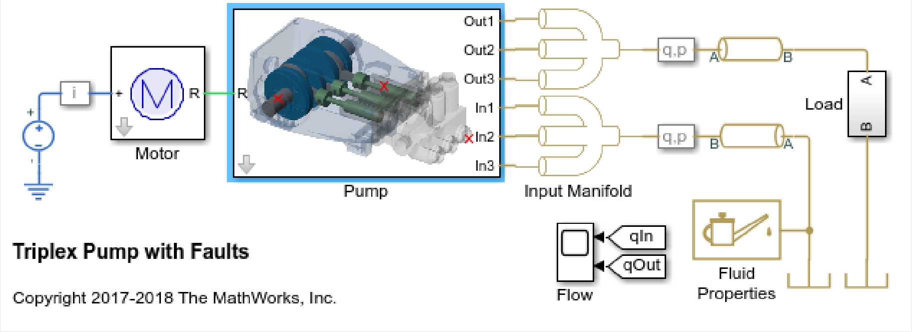
# Generate Synthetic Data for various applications in MATLAB

## Simulate data using Simulink models

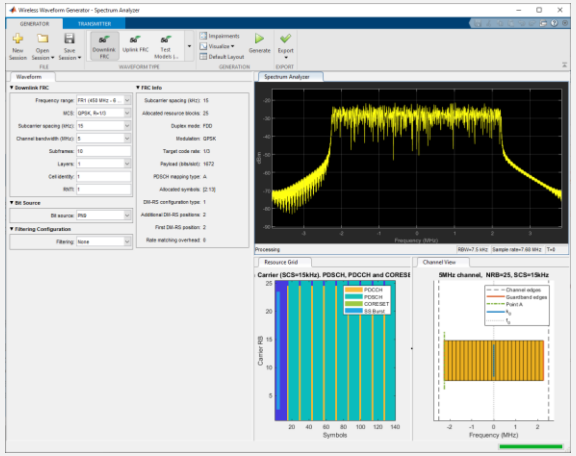


# Generate Synthetic Data for various applications in MATLAB

## Simulate data using Simulink models

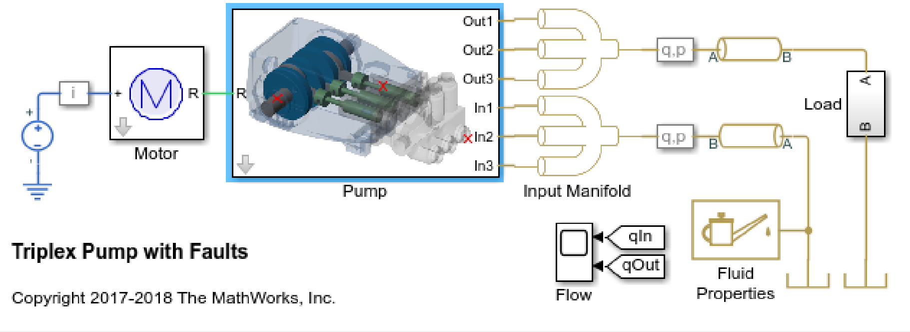



## Generate wireless waveforms

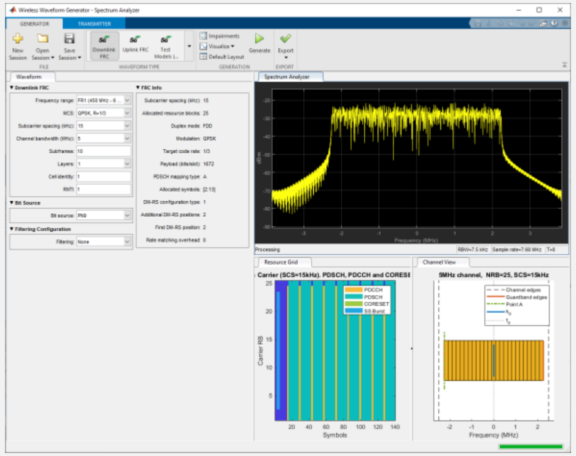


# Generate Synthetic Data for various applications in MATLAB

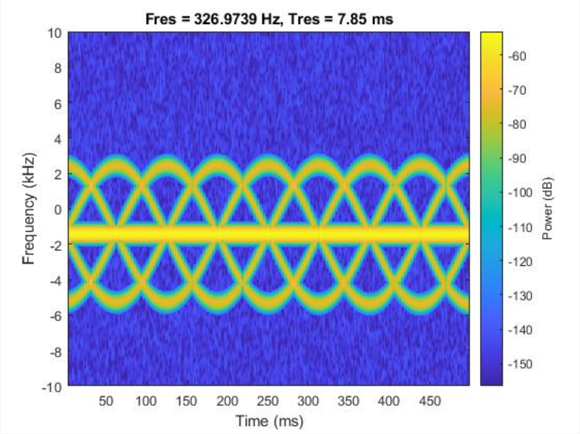
## Simulate data using Simulink models




## Generate wireless waveforms

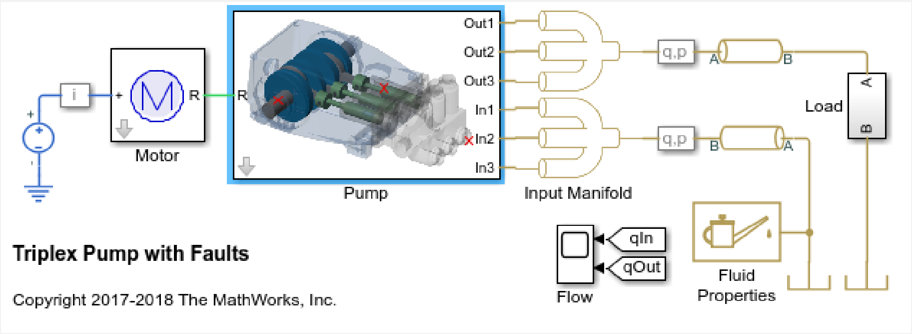


## Generate Radar Returns

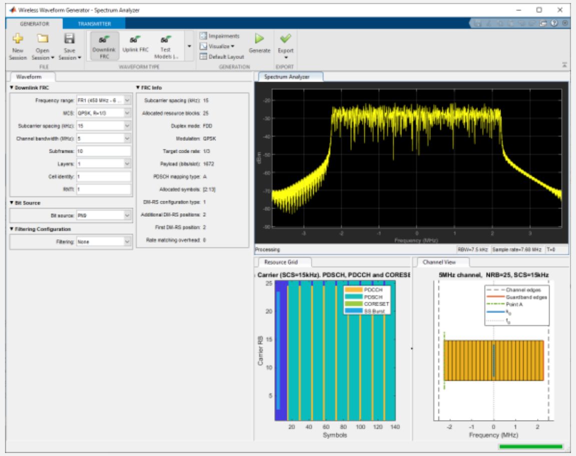


# Generate Synthetic Data for various applications in MATLAB

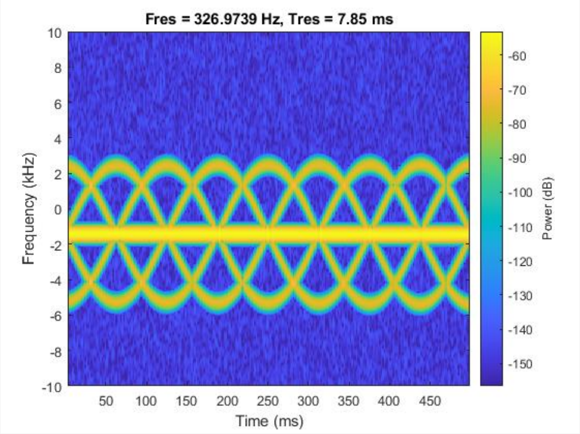
## Simulate data using Simulink models




## Generate wireless waveforms

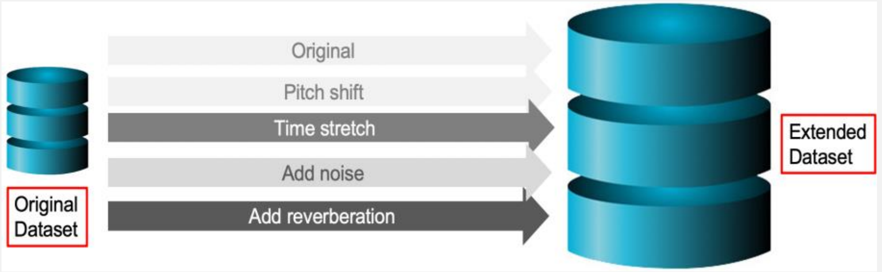


## Generate Radar Returns




## Generate and Augment Audio Data

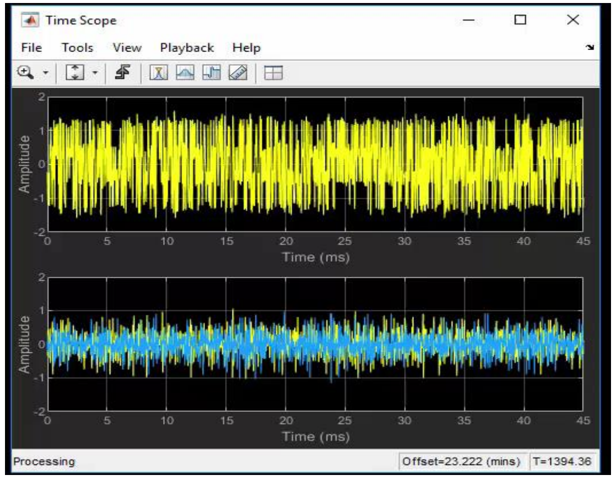
`text2speech`



# Generation of wireless communication waveforms with impairments

# Generation of wireless communication waveforms with impairments

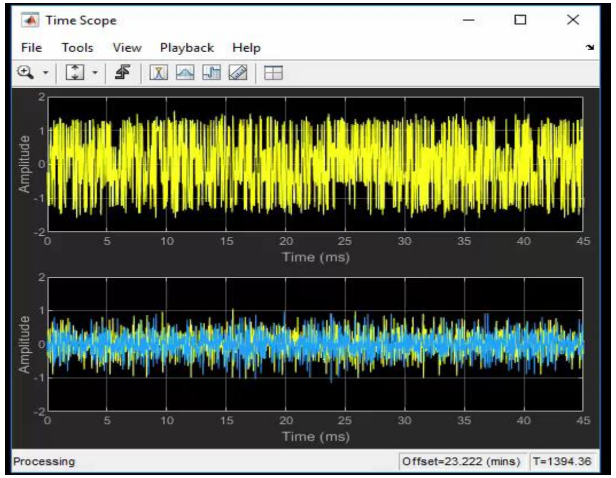
- Modulate digital baseband signals using built-in functions
  - BPSK, QPSK, 8PSK, FM, DSB-AM, SSB-AM, GFSK,PAM4



# Generation of wireless communication waveforms with impairments

- Modulate digital baseband signals using built-in functions
  - BPSK, QPSK, 8PSK, FM, DSB-AM, SSB-AM, GFSK,PAM4

- Easily account for various impairments
  - RF / Hardware impairments (Frequency/ Phase Offsets etc. )
  - Channel Impairments (Multipath Fading Channels)




**Rician Multipath**

```
multipathChannel = comm.RicianChannel(...
  'SampleRate', fs, ...
  'PathDelays', [0 1.8 3.4]/fs, ...
  'AveragePathGains', [0 -2 -10], ...
  'KFactor', 4, ...
  'MaximumDopplerShift', 4)
```

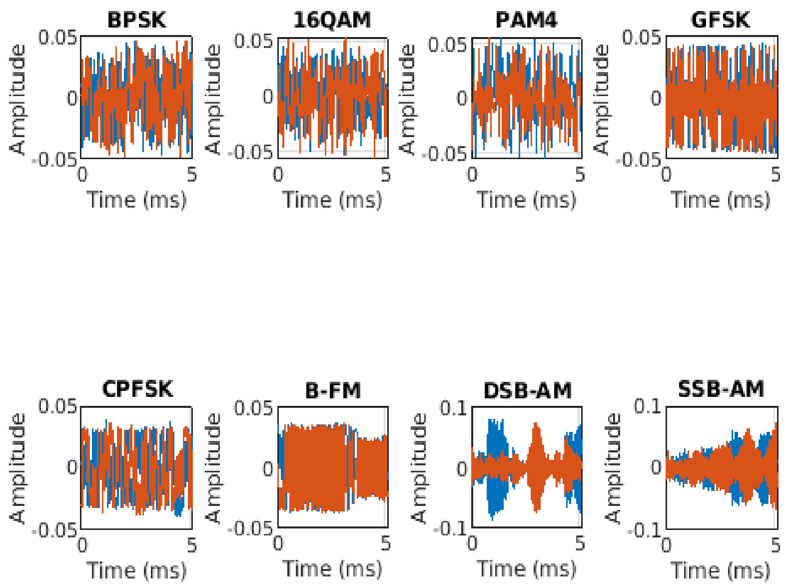
```
multipathChannel =
  comm.RicianChannel with properties:

                SampleRate: 200000
                PathDelays: [0 9.0000e-06 1.7000e-05]
          AveragePathGains: [0 -2 -10]
        NormalizePathGains: true
                   KFactor: 4
    DirectPathDopplerShift: 0
    DirectPathInitialPhase: 0
       MaximumDopplerShift: 4
           DopplerSpectrum: [1x1 struct]

  Show all properties
```

# Generation of wireless communication waveforms with impairments

- Modulate digital baseband signals using built-in functions
  - BPSK, QPSK, 8PSK, FM, DSB-AM, SSB-AM, GFSK,PAM4
- Easily account for various impairments
  - RF / Hardware impairments (Frequency/ Phase Offsets etc. )
  - Channel Impairments (Multipath Fading Channels)
- Generate Datasets for Deep Learning
  - 5000 frames generated for each modulation type
  - 80% data – Training; 10% data – Validation; 10% data - Test



# Feature Extraction

## Data Preparation

- Data cleansing and preparation
- Human insight
- Simulation-generated data

# Feature Extraction

**Data Preparation**

- Data cleansing and preparation
- Human insight
- Simulation-generated data

Q. Can I use raw data?

# Feature Extraction

**Data Preparation**

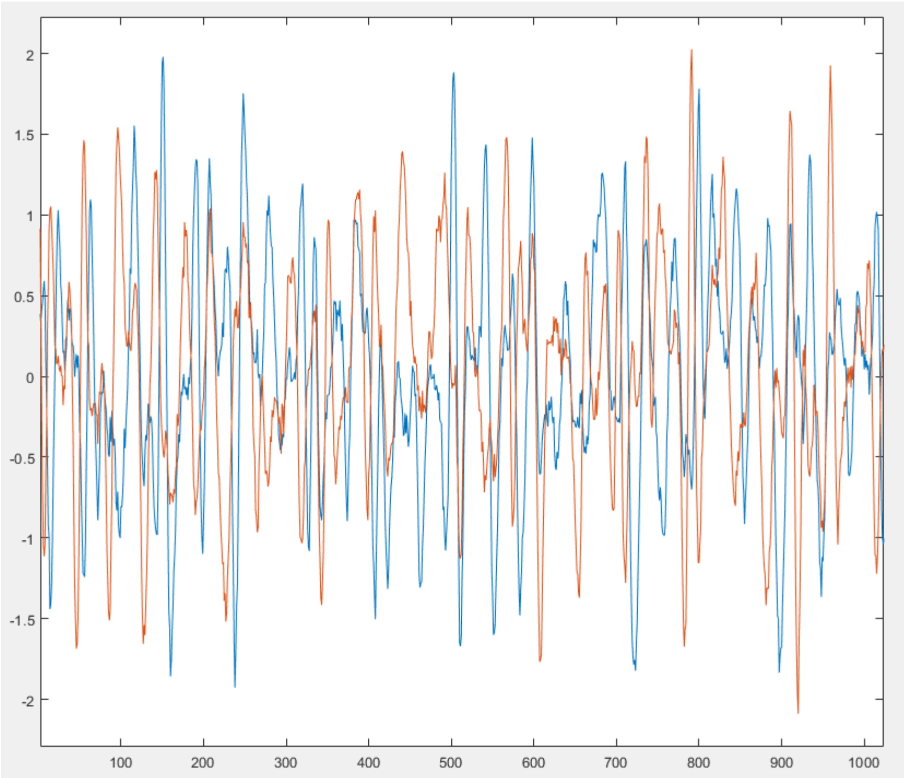
- Data cleansing and preparation
- Human insight
- Simulation-generated data

Q. Can I use raw data?

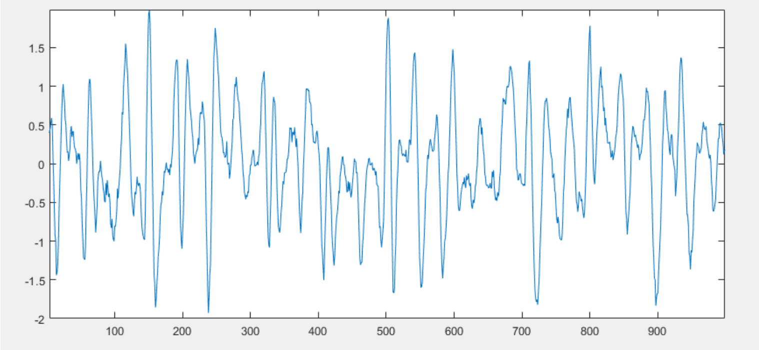
Q. How do I extract the right features for my data?

# Use of raw data for AI models

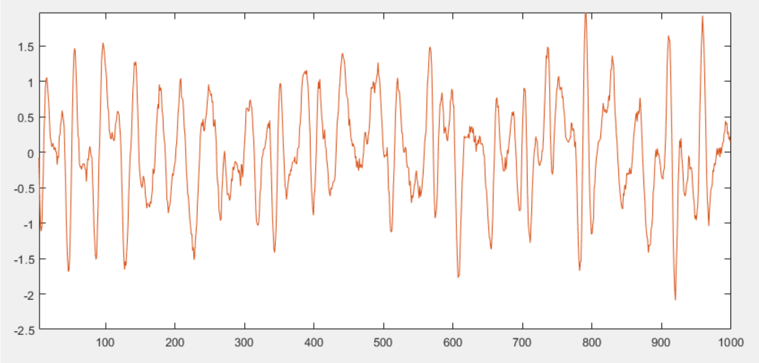
# Use of raw data for AI models




IQ waveform

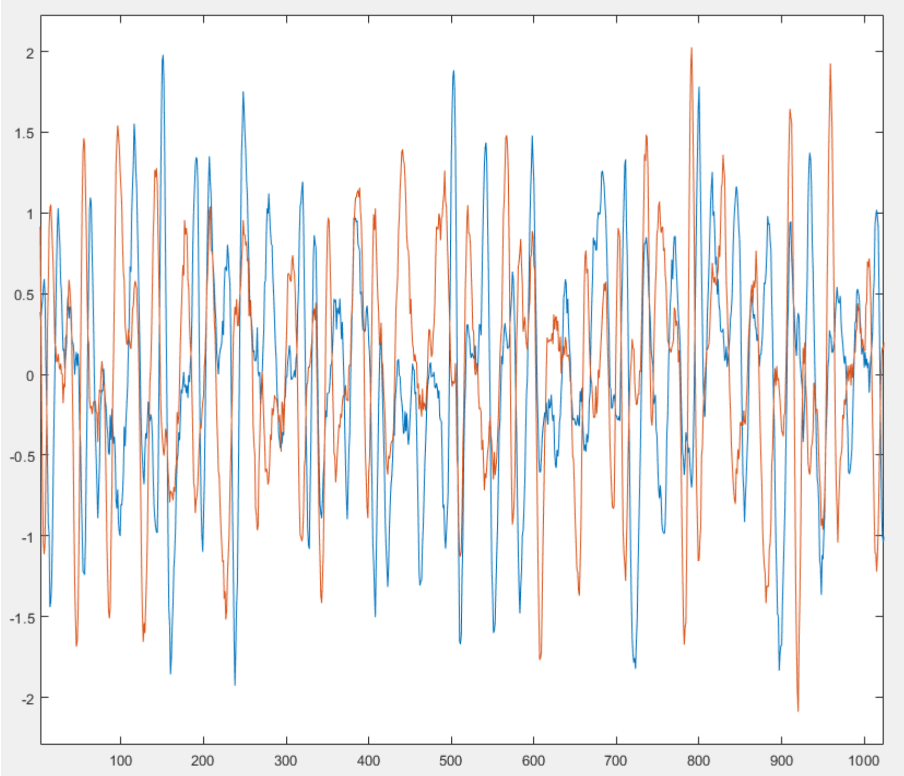



I waveform

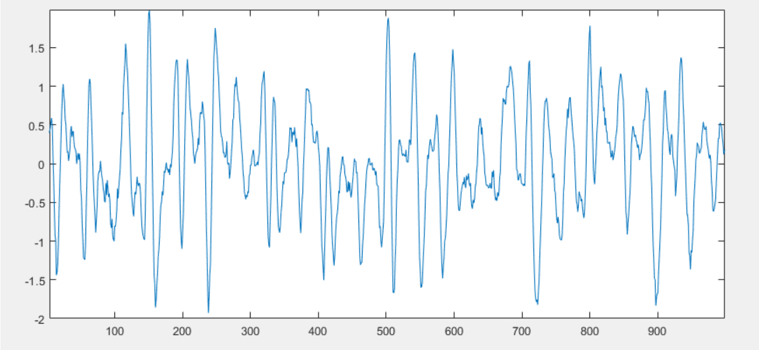



Q waveform

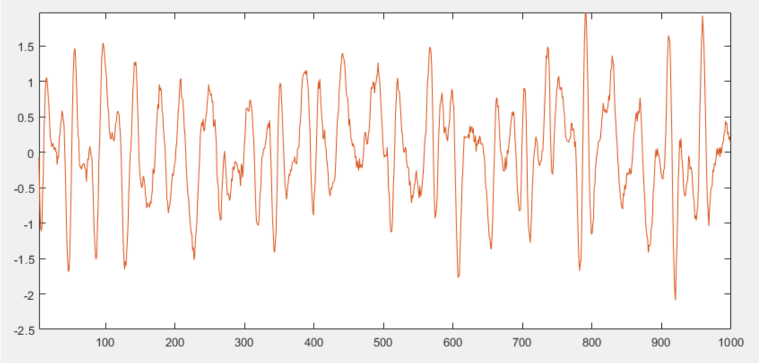
# Use of raw data for AI models




IQ waveform




I waveform




Q waveform

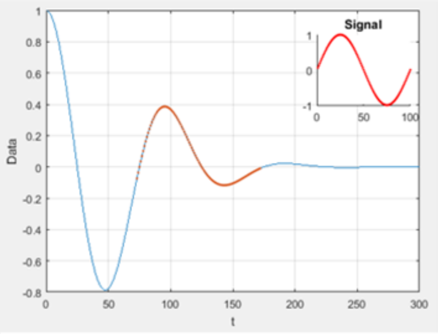
## Challenges with Raw Data

- High Dimensionality
- Need for more data
- Need for specialized models

# Feature extraction with signal processing techniques

# Feature extraction with signal processing techniques

## Time-Domain Features

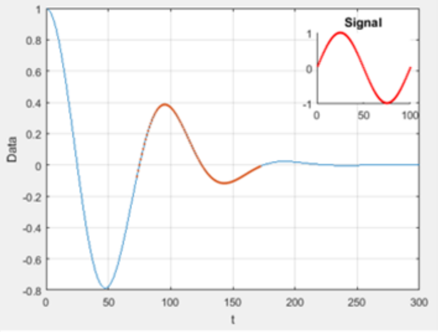



- Signal Patterns
- Changepoints
- Peaks
- Signal Envelope

……

# Feature extraction with signal processing techniques

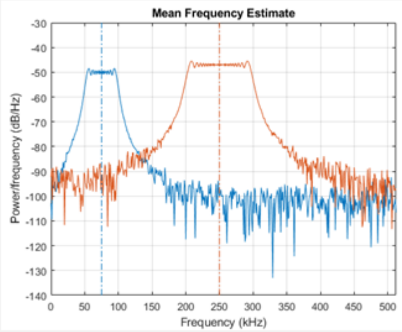
## Time-Domain Features




- Signal Patterns
- Changepoints
- Peaks
- Signal Envelope

……

## Frequency-Domain Features

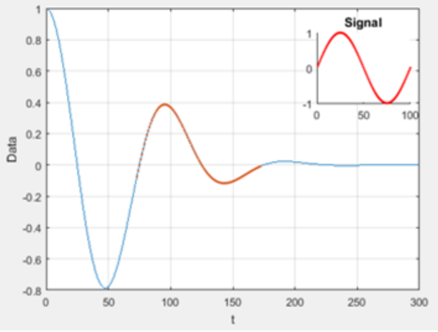



- BW measurements
- Spectral Statistics
- Octave Spectrum

……

# Feature extraction with signal processing techniques

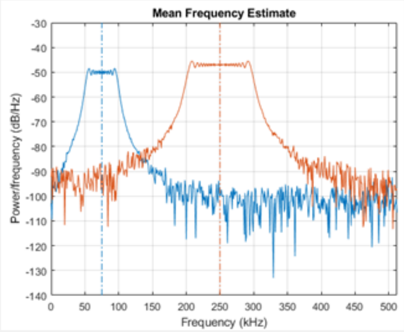
## Time-Domain Features




- Signal Patterns
- Changepoints
- Peaks
- Signal Envelope

……

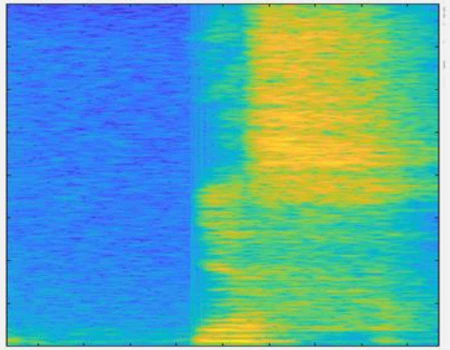
## Frequency-Domain Features




- BW measurements
- Spectral Statistics
- Octave Spectrum

……

## Time-Frequency features

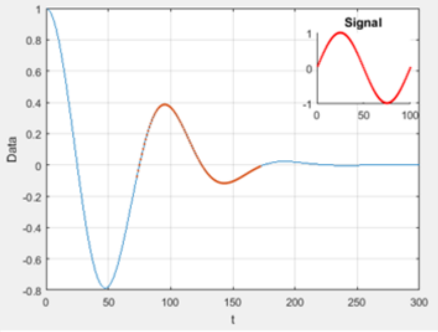


- STFT
- CWT
- Constant-Q Transform

……

# Feature extraction with signal processing techniques

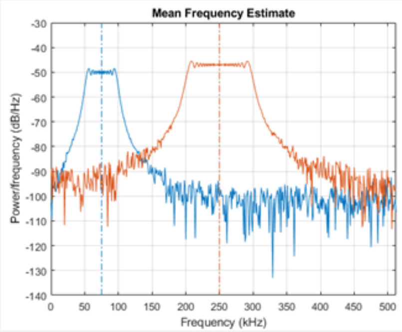
## Time-Domain Features




- Signal Patterns
- Changepoints
- Peaks
- Signal Envelope

……

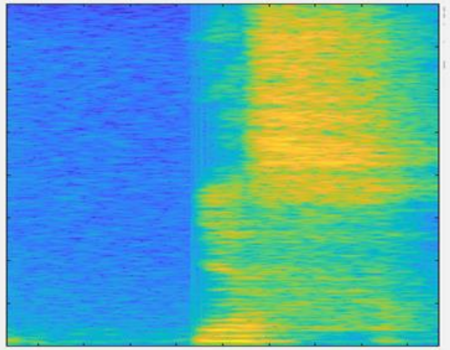
## Frequency-Domain Features




- BW measurements
- Spectral Statistics
- Octave Spectrum

……

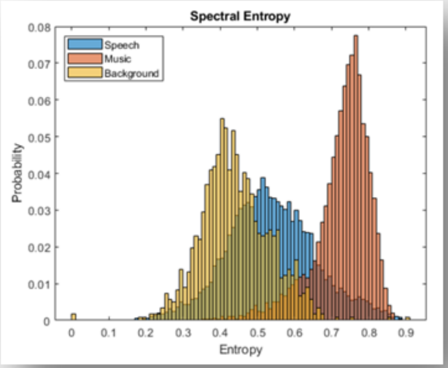
## Time-Frequency features



- STFT
- CWT
- Constant-Q Transform

……

## Domain-Specific Features




- Speech and audio
- Navigation and Sensor Fusion
- Radar
- Communication

……

# Building the AI models

## AI Modeling

- Model design and tuning
- Hardware accelerated training
- Interoperability

# Building the AI models

**AI Modeling**

- Model design and tuning
- Hardware accelerated training
- Interoperability

Q. How do I select the right model for my application:

# Building the AI models

**AI Modeling**

- Model design and tuning
- Hardware accelerated training
- Interoperability

Q. How do I select the right model for my application:

- If I do not have enough data?
- If I do not have domain expertise?
- If I need an easily interpretable model?

…..

# Start by using published literature and MATLAB examples

# Start by using published literature and MATLAB examples

Deep Neural Network Architectures for Modulation Classification

Xiaoyu Liu, Diyu Yang, and Aly El Gamal
School of Electrical and Computer Engineering
Purdue University
Email: {liu1962, yang1467, elgamala}@purdue.edu

*Abstract*—In this work, we investigate the value of employing deep learning for the task of wireless signal modulation recognition. Recently in [1], a framework has bee[n introduced] by generating a dataset using GNU radio that [mimics] imperfections in a real wireless channel, and uses [10 different] modulation types. Further, a convolutional neural ne[twork (CNN)] architecture was developed and shown to deliver [performance] that exceeds that of expert-based approaches. He[re, we follow] the framework of [1] and find deep neural network [architectures] that deliver higher accuracy than the state of the a[rt. We tested] the architecture of [1] and found it to achieve an [accuracy of] approximately 75% of correctly recognizing the mo[dulations.] We first tune the CNN architecture of [1] and f[ind a design] with four convolutional layers and two dense lay[ers that gives] an accuracy of approximately 83.8% at high SN[R. We then] develop architectures based on the recently introd[uced ideas of] Residual Networks (ResNet [2]) and Densely Connec[ted Networks] (DenseNet [3]) to achieve high SNR accuracies of a[pproximately] 83.5% and 86.6%, respectively. Finally, we introd[uce a Convo]lutional Long Short-term Deep Neural Network (C[LDNN] to) achieve an accuracy of approximately 88.5% at hi[gh SNR.]

arXiv:1712.00443v3 [cs.LG] 5 Jan 2018

Automatic Modulation Recognition Using Wavelet Transform and Neural Networks in Wireless Systems

K. Hassan, I. Dayoub, W. Hamouda & M. Berbineau

2010, Article number: 532898 (2010) | Cite this article
Metrics

Time-Frequency Analysis based Blind Modulation Classification for Multiple-Antenna Systems

Weiheng Jiang[a], Xiaogang Wu[a], Bolin Chen[a], Wenjiang Feng[a], Yi Jin[b]

[a]*School of Microelectronics and Communication Engineering, Chongqing University, Chongqing 400044, China.*
[b]*Xi'an Branch of China Academy of Space Technology, Xi'an 710100, China.*

**Abstract**

Blind modulation classification is an important step to implement cognitive radio networks. The multiple-input multiple-output (MIMO) technique is widely used in military and civil communication systems. Due to the lack of prior information about channel parameters and the overlapping of signals in the MIMO systems, the traditional likelihood-based and feature-based approaches cannot be applied in these scenarios directly. Hence, in this paper, to resolve the problem of blind modulation classification in MIMO systems, the time-frequency analysis method based on the windowed short-time Fourier transform is used to analyse the time-frequency characteristics of time-domain modulated signals. Then the extracted time-frequency characteristics are converted into RGB spectrogram images, and the convolutional neural network based on transfer learning is applied to classify the modulation types according to the RGB spectrogram images. Finally, a decision fusion module is used to fuse the classification results of all the receive antennas. Through simulations, we analyse the classification performance at different signal-to-noise ratios (SNRs), the results indicate that, for the single-input single-output (SISO) network, our proposed scheme can achieve 92.37% and 99.12% average classification accuracy at SNRs of -4 dB and 10 dB, respectively. For the MIMO network, our scheme achieves 80.42% and 87.92% average classification accuracy at -4 dB and 10 dB, respectively. This outperforms the existing classification methods based on baseband signals.

*Keywords:* Time-Frequency Analysis, Blind Modulation Classification, Multiple-Antenna Systems, RGB Spectrogram Image

arXiv:2004.00378v1 [cs.LG] 1 Apr 2020

**1. Introduction**

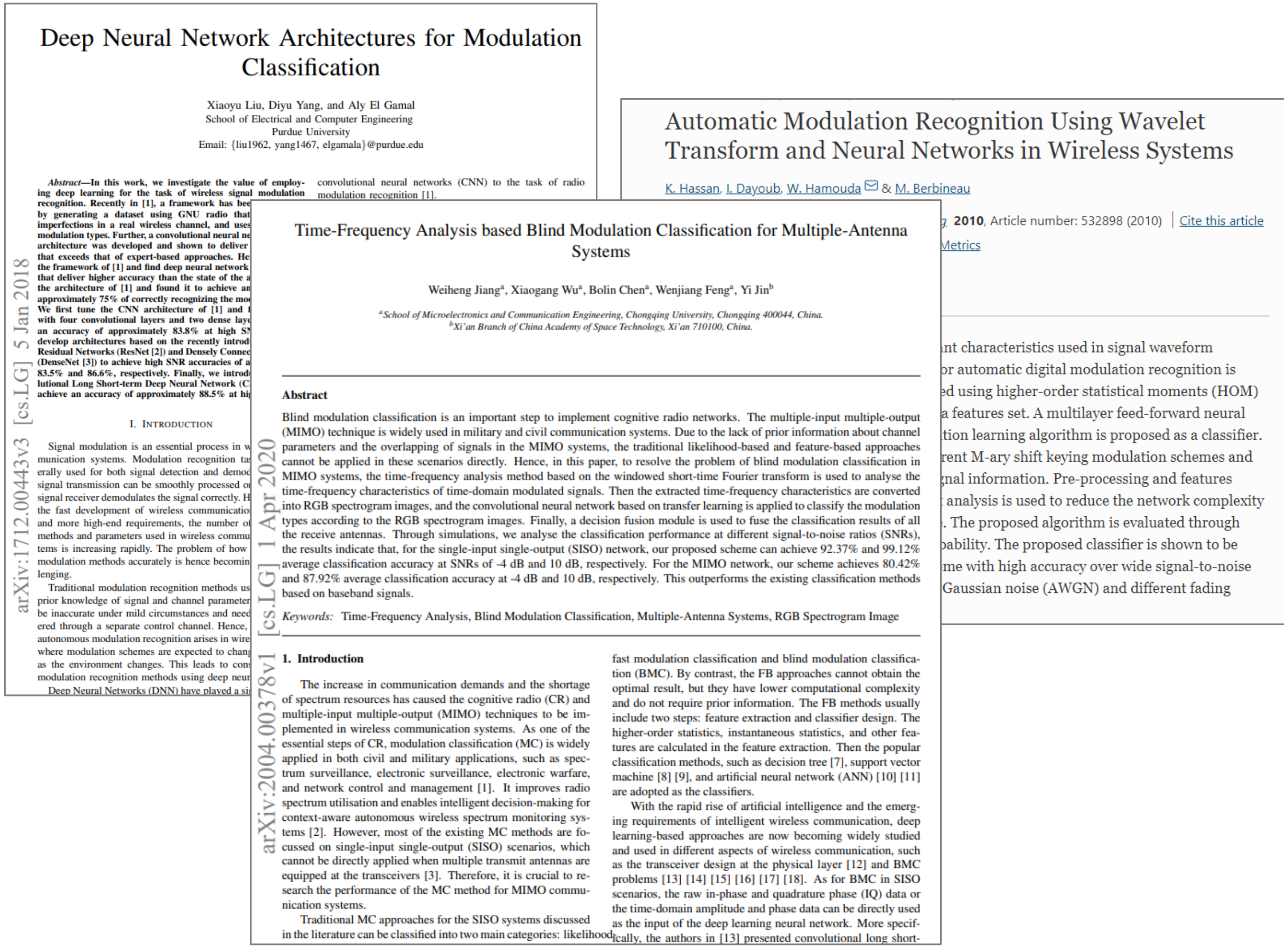
The increase in communication demands and the shortage of spectrum resources has caused the cognitive radio (CR) and multiple-input multiple-output (MIMO) techniques to be implemented in wireless communication systems. As one of the essential steps of CR, modulation classification (MC) is widely applied in both civil and military applications, such as spectrum surveillance, electronic surveillance, electronic warfare, and network control and management [1]. It improves radio spectrum utilisation and enables intelligent decision-making for context-aware autonomous wireless spectrum monitoring systems [2]. However, most of the existing MC methods are focussed on single-input single-output (SISO) scenarios, which cannot be directly applied when multiple transmit antennas are equipped at the transceivers [3]. Therefore, it is crucial to research the performance of the MC method for MIMO communication systems.

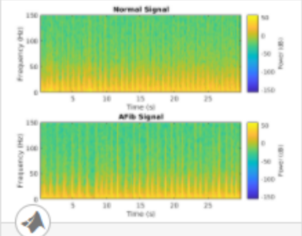
Traditional MC approaches for the SISO systems discussed in the literature can be classified into two main categories: likelihood

fast modulation classification and blind modulation classification (BMC). By contrast, the FB approaches cannot obtain the optimal result, but they have lower computational complexity and do not require prior information. The FB methods usually include two steps: feature extraction and classifier design. The higher-order statistics, instantaneous statistics, and other features are calculated in the feature extraction. Then the popular classification methods, such as decision tree [7], support vector machine [8] [9], and artificial neural network (ANN) [10] [11] are adopted as the classifiers.

With the rapid rise of artificial intelligence and the emerging requirements of intelligent wireless communication, deep learning-based approaches are now becoming widely studied and used in different aspects of wireless communication, such as the transceiver design at the physical layer [12] and BMC problems [13] [14] [15] [16] [17] [18]. As for BMC in SISO scenarios, the raw in-phase and quadrature phase (IQ) data or the time-domain amplitude and phase data can be directly used as the input of the deep learning neural network. More specifically, the authors in [13] presented convolutional long short-

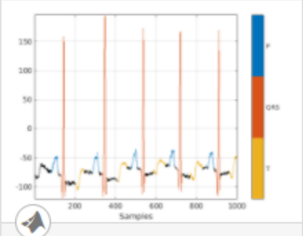
# Start by using published literature and MATLAB examples



**Classify ECG Signals Using Long Short-Term Memory Networks**

Classify heartbeat electrocardiogram data using deep learning and signal processing.

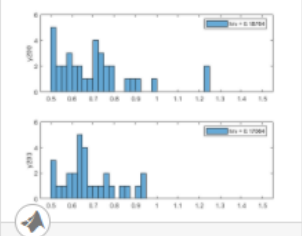


**Waveform Segmentation Using Deep Learning**

Segment human electrocardiogram signals using time-frequency analysis and deep learning.

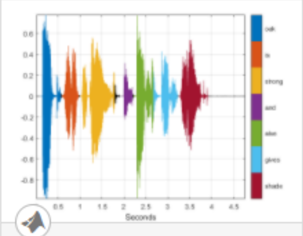
**Iterative Approach for Creating Labeled Signal Sets with Reduced Huma…**

Use deep learning to decrease the human effort required to label signals.



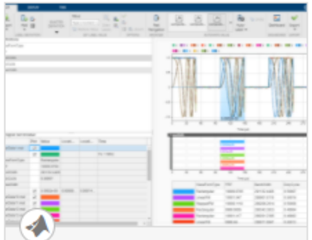
**Label QRS Complexes and R Peaks of ECG Signals Using Deep Learning…**

Use **Signal Labeler** to locate and label QRS complexes and R peaks of ECG signals.



**Label Spoken Words in Audio Signals Using External API**

Use **Signal Labeler** to label spoken words in an audio signal.

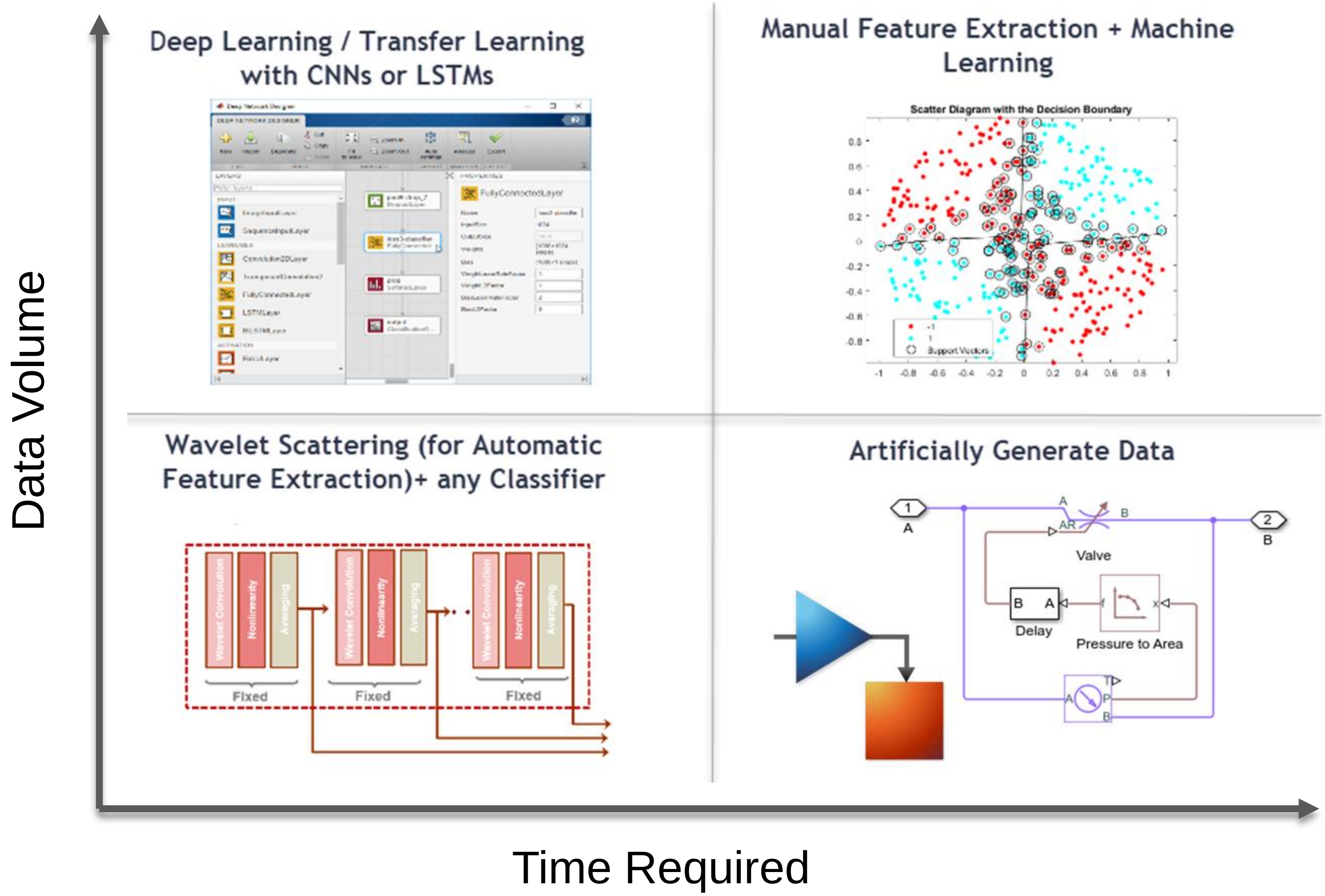


**Labeling Radar Signals with Signal Labeler**

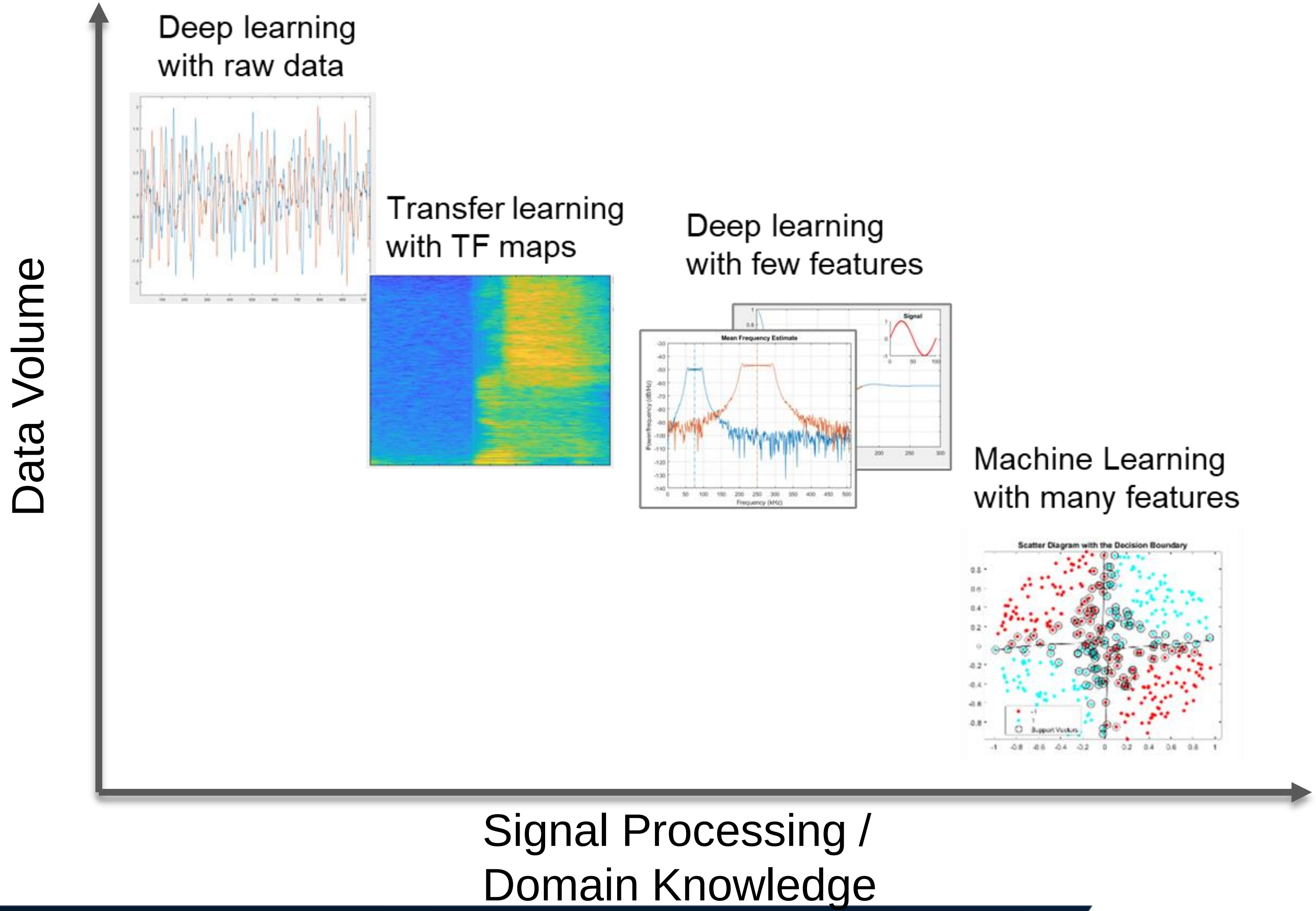
Label the time and frequency features of pulse radar signals with added noise.

# Understanding tradeoffs for model selection

# Understanding tradeoffs for model selection



# Understanding tradeoffs for model selection



# There are three ways to build AI models in MATLAB

# There are three ways to build AI models in MATLAB

```
imageInputLayer([2 spf 1], 'N
'Input Layer')

convolution2dLayer(filterSize
'Name', 'CNN1')

batchNormalizationLayer('Name
reluLayer('Name', 'ReLU1')
maxPooling2dLayer(poolSize, '
'MaxPool1')
```

**fitcauto/fitrauto**

**Writing code**

# There are three ways to build AI models in MATLAB

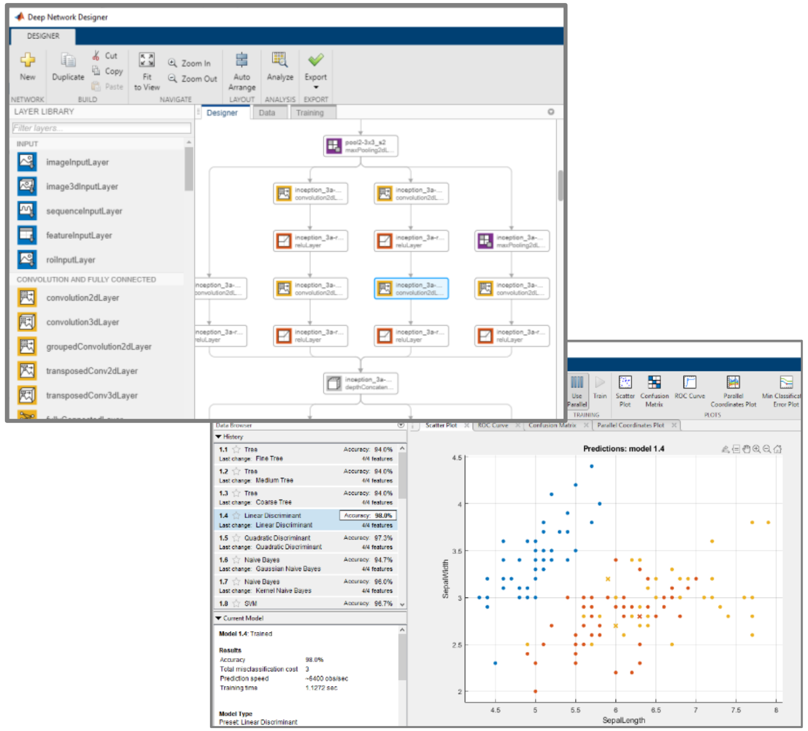
```
imageInputLayer([2 spf 1], 'N
'Input Layer')

convolution2dLayer(filterSize
'Name', 'CNN1')

batchNormalizationLayer('Name
reluLayer('Name', 'ReLU1')
maxPooling2dLayer(poolSize, '
'MaxPool1')
```

**fitcauto/fitrauto**

**Writing code**




**Interactively Design Models with Apps**

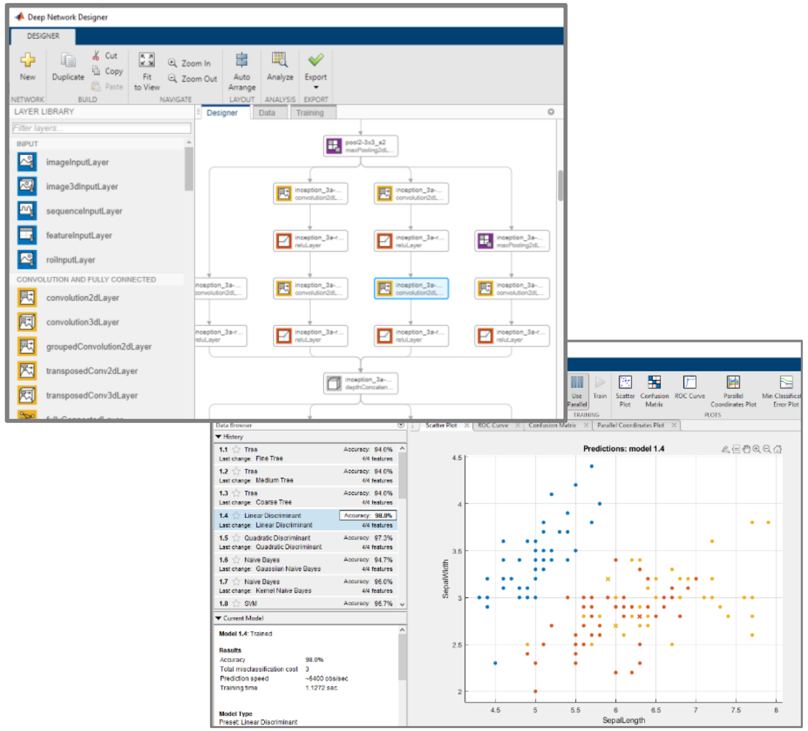
# There are three ways to build AI models in MATLAB

```
imageInputLayer([2 spf 1], 'N
'Input Layer')

convolution2dLayer(filterSize
'Name', 'CNN1')

batchNormalizationLayer('Name
reluLayer('Name', 'ReLU1')
maxPooling2dLayer(poolSize, '
'MaxPool1')
```

**fitcauto/fitrauto**




Inception-v3 | ResNet-101 | VGG-16 | Inception-ResNet-v2
ResNet-18 | GoogLeNet | DenseNet-201 | VGG-19
SqueezeNet | AlexNet | ResNet-50

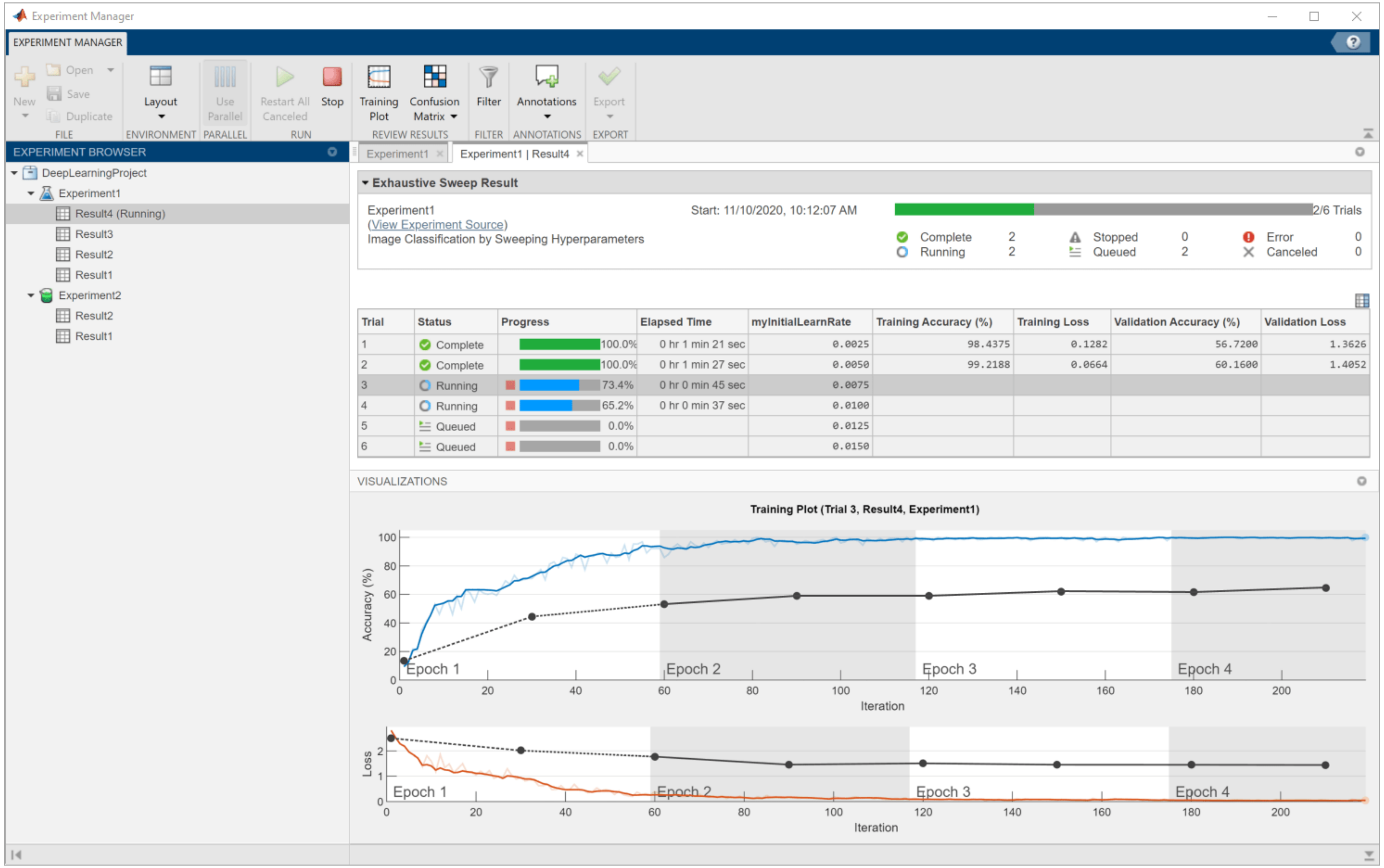
## Writing code

## Interactively Design Models with Apps

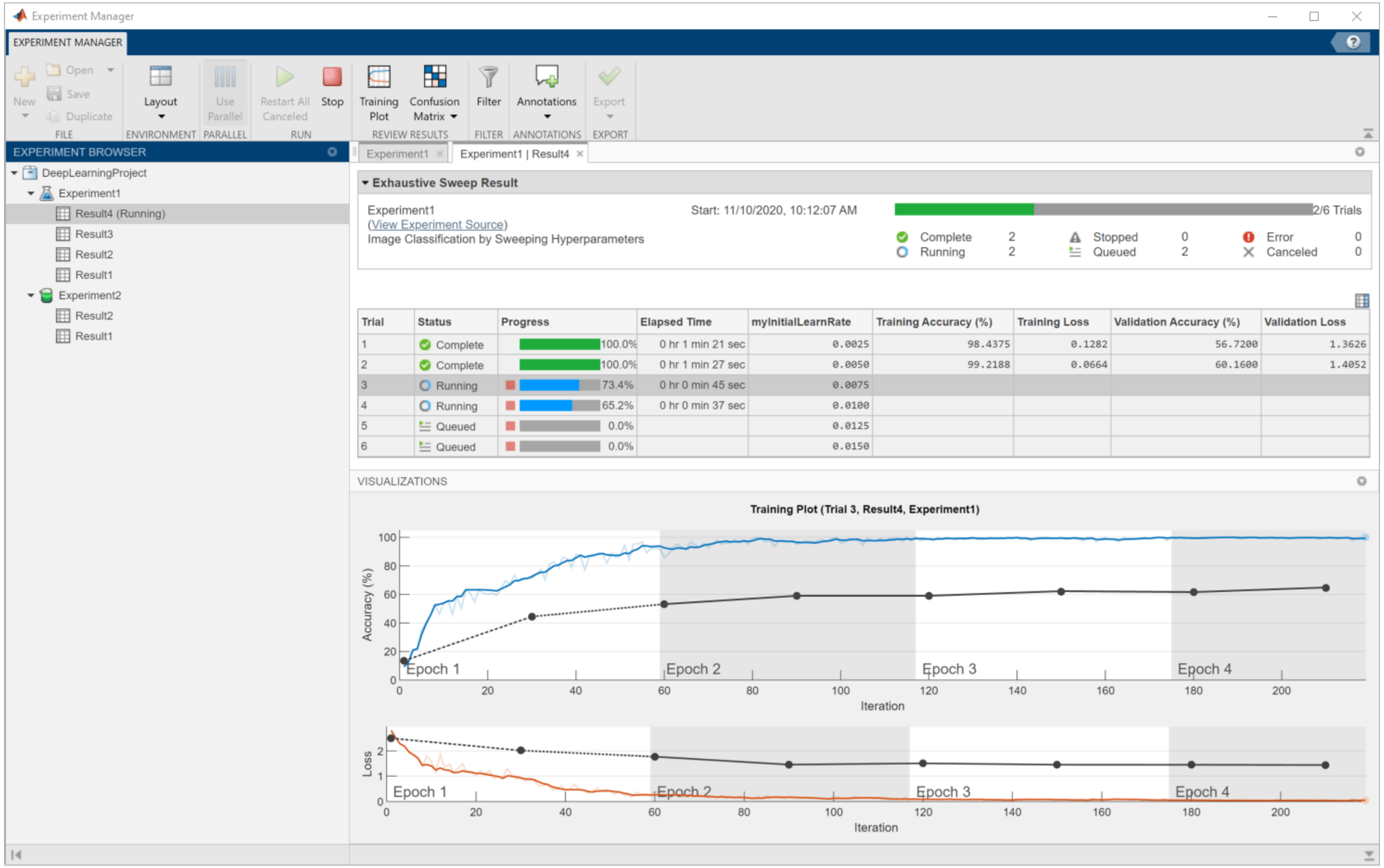
## Use Transfer Learning for Deep Learning

# Iterate to find the best model with Experiment Manager App

# Iterate to find the best model with Experiment Manager App

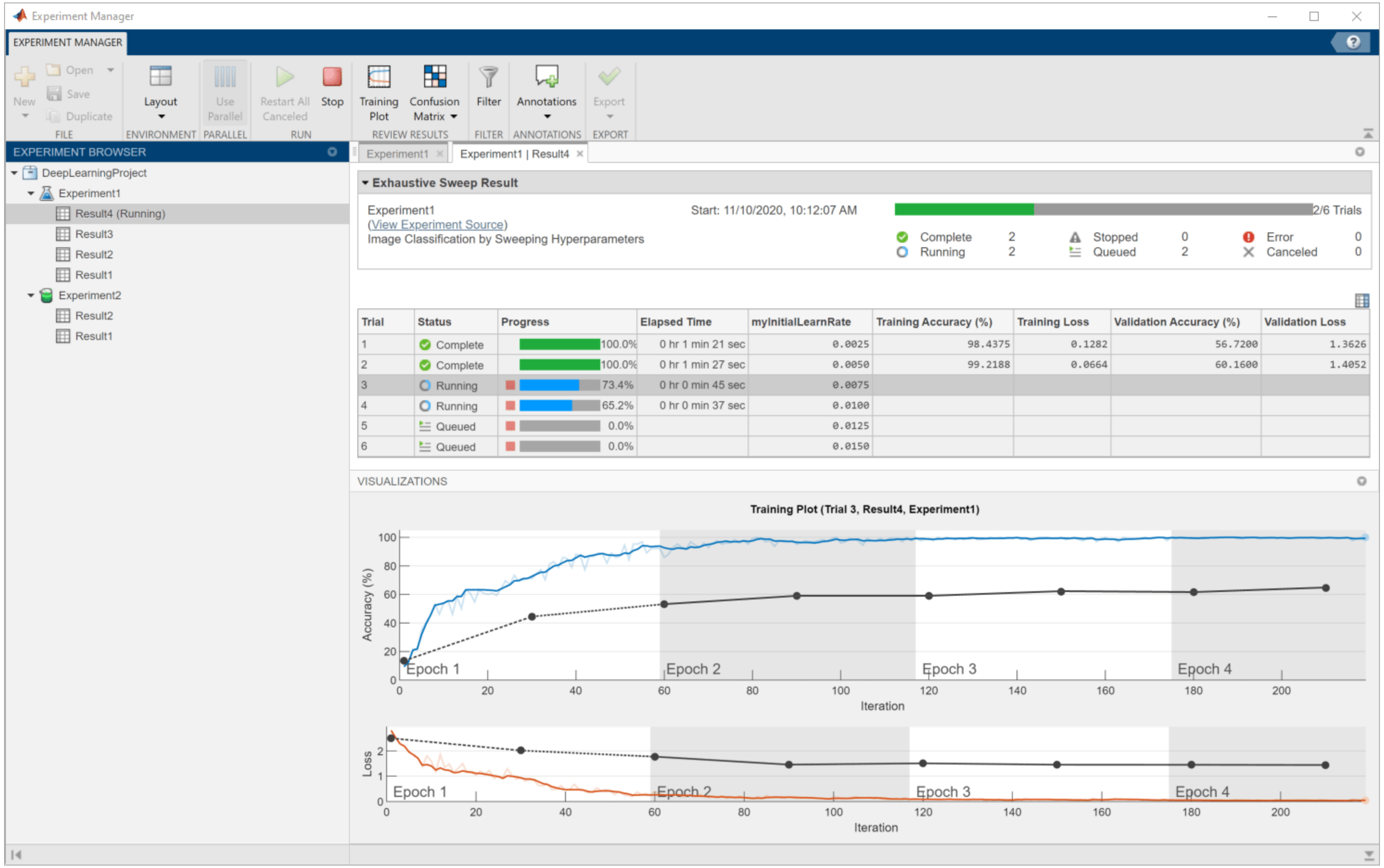


# Iterate to find the best model with Experiment Manager App




Find optimal training options

# Iterate to find the best model with Experiment Manager App



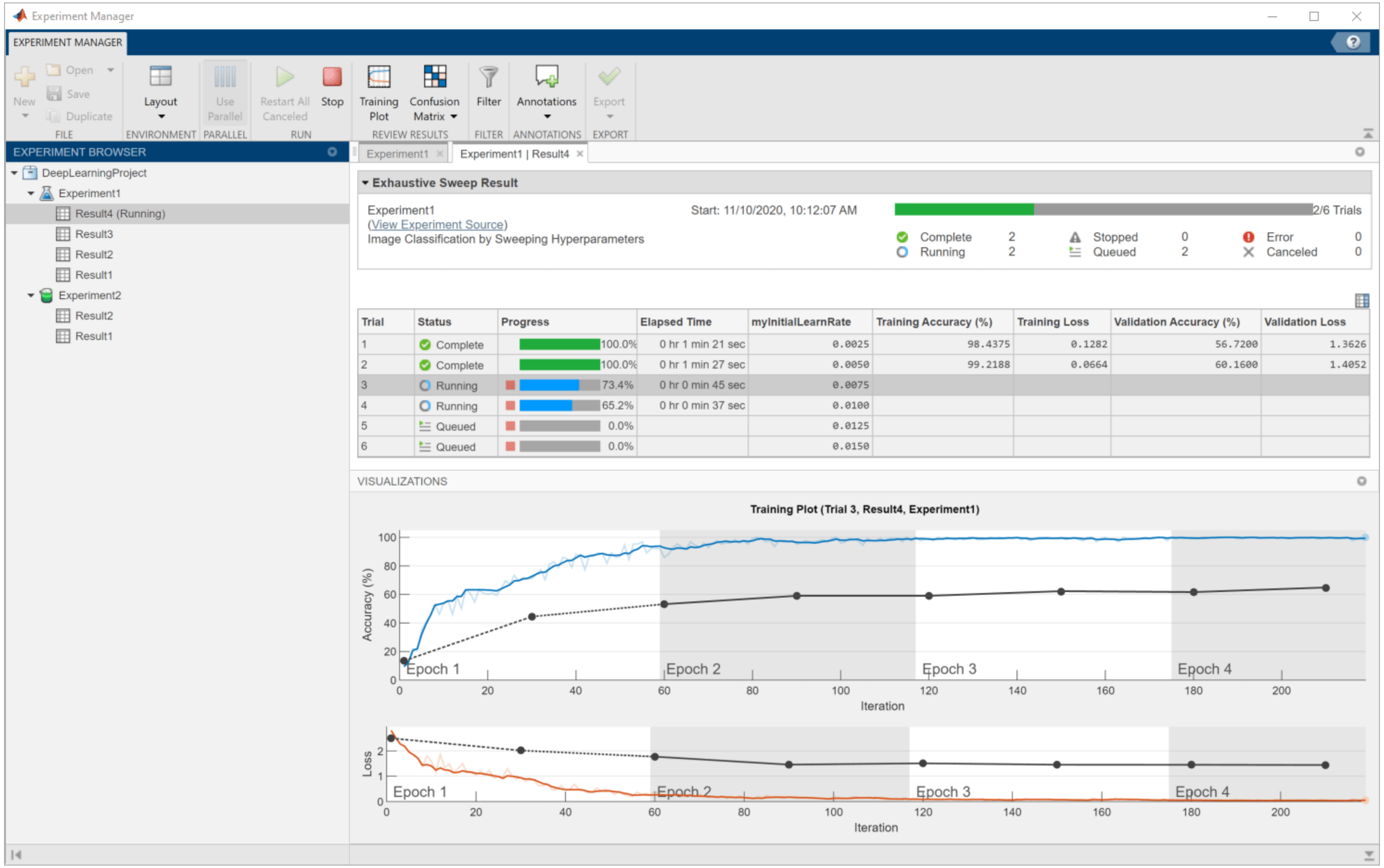

**Find optimal training options**

**Compare the results of using different data sets**

MATLAB EXPO

MathWorks

# Iterate to find the best model with Experiment Manager App

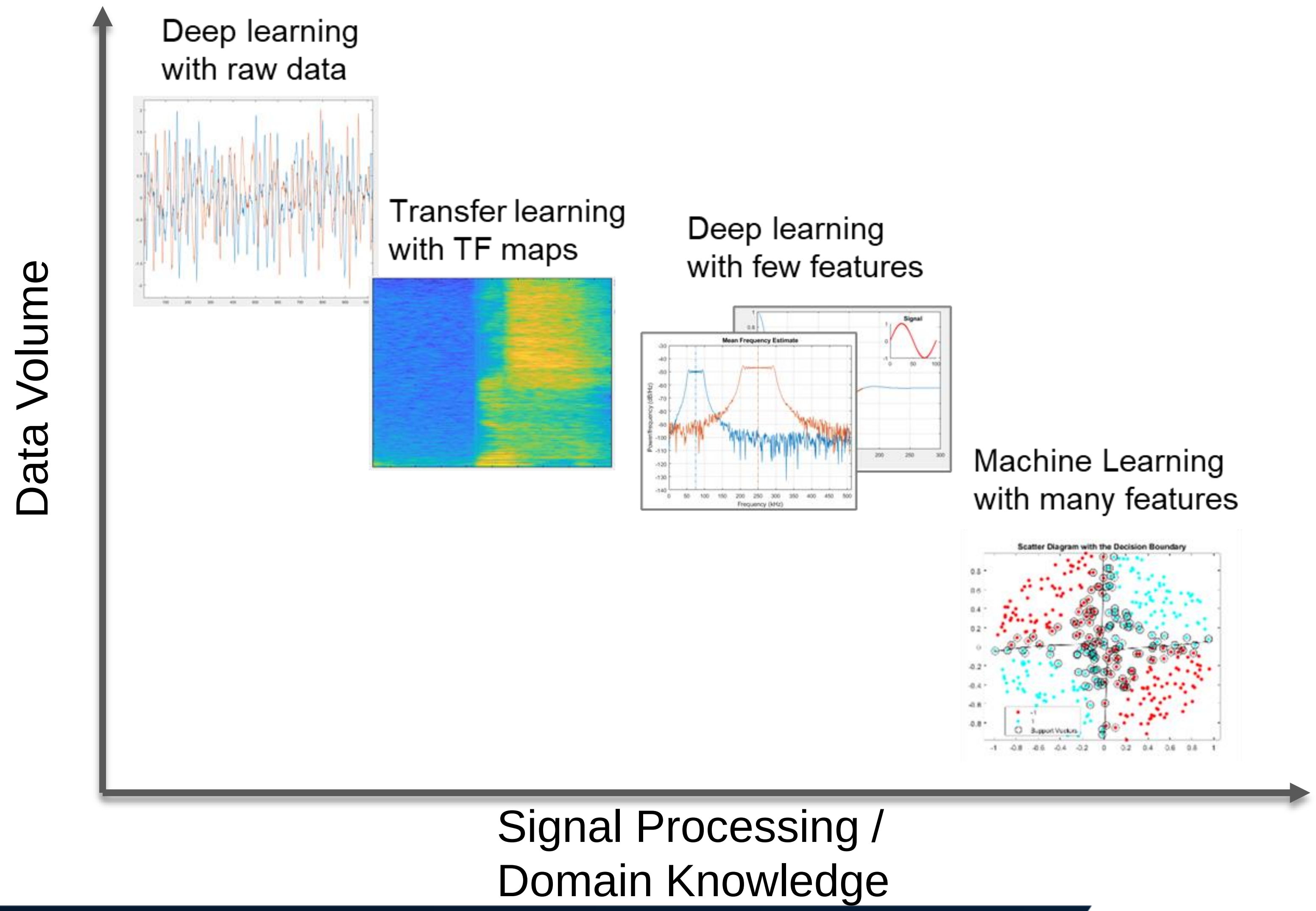



Find optimal training options

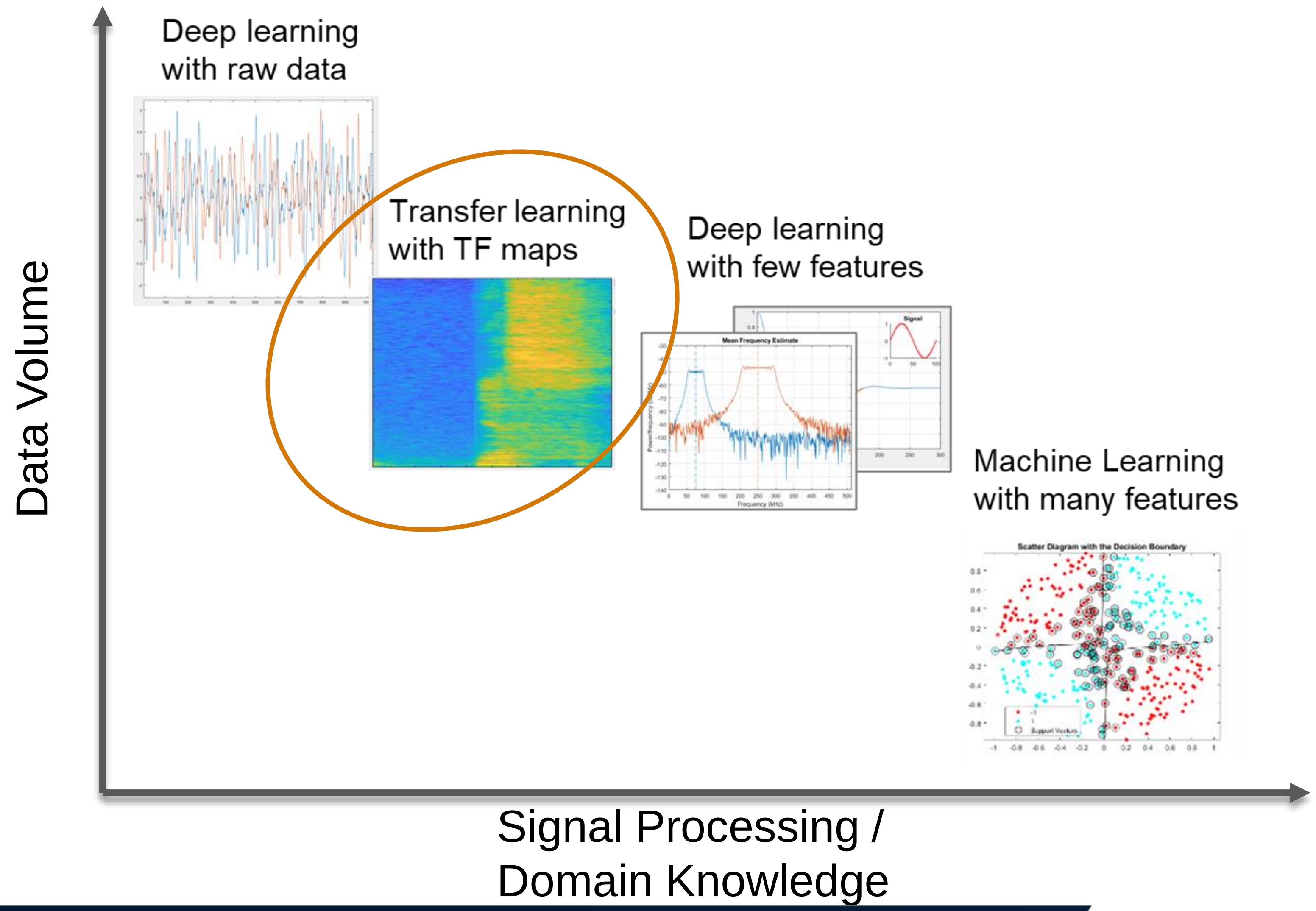
Compare the results of using different data sets

Compare the results of using different models

# Selecting the Right Model : Understanding Tradeoffs



# Selecting the Right Model : Understanding Tradeoffs

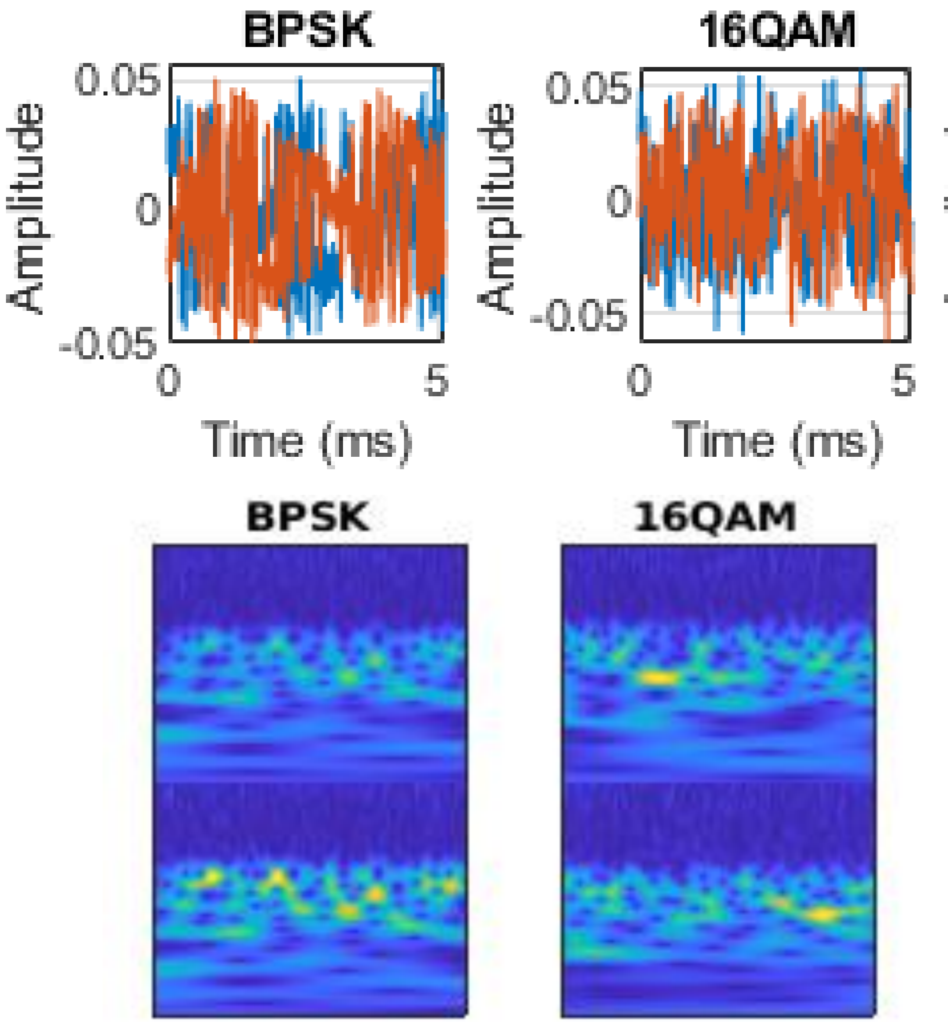


# Continuous Wavelet Transform is used to extract the Time-Frequency maps

# Continuous Wavelet Transform is used to extract the Time-Frequency maps

- One line of code for generating wavelet time-frequency visualization in MATLAB. Works for any signal

```
>> cwt(inputSignal)
```

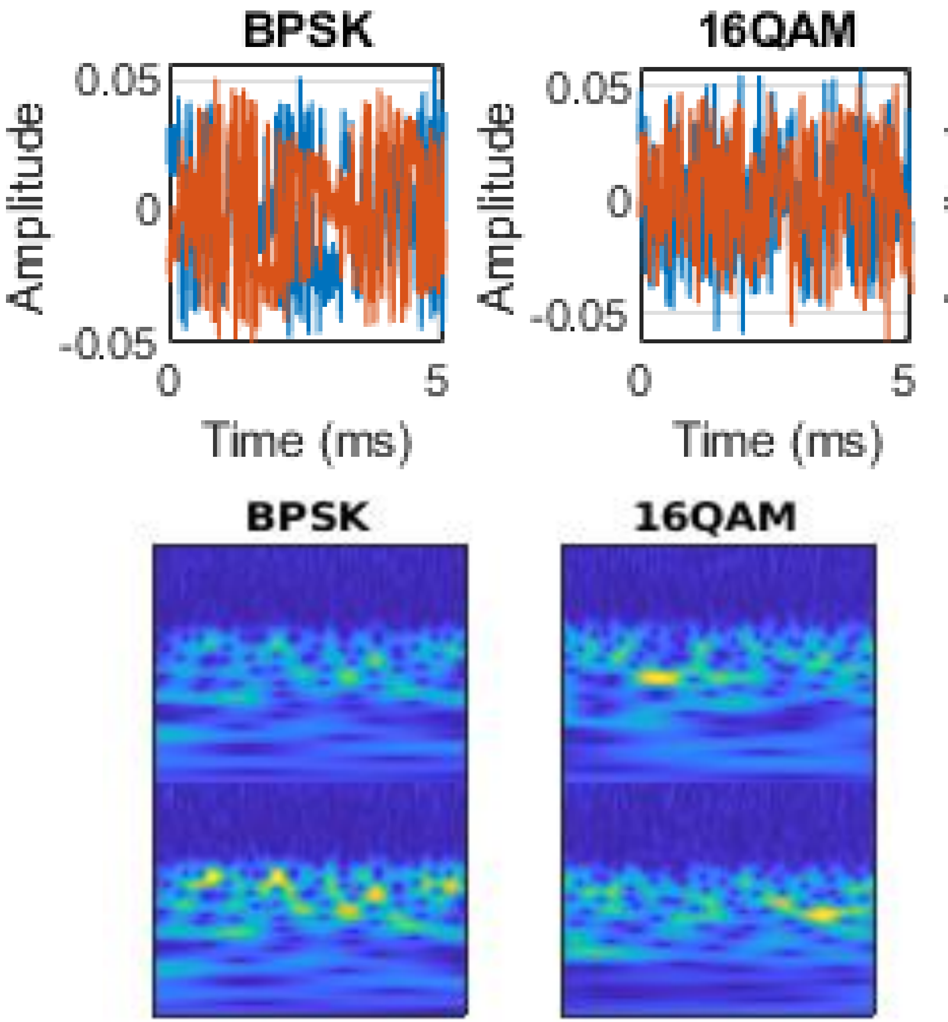


# Continuous Wavelet Transform is used to extract the Time-Frequency maps

- One line of code for generating wavelet time-frequency visualization in MATLAB. Works for any signal

```
>> cwt(inputSignal)
```

- Localizes sharp transients and slowly varying oscillations simultaneously



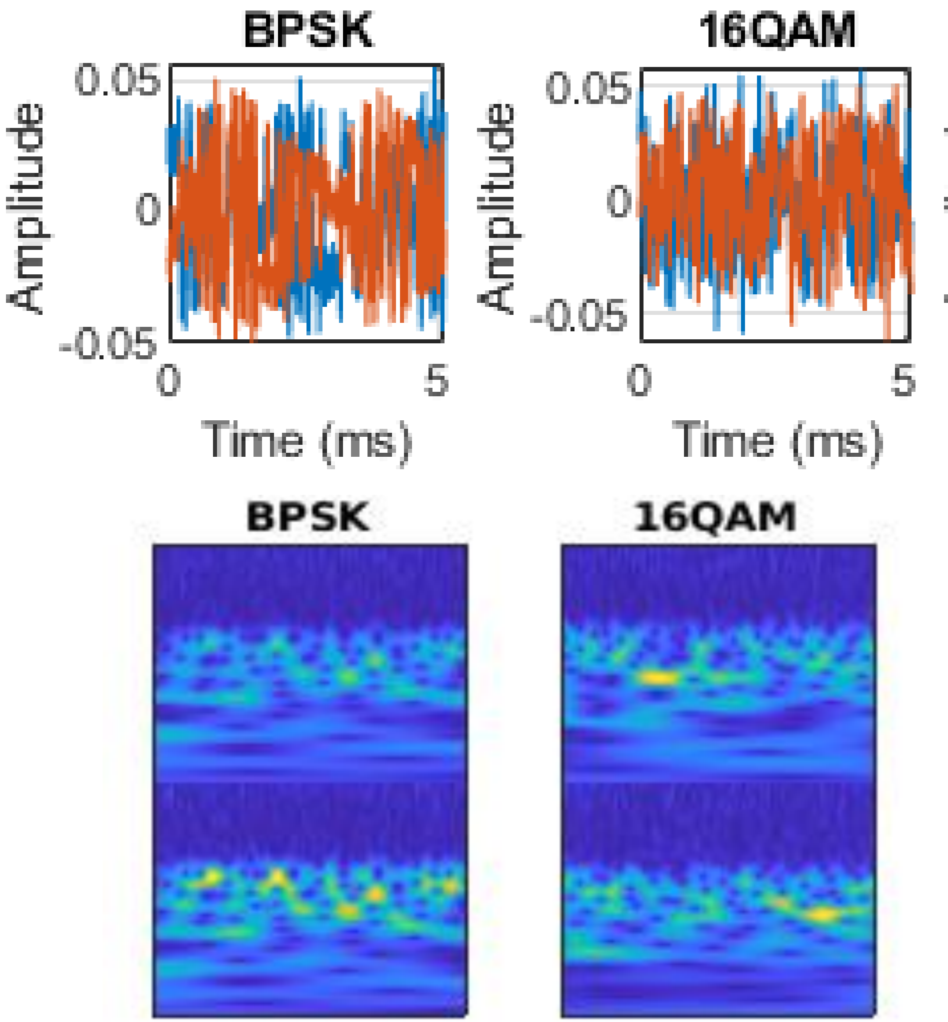
# Continuous Wavelet Transform is used to extract the Time-Frequency maps

- One line of code for generating wavelet time-frequency visualization in MATLAB. Works for any signal

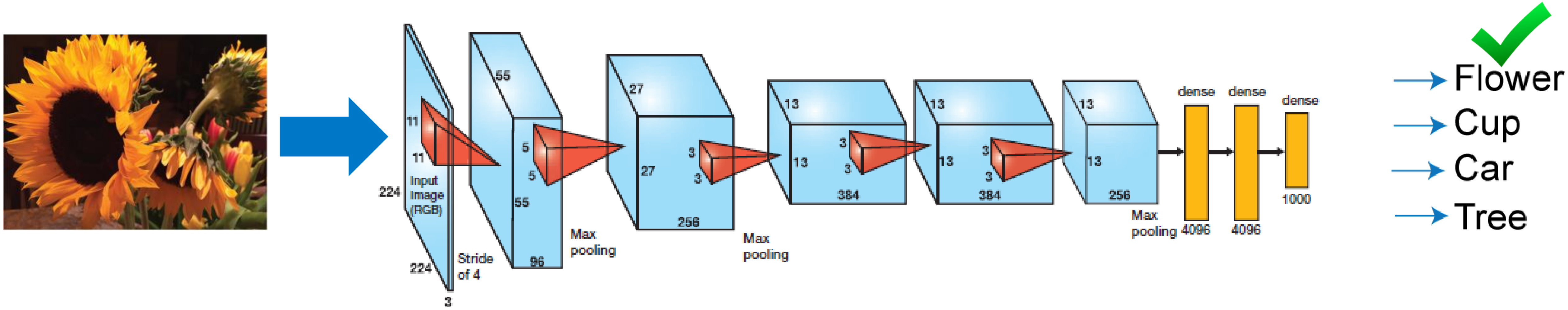
```
>> cwt(inputSignal)
```

- Localizes sharp transients and slowly varying oscillations simultaneously

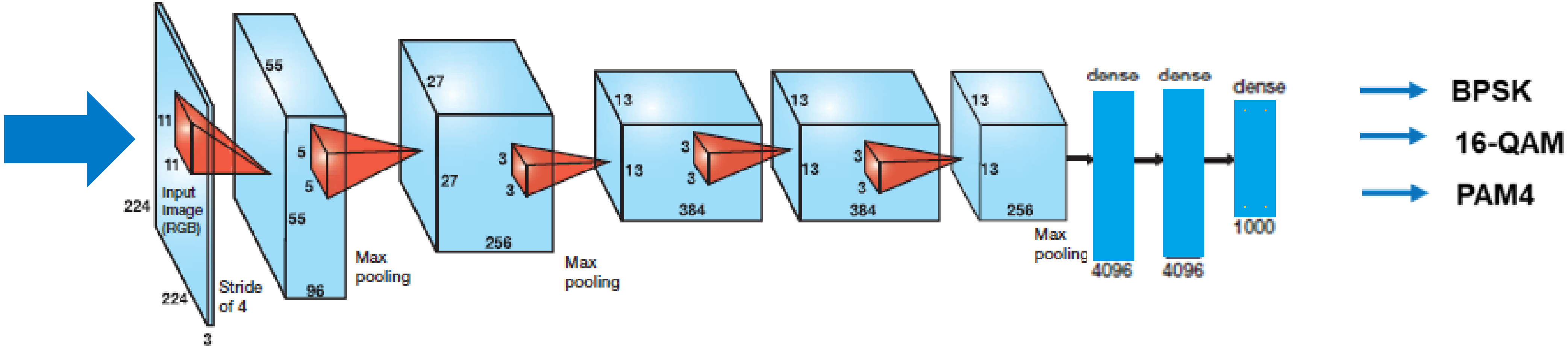
- Works with complex data



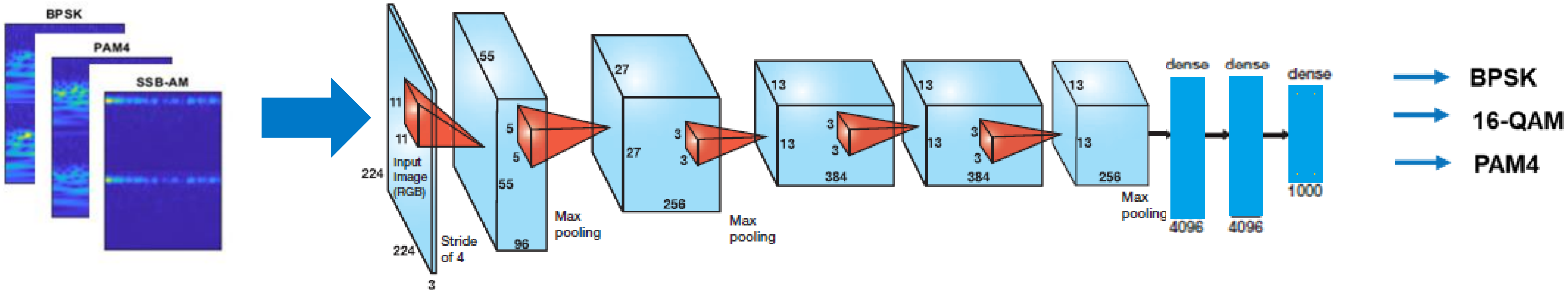
# Using time-frequency maps as inputs to a pretrained CNN



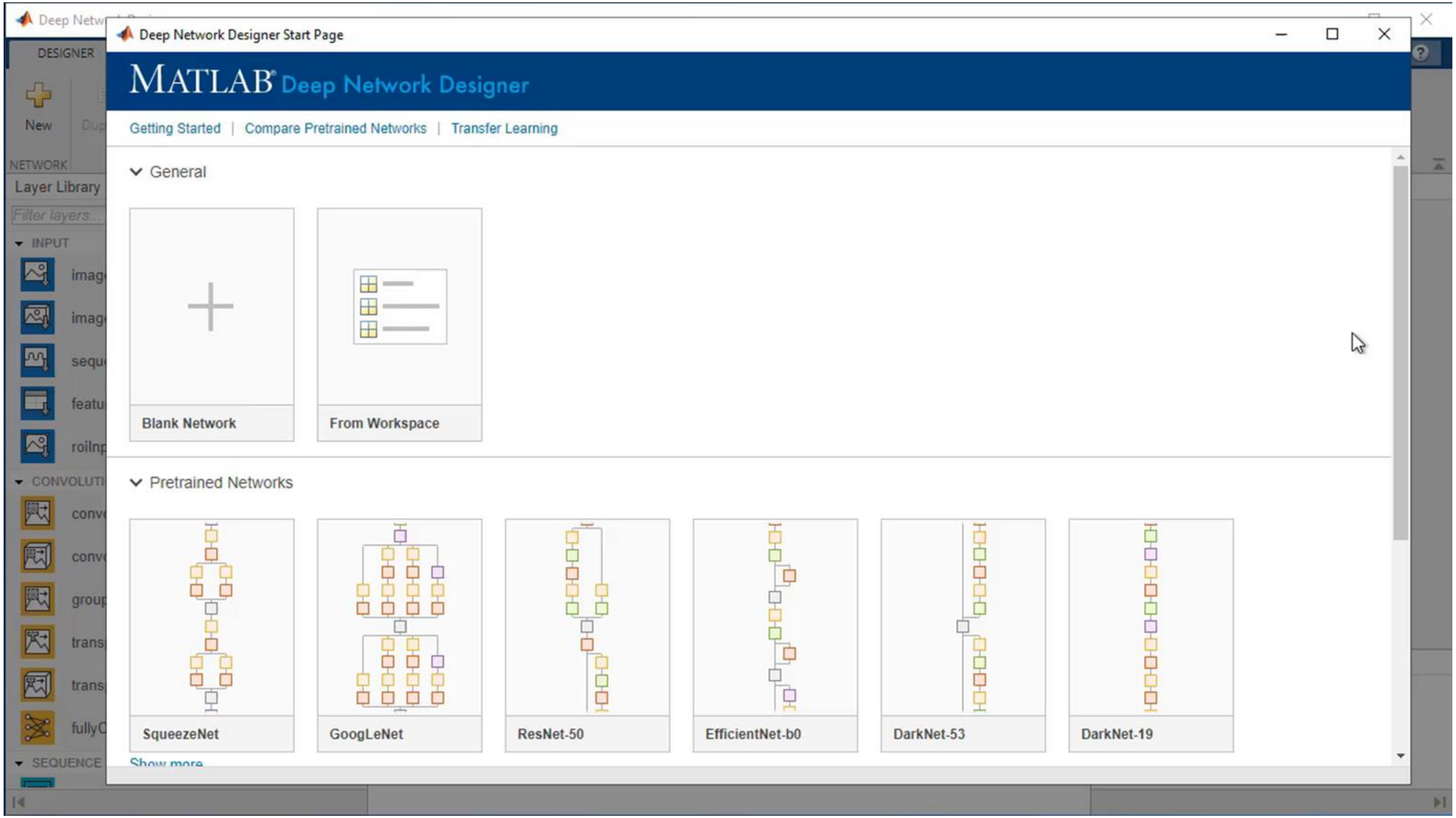
# Using time-frequency maps as inputs to a pretrained CNN



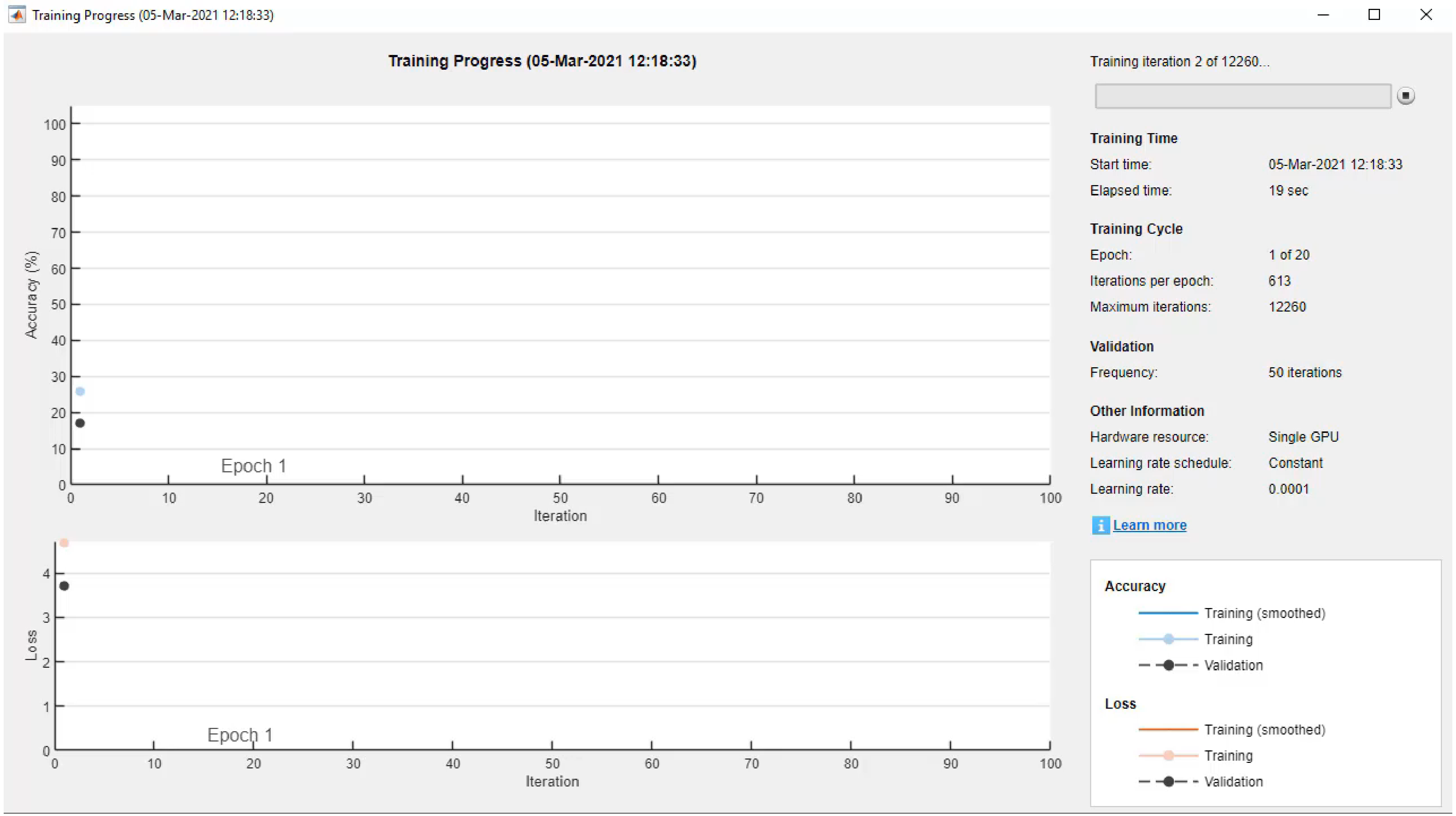
# Using time-frequency maps as inputs to a pretrained CNN



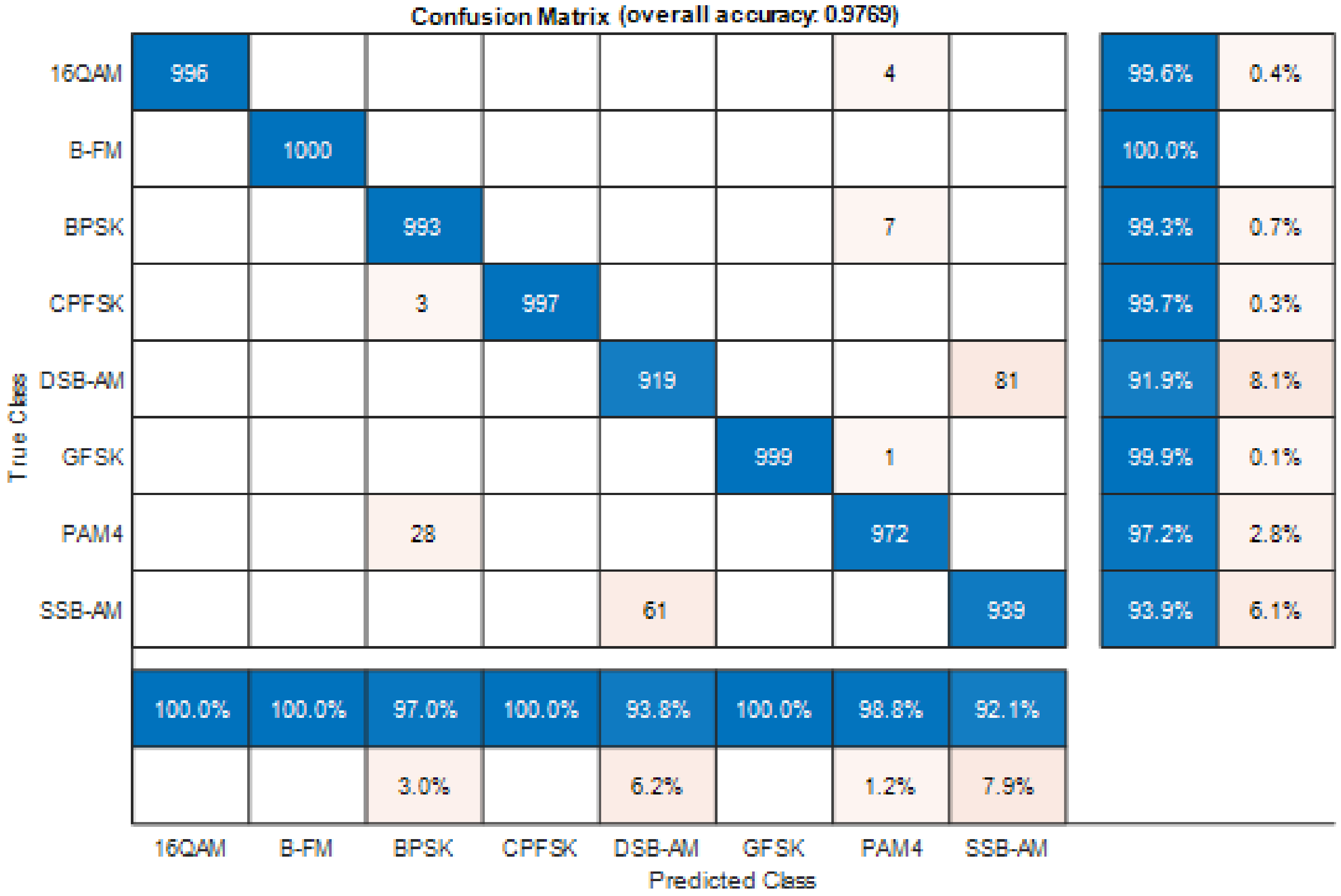
# Transfer Learning with Deep Network Designer App



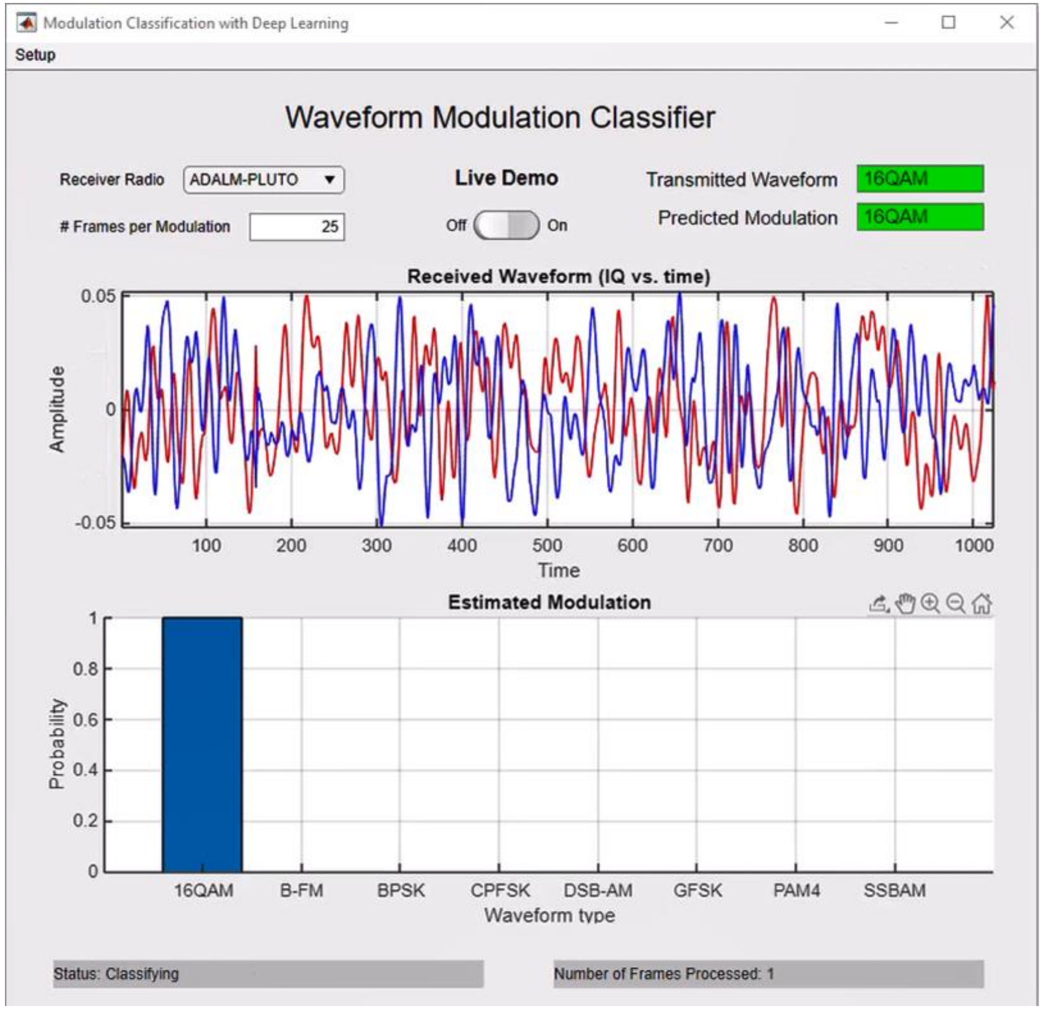
# Train and Test Deep Network



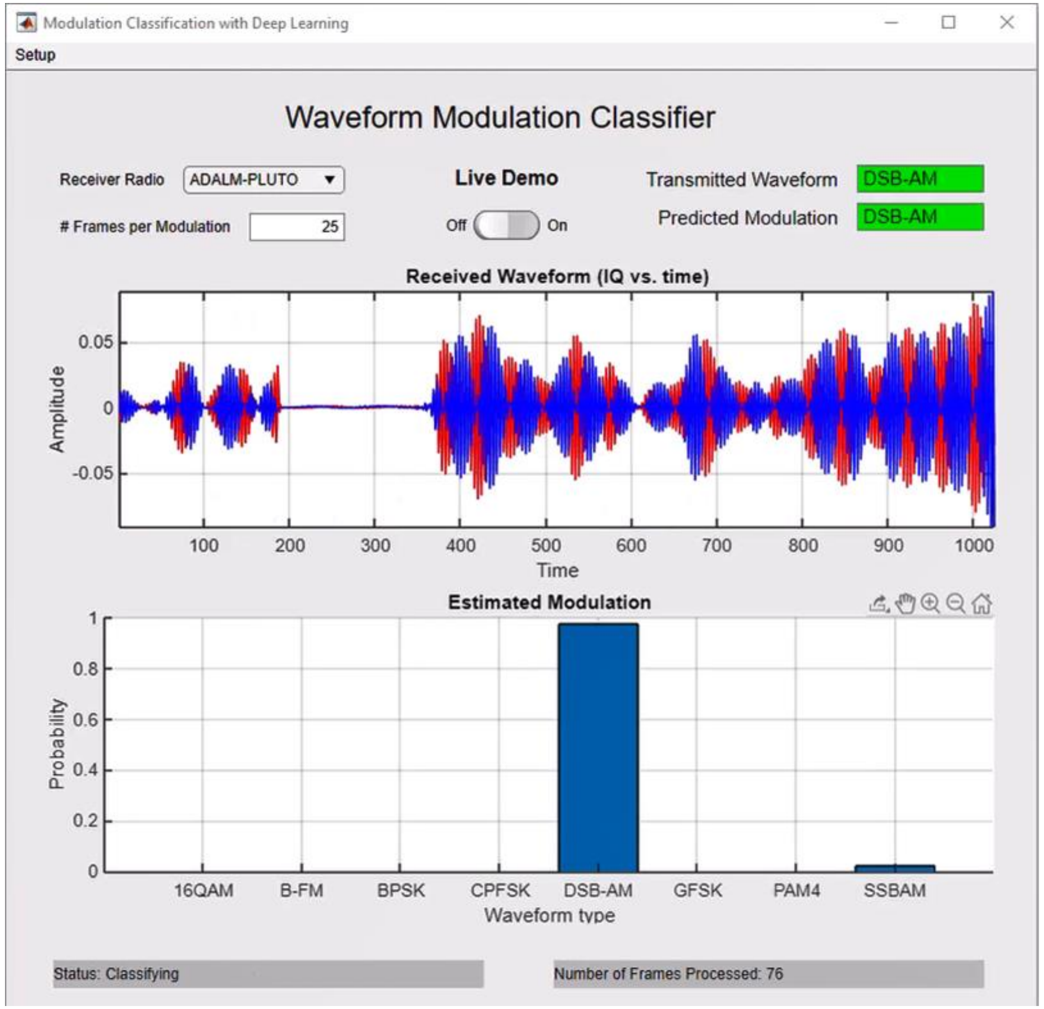
# Test Deep Network



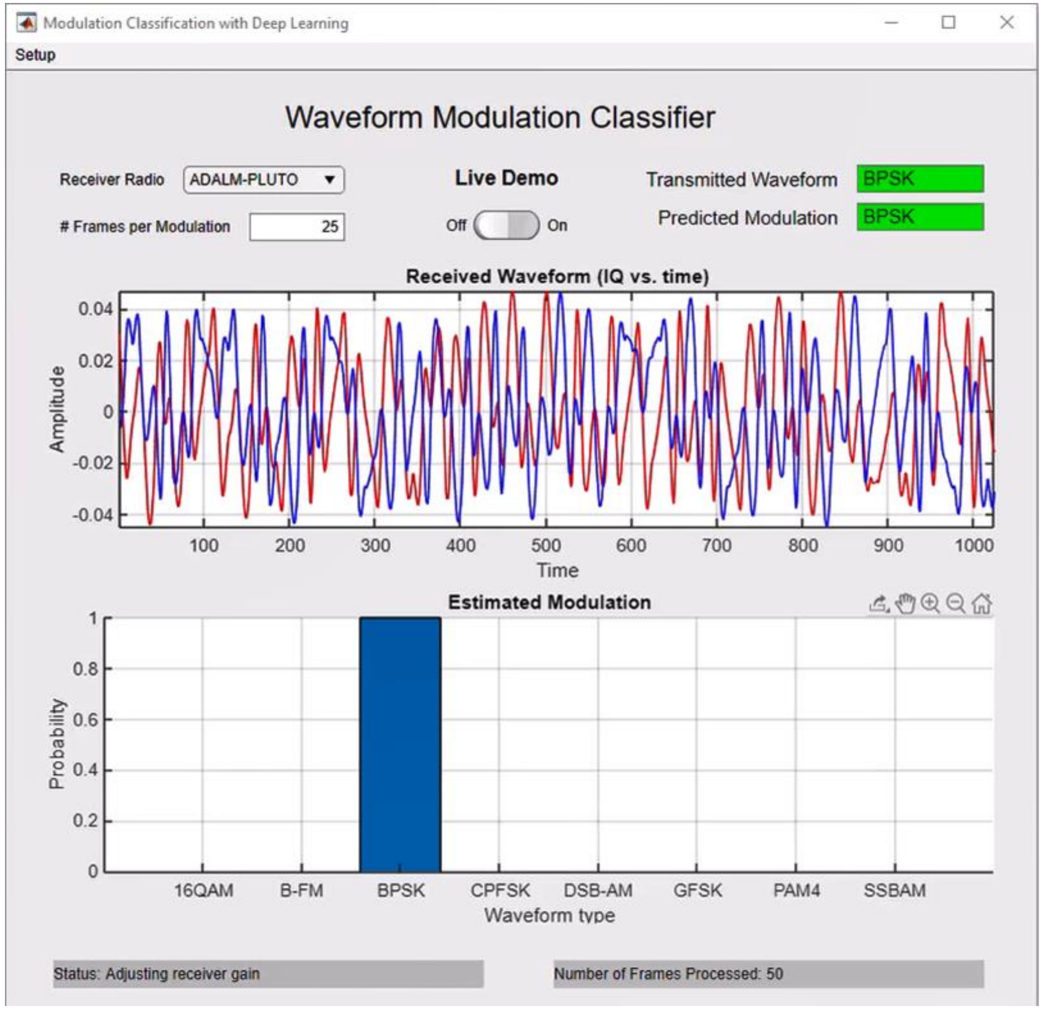
# Testing network with connected hardware

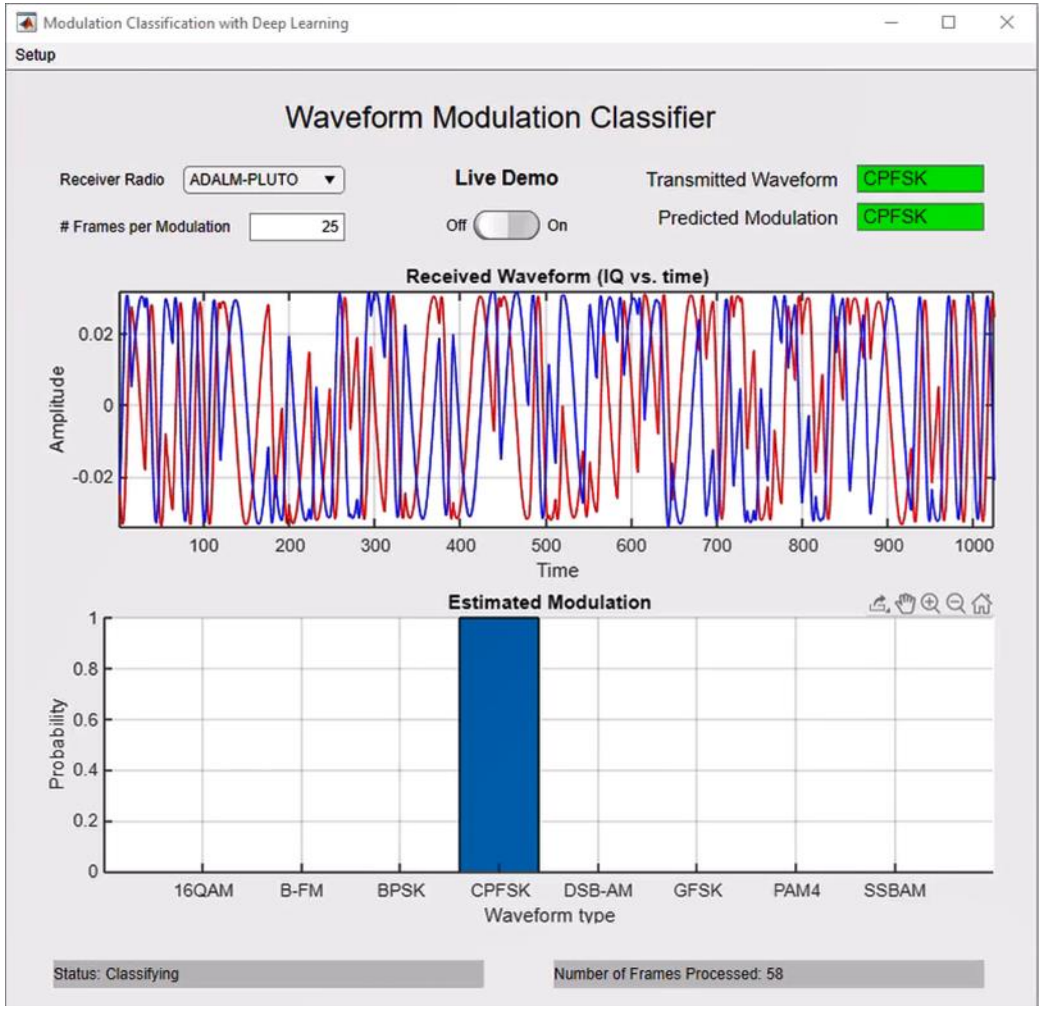
# Testing network with connected hardware

# Testing network with connected hardware

# Testing network with connected hardware

# AI-assisted system design

| Data Preparation | AI Modeling | Deployment |
|---|---|---|
| Data cleansing and preparation | Model design and tuning | Embedded devices |
| Human insight | Hardware accelerated training | Enterprise systems |
| Simulation-generated data | Interoperability | Edge, cloud, desktop |

# AI-assisted system design

## Data Preparation

- Data cleansing and preparation
- Human insight
- Simulation-generated data

## AI Modeling

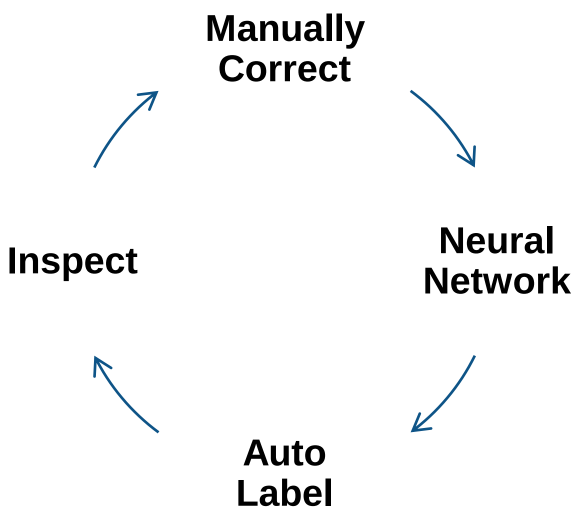
- Model design and tuning
- Hardware accelerated training
- Interoperability

## Deployment

- Embedded devices
- Enterprise systems
- Edge, cloud, desktop

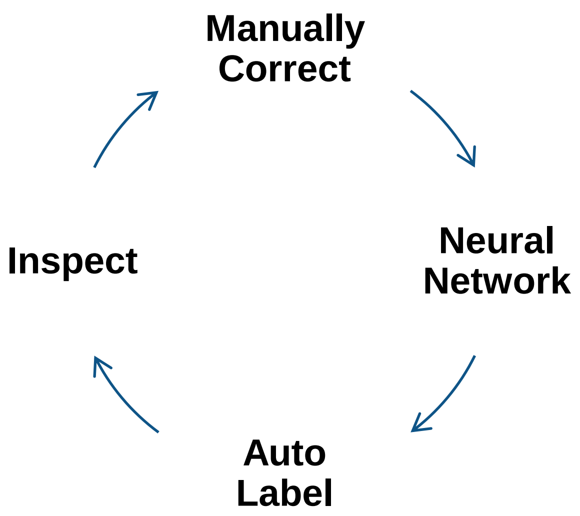
# Deep Learning can be used in each step of the AI workflow

# Deep Learning can be used in each step of the AI workflow




Labeling assistance

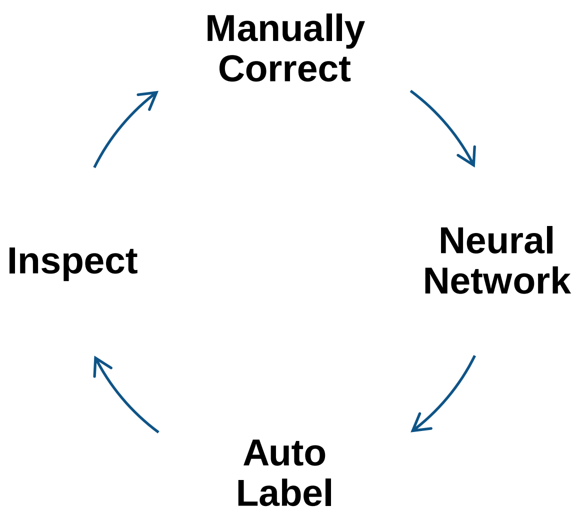
# Deep Learning can be used in each step of the AI workflow




Labeling assistance

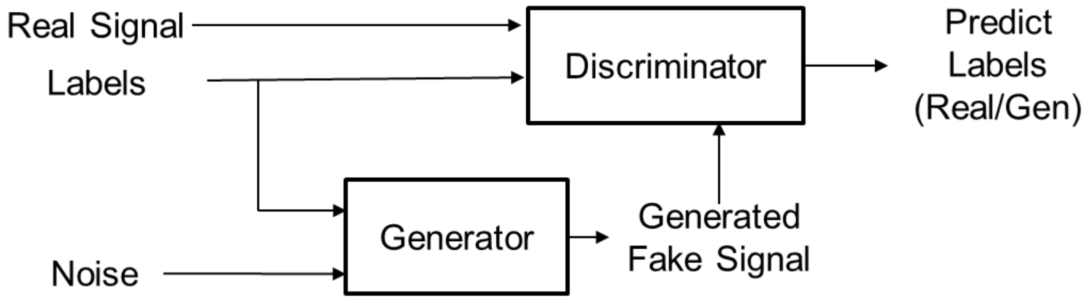
`classifySound (YAMNet),GoogLeNet, fitcecoc(ResNet18)`

# Deep Learning can be used in each step of the AI workflow



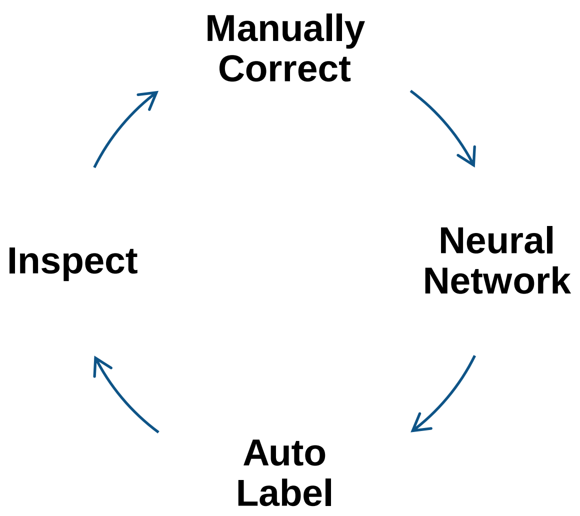

Labeling assistance

`classifySound (YAMNet),GoogLeNet,`
`fitcecoc(ResNet18)`



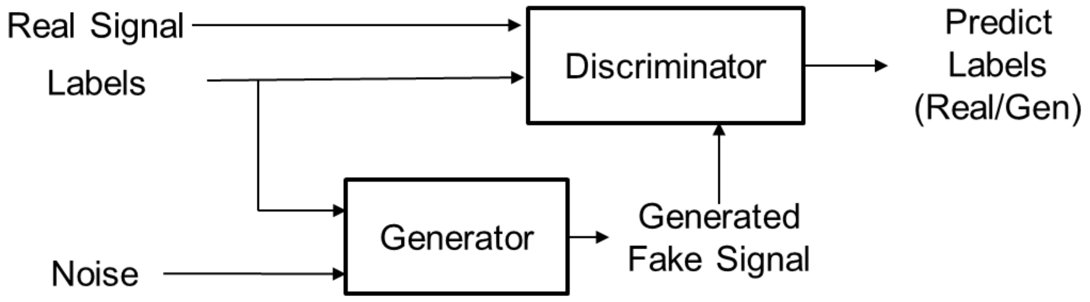

Synthetic Data Generation

# Deep Learning can be used in each step of the AI workflow




Labeling assistance

`classifySound (YAMNet),GoogLeNet, fitcecoc(ResNet18)`




Synthetic Data Generation

`Generative Adversarial Networks (GANs)`

# Deep Learning can be used in each step of the AI workflow

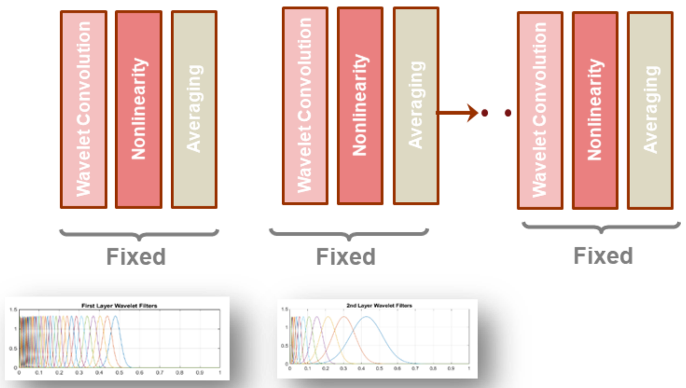
# Deep Learning can be used in each step of the AI workflow


```
features = vggFeatures(audioIn,fs);)
```

Feature Extraction

# Deep Learning can be used in each step of the AI workflow

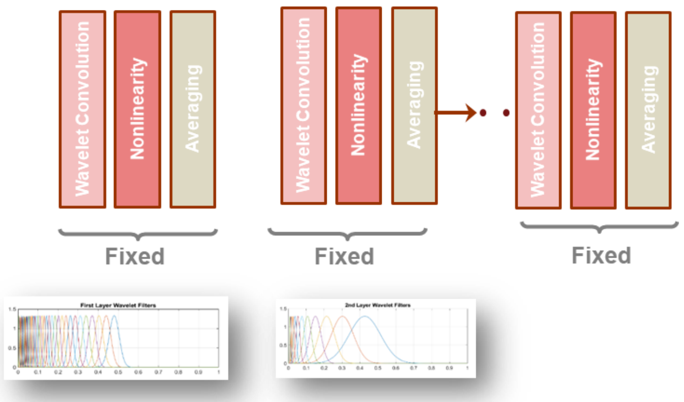

```
features = vggFeatures(audioIn,fs);)
```




Feature Extraction

# Deep Learning can be used in each step of the AI workflow


```
features = vggFeatures(audioIn,fs);)
```

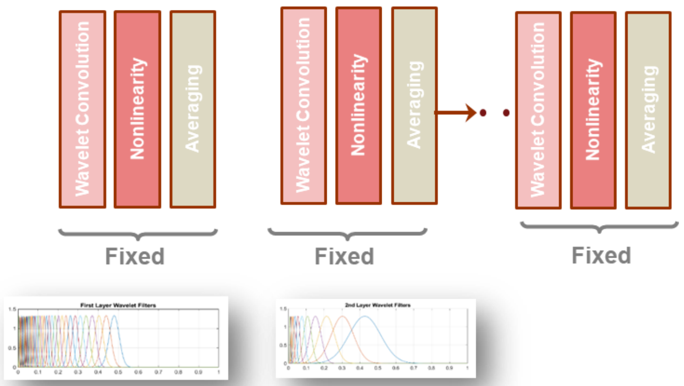



Feature Extraction

**`vggFeatures, waveletScattering`**

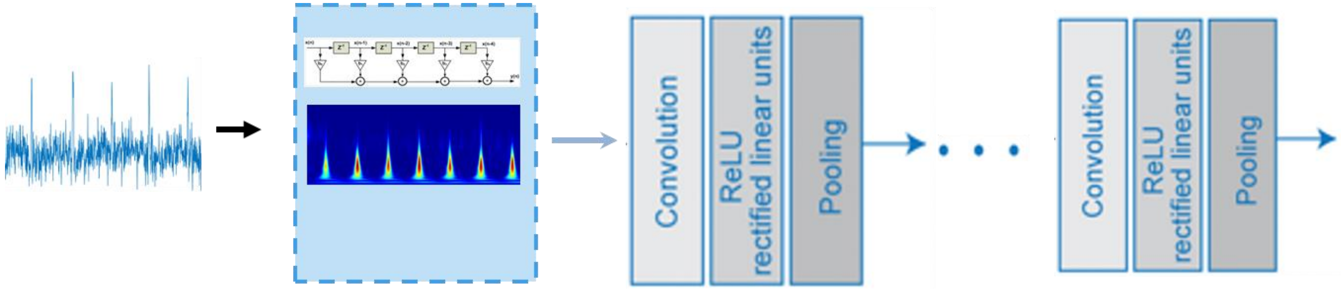
# Deep Learning can be used in each step of the AI workflow


```
features = vggFeatures(audioIn,fs);)
```




Feature Extraction

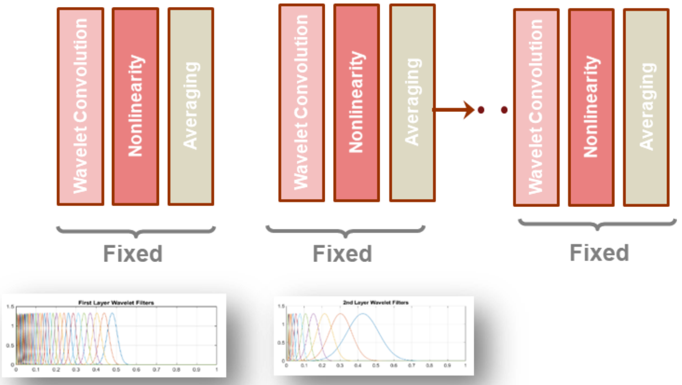
**vggFeatures, waveletScattering**




Differentiable Signal Processing

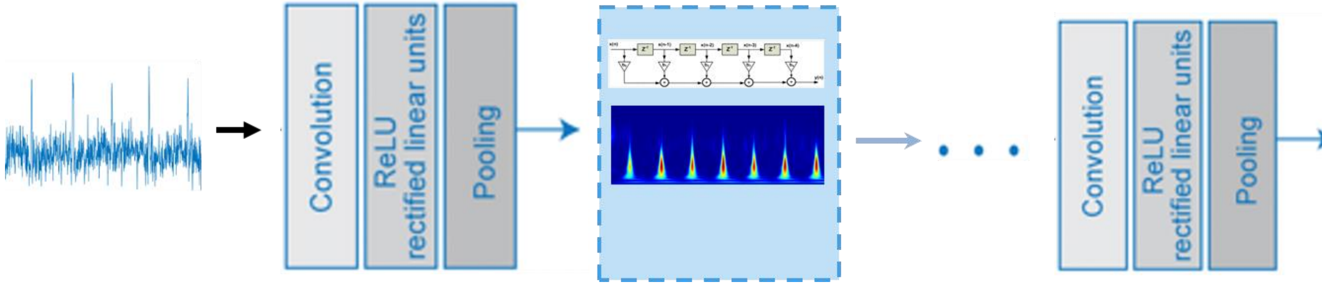
# Deep Learning can be used in each step of the AI workflow


```
features = vggFeatures(audioIn,fs););
```




Feature Extraction

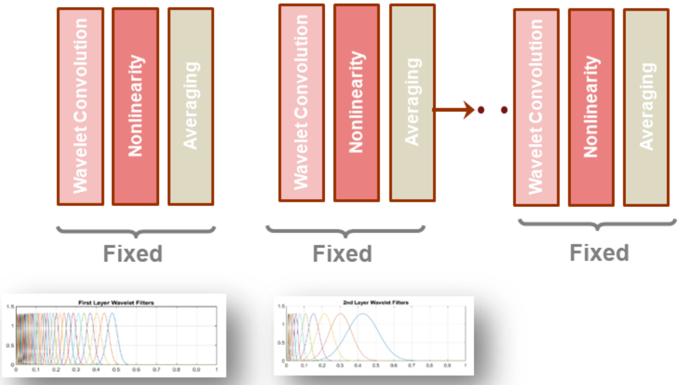
**vggFeatures, waveletScattering**




Differentiable Signal Processing

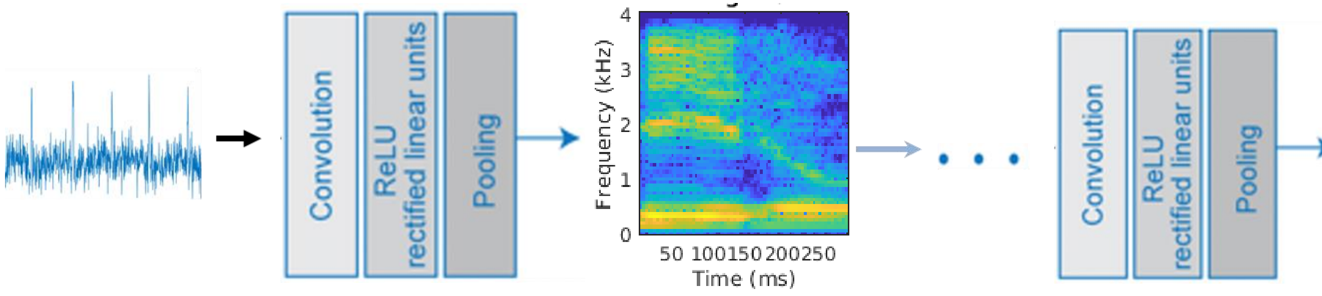
# Deep Learning can be used in each step of the AI workflow


```
features = vggFeatures(audioIn,fs););
```




Feature Extraction

**`vggFeatures, waveletScattering`**




Differentiable Signal Processing

**`dlstft (Differentiable STFT)`**

# AI-driven system design

## Data Preparation

- Data cleansing and preparation
- Human insight
- Simulation-generated data

## AI Modeling

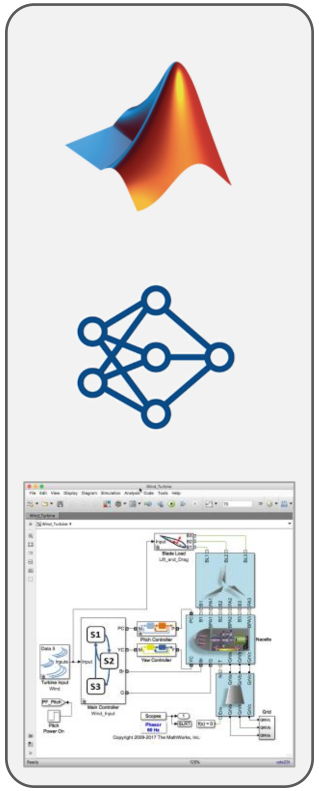
- Model design and tuning
- Hardware accelerated training
- Interoperability

## Deployment

- Embedded devices
- Enterprise systems
- Edge, cloud, desktop

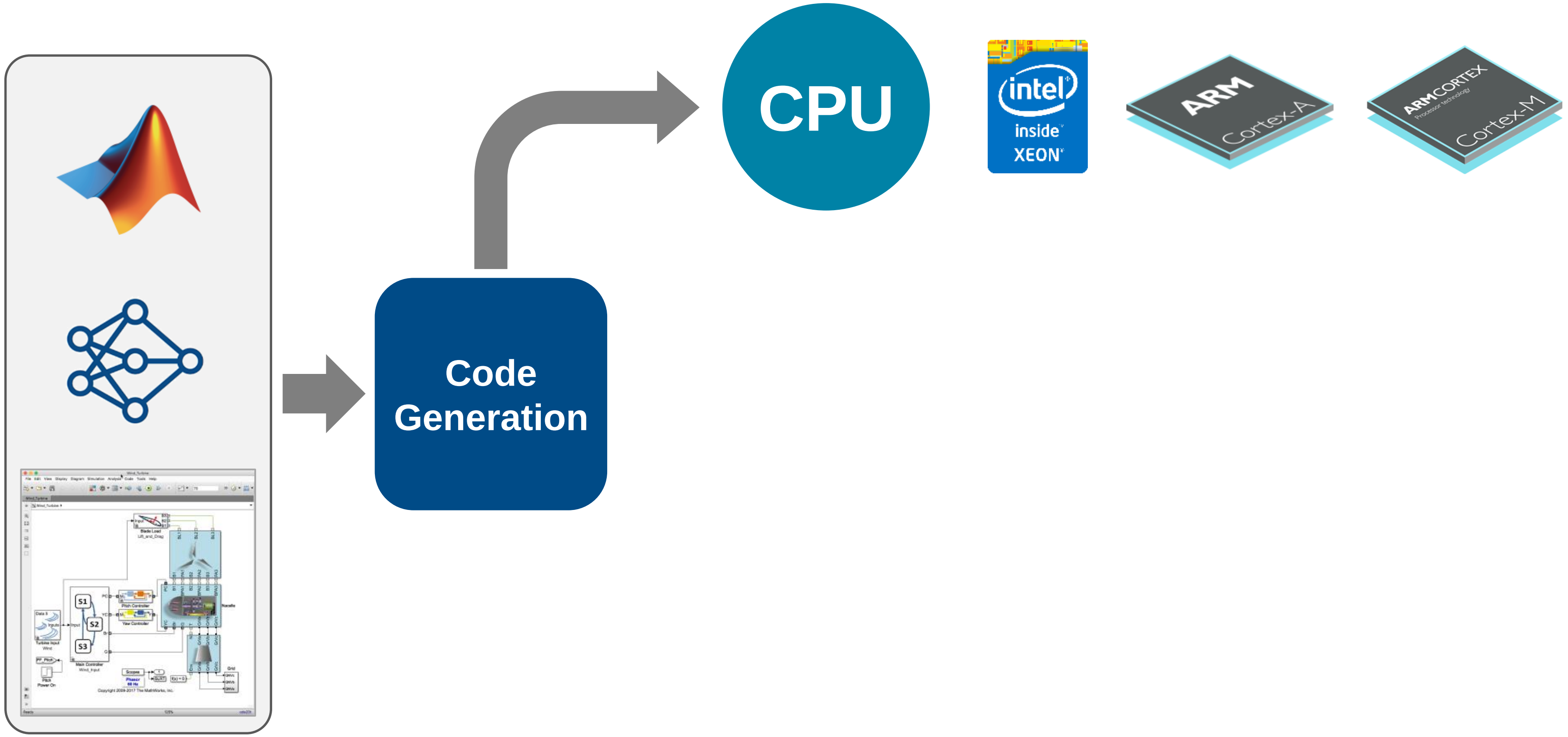
# Deploy to any processor with best-in-class performance

# Deploy to any processor with best-in-class performance



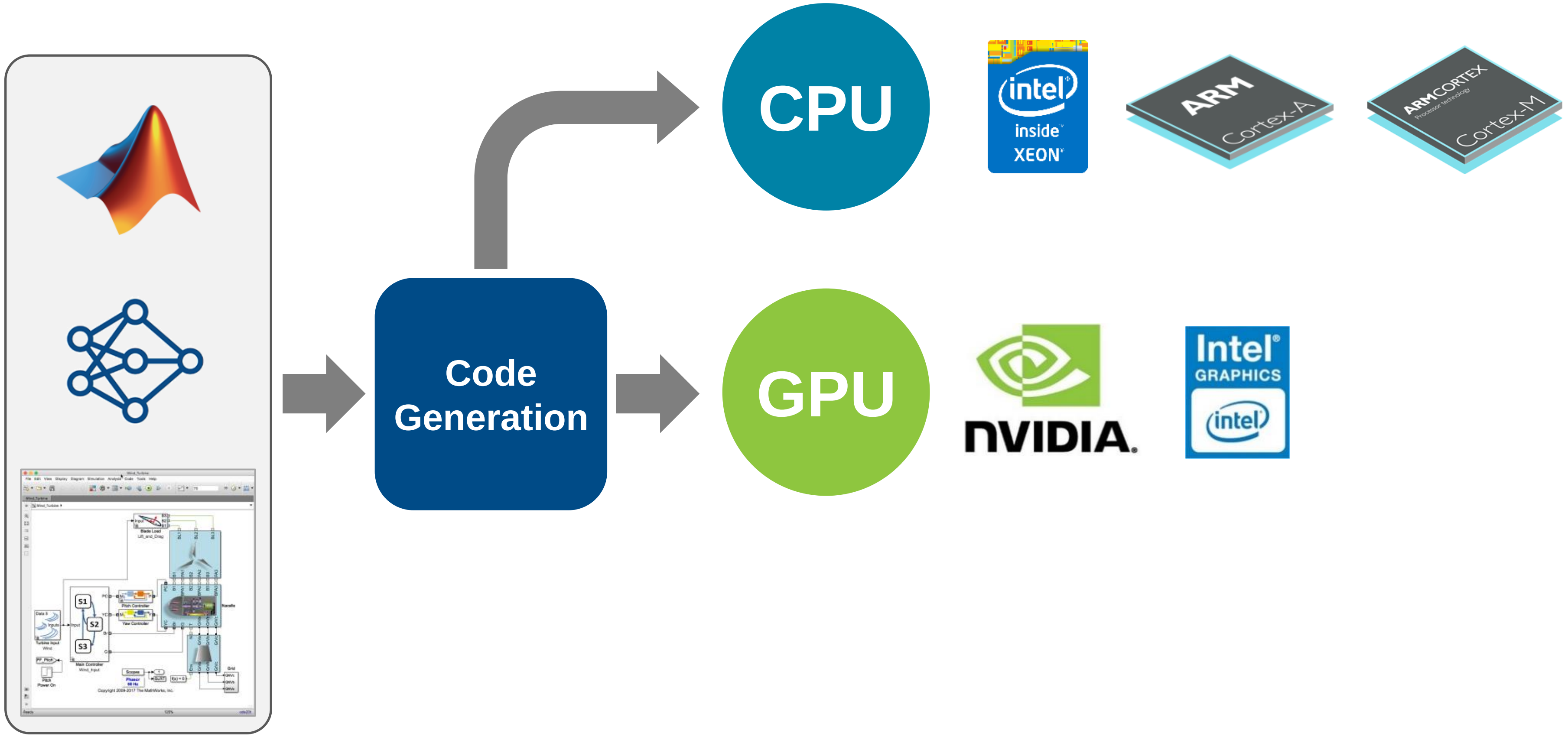
**Preprocessing, Feature Extraction, AI Model**

# Deploy to any processor with best-in-class performance



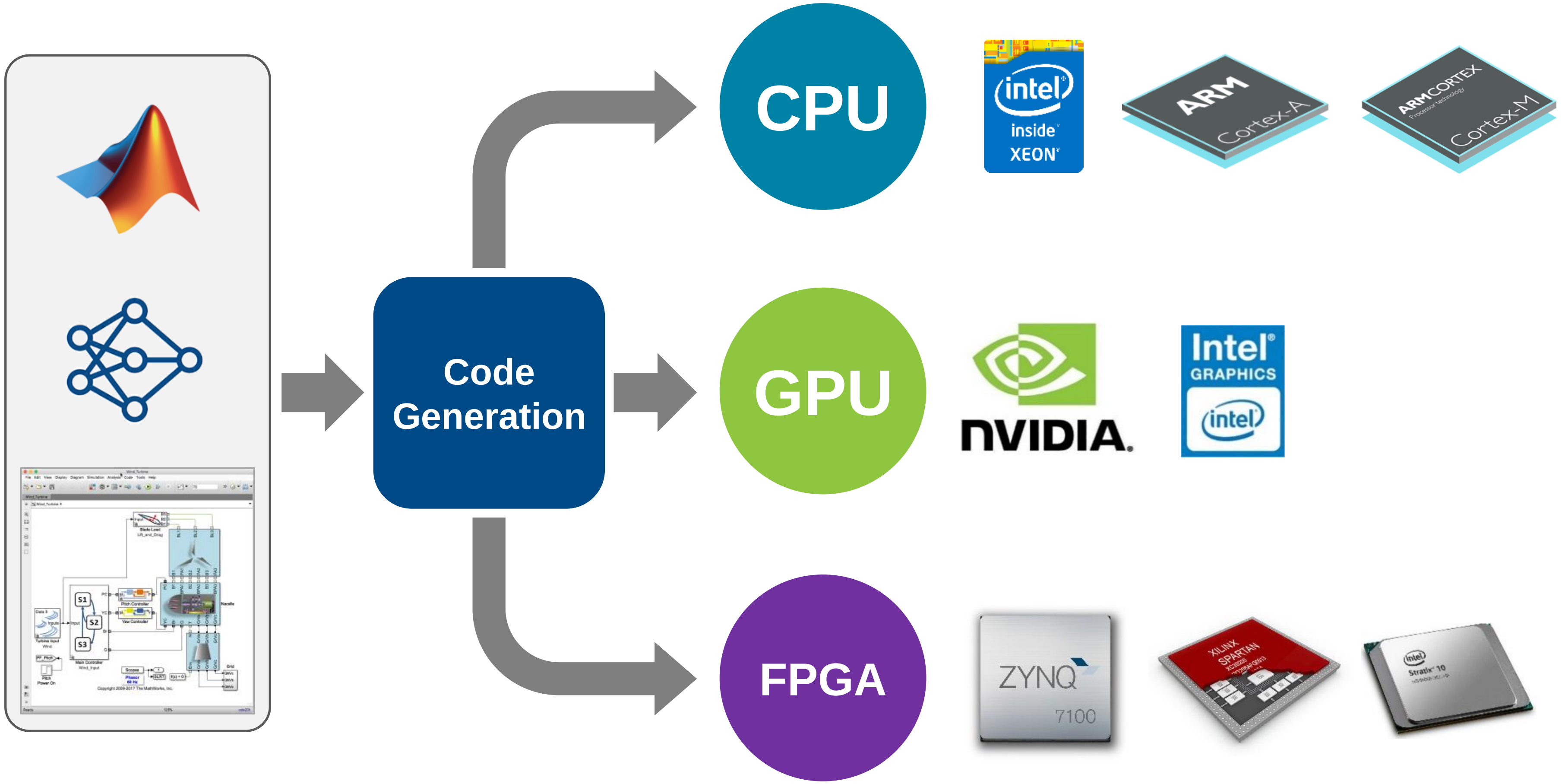

**Preprocessing, Feature Extraction, AI Model**

# Deploy to any processor with best-in-class performance



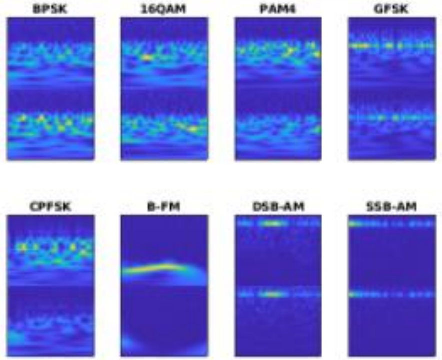

**Preprocessing, Feature Extraction, AI Model**

# Deploy to any processor with best-in-class performance




Preprocessing, Feature Extraction, AI Model

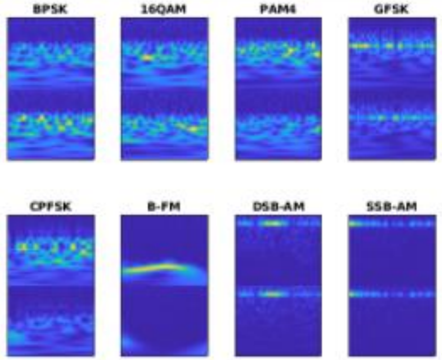
# Deploying complete AI algorithms to embedded processors, GPUs and FPGAs




**Modulation Classification Using Wavelet Analysis on NVIDIA Jetson**

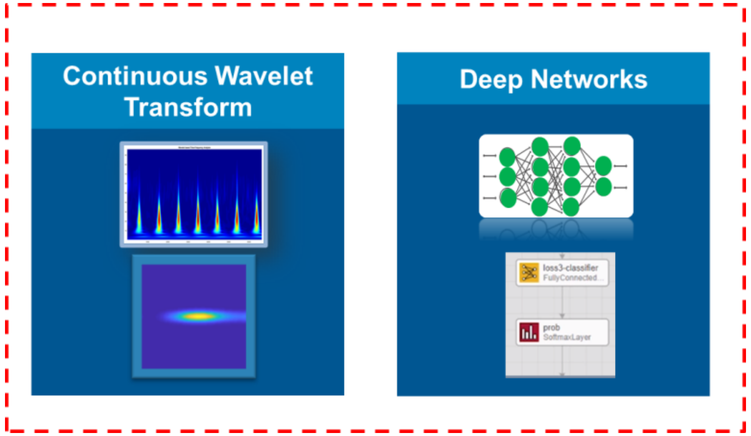
Generate and deploy a CUDA® executable that performs modulation classification using features extracted by the continuous wavelet

# Deploying complete AI algorithms to embedded processors, GPUs and FPGAs

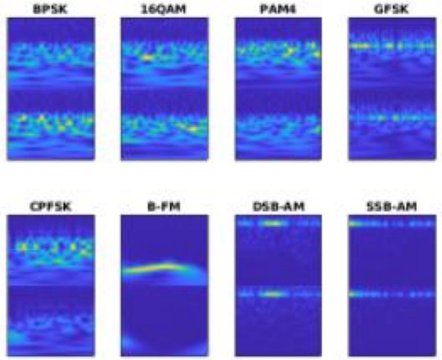



**Modulation Classification Using Wavelet Analysis on NVIDIA Jetson**

Generate and deploy a CUDA® executable that performs modulation classification using features extracted by the continuous wavelet

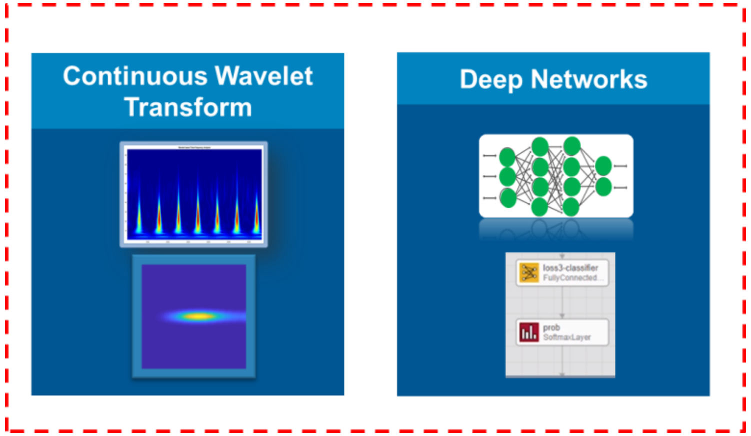


# Deploying complete AI algorithms to embedded processors, GPUs and FPGAs




**Modulation Classification Using Wavelet Analysis on NVIDIA Jetson**

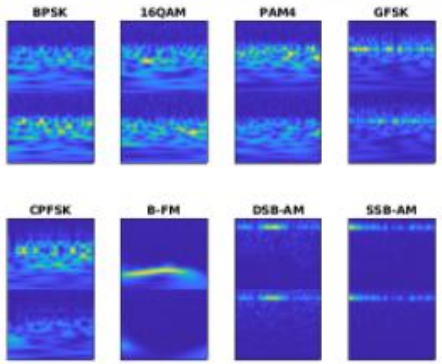
Generate and deploy a CUDA® executable that performs modulation classification using features extracted by the continuous wavelet




NVIDIA
CUDA
C/C++

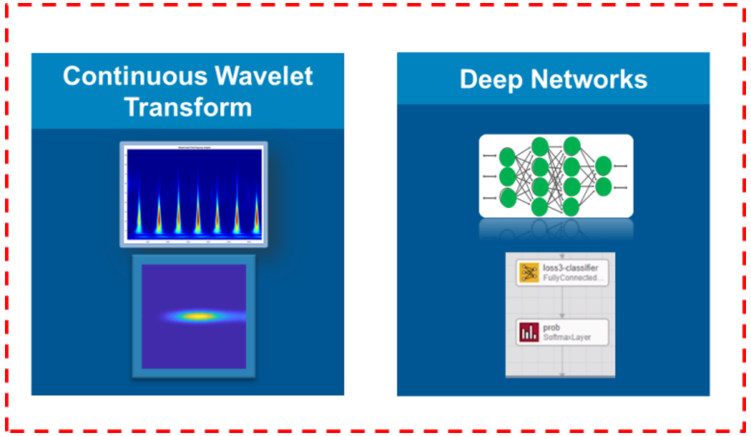
GPU Coder

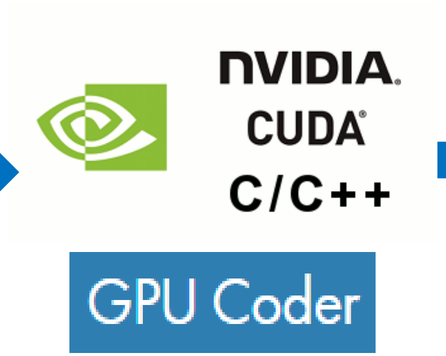
# Deploying complete AI algorithms to embedded processors, GPUs and FPGAs

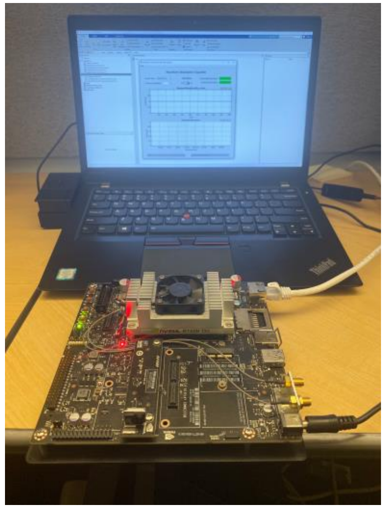



**Modulation Classification Using Wavelet Analysis on NVIDIA Jetson**

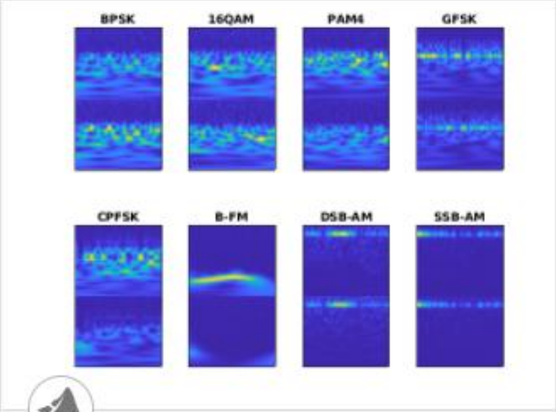
Generate and deploy a CUDA® executable that performs modulation classification using features extracted by the continuous wavelet



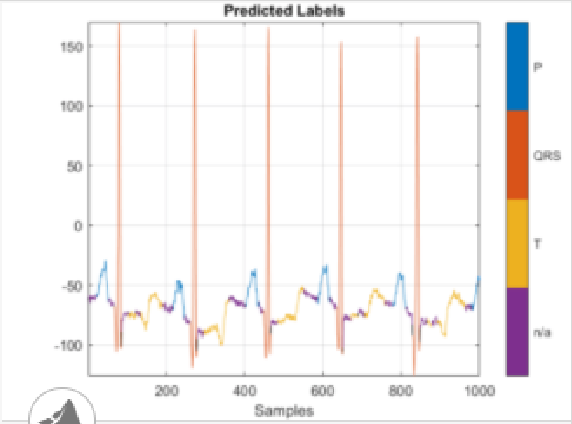



# Deploying complete AI algorithms to embedded processors, GPUs and FPGAs



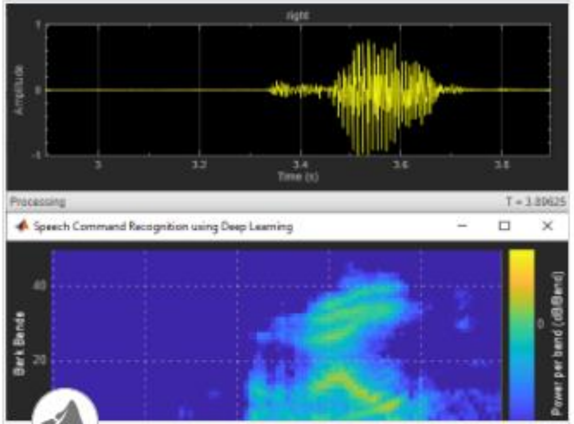

**Modulation Classification Using Wavelet Analysis on NVIDIA Jetson**

Generate and deploy a CUDA® executable that performs modulation classification using features extracted by the continuous wavelet



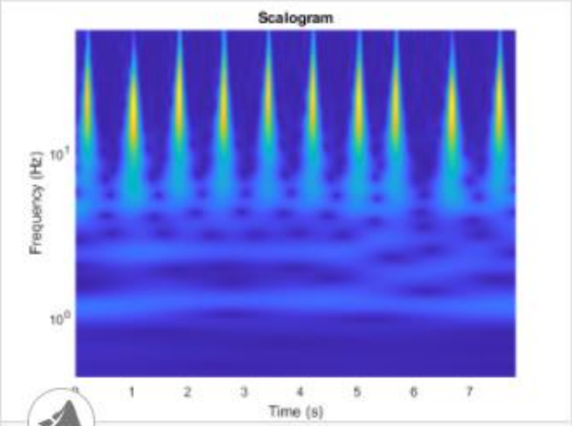

**Deploy Signal Segmentation Deep Network on Raspberry Pi**

Generate a MEX function and a standalone executable to perform waveform segmentation on a Raspberry Pi™.




**Speech Command Recognition Code Generation with Intel MK…**

Deploy feature extraction and a convolutional neural network (CNN) for speech command recognition on Intel® processors. To generate the




**Classify ECG Signals Using DAG Network Deployed To FPGA**

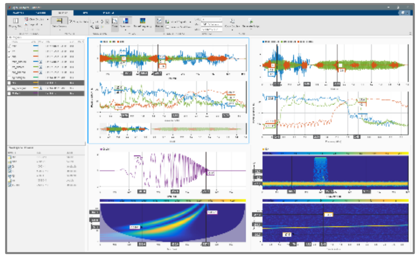
Classify human electrocardiogram (ECG) signals by deploying a trained directed acyclic graph (DAG) network.

# MATLAB supports the entire AI-driven system design

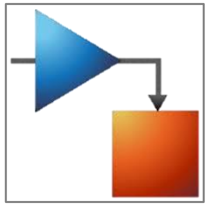
| Data Preparation | AI Modeling | Deployment |
| --- | --- | --- |
| Data cleansing and preparation | Model design and tuning | Embedded devices |
| Human insight | Hardware accelerated training | Enterprise systems |
| Simulation-generated data | Interoperability | Edge, cloud, desktop |

# MATLAB supports the entire AI-driven system design

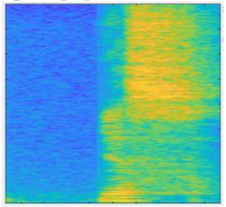
| Data Preparation | AI Modeling | Deployment |
| --- | --- | --- |
| Data cleansing and preparation | Model design and tuning | Embedded devices |
| Human insight | Hardware accelerated training | Enterprise systems |
| Simulation-generated data | Interoperability | Edge, cloud, desktop |



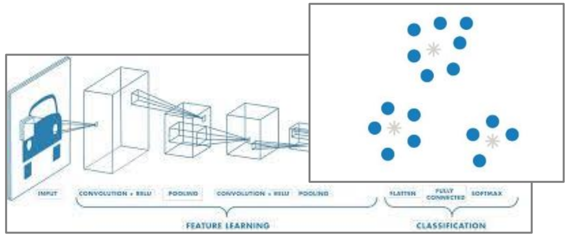
**Signal Processing apps**



**Generate Data**

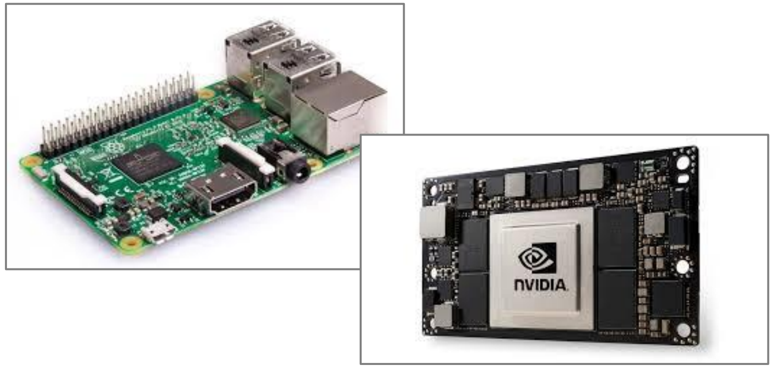


**Feature Extraction Techniques**




**Quickly build models**

**Accelerate training**



**Deploy to targets with code generation**

mathworks.com