Introduction to Deep Learning in Signal Processing & Communications with MATLAB

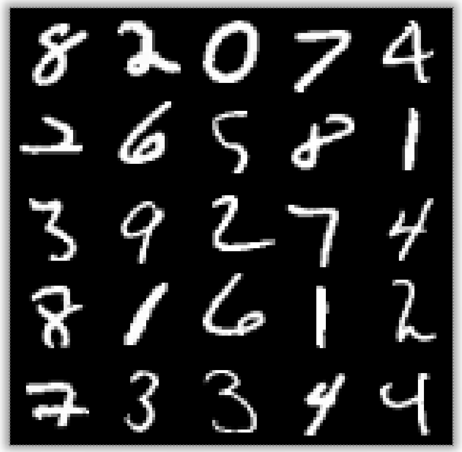
Dr. Amod Anandkumar
Pallavi Kar
Application Engineering Group, Mathworks India

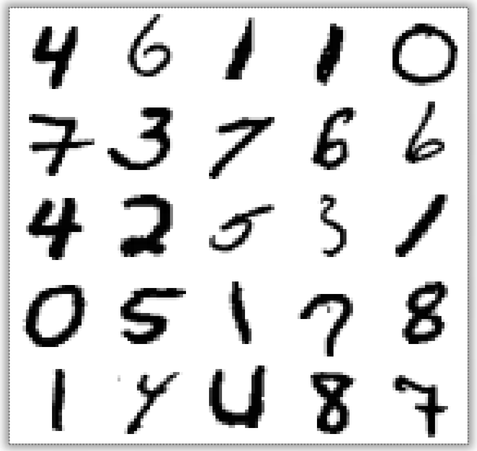
Different Types of Machine Learning

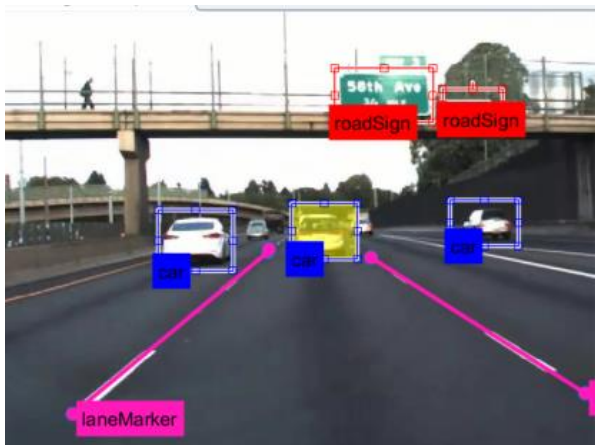
Type of Machine Learning | **Categories of Algorithms**

Machine Learning

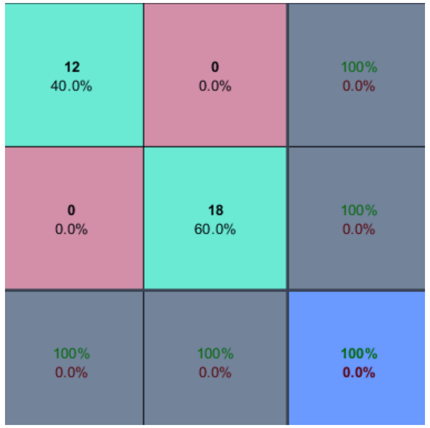
- **Supervised Learning**
  - Develop **predictive model** based on both **input and output** data
  - **Classification**
    - Output is a choice between classes *(True, False) (Red, Blue, Green)*
  - **Regression**
    - Output is a real number *(temperature, stock prices)*
- **Unsupervised Learning**
  - Discover an **internal representation** from **input data** only
  - **Clustering**
    - No output - *find natural groups and patterns from input data only*







What is Deep Learning?

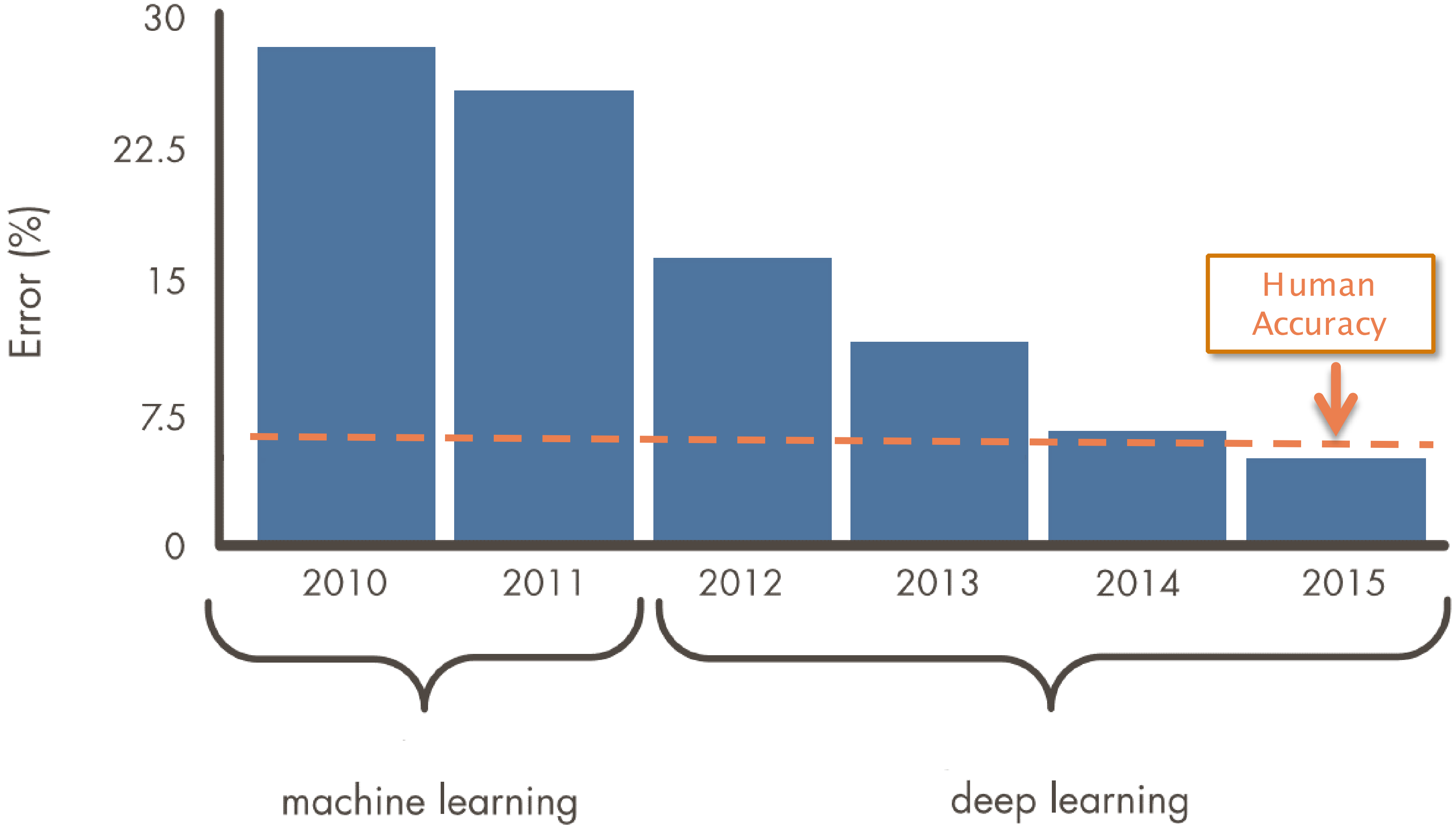


Deep learning is a type of supervised machine learning in which a model learns to perform classification tasks directly from images, text, or sound.

Deep learning is usually implemented using a **neural network.**

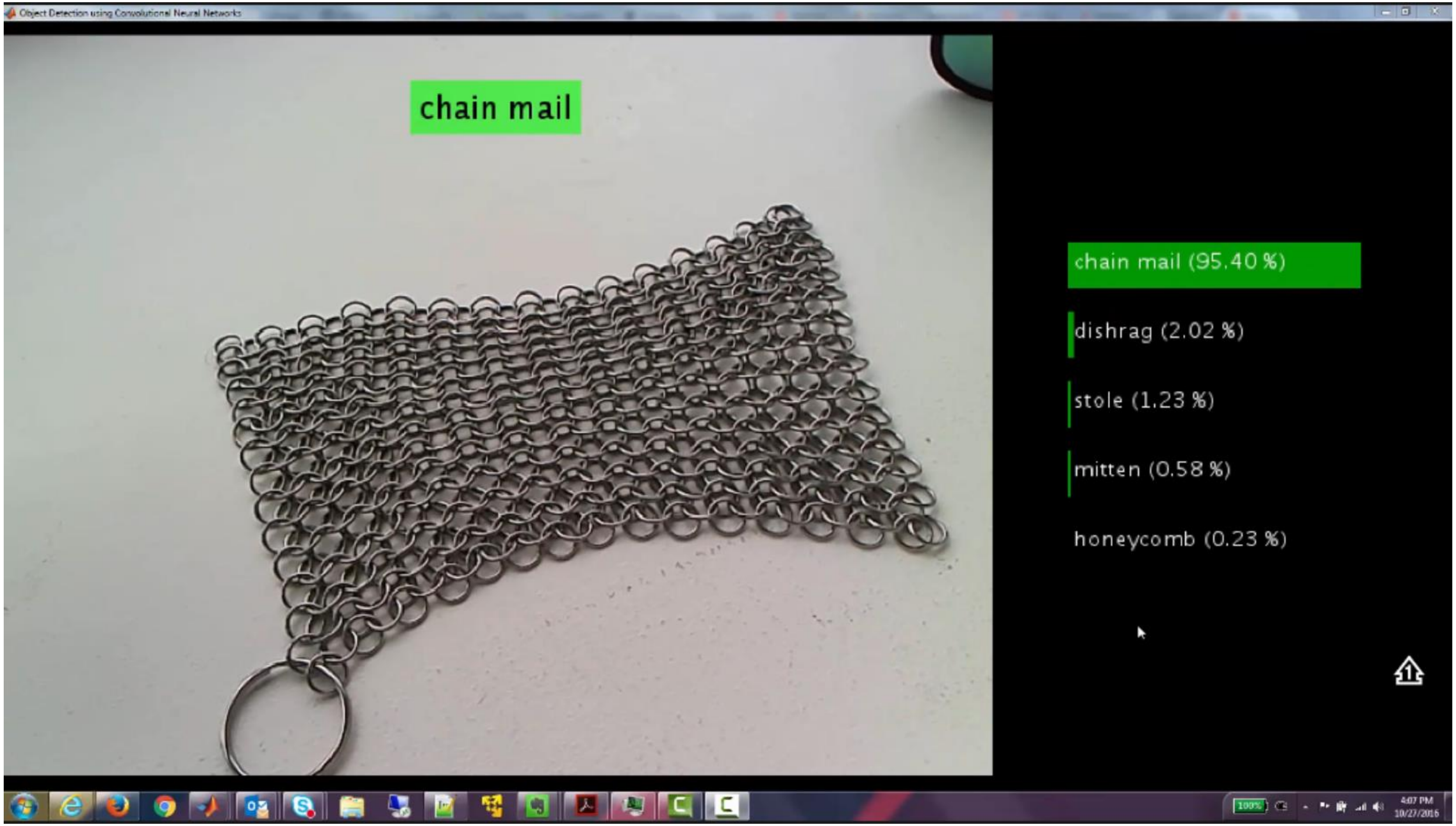
The term “deep” refers to the **number of layers** in the network—the more layers, the deeper the network.

Why is Deep Learning So Popular Now?



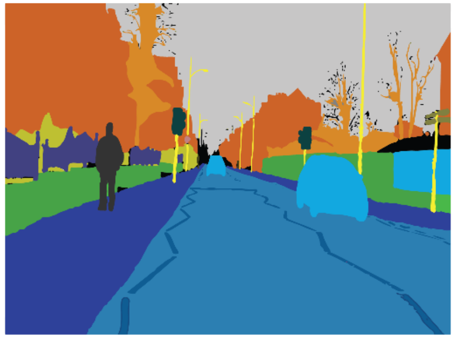
Source: ILSVRC Top-5 Error on ImageNet

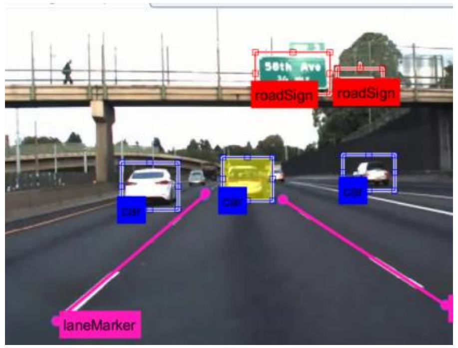
Vision applications have been driving the progress in deep learning producing surprisingly accurate systems

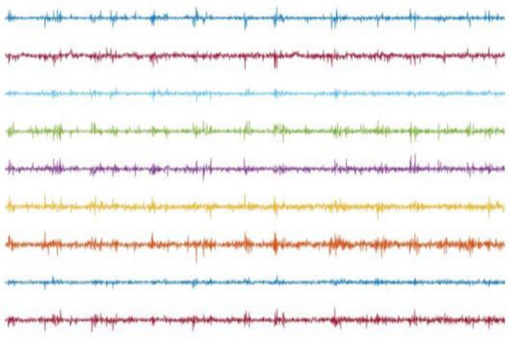


Deep Learning success enabled by:

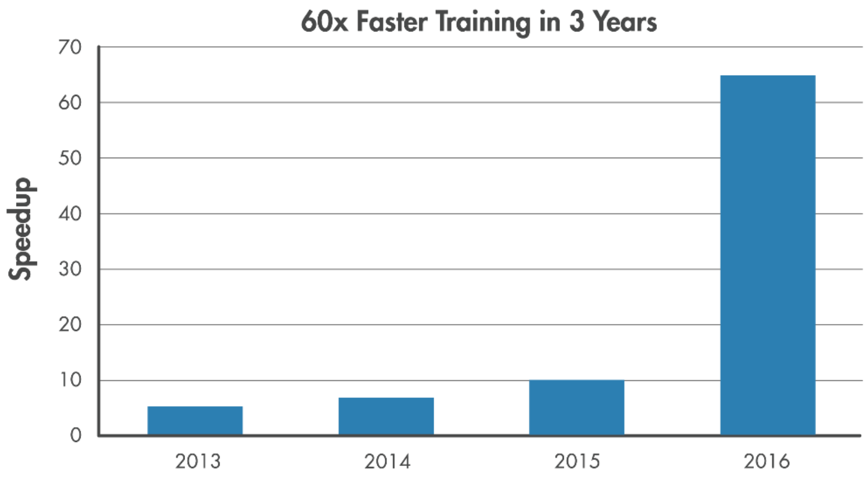
- Labeled public datasets







- Progress in GPU for acceleration



- World-class models and connected community

AlexNet
PRETRAINED MODEL

VGG-16
PRETRAINED MODEL

ResNet-50
PRETRAINED MODEL

ONNX Converter
MODEL CONVERTER

Caffe
IMPORTER

GoogLeNet
PRETRAINED MODEL

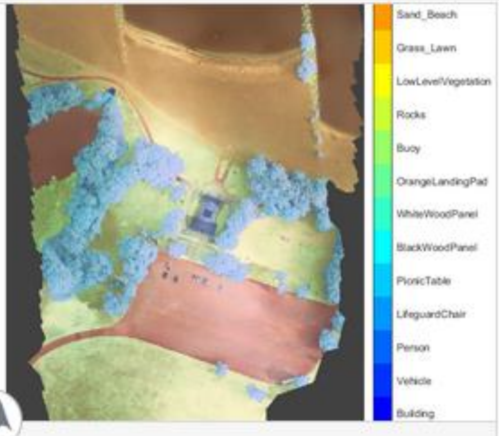
TensorFlow-Keras
IMPORTER

Inception-v3
MODELS

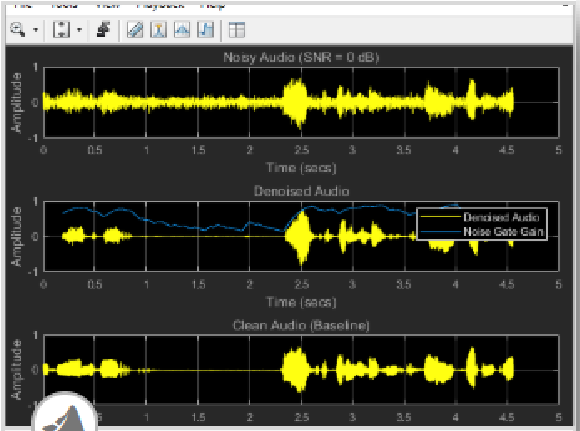
Deep Learning is Versatile

MATLAB Examples Available Here

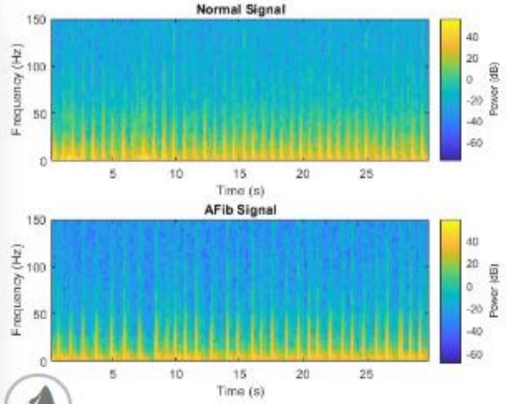
Object Detection Using Faster R-CNN Deep Learning



Semantic Segmentation of Multispectral Images Using Deep Learning

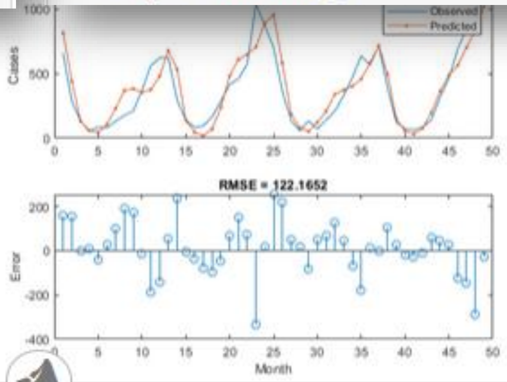


Denoise Speech Using Deep Learning Networks

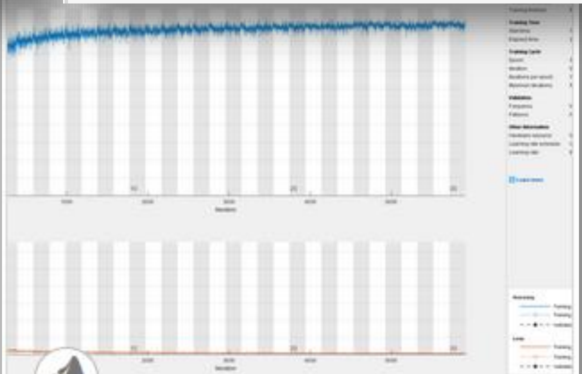


Classify ECG Signals Using Long Short-Term Memory Networks

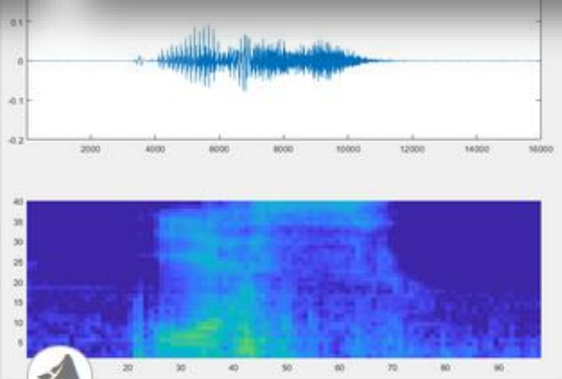
Classify Image Using GoogLeNet



Time Series Forecasting Using Deep Learning



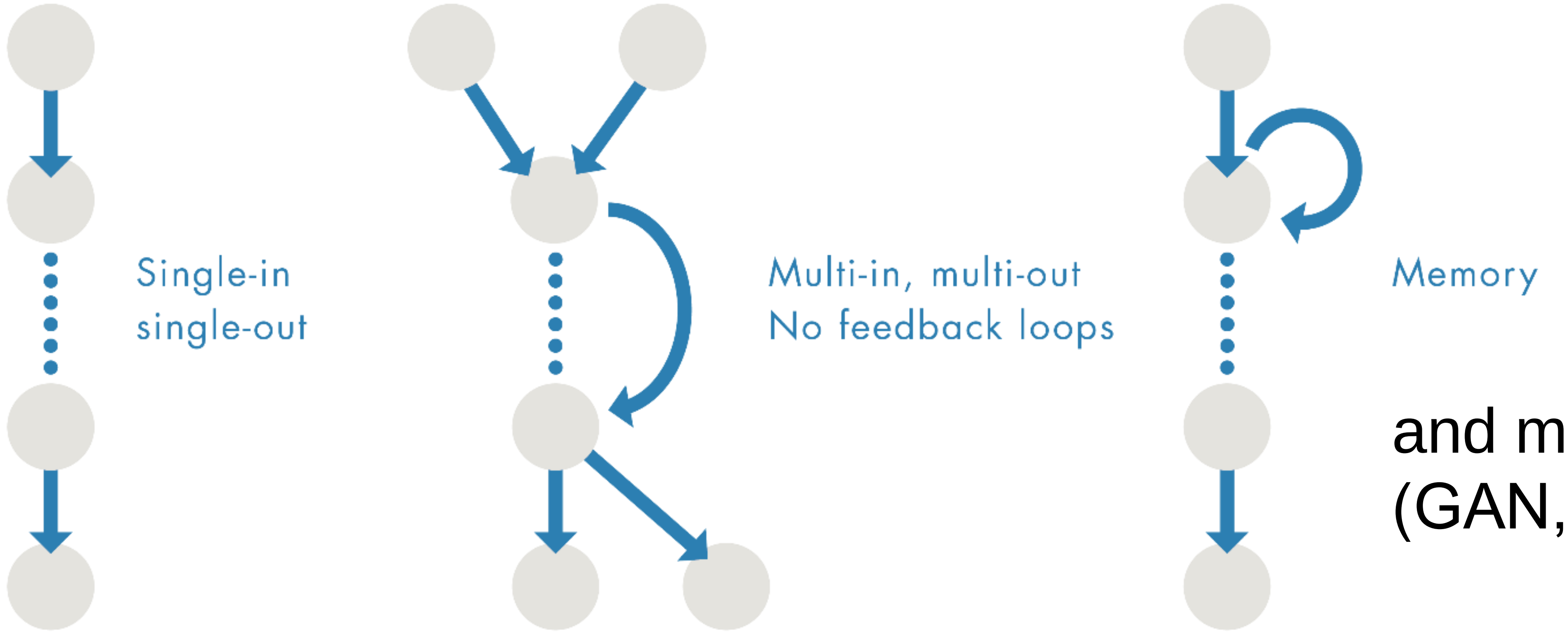
Classify Text Data Using Deep Learning



Deep Learning Speech Recognition

Many Network Architectures for Deep Learning

Series Network | Directed Acyclic Graph Network | Recurrent Network



and more
(GAN, DQN,…)

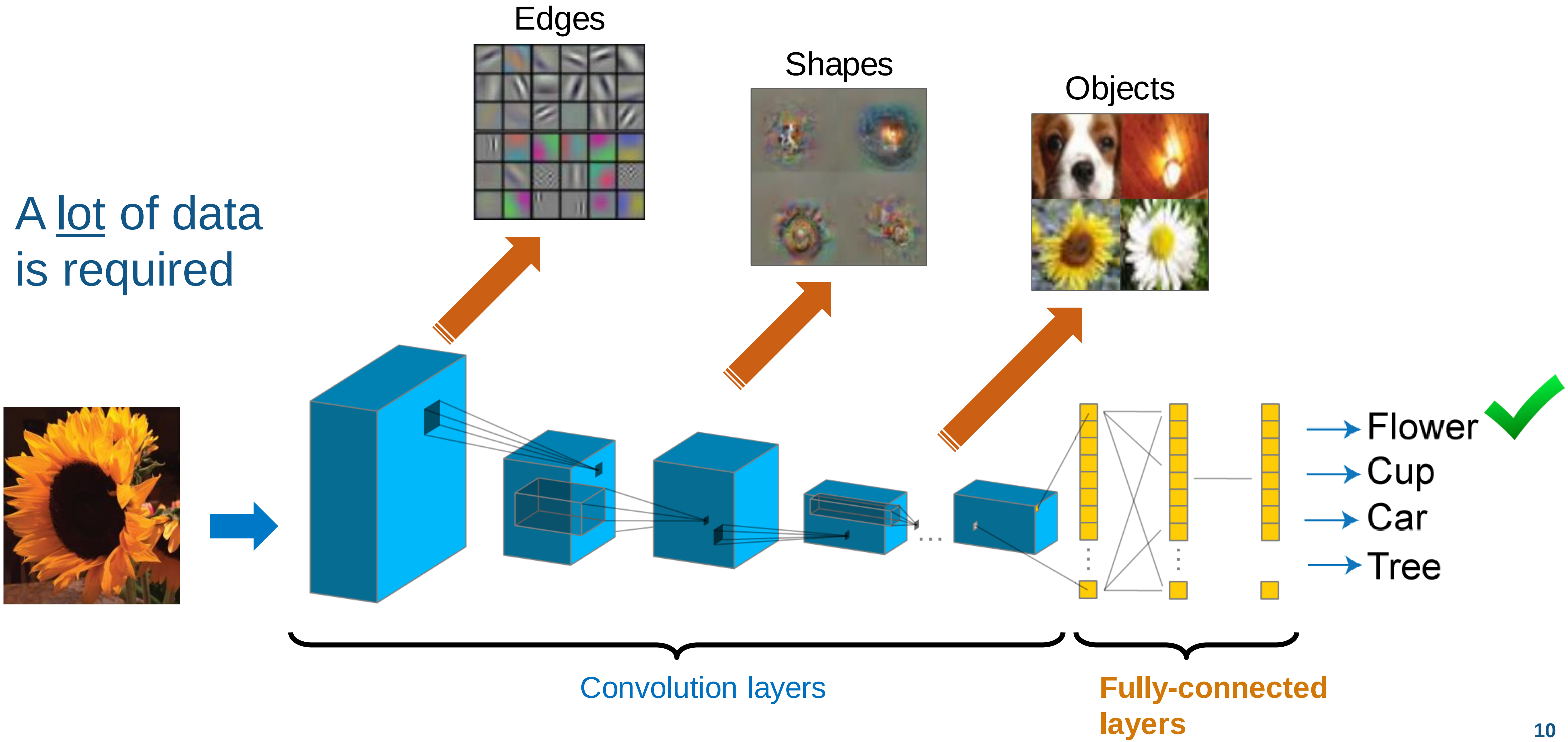
AlexNet
YOLO

ResNet
R-CNN

LSTM

Convolutional Neural Networks

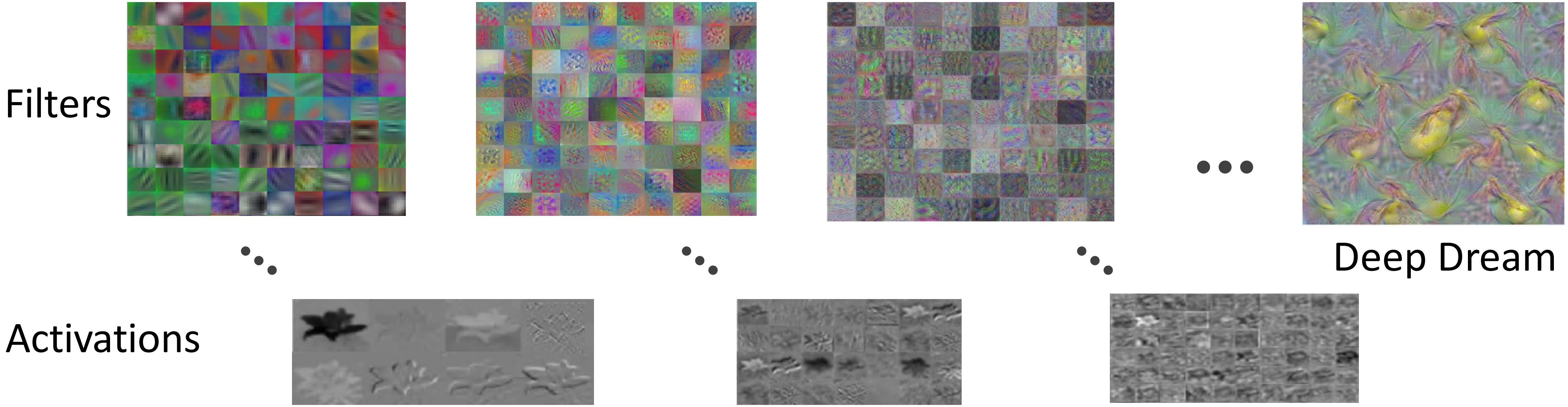
A lot of data is required



Deep Learning Inference in 4 Lines of Code

```
>> net = alexnet;
>> I = imread('peacock.jpg')
>> I1 = imresize(I,[227 227]);
>> classify(net,I1)
ans =
   categorical
    peacock
```

Understanding network behavior using visualizations



- Custom visualizations
  - Example: Class Activation Maps
    (See blog post)

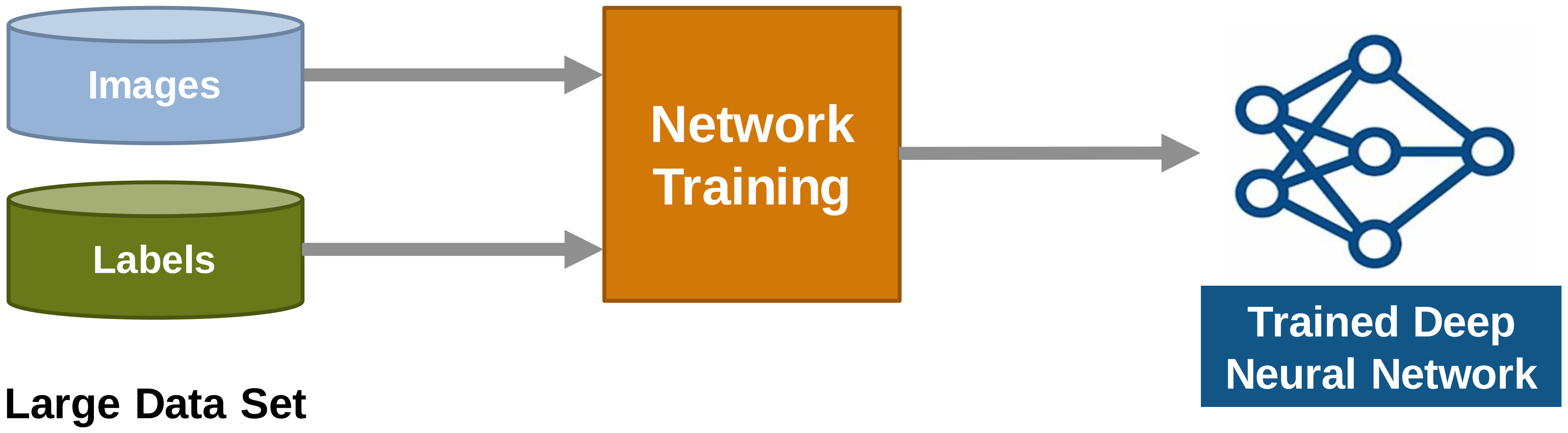
Visualization Technique – Deep Dream

```
deepDreamImage(...
      net, ‘fc8', channel,
     'NumIterations', 50, ...
     'PyramidLevels', 4,...
     'PyramidScale', 1.25);
```

Synthesizes images that strongly activate a channel in a particular layer

[Example Available Here]

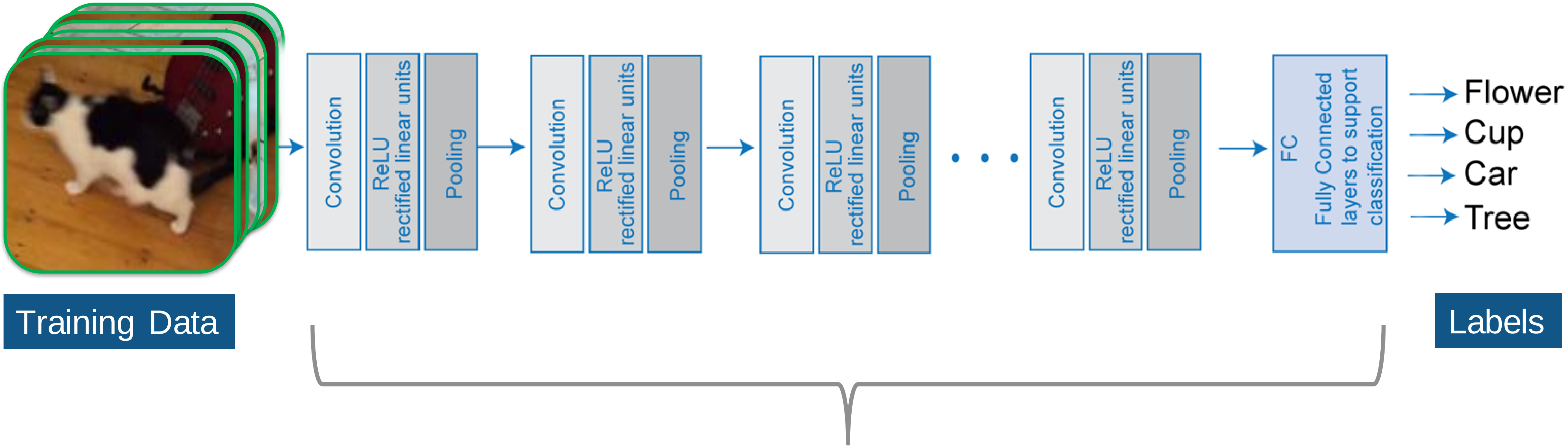
What is Training?



During training, neural network architectures learn features directly from the data without the need for manual feature extraction

What Happens During Training?

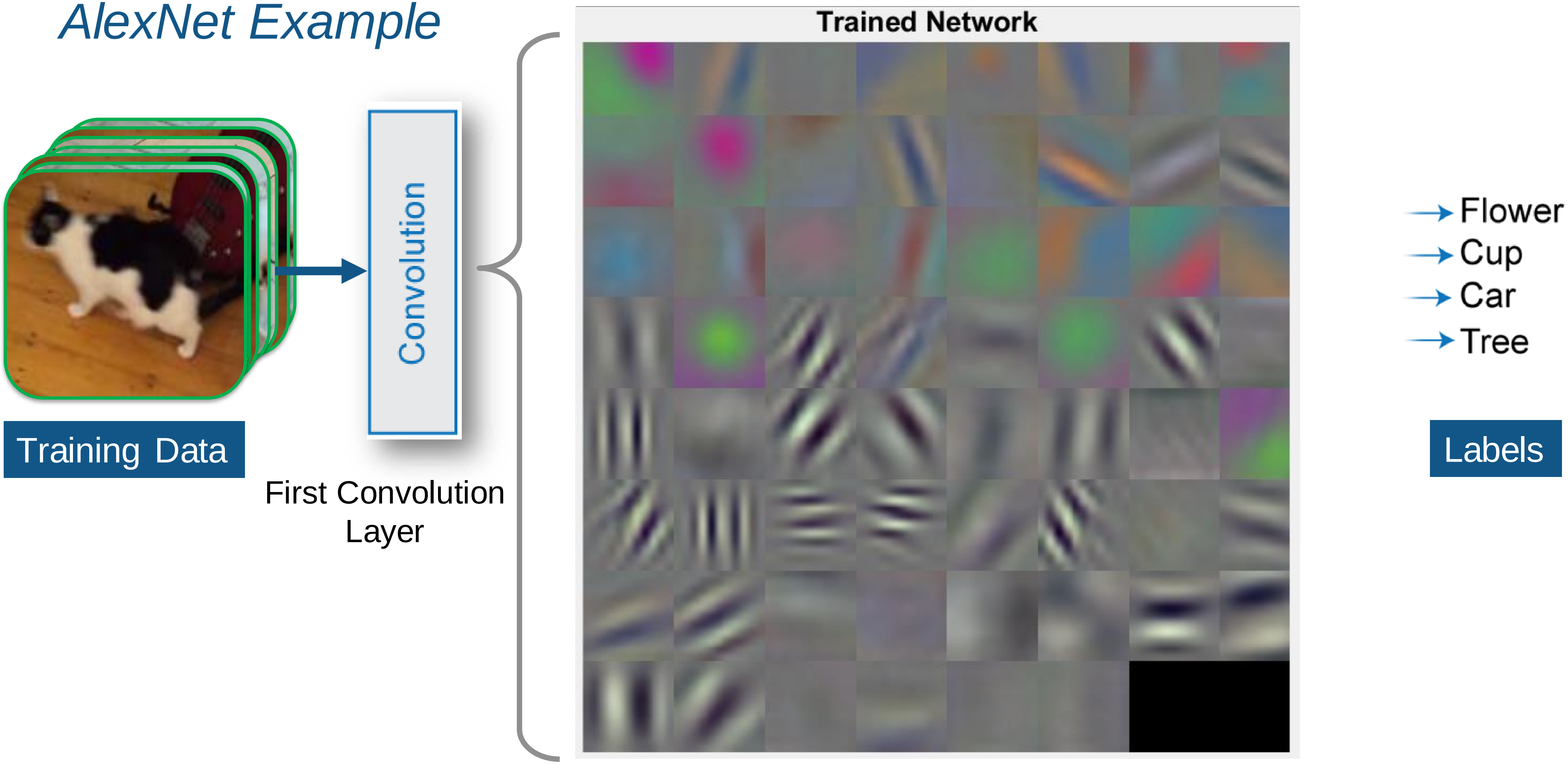
AlexNet Example



Layer weights are learned during training

Visualize Network Weights During Training

AlexNet Example



Visualize Features Learned During Training

AlexNet Example

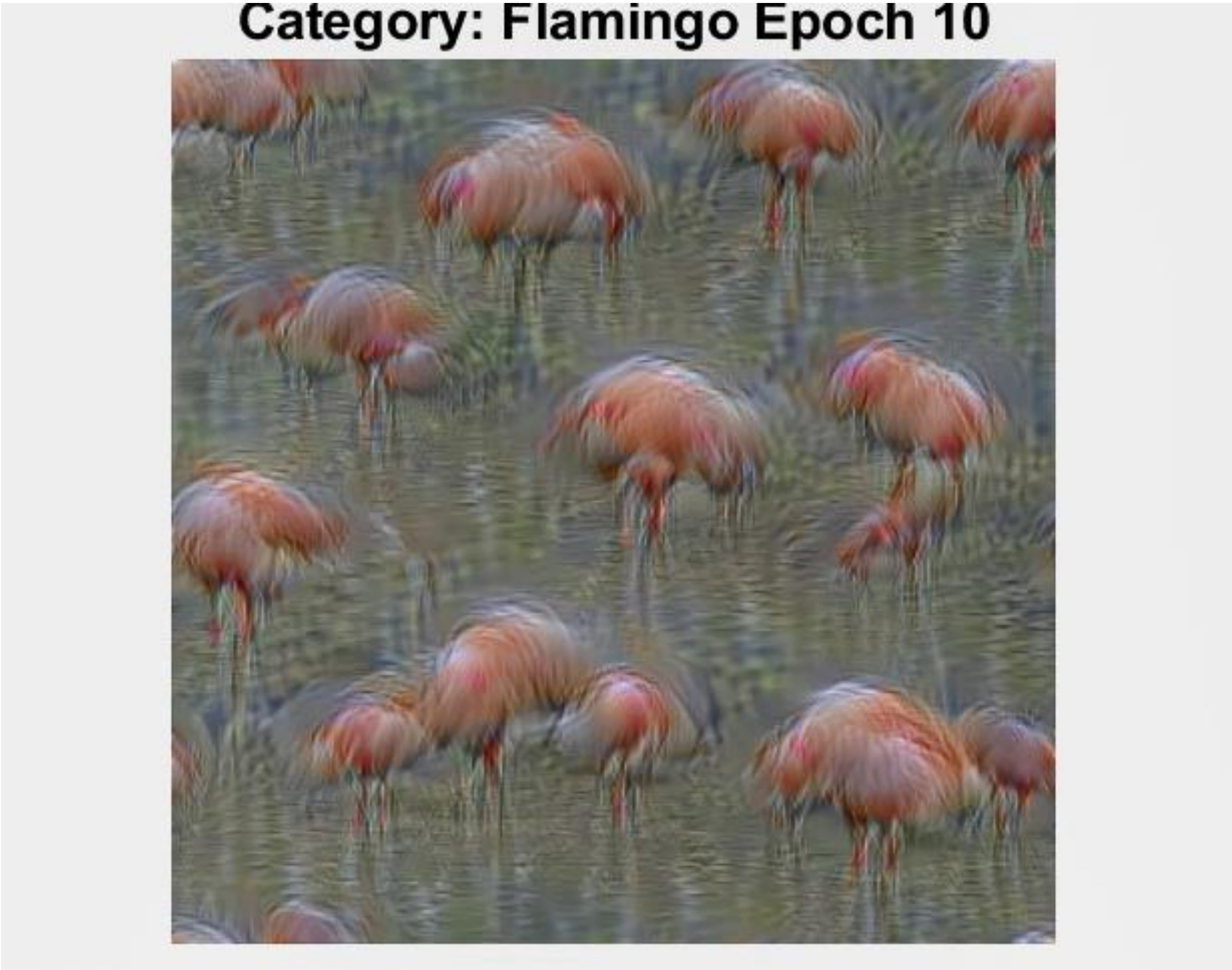
Sample Training Data

Features Learned by Network

Visualize Features Learned During Training

AlexNet Example

Sample Training Data



Features Learned by Network

Deep Learning Workflow

ACCESS AND EXPLORE DATA → **LABEL AND PREPROCESS DATA** → **DEVELOP PREDICTIVE MODELS** → **INTEGRATE MODELS WITH SYSTEMS**

Access and Explore Data

Files

Databases

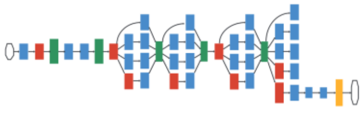
Sensors

Label and Preprocess Data

Data Augmentation/ Transformation

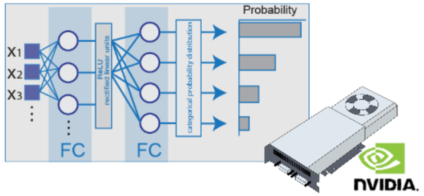
Labeling Automation

Import Reference Models



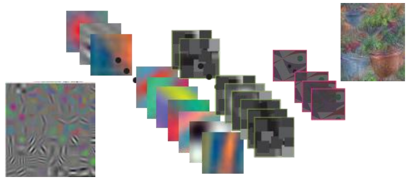
Develop Predictive Models

Hardware-Accelerated Training



Hyperparameter Tuning

Network Visualization

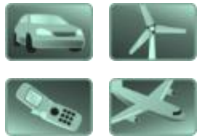


Integrate Models with Systems

Desktop Apps

Enterprise Scale Systems

Embedded Devices and Hardware



Deep Learning Challenges

Data

- Handling large amounts of data
- Labeling thousands of signals, images & videos
- Transforming, generating, and augmenting data (for different domains)

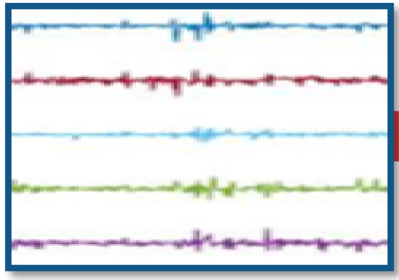
Training and Testing Deep Neural Networks

- Accessing reference models from research
- Understanding network behaviour
- Optimizing hyperparameters
- Training takes hours-days

Rapid and Optimized Deployment

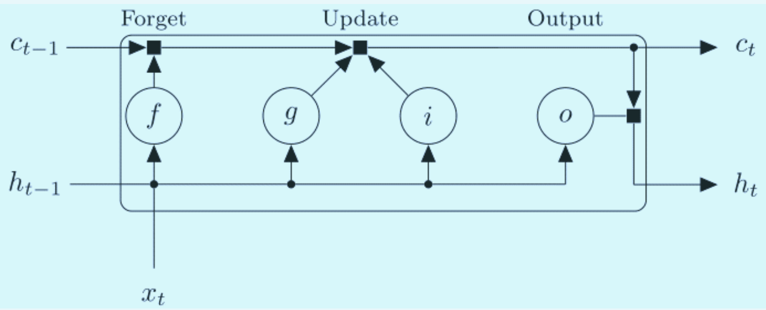
- Desktop, web, cloud, and embedded hardware

Working with Signal Data

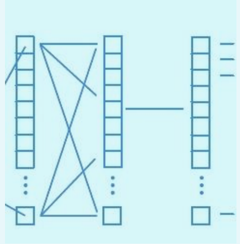


Signals

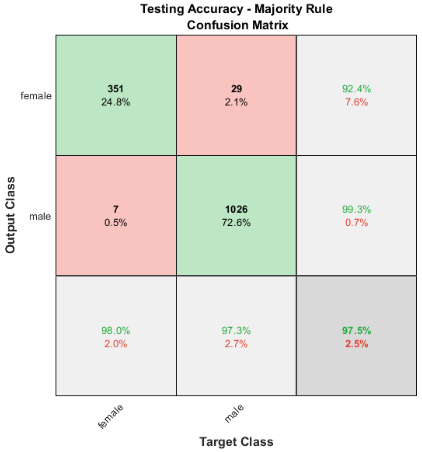
Labels



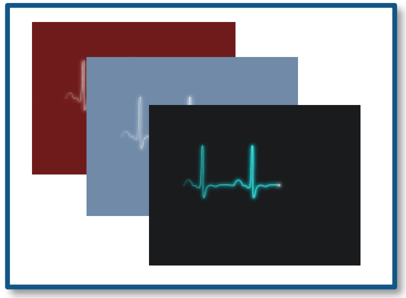
Sequence Layers

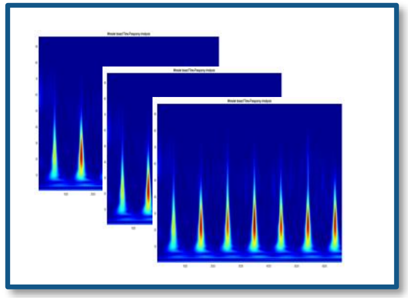


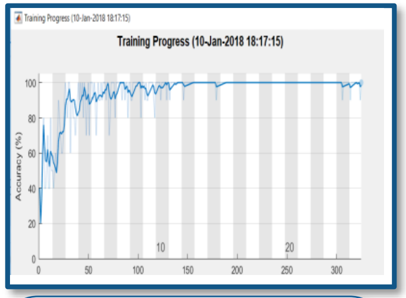
Fully Connected Layers

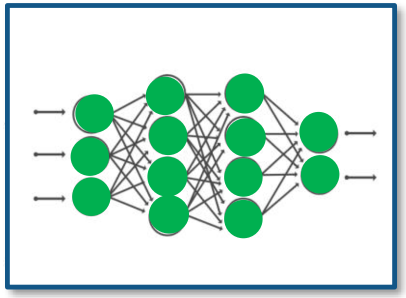


Trained Model









Signals

Transform

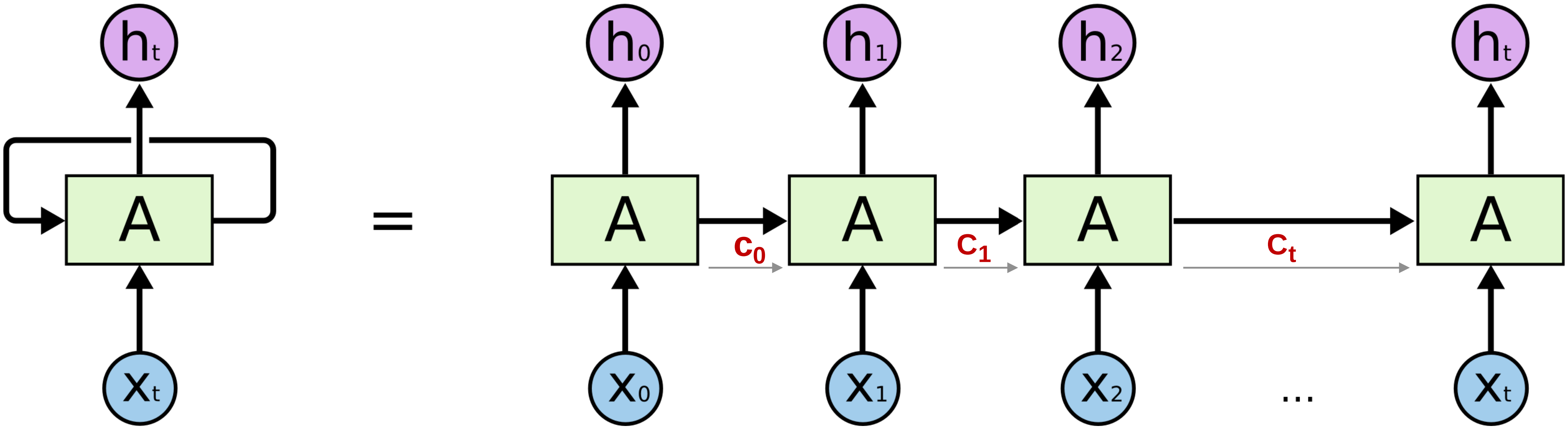
Labels

Train Convolutional Network (CNN)

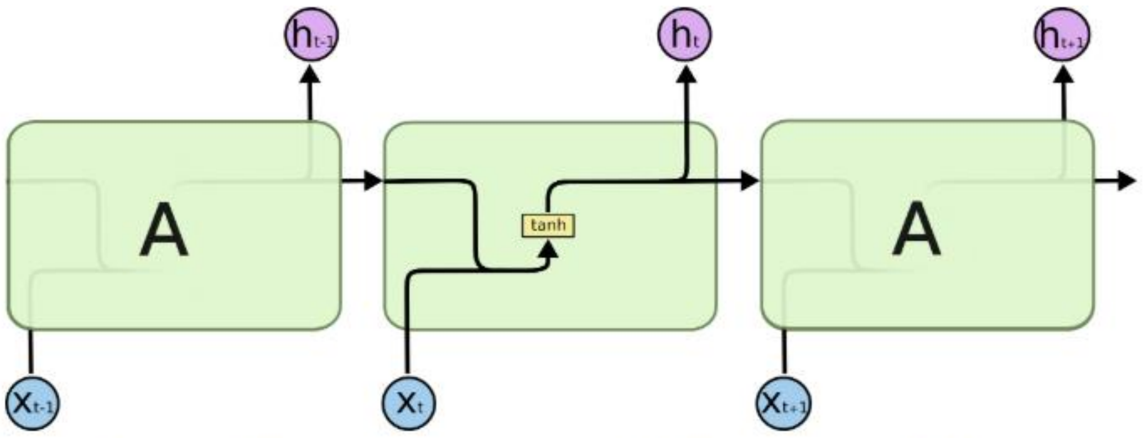
Trained Model

Long Short Term Memory Networks from RNNs

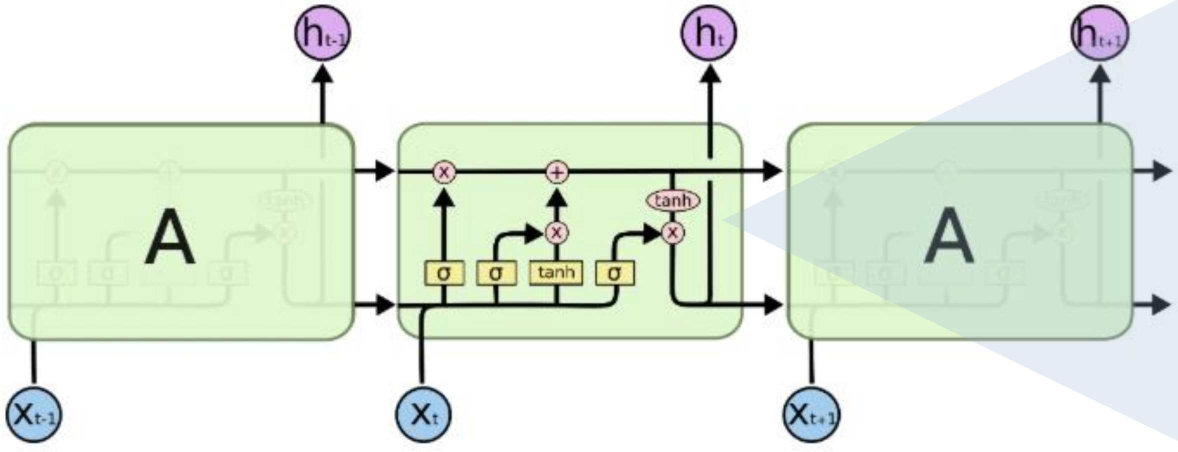
- Recurrent Neural Network that carries a memory cell throughout the process
- Sequence Problems – Long term dependency does not work well with RNNs



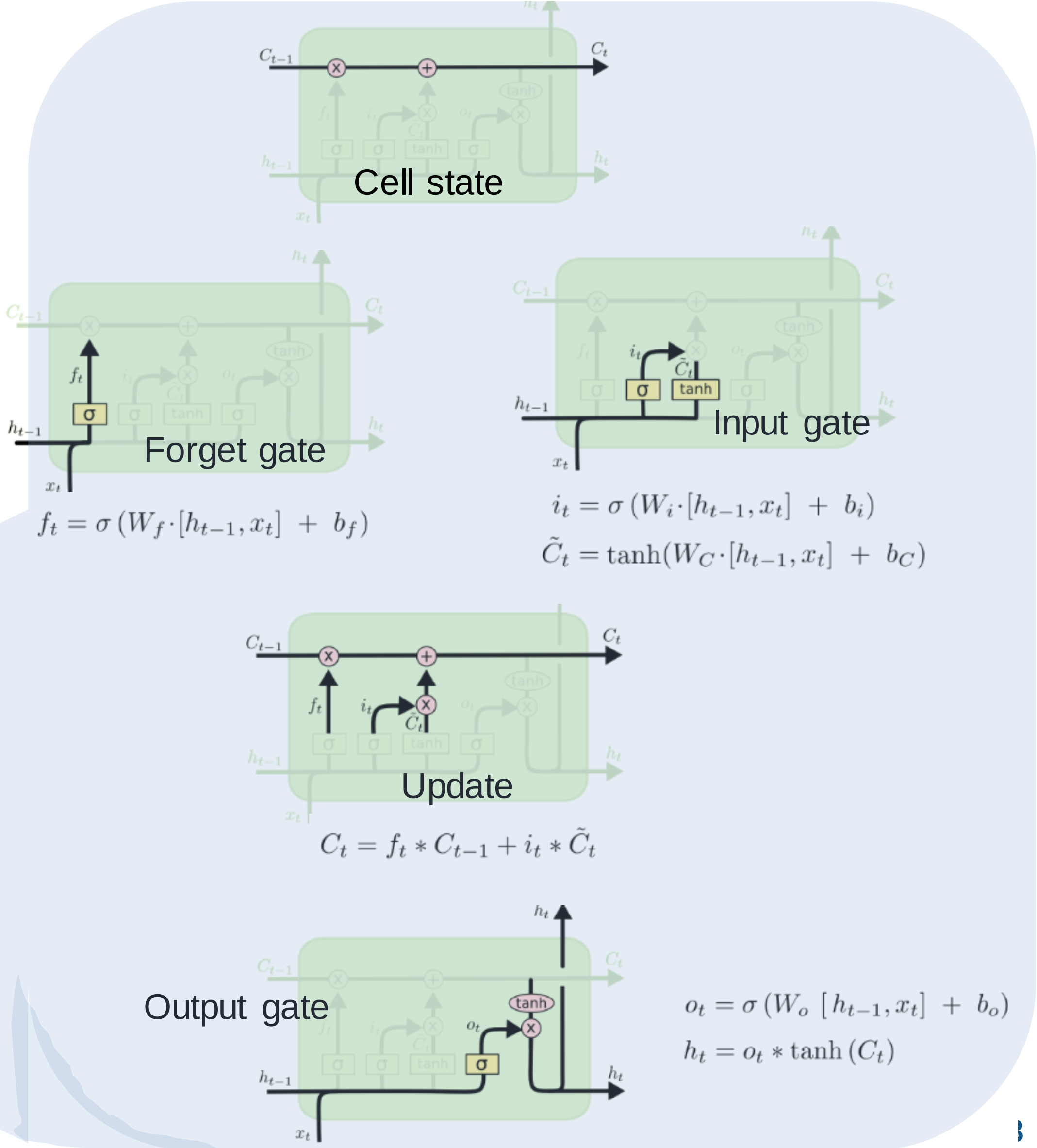
RNN to LSTM



The repeating module in a standard RNN contains a single layer.



The repeating module in an LSTM contains four interacting layers.



$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

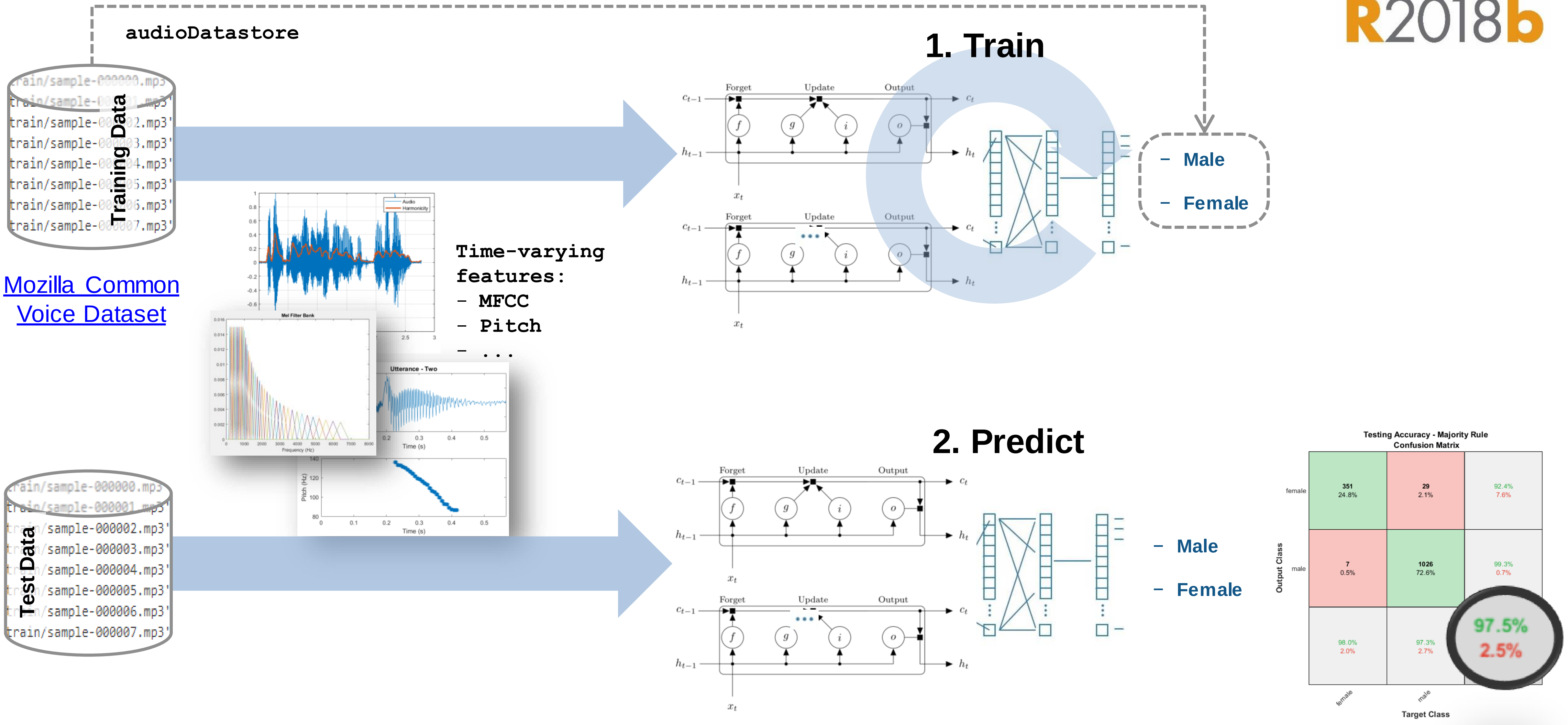
$$o_t = \sigma\left(W_o \; [h_{t-1}, x_t] \; + \; b_o\right)$$

$$h_t = o_t * \tanh\left(C_t\right)$$

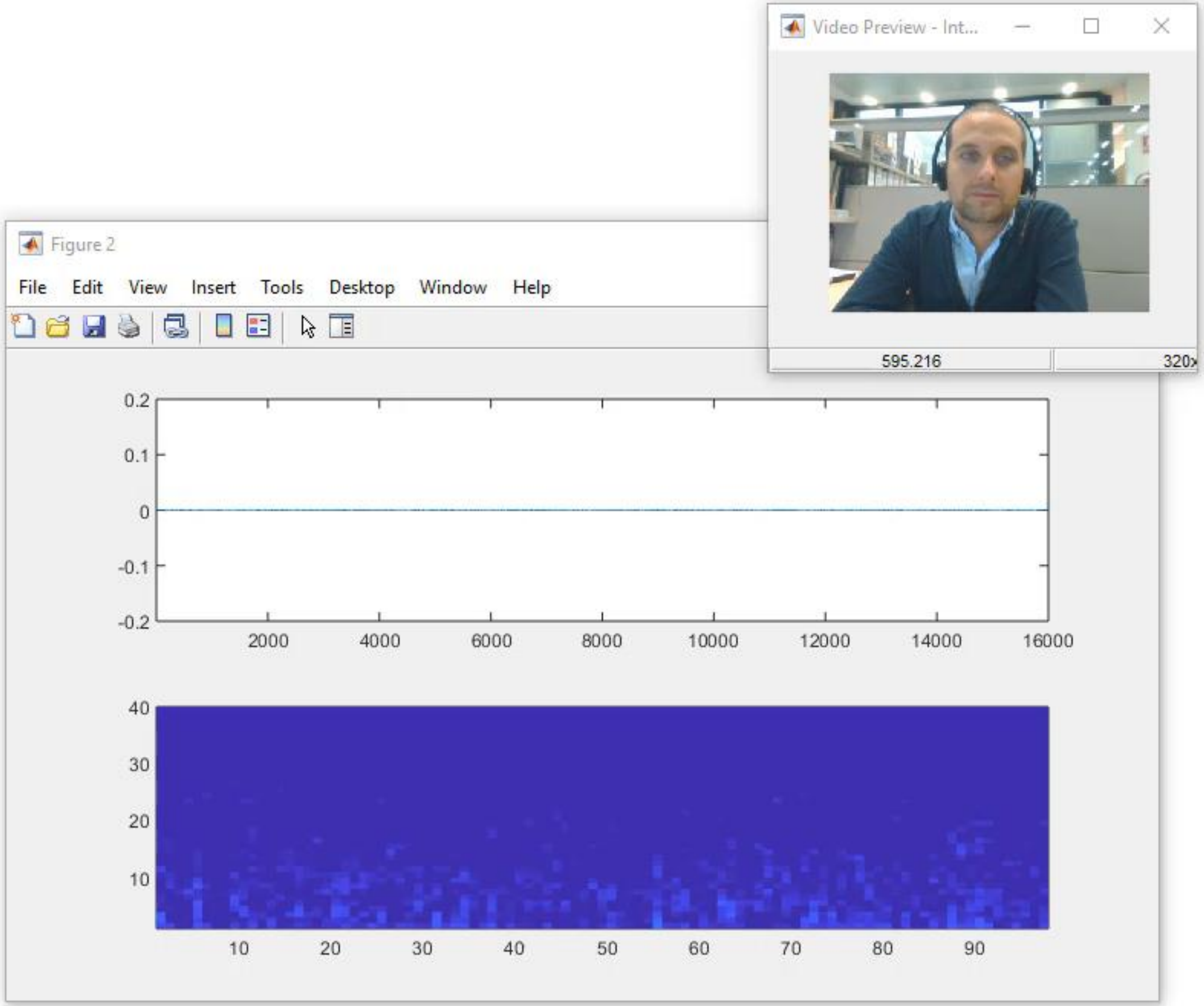
Example: Speaker Gender Recognition Using Deep Learning

LSTM Network for Audio based Speaker Classification

R2018b



Some audio and speech applications following CNN workflows



INTERSPEECH 2015

Convolutional Neural Networks for Small-footprint Keyword Spotting

Tara N. Sainath, Carolina Parada

Google, Inc. New York, NY, U.S.A
{tsainath,

Abstract

We explore using Convolutional Neural Networks (CNNs)
a small-footprint keyword spotting (KWS) task. CNNs ar
tractive for KWS since they have been shown to outperf
DNNs with far fewer parameters. We consider two diffe
applications in our work, one where we limit the numbe
multiplications of the KWS system, and another where we l
the number of parameters. We present new CNN architect
to address the constraints of each applications. We find tha
CNN architectures offer between a 27-44% relative impr
ment in false reject rate compared to a DNN, while fitting
the constraints of each application.

1. Introduction

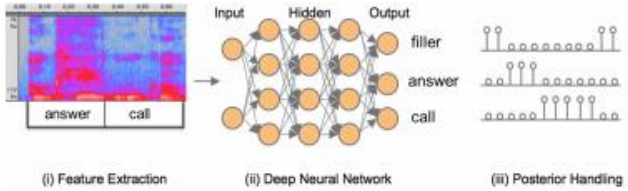


Figure 1: Framework of Deep KWS system, components from left to right: (i) Feature Extraction (ii) Deep Neural Network (iii) Posterior Handling

3. CNN Architectures

In this section, we describe CNN architectures as an alternative to the DNN described in Section 2. The feature extraction and posterior handling stages remain the same as Section 2.

3.1. CNN Description

A typical CNN architecture is shown in Figure 2. First, we are given an input signal $\mathbf{V} \in \Re^{t \times f}$, where t and f are the input feature dimension in time and frequency respectively. A weight

The second convolutional filter has a filte
quency, and no max-pooling is performed
For example, in our task if we want
of parameters below 250K, a typical arc
tecture is shown in Table 1. We will refe
as `cnn-trad-fpool3` in this paper. T
convolutional, one linear low-rank and on
tion 5, we will show the benefit of this a
particularly the pooling in frequency, com
However, a main issue with this arc
number of multiplies in the convolutional
acerbated in the second layer because of
put, spanning across time, frequency and
type of architecture is infeasible for pow
footprint KWS tasks where multiplies are
even if our application is limited by para
tiplies, other architectures which pool in
suited for KWS. Below we present alter
tures to address the tasks of limiting param

type	m	r	n	p	q
conv	20	8	64	1	3

model	layer	m	r	n	s	q	Params
`cnn-tstride2`	conv	16	8	78	**2**	3	10.0K
	conv	9	4	78	1	1	219.0K
	lin	-	-	32	-	-	20.0K
`cnn-tstride4`	conv	16	8	100	**4**	3	12.8K
	conv	5	4	78	1	1	200.0K
	lin	-	-	32	-	-	25.6K
`cnn-tstride8`	conv	16	8	126	**8**	3	16.1K
	conv	5	4	78	1	1	190.5K
	lin	-	-	32	-	-	32.2K

Table 4: CNNs for Striding in Time

3.4.2. Pooling in Time

An alternative to striding the filter in time is to pool in time, by a non-overlapping amount. Table 5 shows configurations as we vary the pooling in time p. We will refer to these architectures as `cnn-tpool2` and `cnn-tpool4`. For simplicity, we have omitted certain variables held constant for all experiments, namely time and frequency stride $s = 1$ and $v = 1$. Notice that by pooling in time, we can increase the number of feature maps n to keep the total number of parameters constant.

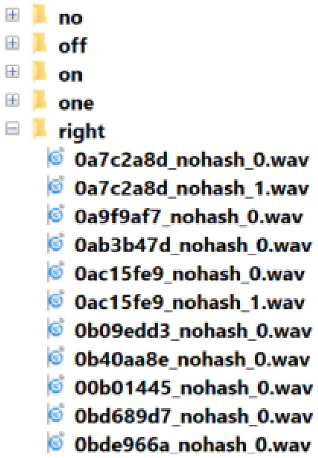
model	layer	m	r	n	p	q	Params
`cnn-tpool2`	conv	21	8	94	**2**	3	5.6M
	conv	6	4	94	1	1	1.8M
	lin	-	-	32	-	-	65.5K
`cnn-tpool3`	conv	15	8	94	**3**	3	7.1M
	conv	6	4	94	1	1	1.6M
	lin	-	-	32	-	-	65.5K

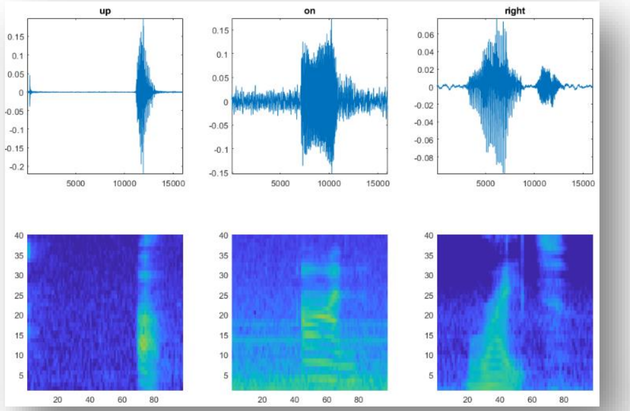
Table 5: CNNs for Pooling in Time

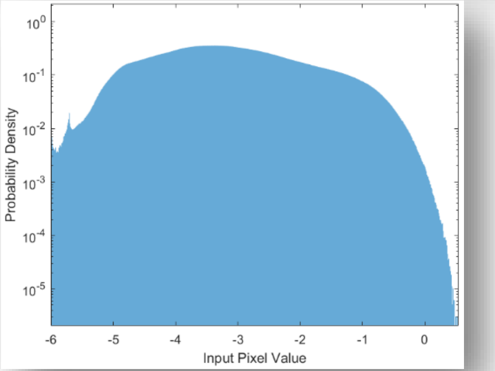
https://www.isca-speech.org/archive/interspeech_2015/papers/i15_1478.pdf

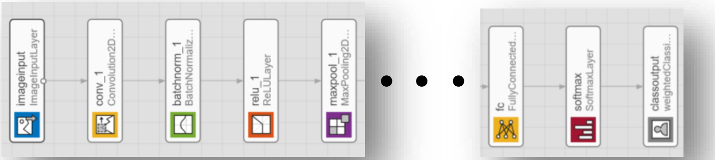
Solution2: Speech Command Recognition with Deep Learning(MATLAB)

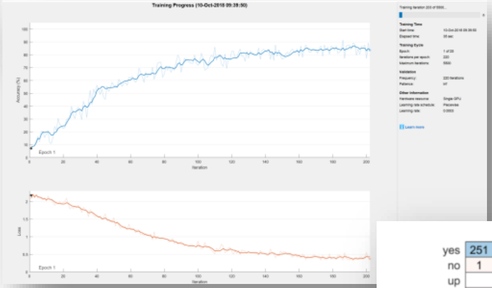
Raw Data → **Preprocessed Data** → **Models and Algorithms** → **Integrated Analytic Systems**

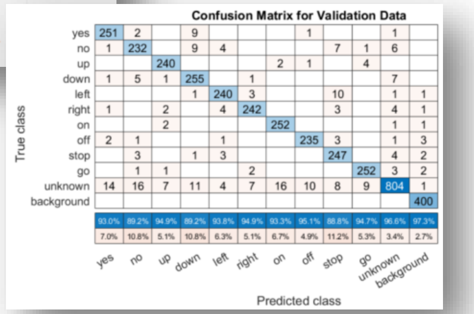


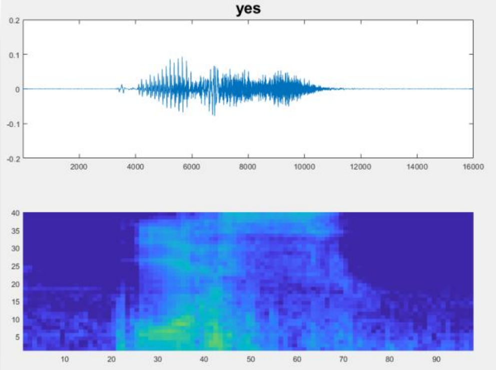


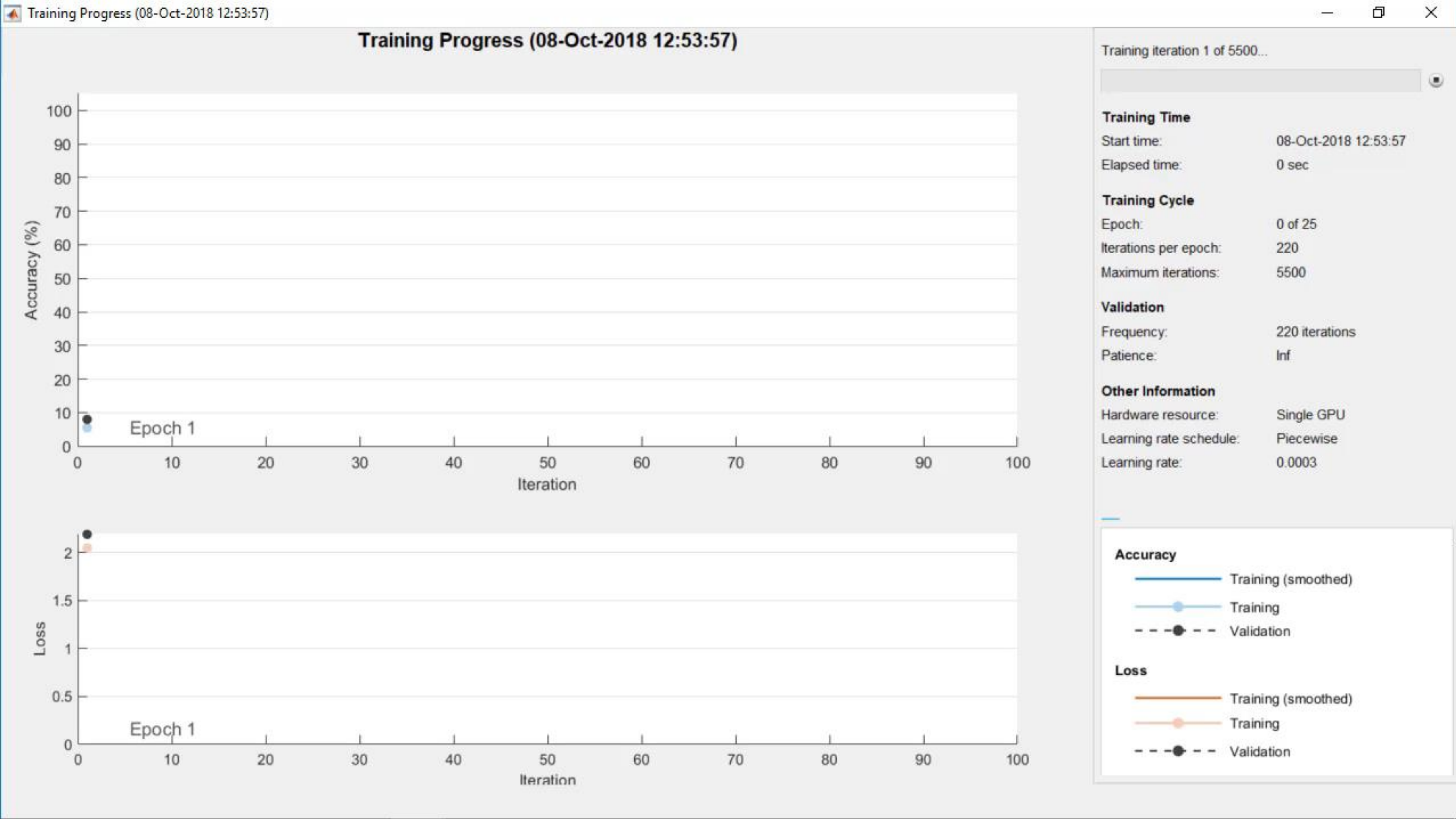


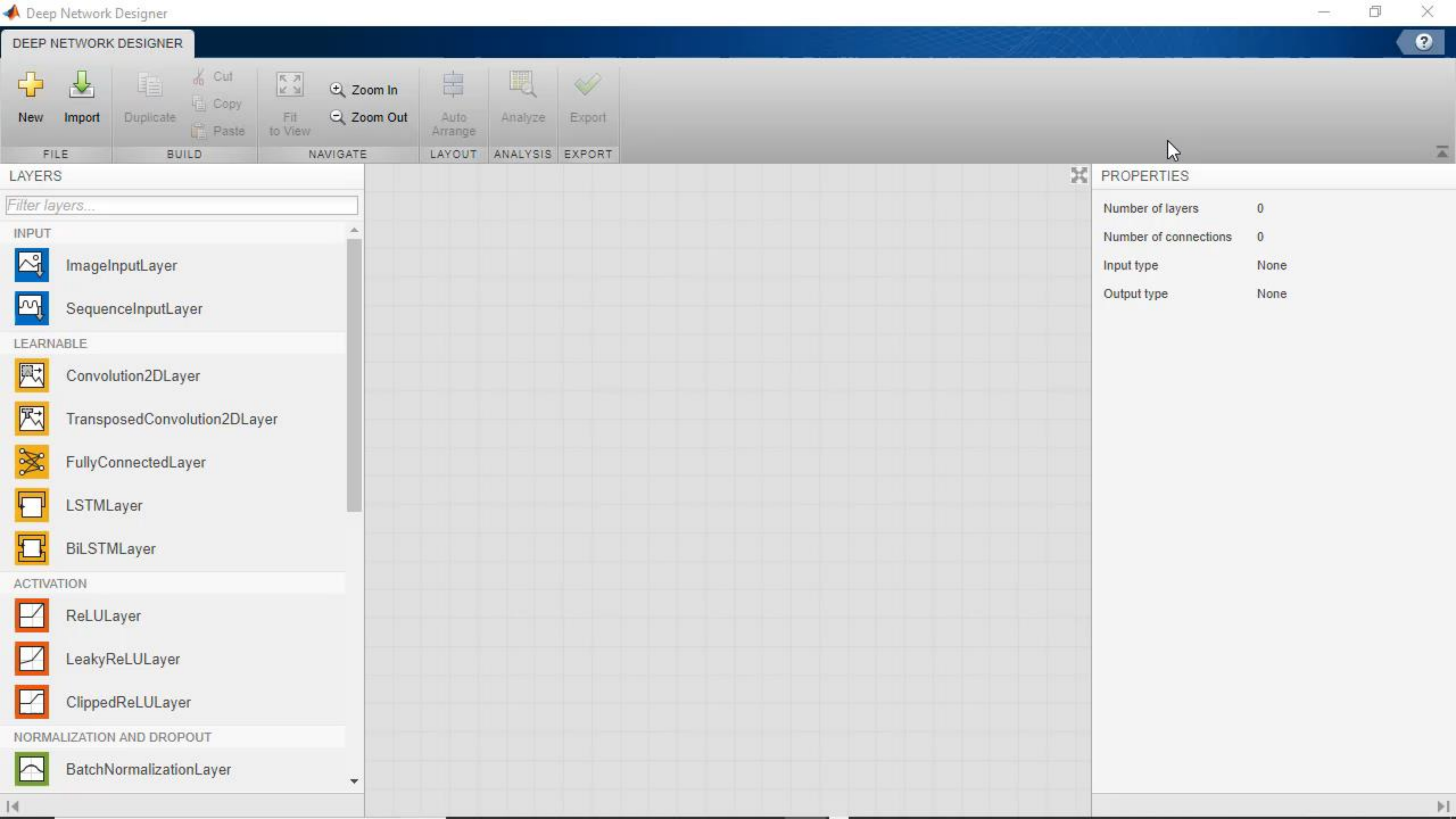


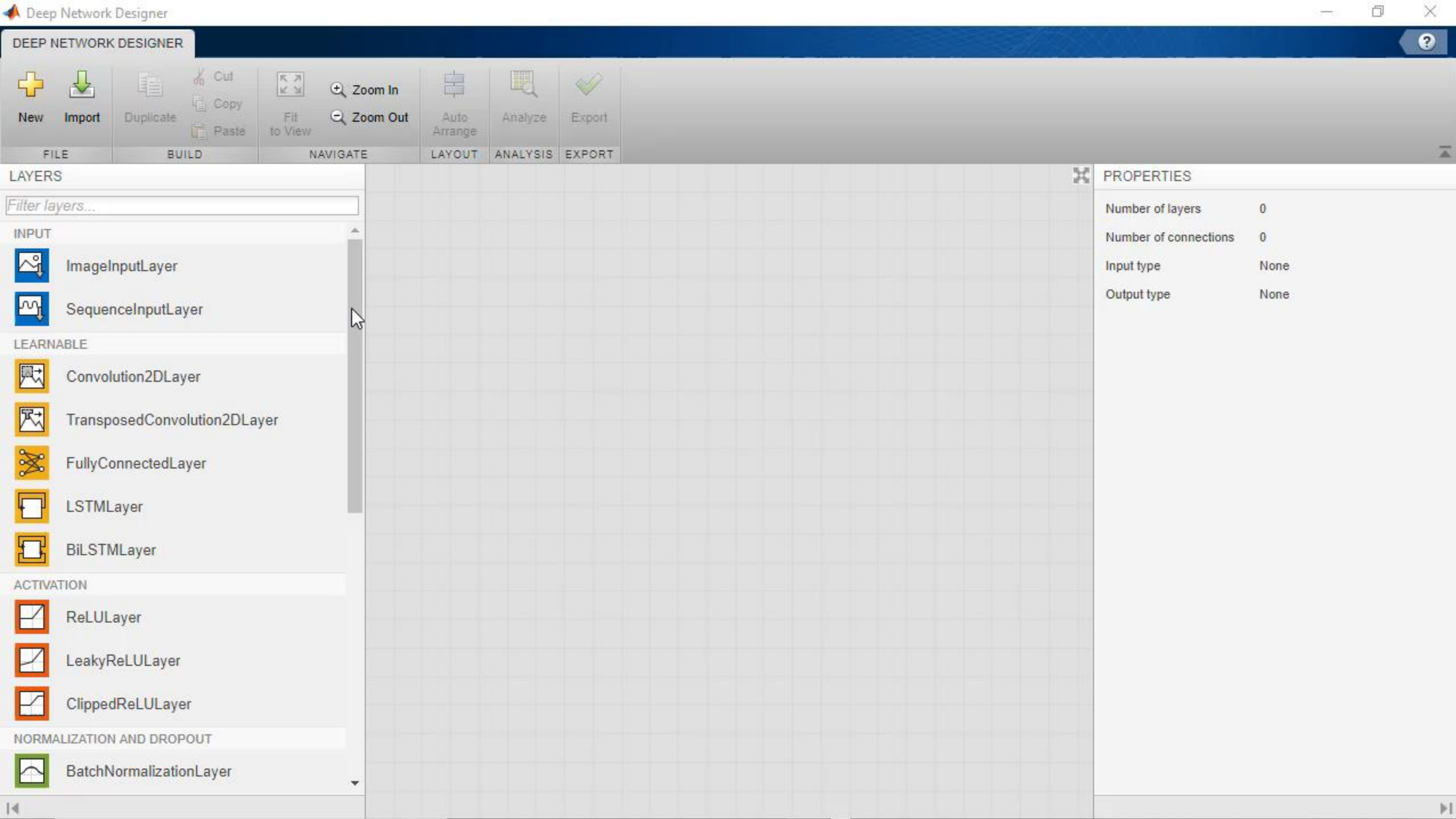


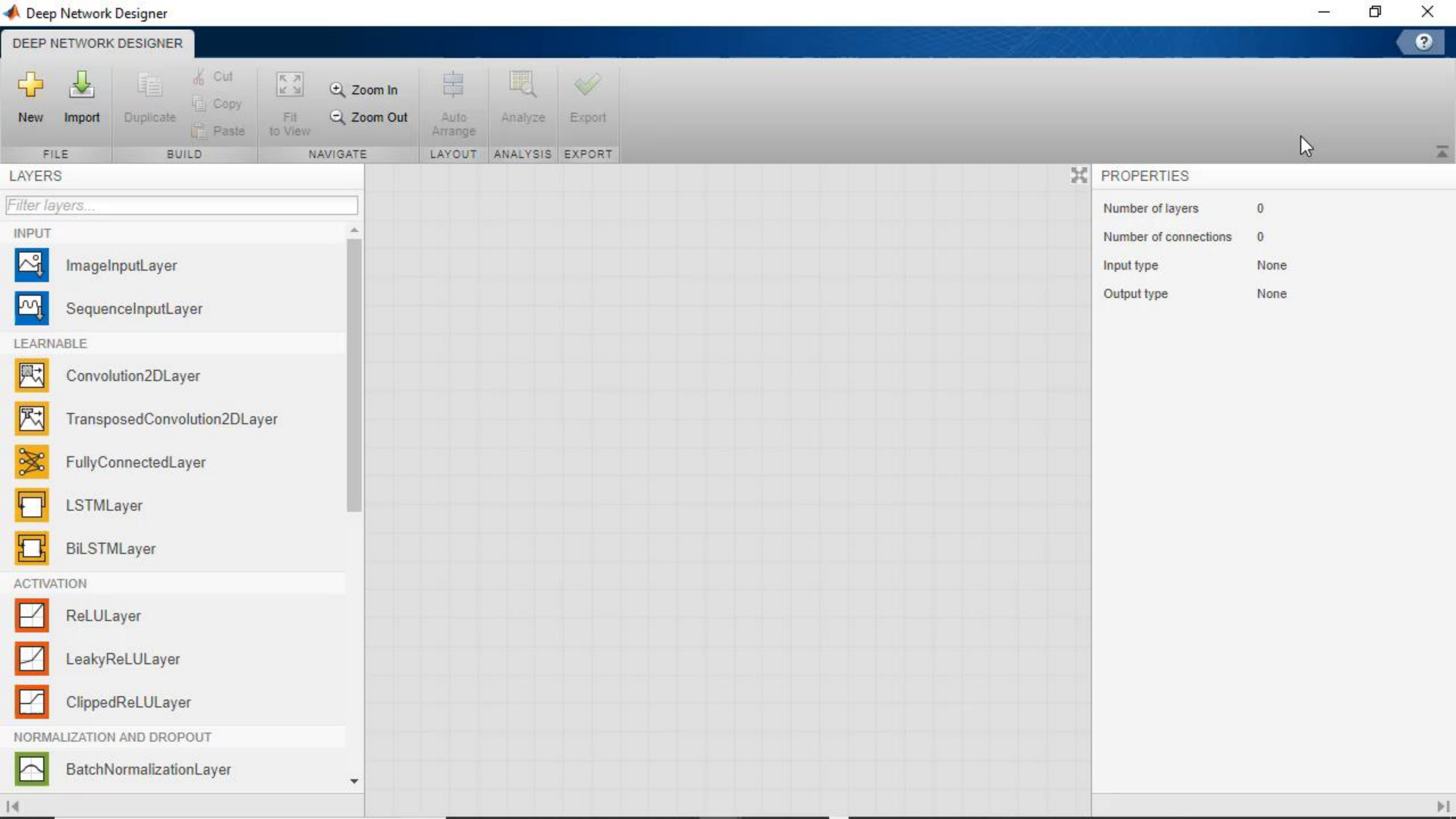




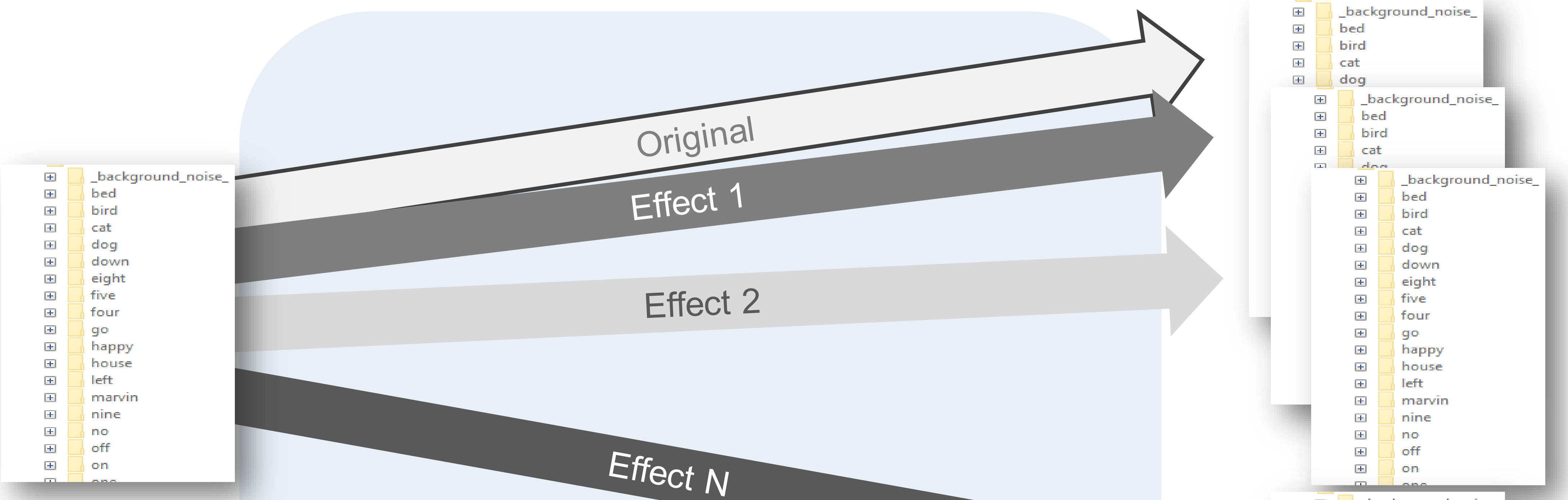

Training Progress (08-Oct-2018 12:53:57)
Training Progress (08-Oct-2018 12:53:57)
Accuracy (%)
Epoch 1
Iteration
Loss
Epoch 1
Iteration
Training iteration 1 of 5500...
Training Time
Start time: 08-Oct-2018 12:53:57
Elapsed time: 0 sec
Training Cycle
Epoch: 0 of 25
Iterations per epoch: 220
Maximum iterations: 5500
Validation
Frequency: 220 iterations
Patience: Inf
Other Information
Hardware resource: Single GPU
Learning rate schedule: Piecewise
Learning rate: 0.0003
Accuracy
Training (smoothed)
Training
Validation
Loss
Training (smoothed)
Training
Validation

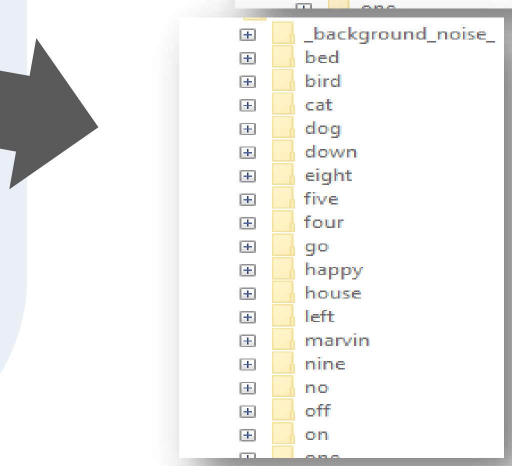

Deep Network Designer
DEEP NETWORK DESIGNER
New
Import
Duplicate
Cut
Copy
Paste
Fit to View
Zoom In
Zoom Out
Auto Arrange
Analyze
Export
FILE
BUILD
NAVIGATE
LAYOUT
ANALYSIS
EXPORT
LAYERS
Filter layers...
INPUT
ImageInputLayer
SequenceInputLayer
LEARNABLE
Convolution2DLayer
TransposedConvolution2DLayer
FullyConnectedLayer
LSTMLayer
BiLSTMLayer
ACTIVATION
ReLULayer
LeakyReLULayer
ClippedReLULayer
NORMALIZATION AND DROPOUT
BatchNormalizationLayer
PROPERTIES
Number of layers 0
Number of connections 0
Input type None
Output type None


Deep Network Designer
DEEP NETWORK DESIGNER
New
Import
Duplicate
Cut
Copy
Paste
Fit to View
Zoom In
Zoom Out
Auto Arrange
Analyze
Export
FILE
BUILD
NAVIGATE
LAYOUT
ANALYSIS
EXPORT
LAYERS
Filter layers...
INPUT
ImageInputLayer
SequenceInputLayer
LEARNABLE
Convolution2DLayer
TransposedConvolution2DLayer
FullyConnectedLayer
LSTMLayer
BiLSTMLayer
ACTIVATION
ReLULayer
LeakyReLULayer
ClippedReLULayer
NORMALIZATION AND DROPOUT
BatchNormalizationLayer
PROPERTIES
Number of layers
0
Number of connections
0
Input type
None
Output type
None


Deep Network Designer
DEEP NETWORK DESIGNER
New
Import
Duplicate
Cut
Copy
Paste
Fit to View
Zoom In
Zoom Out
Auto Arrange
Analyze
Export
FILE
BUILD
NAVIGATE
LAYOUT
ANALYSIS
EXPORT
LAYERS
Filter layers...
INPUT
ImageInputLayer
SequenceInputLayer
LEARNABLE
Convolution2DLayer
TransposedConvolution2DLayer
FullyConnectedLayer
LSTMLayer
BiLSTMLayer
ACTIVATION
ReLULayer
LeakyReLULayer
ClippedReLULayer
NORMALIZATION AND DROPOUT
BatchNormalizationLayer
PROPERTIES
Number of layers 0
Number of connections 0
Input type None
Output type None

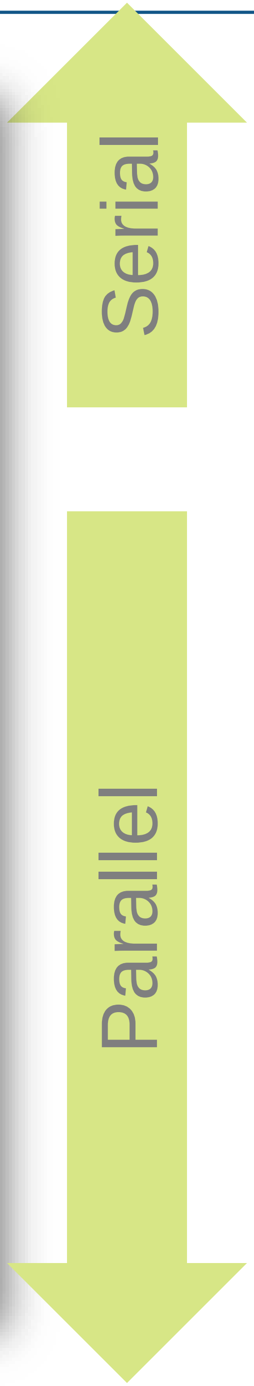
Data augmentation allows training more advanced networks and generating more robust models





Original Dataset

Simulate:
- lower voice
- varying level of environmental interference
- recording hardware effects
- distortions due to use of different microphones
- different speaker vs microphone positions or room sizes in enclosed environments
- ...

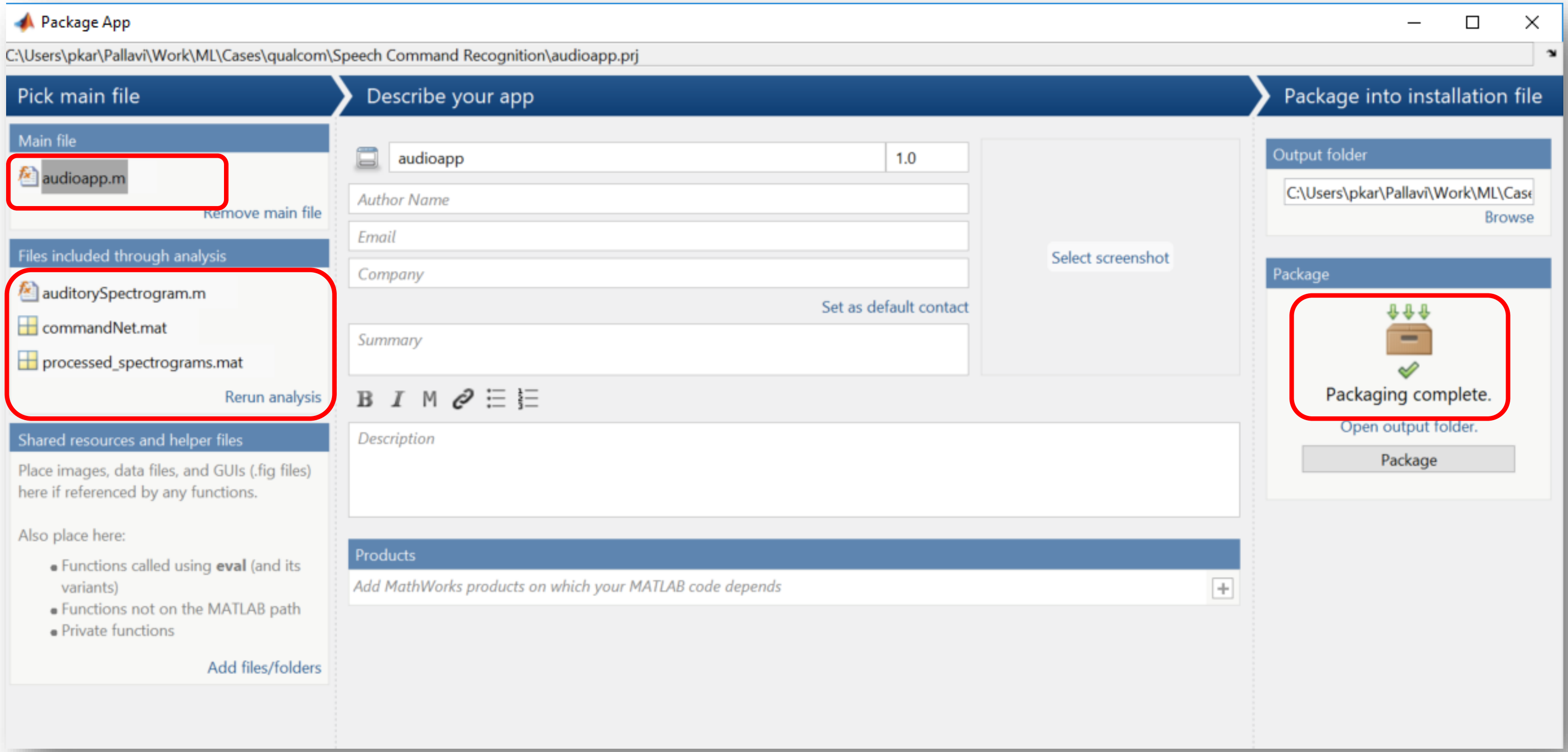


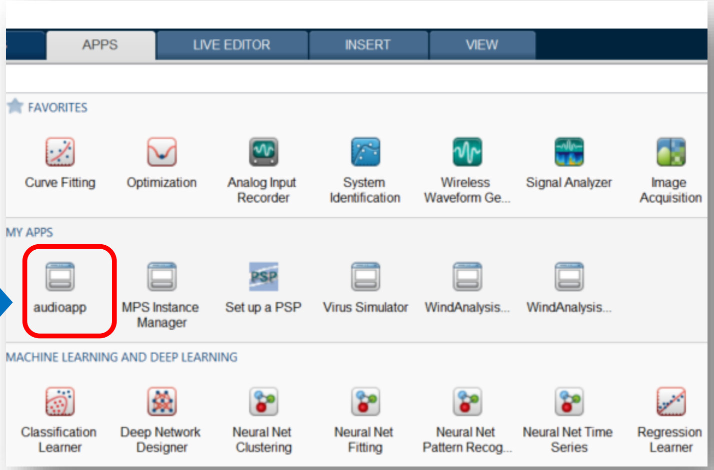

```
% Cycle continuously and automatically through files in datastore
mfccFile = zeros(numel(ads.Files),numMfcc)
while hasdata(ads)
         [data,info] = read(ads);
          % do something with data
end
```

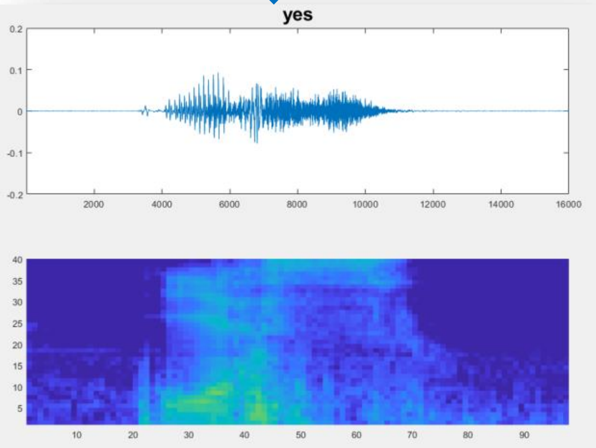
```
% Partition and explicit parfor parallel processing pattern
N = numpartitions(ads);

parfor index=1:N
    subads = partition(ads,N,index);
    while hasdata(subads)
        data = read(subads);
                  % do something with data
    end
end
```

```
% Tall array and "implicit" parallel processing pattern
T = tall(ads);

dcOffsets = cellfun(@(x) mean(x), T,'UniformOutput',false);
gather(dcOffsets);
```

Package Speech Recognition App





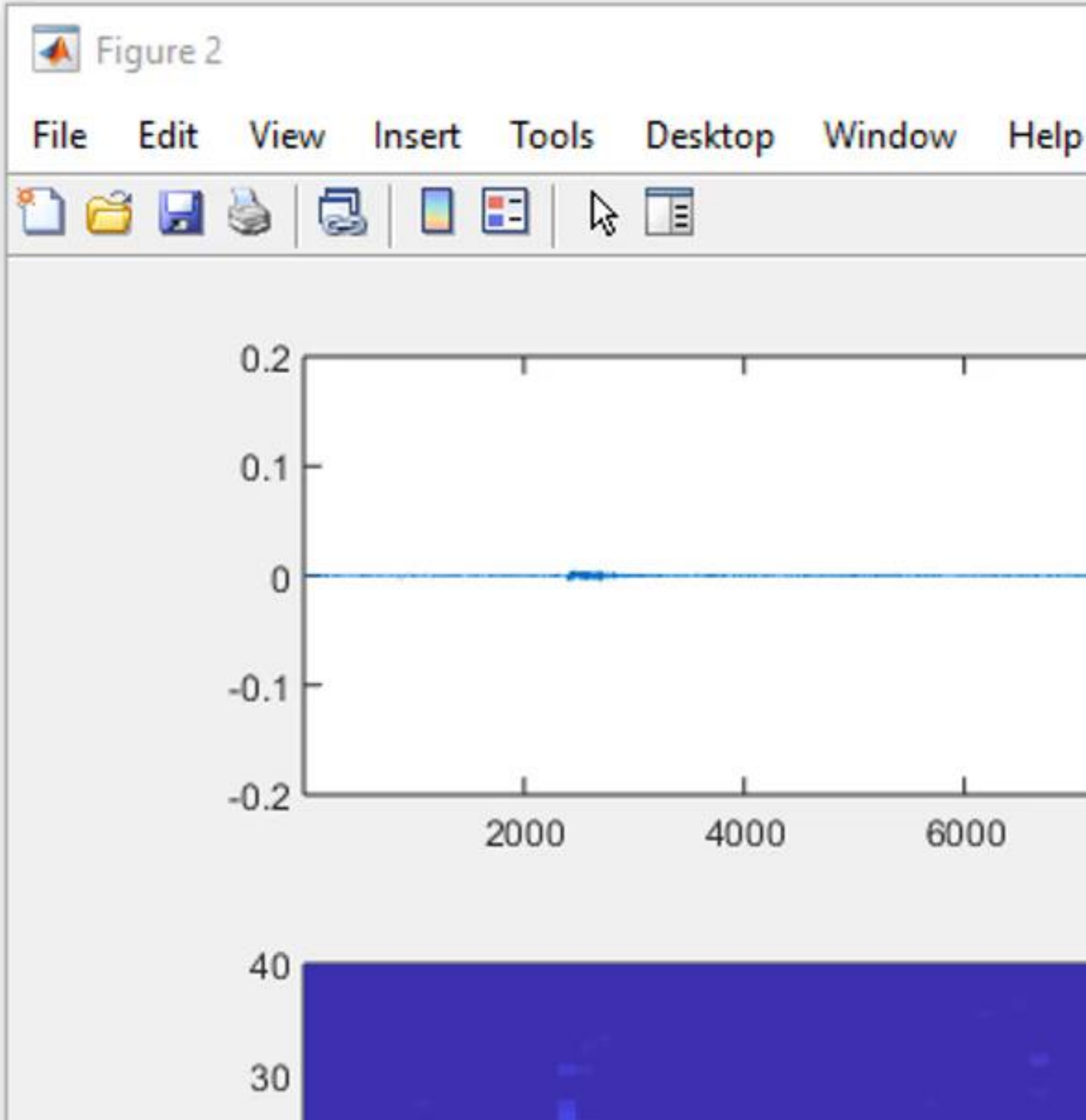


Commands

- Yes
- No
- Up
- Down
- Left
- Right
- On
- Off
- Stop
- Go

Non-Commands
(= Unknown)

- Bed
- Bird
- Cat
- Dog
- Happy
- House
- Marvin
- Sheila
- Tree
- Wow
- Zero
- One
- Two
- Three
- Four



BREAK

Deep Learning Challenges

Not a deep learning expert

Data

- ✓ Handling large amounts of data
- Labeling thousands of signals, images & videos
- Transforming, generating, and augmenting data (for different domains)

Training and Testing Deep Neural Networks

- ✓ Understanding network behavior
- Accessing reference models from research
- Optimizing hyperparameters
- Training takes hours-days

Rapid and Optimized Deployment

- Desktop, web, cloud, and embedded hardware

BREAK

Segment and label audio signals automatically

Read speech recording

Load speech recording from (.wav) file

```
fileName = 'Counting-16-44p1-mono-15secs.wav';
pathName = fullfile(matlabroot,'toolbox','audio','samples', fileName);
[x,fs] = audioread(pathName);
```

Plot samples over time

```
t = (1/fs)*(0:length(x)-1);
hpl = plot(t, x);
xlabel('time (s)')
```

Playback content

```
soundsc(x,fs)
```

Segment automatically

Use a custom function based on combined thresholding of signal energy and spectral centroid

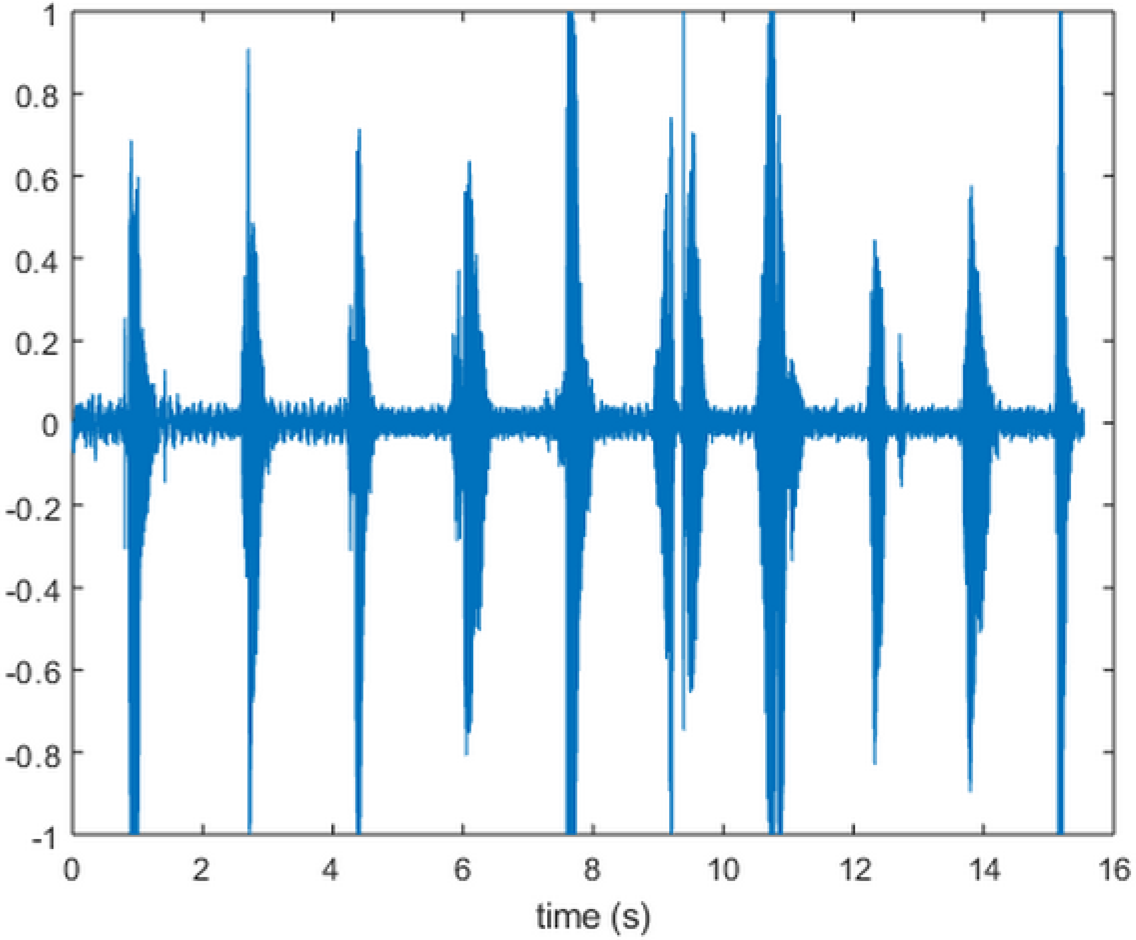
[redacted]

```
[segm, ~] = findSpeechSegments(x,fs);
```

Plot segmented time intervals

```
hold on
hax = hpl.Parent;
xlr = segm(:,:);
```

```
hpl = plot(t, x);
xlabel('time (s)')
```



Playback content

```
soundsc(x,fs)
```

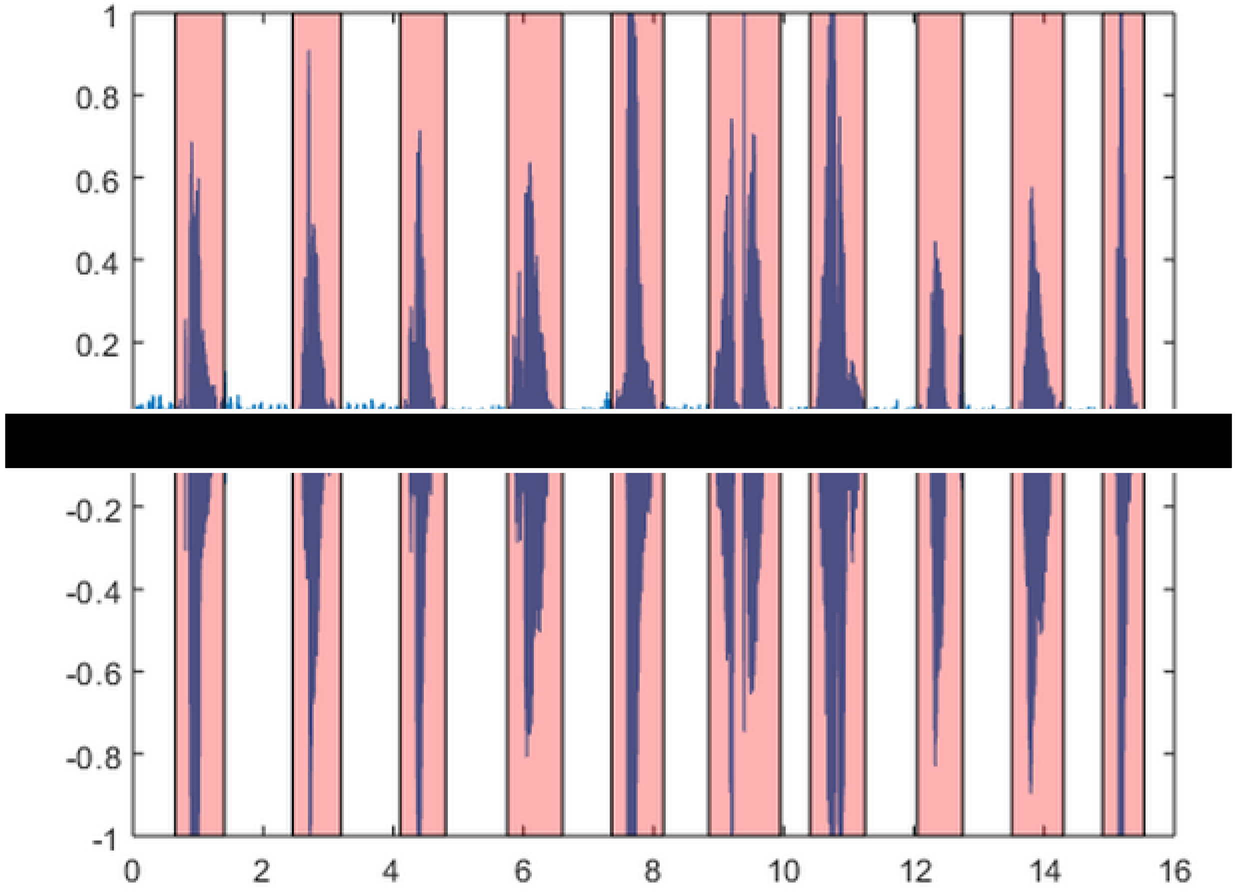
Segment automatically

Use a custom function based on combined thresholding of signal energy and spectral centroid

```
[segm, ~] = findSpeechSegments(x,fs);
```

Plot segmented time intervals

```
plotSegments(hpl, segm/fs)
```



Segment automatically

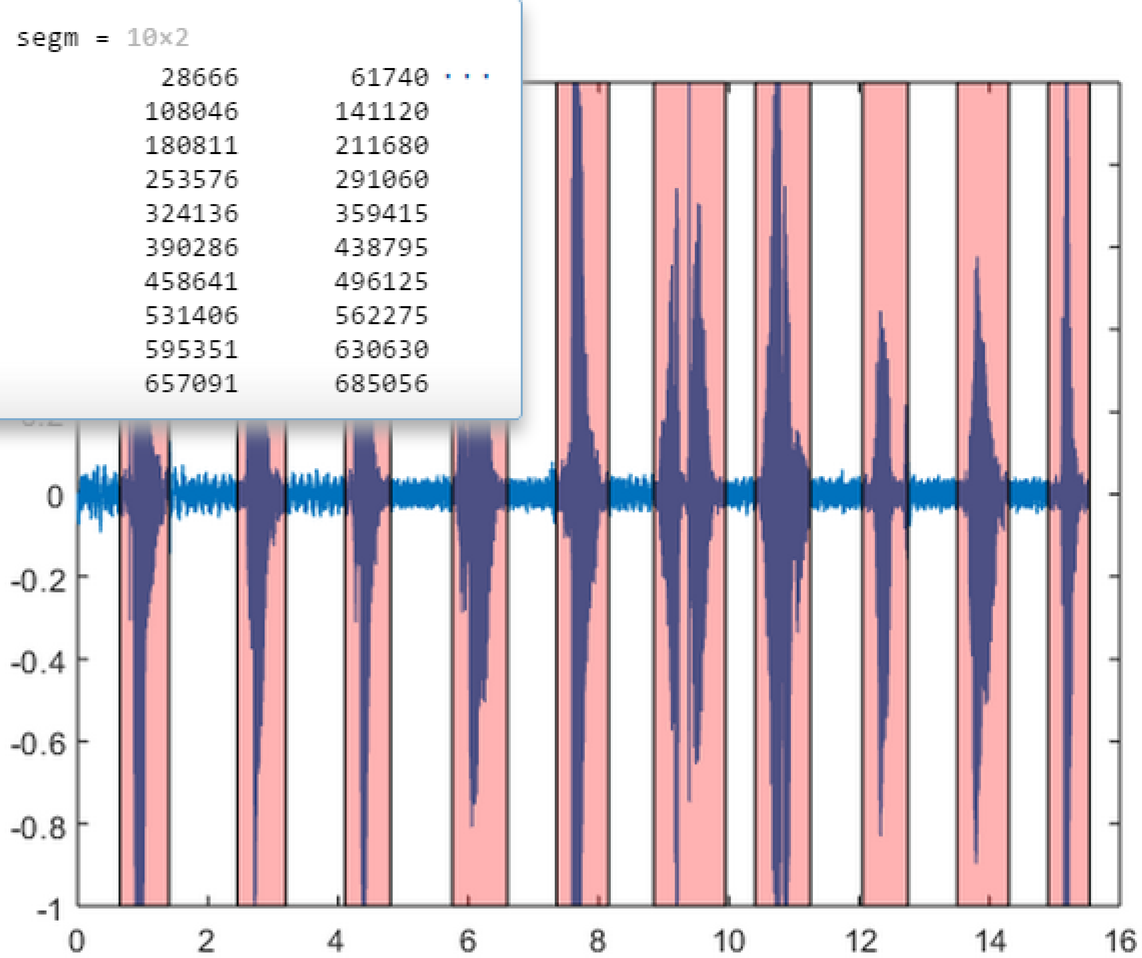
Use a custom function based on combined thresholding of signal energy and spectral centroid

```
[segm, ~] = findSpeechSegments(x,fs);
```

Plot segmented time intervals

```
plotSegments(hpl, segm/fs)
```

```
segm = 10×2
       28666       61740 ...
      108046      141120
      180811      211680
      253576      291060
      324136      359415
      390286      438795
      458641      496125
      531406      562275
      595351      630630
      657091      685056
```



Automate labeling with speech content using speech-to-text services

Initialize speech transcription wrapper

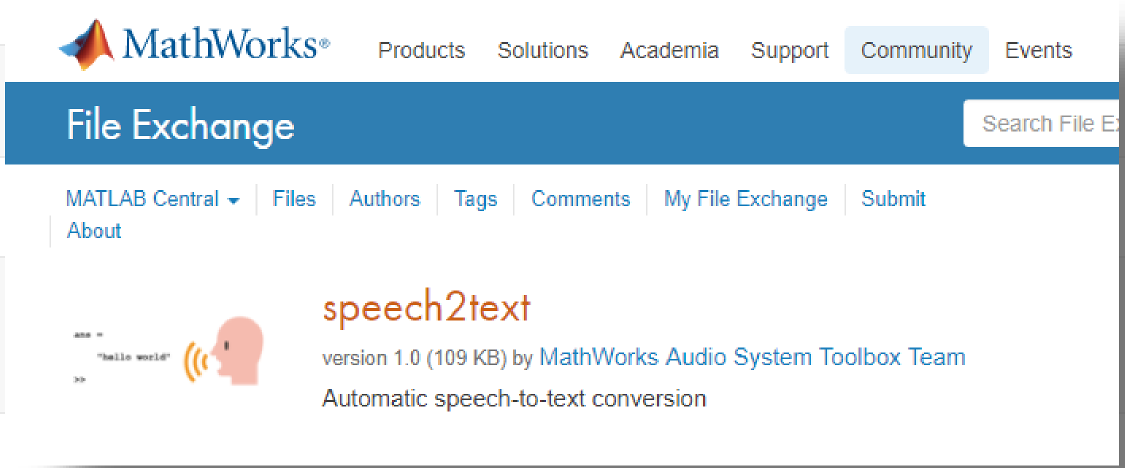
```
speechObject = speechClient('Google','languageCode','en-GB');
```

Loop over segments

```
autoLabels = strings(numSegments,1);
for idx = 1:numSegments
```

Get segment boundary

```
    start = segm(idx,1);
    stop = segm(idx,2);
    fprintf('Querying transcript for segment %02d [%5.02f,
```



Get speech transcription using the Google Speech API

```
    tableOut = speech2text(speechObject, x(start:stop,1), fs);
```

Store output

```
    if(ismember('TRANSCRIPT',tableOut.Properties.VariableNames))
        autoLabels(idx) = tableOut.TRANSCRIPT(1);
    end

end
```

Visualize results in Audio Labeler

Create label definition

Automate labeling with speech content using speech-to-text services

Initialize speech transcription wrapper

```
speechObject = speechClient('Google','languageCode','en-GB');
```

Loop over segments

```
autoLabels = strings(numSegments,1);
for idx = 1:numSegments
```

Get segment boundary

```
    start = segm(idx,1);
    stop = segm(idx,2);
    fprintf('Querying transcript for segment %02d [%5.02f, %5.02f]s of file "%s"\n', idx, start/fs, stop/fs,fileName)
```

Get speech transcription using the Google Speech API

```
    tableOut = speech2text(speechObject, x(start:stop,1), fs);
```

Store output

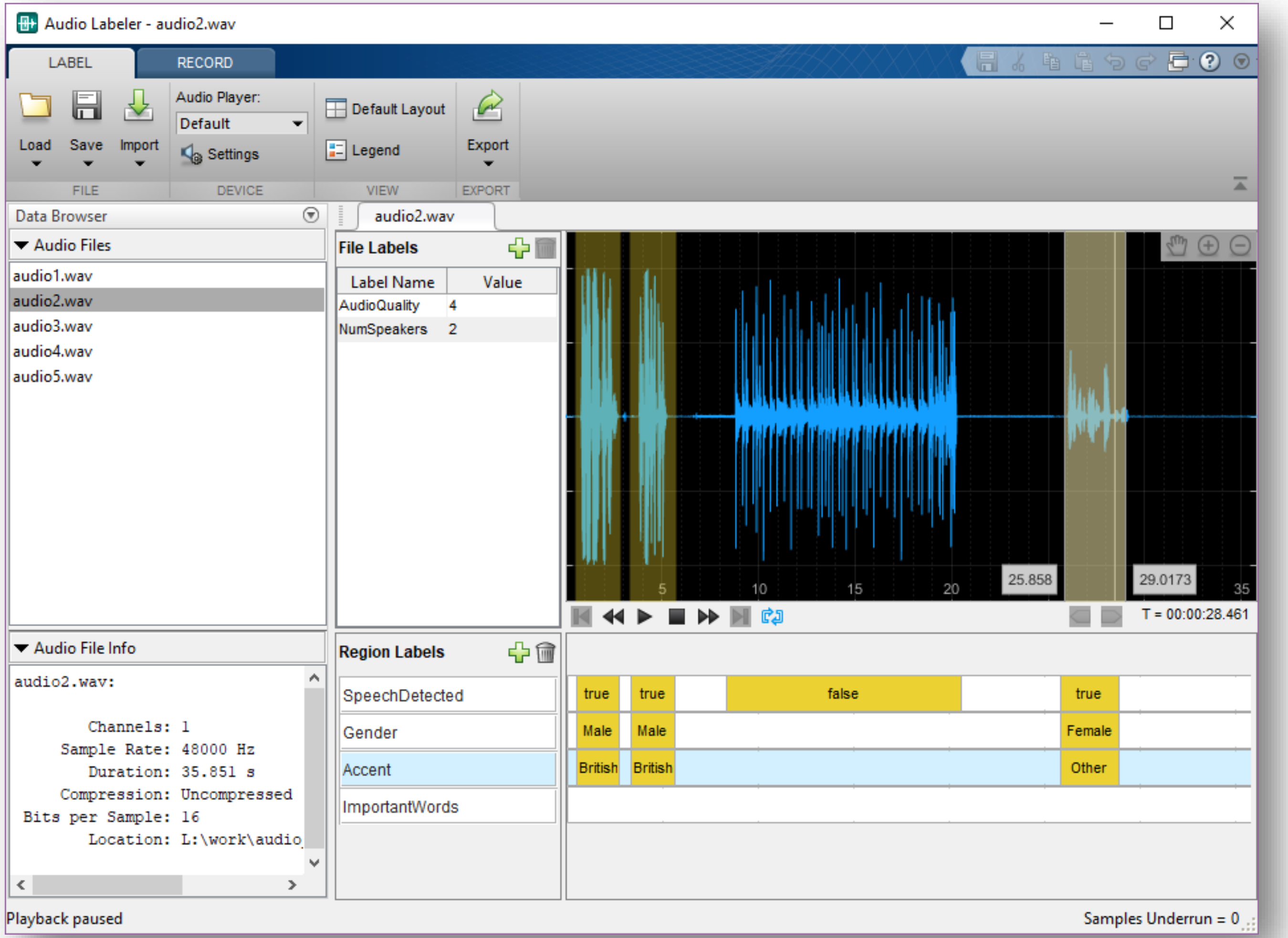
```
    if(ismember('TRANSCRIPT',tableOut.Properties.VariableNames))
        autoLabels(idx) = tableOut.TRANSCRIPT(1);
    end

end
```

Visualize results in Audio Labeler

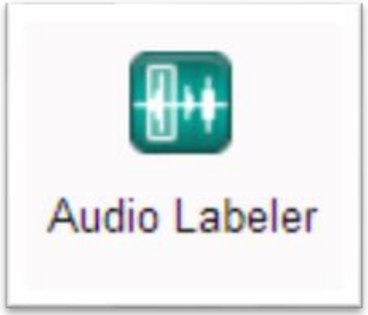
Create label definition

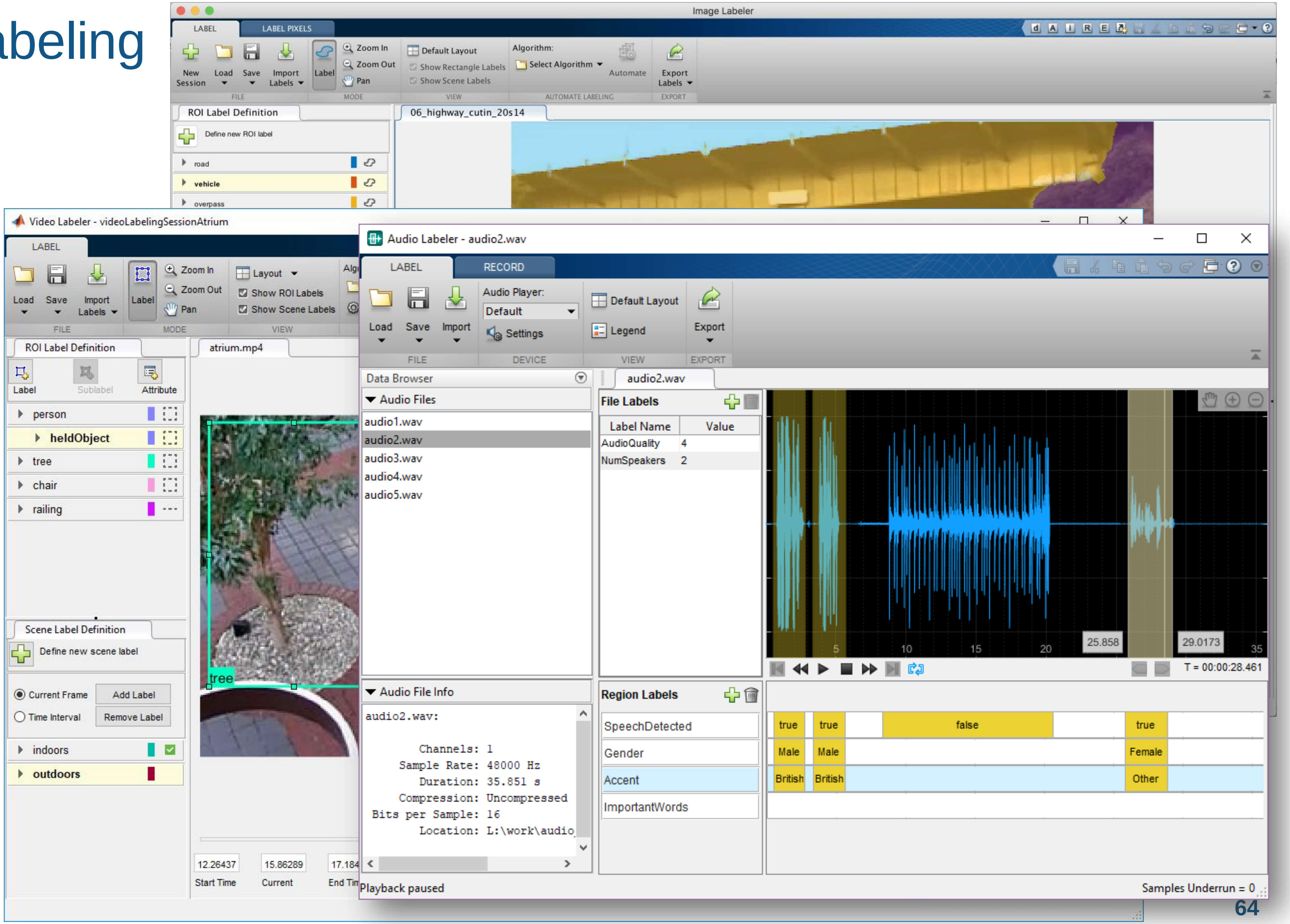
Audio Labeler

- Work on collections of recordings or record new audio directly within the app
- Navigate dataset and playback interactively
- Define and apply labels to
  - Entire files
  - Regions within files
- Import and export audio folders, label definitions and datastores

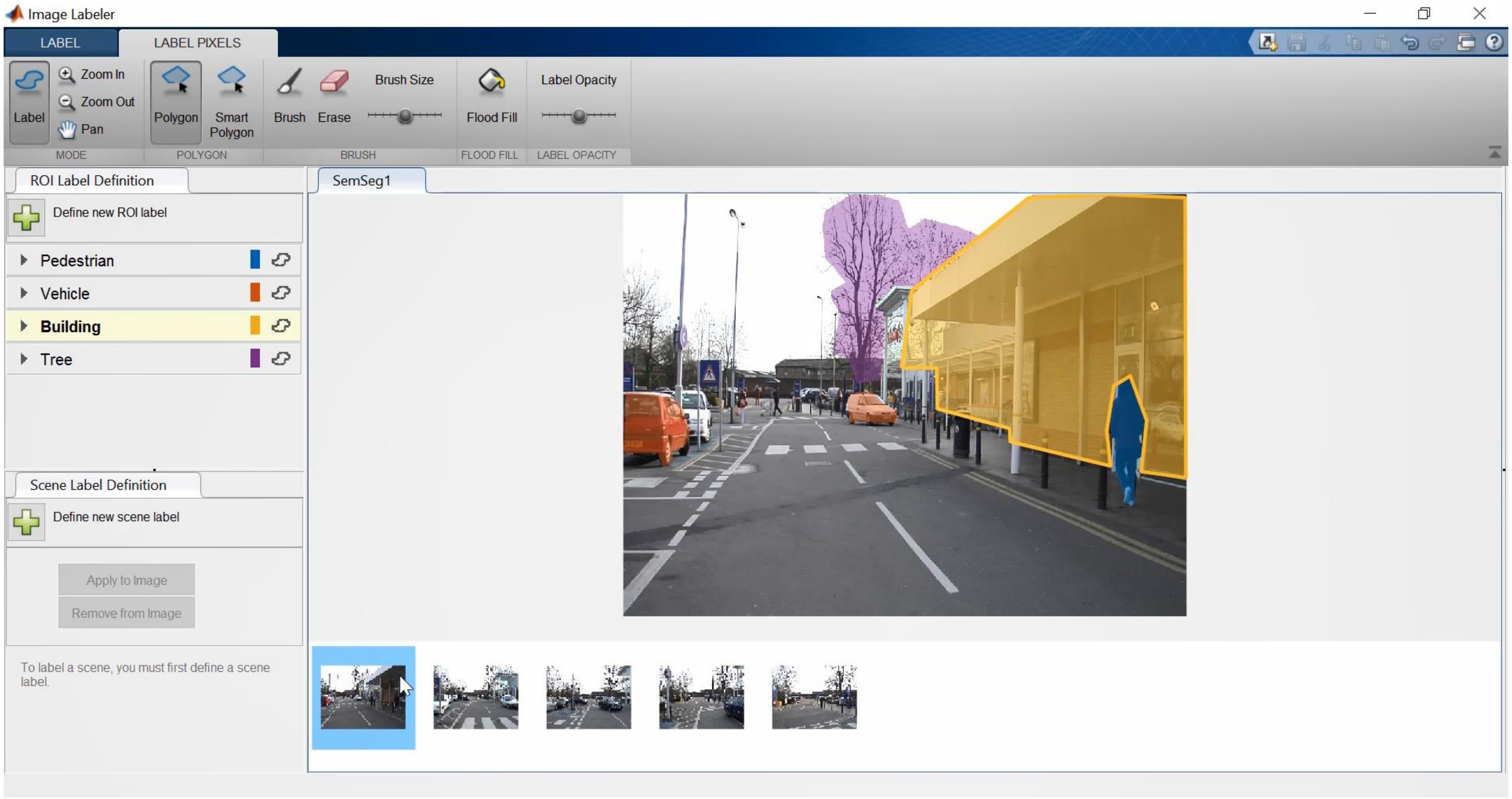


Apps for Data Labeling

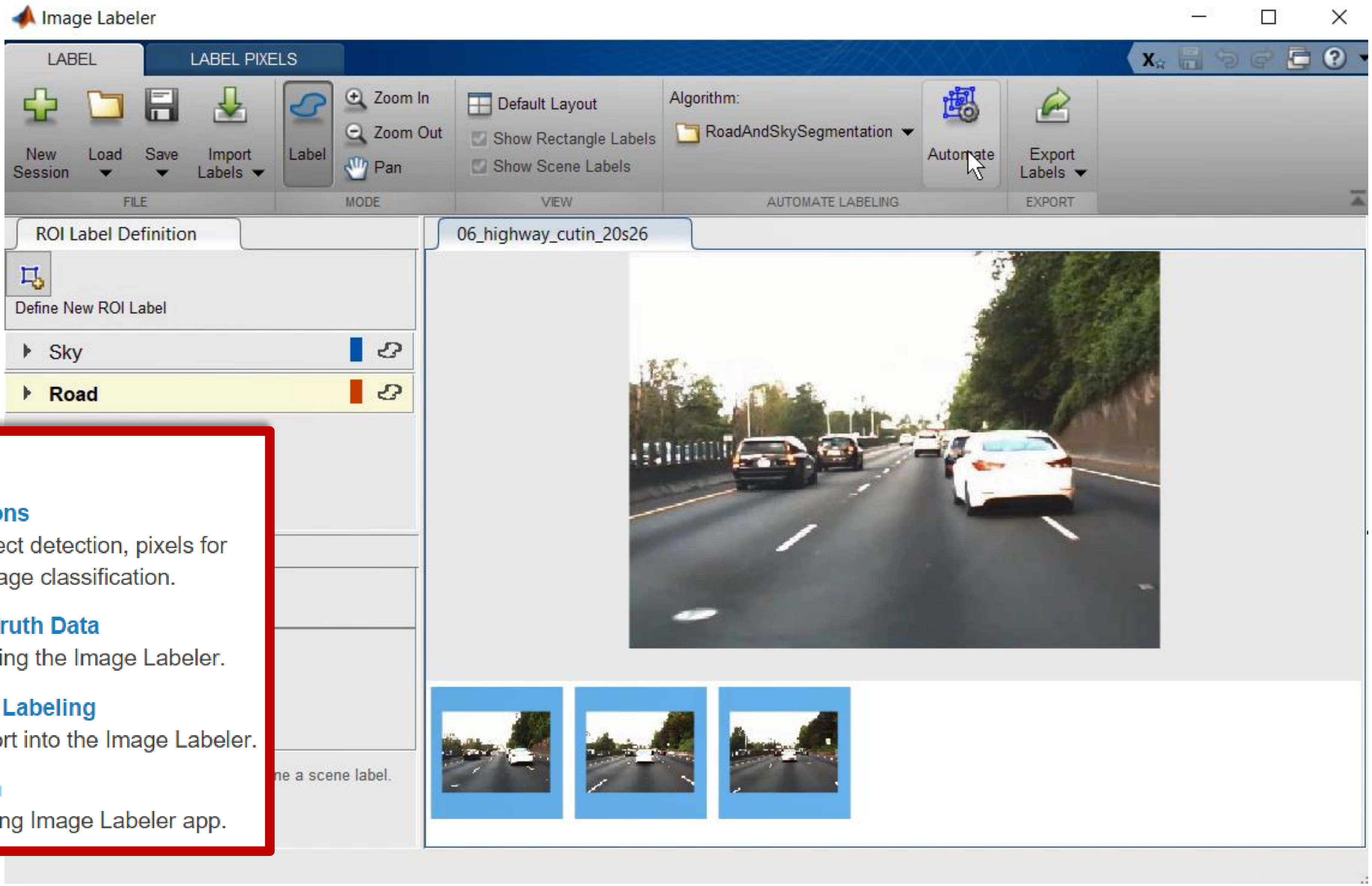




Label Images Using Image Labeler App



Accelerate Labeling With Automation Algorithms



[Learn More]

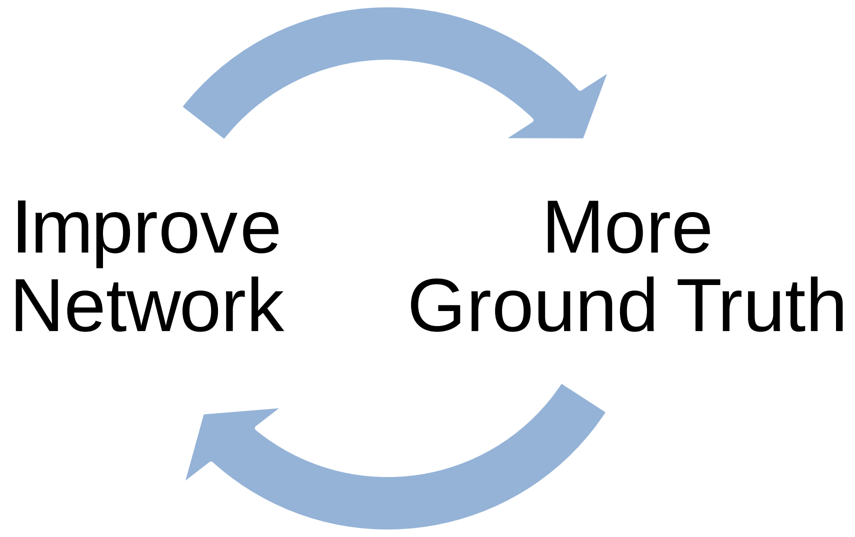
Define Ground Truth for Image Collections
Interactively label rectangular ROIs for object detection, pixels for semantic segmentation, and scenes for image classification.

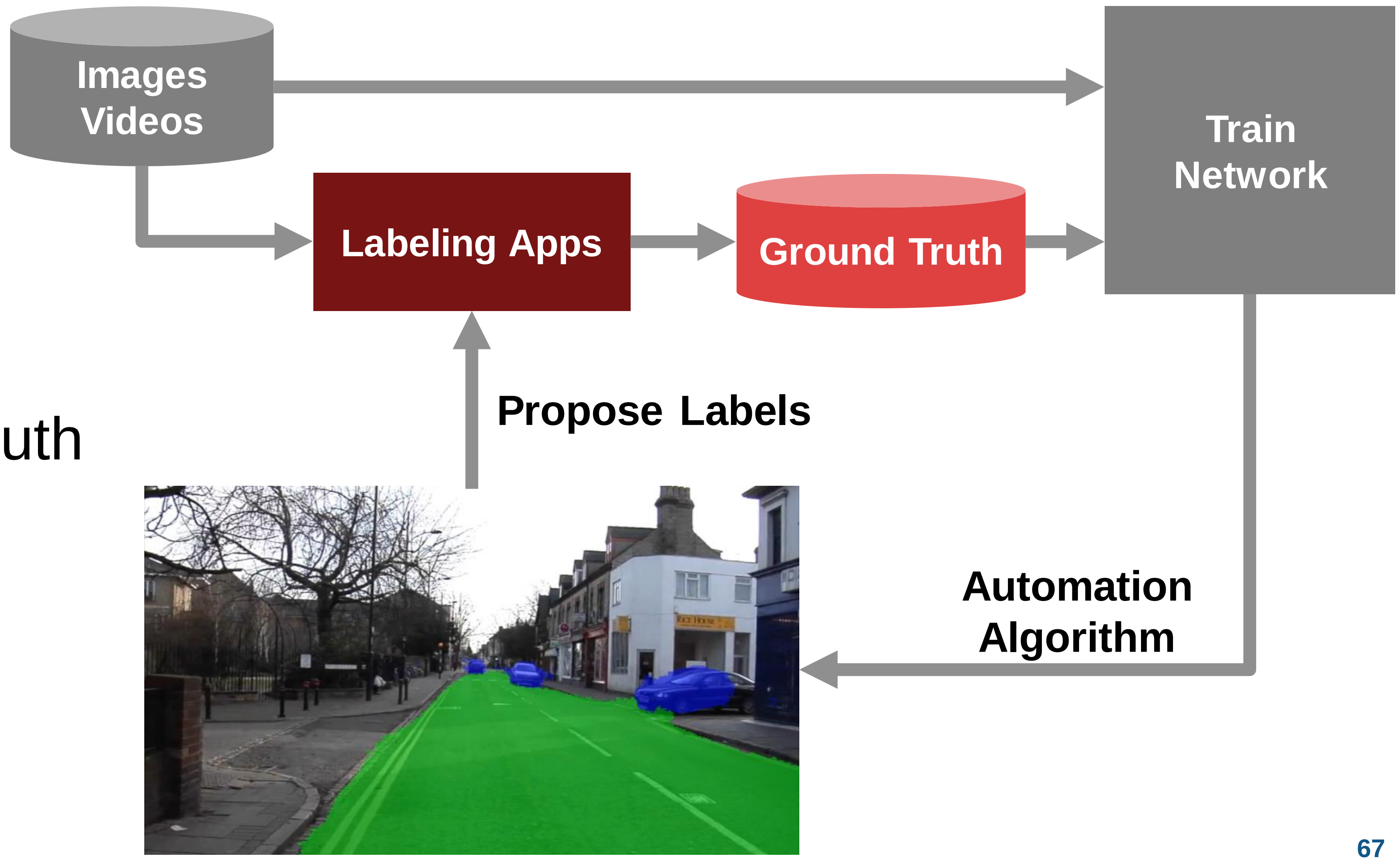
Train an Object Detector from Ground Truth Data
Create training data for object detection using the Image Labeler.

Create Automation Algorithm for Image Labeling
Create a custom tracking algorithm to import into the Image Labeler.

Label Pixels for Semantic Segmentation
Label pixels for semantic segmentation using Image Labeler app.

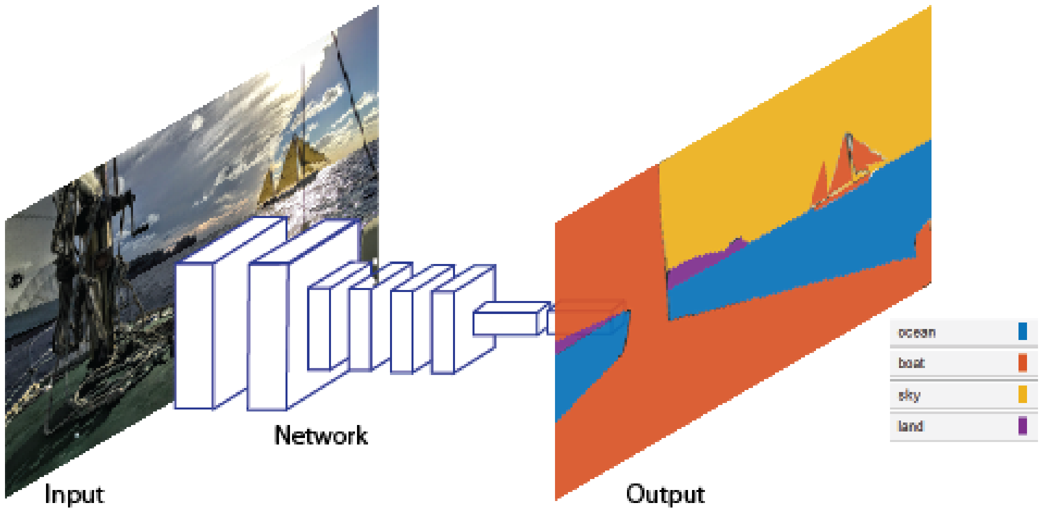
Perform Bootstrapping to Label Large Datasets





Example – Semantic Segmentation

Available Here



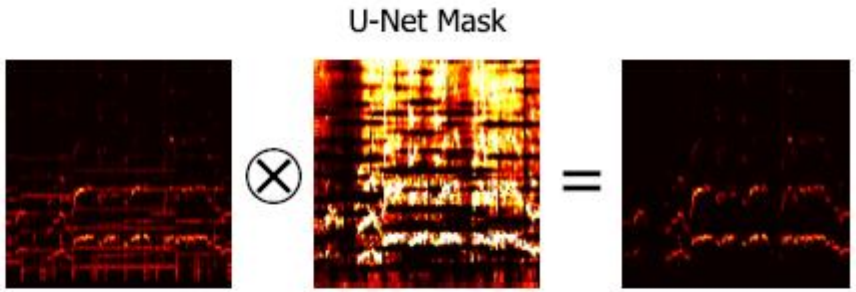
- Classify pixels into 11 classes
  – Sky, Building, Pole, Road, Pavement, Tree, SignSymbol, Fence, Car, Pedestrian, Bicyclist

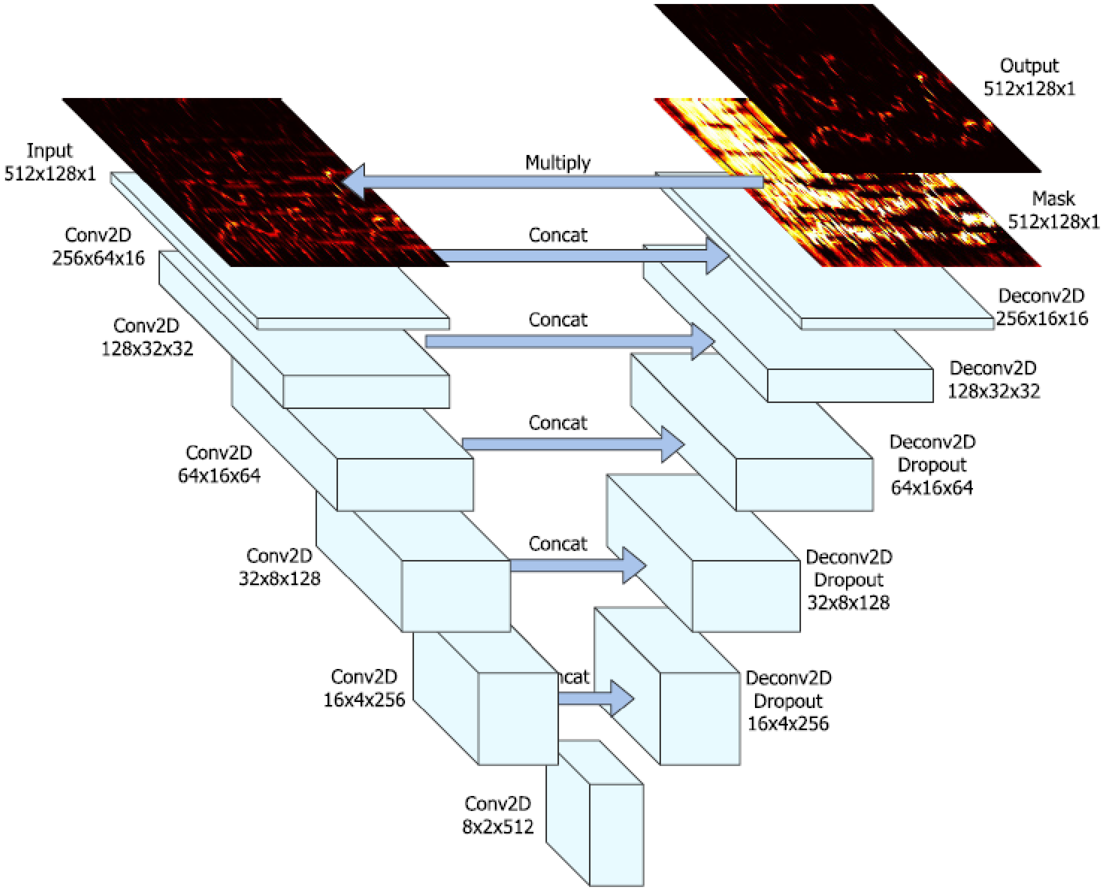
- CamVid dataset

Brostow, Gabriel J., Julien Fauqueur, and Roberto Cipolla. "Semantic object classes in video: A high-definition ground truth database." Pattern Recognition Letters Vol 30, Issue 2, 2009, pp 88-97.

Example: Singing Voice Separation

- Source separation
- Based on U-Net architecture

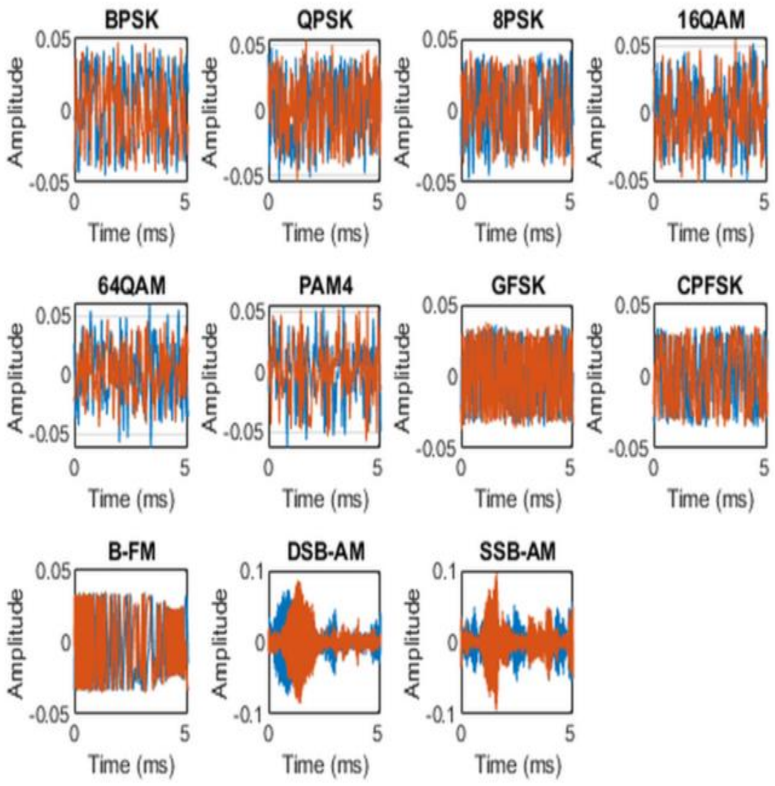


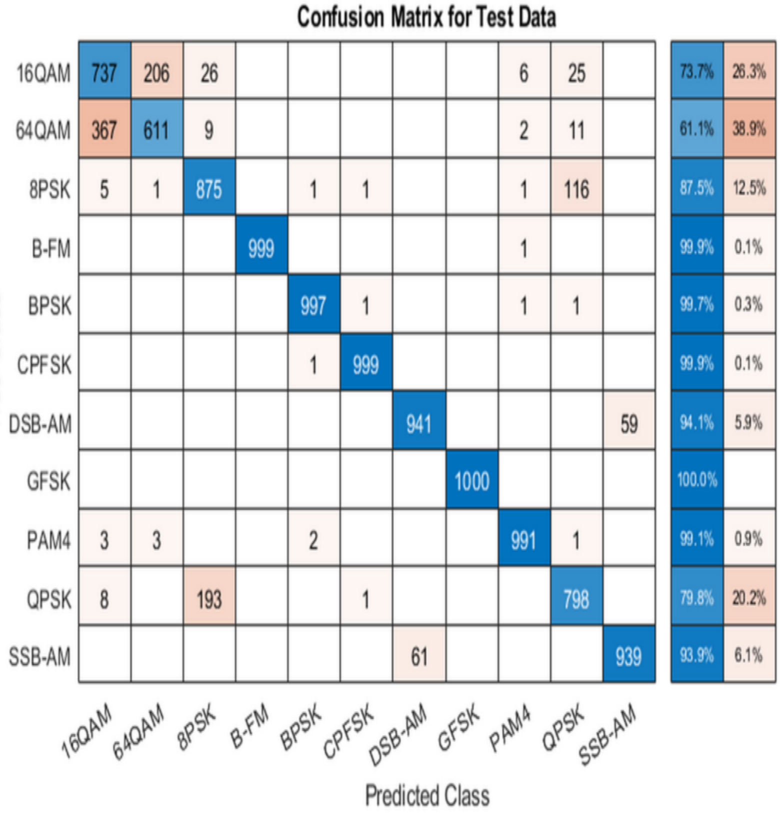


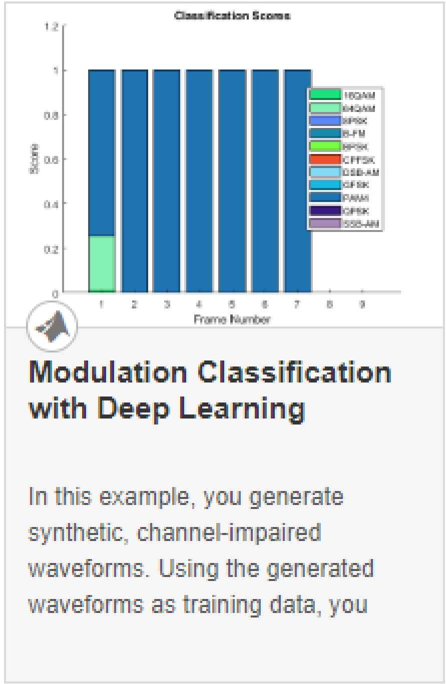
Synthetically generating labeled data

Modulation Classification with Deep Learning

- Generate synthetic modulated signals
- Apply channel impairments
- Train a CNN to classify modulation types







R2019a

Deep Learning Challenges

Data

- ✓ Handling large amounts of data
- ✓ Labeling thousands of signals, images & videos
- ✓ Transforming, generating, and augmenting data (for different domains)

Training and Testing Deep Neural Networks

- ✓ Understanding network behavior
- Accessing reference models from research
- Optimizing hyperparameters
- Training takes hours-days

Rapid and Optimized Deployment

- Desktop, web, cloud, and embedded hardware

Transfer Learning

8 lines of MATLAB Code

Load Training Data

Load Pre-Trained Network

Replace Final Layers

Train Network on New Data

```
%% Create a datastore
1 imds = imageDatastore('Data',...
   'IncludeSubfolders',true,'LabelSource','foldernames');
2 num_classes = numel(unique(imds.Labels));
%% Load Referece Network
3 net = alexnet;
4 layers = net.Layers
%% Replace Final Layers
5 layers(end-2) =
  fullyConnectedLayer(num_classes,'Name',['fc5']);
6 layers(end) = classificationLayer('Name','classOut');
%% Set Training Options & Train Network
7 trainOpts = trainingOptions('sgdm',...
                    'MiniBatchSize', 64);
8 net = trainNetwork(imds, layers, trainOpts);
```

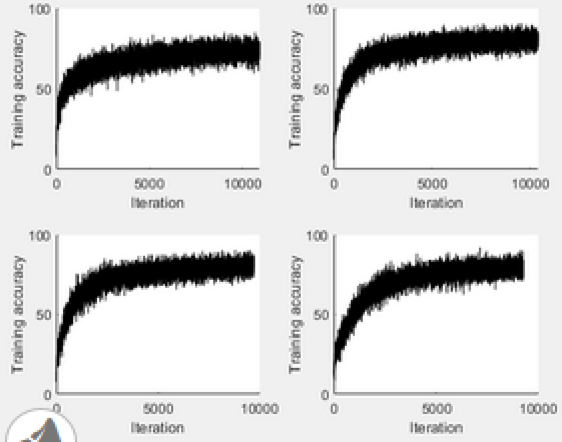
Tune Hyperparameters to Improve Training

Many hyperparameters

- depth, layers, solver options, learning rates, regularization, …

Techniques

- Parameter sweep
- Bayesian optimization



Use parfeval to Train Multiple Deep Learning Networks

Use parfeval for a parameter sweep on the depth of the network architecture. Deep Learning training often takes hours or days, and

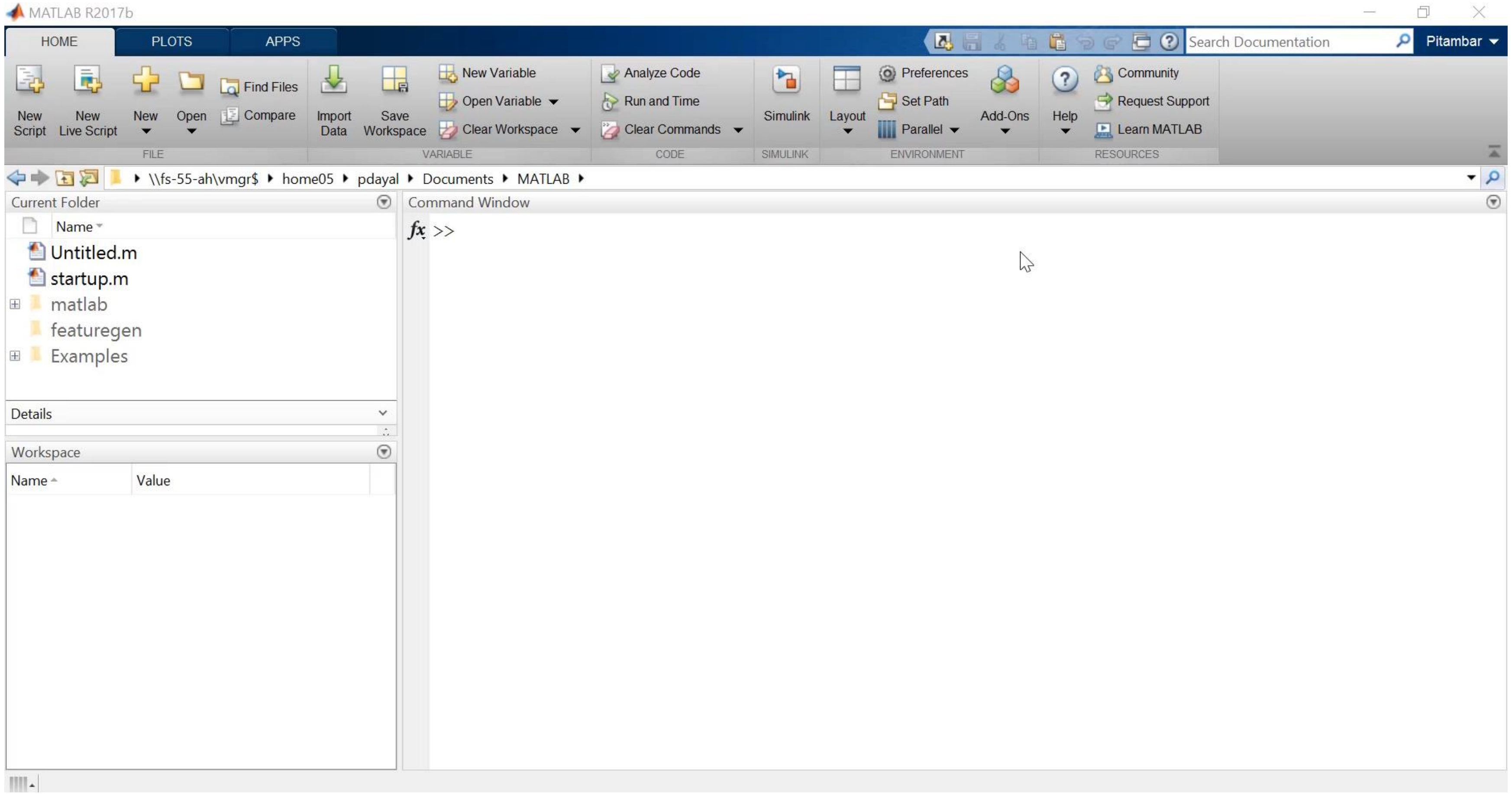
Open Script

Deep Learning Using Bayesian Optimization

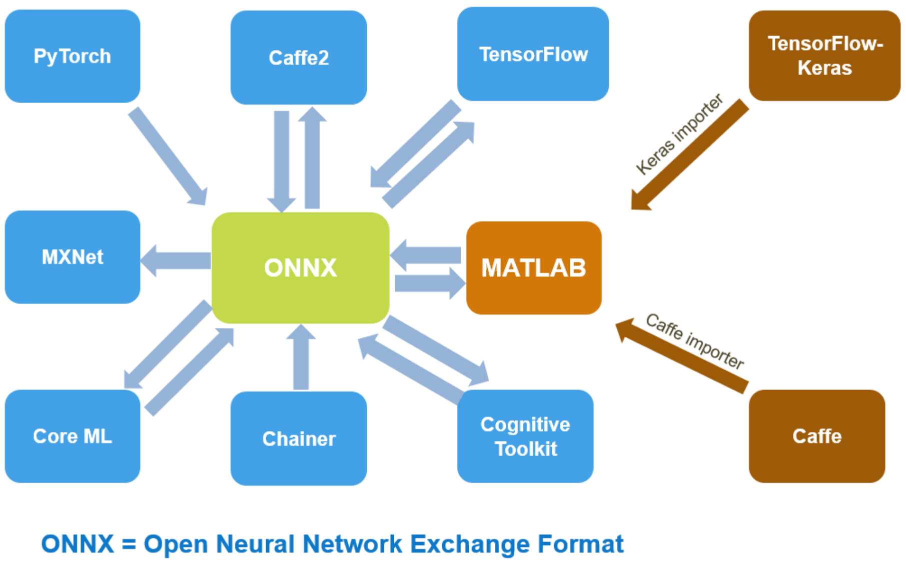
Apply Bayesian optimization to deep learning and find optimal network parameters and training options for convolutional neural networks.

Open Live Script

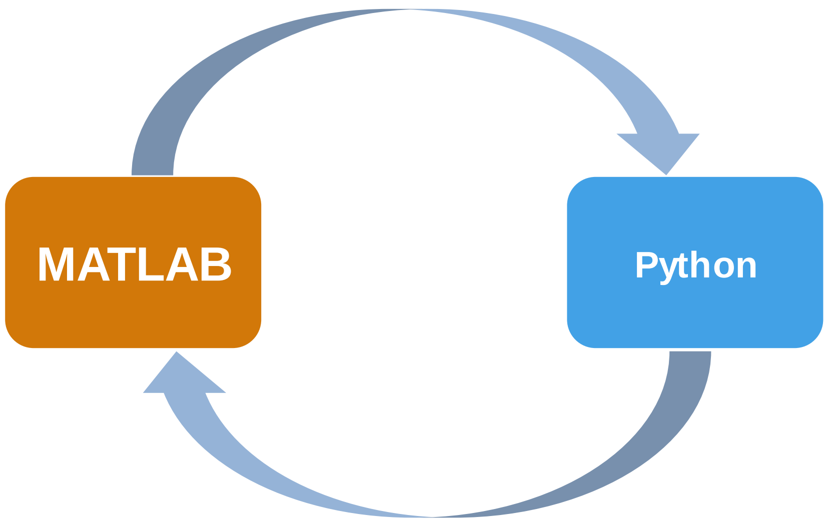
Keras-Tensorflow Importer



Model Exchange and Co-execution

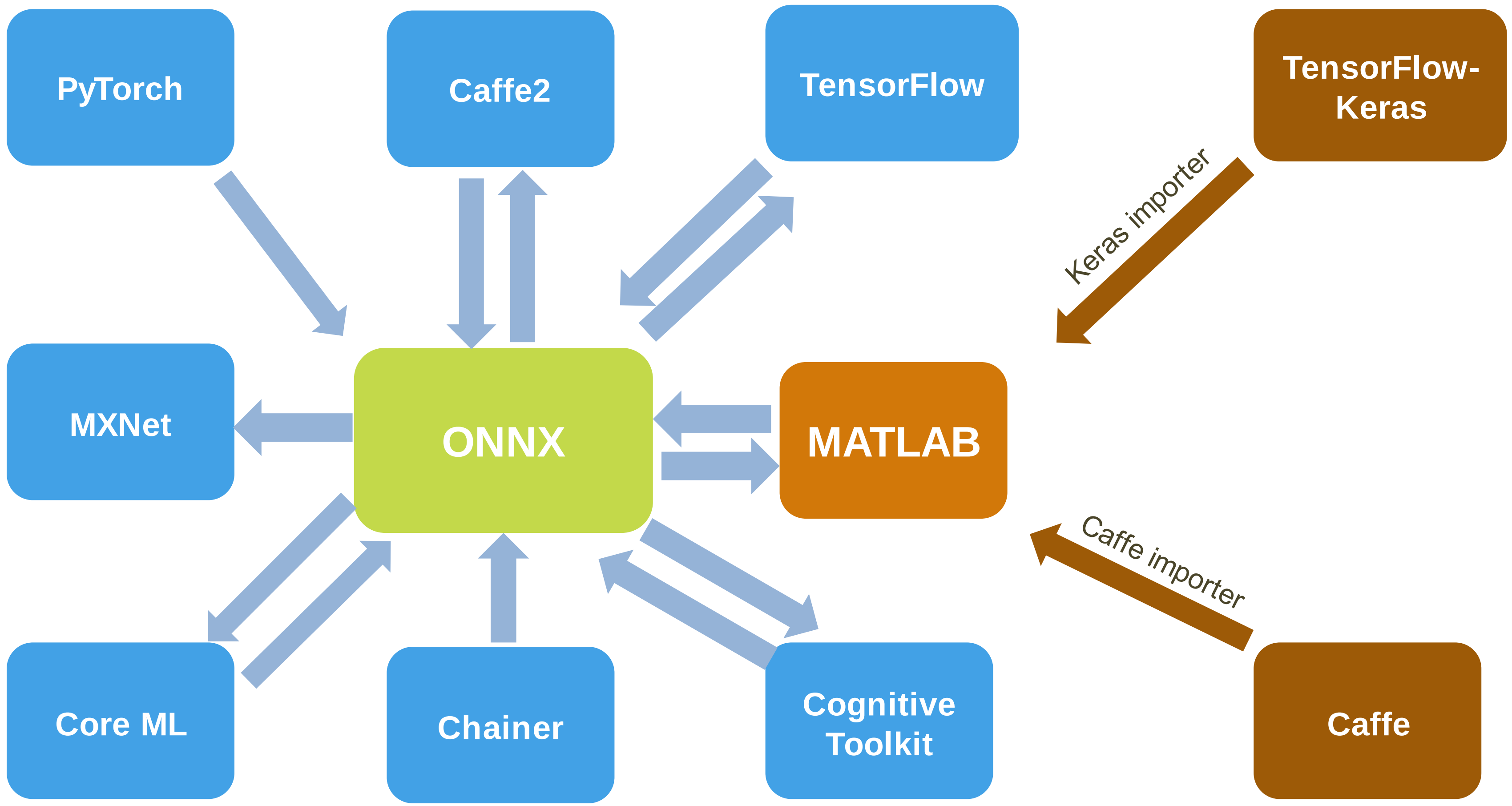


Model Exchange



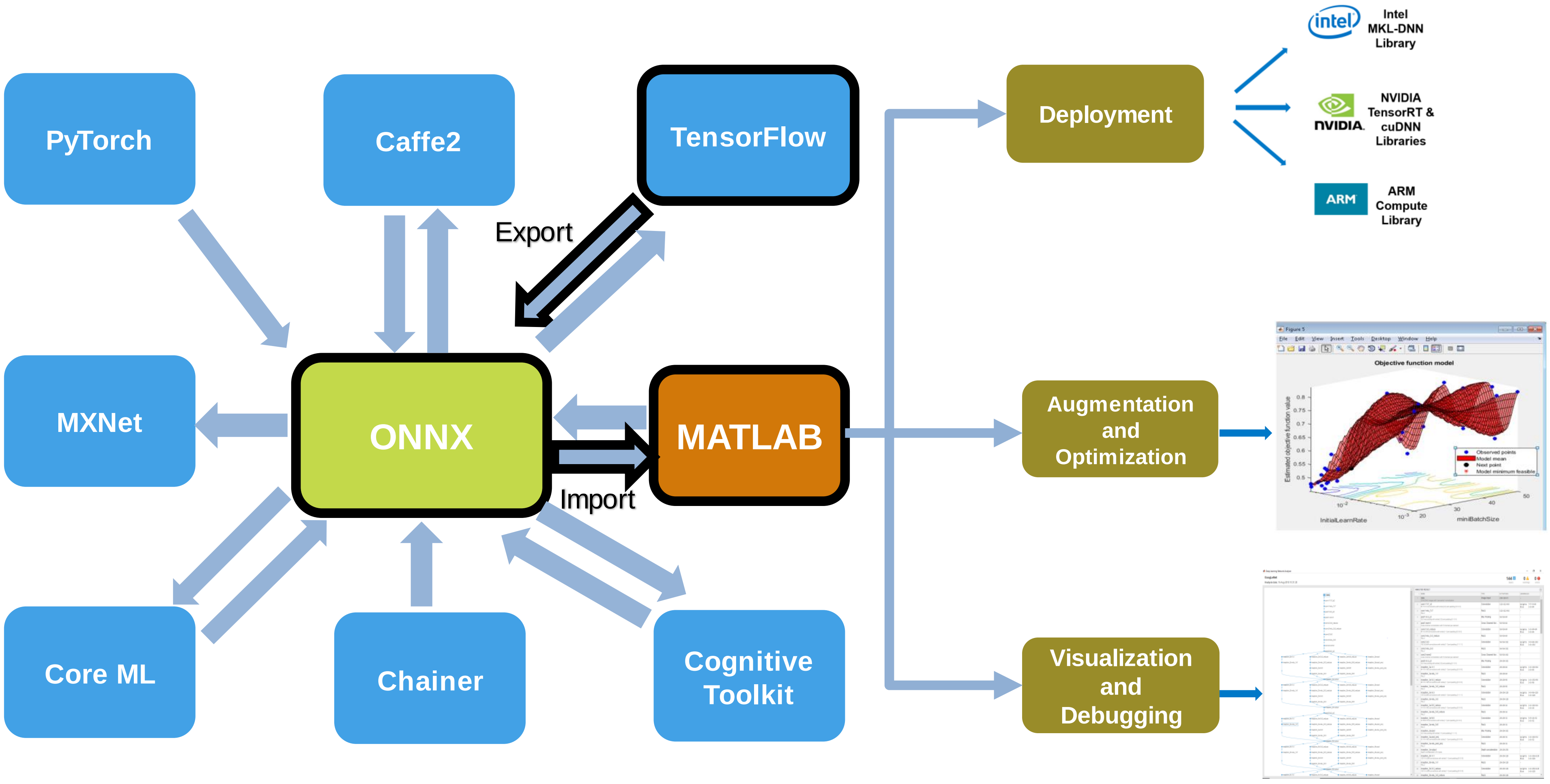
Co-execution

Model Exchange With Deep Learning Frameworks



ONNX = Open Neural Network Exchange Format

Interoperate With Deep Learning Frameworks – Use Cases



ONNX = Open Neural Network Exchange Format

Model Exchange With Deep Learning Frameworks

Caffe Model Importer

- `importCaffeLayers`
- `importCaffeNetwork`

TensorFlow-Keras Model Importer

- `importKerasLayers`
- `importKerasNetwork`

ONNX Converter

- `importONNXNetwork`
- `exportONNXNetwork`

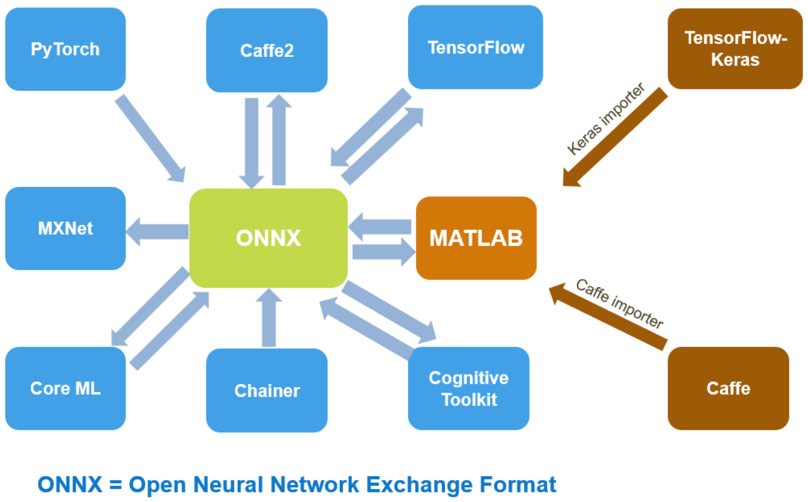
Caffe
MODELS

KERAS IMPORTER

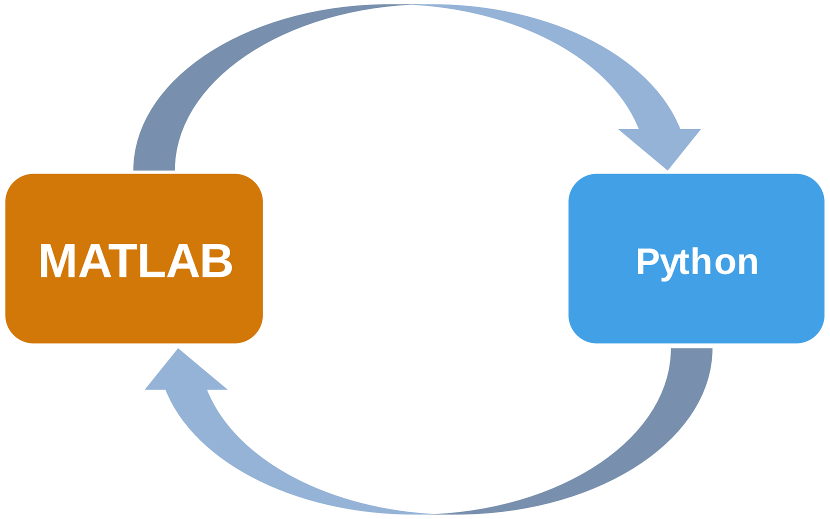
Importer for TensorFlow-Keras Models

ONNX Converter

Model Exchange and **Co-execution**



Model Exchange



Co-execution

MATLAB-Python Co-Execution – The ‘How’

Call Python file from MATLAB

Live Editor - ML_code.mlx *

ML_code.mlx *

Integrating TensorFlow and MATLAB

```
py.TF_code();
```

TF_code.py

Call TensorFlow commands from MATLAB

Live Editor - TFIntegration.mlx

TFIntegration.mlx

Create a model using Keras

Import some necessary Keras classes

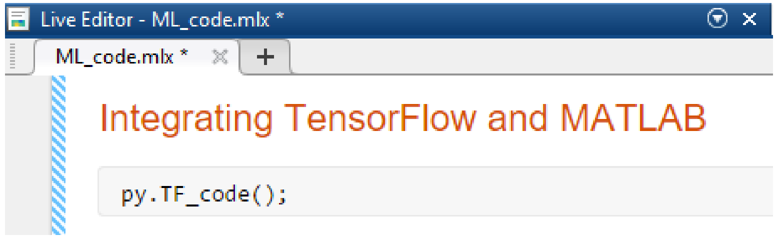
```
import py.tensorflow.keras.Sequential
import py.tensorflow.keras.layers.Dense
```

Run a small Keras example, modified from the example at the following page:

https://www.tensorflow.org/guide/keras

```
model = Sequential();
model.add(Dense(64, 'relu'));
model.add(Dense(64, 'relu'));
model.add(Dense(10, 'softmax'));

model.compile(py.tensorflow.train.AdamOptimizer(0.001),...
    'categorical_crossentropy',...
    py.list({'accuracy'}));
```

MATLAB-Python Co-Execution – Method A



TF_code.py

1. Copy the code into a .PY file
2. Wrap entry point in a function

```
import tensorflow as tf
from tf import keras

def myTFCode():
    for x in y:
        a.foo()

def foo(x):
    return x + 1
```

3. Add module to Python path and then call from MATLAB via:

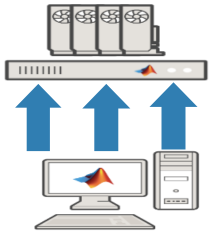
```
py.myModule.myTFCode();
```

Deep Learning on CPU, GPU, Multi-GPU and Clusters

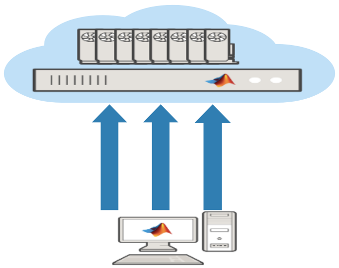
Single CPU

Single CPU Single GPU

Single CPU, Multiple GPUs



On-prem server with GPUs



Cloud GPUs (AWS)

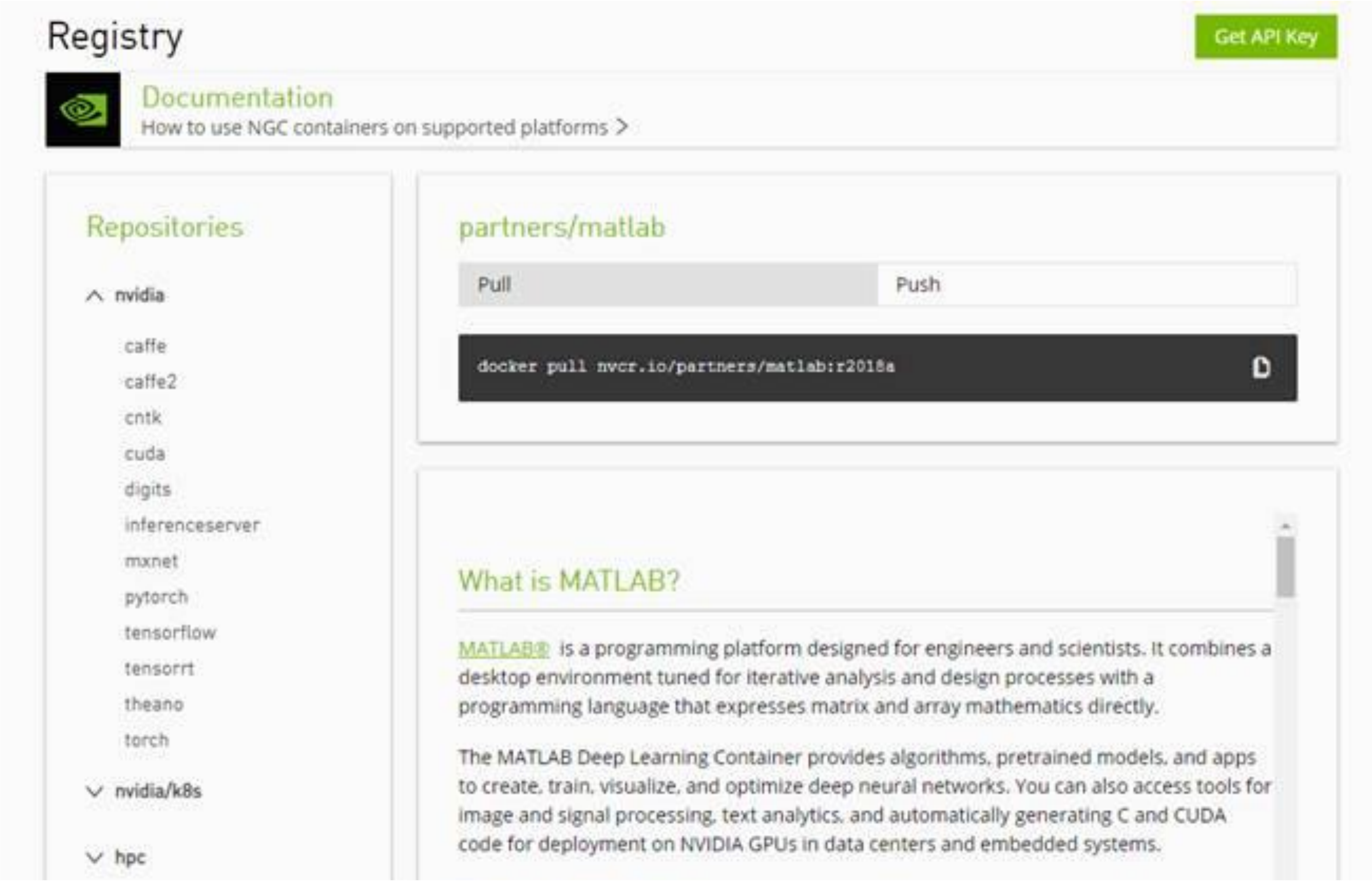
How to target?

```
opts = trainingOptions('sgdm', ...
      'MaxEpochs', 100, ...
      'MiniBatchSize', 250, ...
      'InitialLearnRate', 0.00005, ...
'ExecutionEnvironment', 'auto' );
```

```
opts = trainingOptions('sgdm', ...
      'MaxEpochs', 100, ...
      'MiniBatchSize', 250, ...
      'InitialLearnRate', 0.00005, ...
'ExecutionEnvironment', 'multi-gpu' );
```

```
opts = trainingOptions('sgdm', ...
      'MaxEpochs', 100, ...
      'MiniBatchSize', 250, ...
      'InitialLearnRate', 0.00005, ...
'ExecutionEnvironment', 'parallel' );
```

MATLAB Containers for NVIDIA GPU Cloud & DGX



Deep Learning Challenges

Data

- ✓ Handling large amounts of data
- ✓ Labeling thousands of signals, images & videos
- ✓ Transforming, generating, and augmenting data (for different domains)

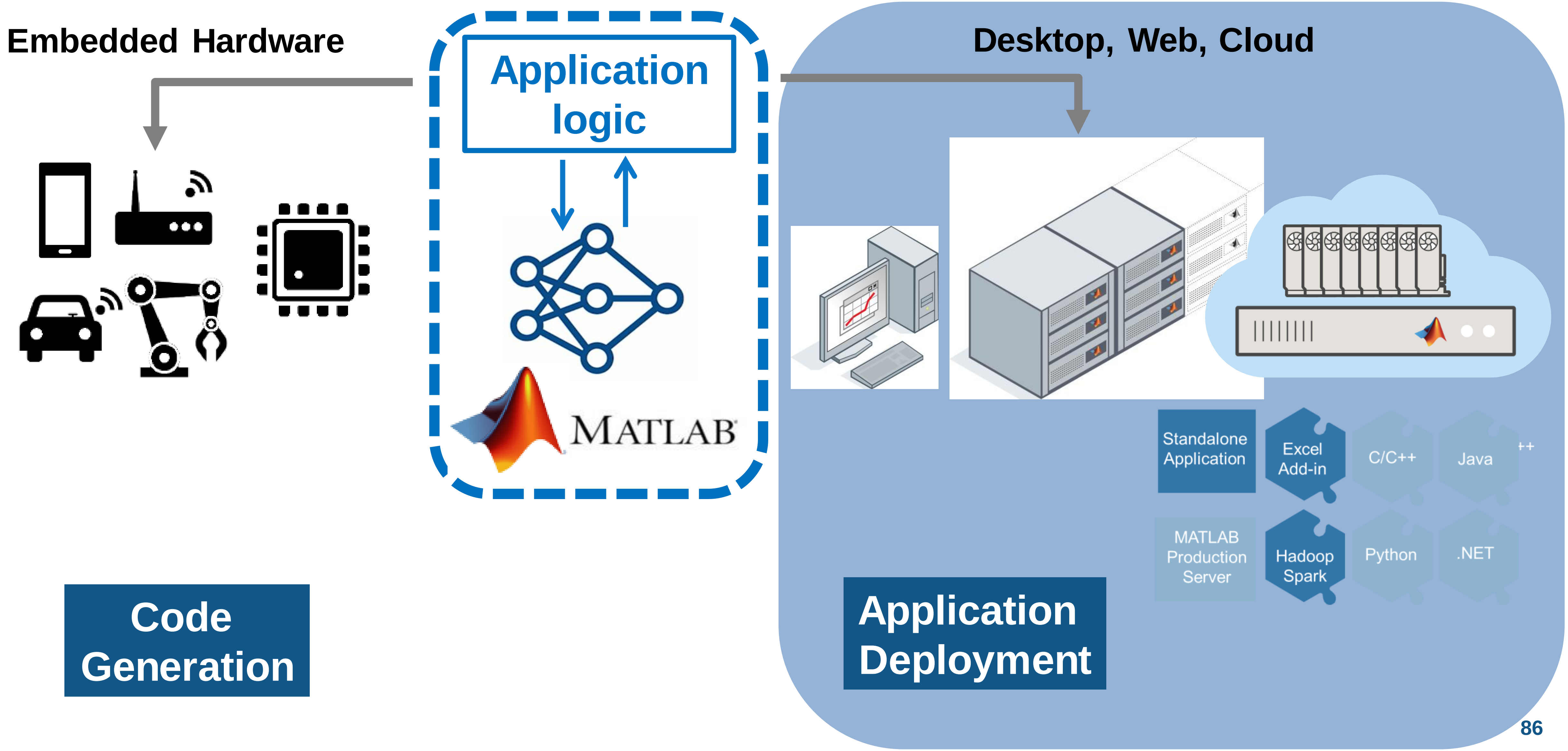
Training and Testing Deep Neural Networks

- ✓ Accessing reference models from research
- ✓ Understanding network behaviour
- ✓ Optimizing hyperparameters
- ✓ Training takes hours-days

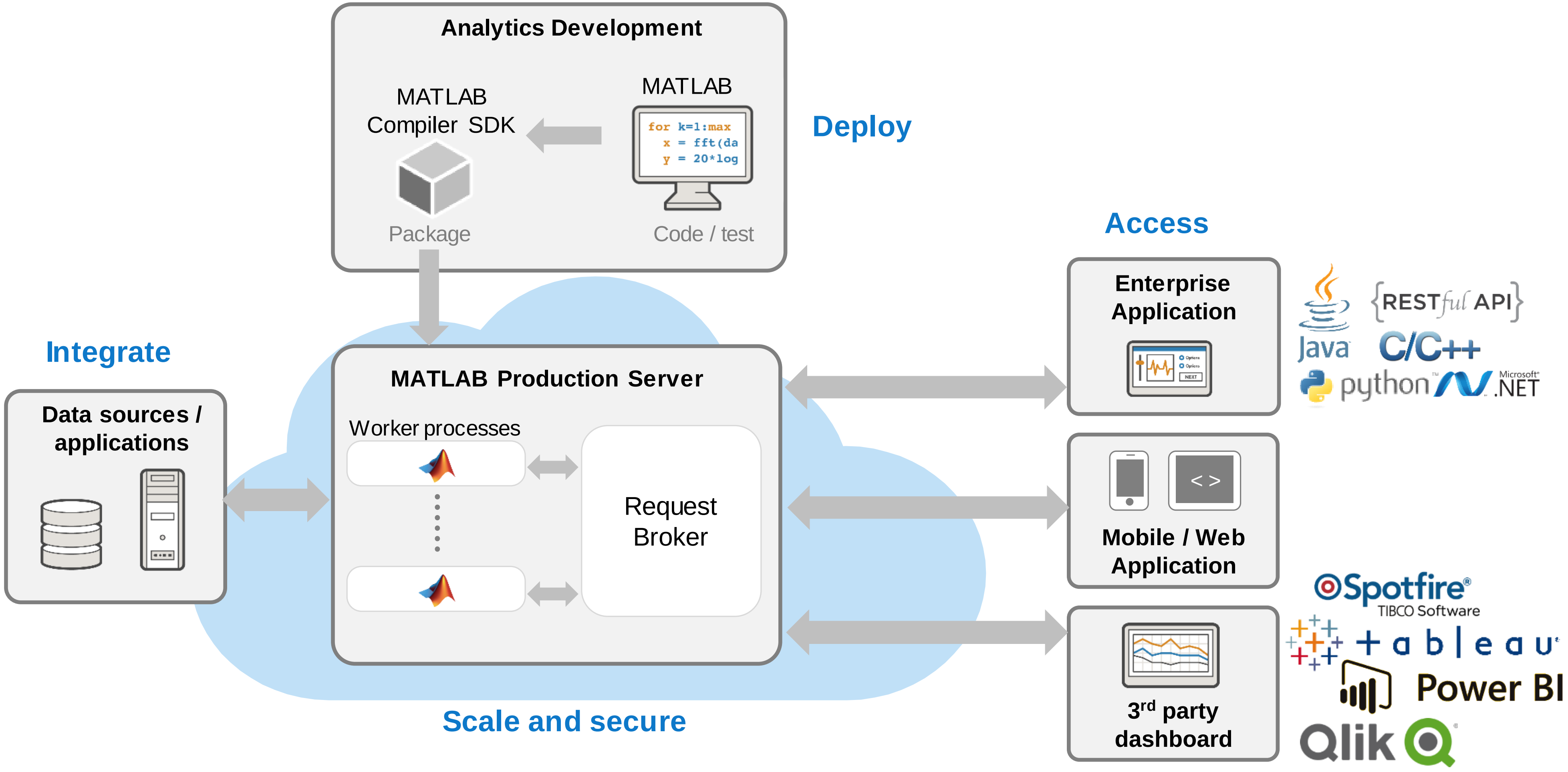
Rapid and Optimized Deployment

- Desktop, web, cloud, and embedded hardware

Deploying Deep Learning Application



MATLAB Production Server is an application server that publishes MATLAB code as APIs that can be called by other applications



MATLAB support for Cloud

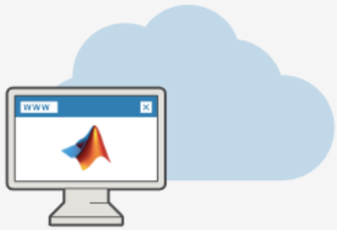
Use MATLAB in the Cloud

Run in different cloud environments from MathWorks Cloud to public clouds including AWS, Azure, and others

MathWorks Cloud

MathWorks Cloud provides you with instant access to MATLAB and other products and services you are licensed for hosted on MathWorks managed cloud infrastructure. With MATLAB Online™, you can use MATLAB in a web browser without installing, configuring, or managing any software. MathWorks Cloud also provides MATLAB Drive™, giving you the ability to store, access, and work with your files from anywhere. You can access MathWorks Cloud solutions anywhere across different devices, use them to teach and learn, and to incorporate MATLAB analytics for a variety of applications.

Learn more about hosted offerings.



Public Clouds

Use MATLAB on virtual machines in public cloud environments like Amazon Web Services (AWS) and Microsoft Azure. These vendors provide access to on-demand computing resources. They also offer wide-ranging, prebuilt services for data storage, data streaming, elastic scaling, load balancing, security, and more.

If you are not a cloud expert, or if you want a head start, use a MathWorks published reference architecture. Templates in these reference architectures automatically create and configure the cloud infrastructure for running MATLAB. You can also adapt or extend the reference architectures to better meet your specific needs.

Learn more about running MATLAB and other products on:

AWS | Azure | Other Clouds

powered by aws

Azure

Features Business Explore Marketplace Pricing Search Sign in or Sign up

MathWorks Reference Architectures

Reference Architectures

http://www.mathworks.com/cloud

Repositories 6 | People 1 | Projects 0

Grow your team on GitHub

GitHub is home to over 28 million developers working together. Join them to grow your own development teams, manage permissions, and collaborate on projects.

Sign up

Dismiss

Search repositories... | Type: All | Language: All

mps-on-azure
Stand up a MATLAB Production Server using Azure Deployment
PowerShell ★ 3 Updated 11 days ago

mps-on-aws
Stand up a MATLAB Production Server using CloudFormation
★ 6 Updated 13 days ago

matlab-on-aws
Stand up a MATLAB desktop with Remote Desktop access using AWS CloudFormation
★ 4 Updated 13 days ago

mdcs-on-aws
Stand up a MATLAB Distributed Computing Server cluster using CloudFormation
Shell ★ 4 1 Updated 14 days ago

mdcs-on-azure
Stand up a MATLAB Distributed Computing Server cluster using Azure Deployment
PowerShell ★ 2 1 Updated 14 days ago

Top languages
PowerShell Shell

People 1
jfluet
John Fluet

Deployment Steps

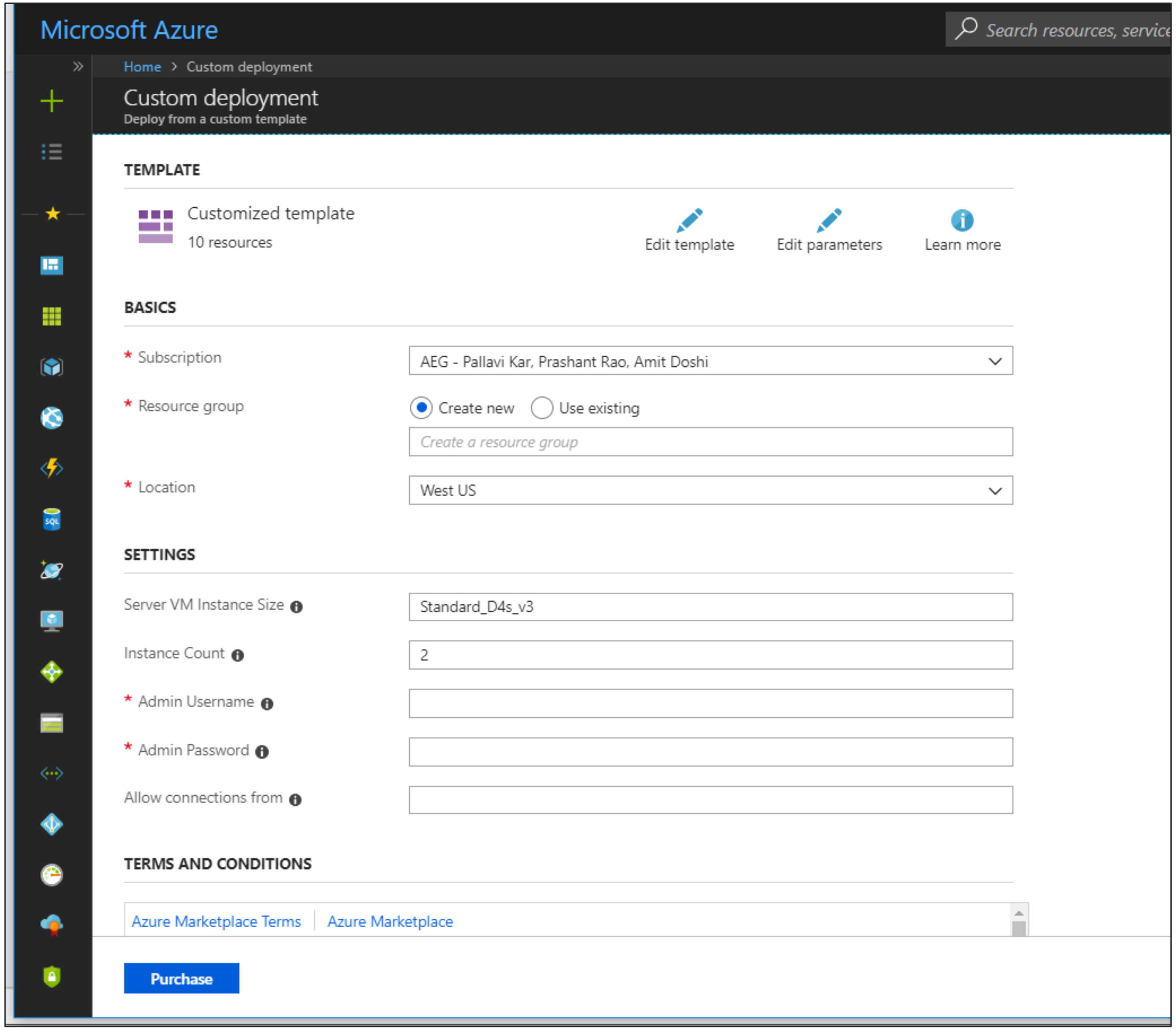
Step 1. Launch the Template

Click the **Deploy to Azure** button to deploy resources on Azure. This will open the Azure Portal in your web browser.

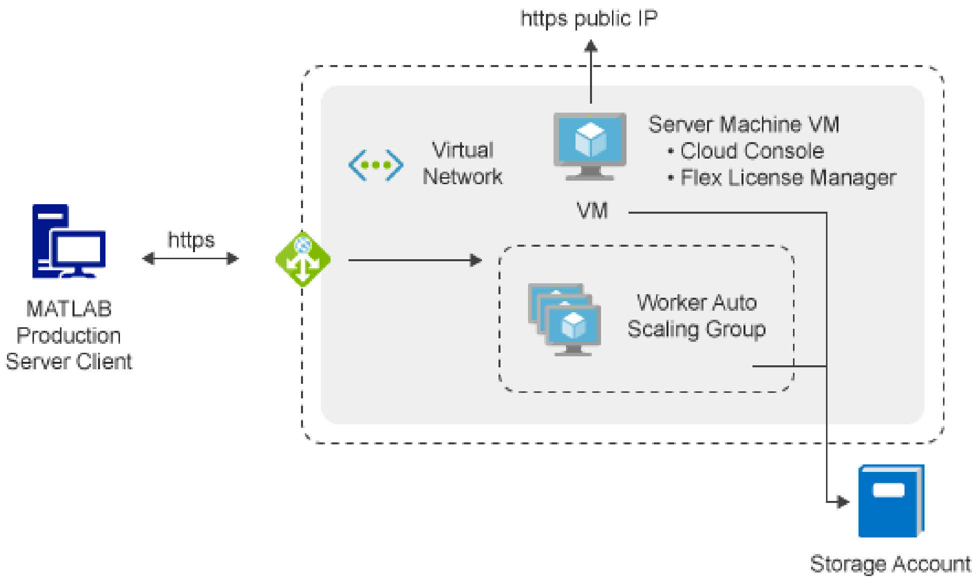
Windows Server 2016 VM	Ubuntu 16.04 VM
Deploy to Azure MATLAB Release: R2018a	Deploy to Azure MATLAB Release: R2018a

Note: Creating resources on Azure can take at least 30 minutes.

IAAS to PAAS

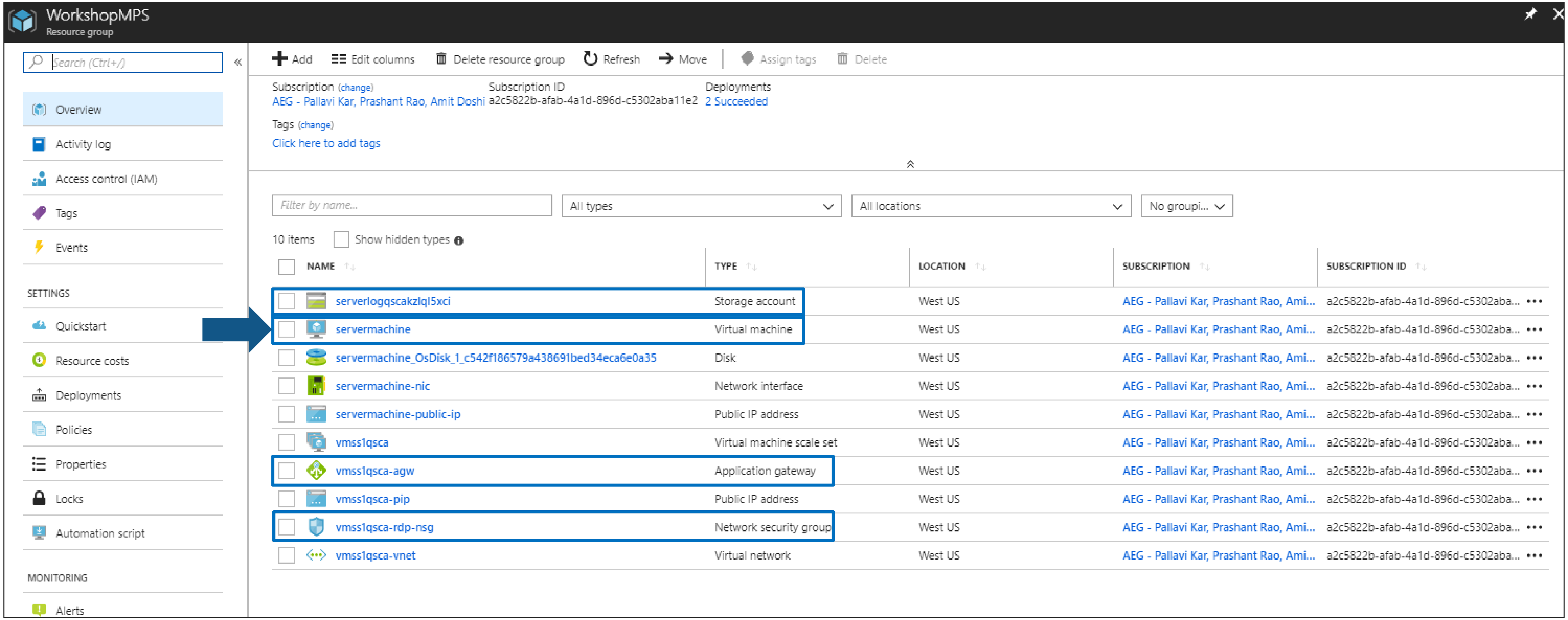


Microsoft Azure
Search resources, service
Home > Custom deployment
Custom deployment
Deploy from a custom template
TEMPLATE
Customized template
10 resources
Edit template
Edit parameters
Learn more
BASICS
* Subscription
AEG - Pallavi Kar, Prashant Rao, Amit Doshi
* Resource group
Create new
Use existing
Create a resource group
* Location
West US
SETTINGS
Server VM Instance Size
Standard_D4s_v3
Instance Count
2
* Admin Username
* Admin Password
Allow connections from
TERMS AND CONDITIONS
Azure Marketplace Terms
Azure Marketplace
Purchase

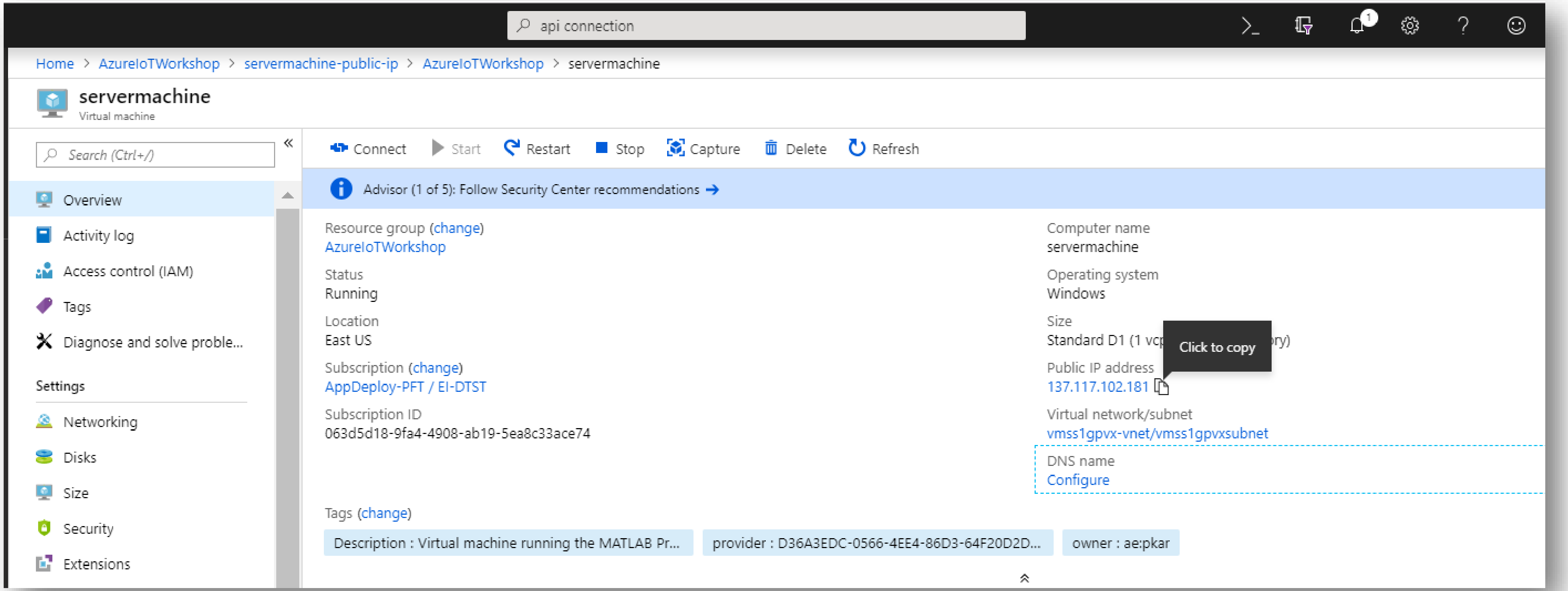


https public IP
Virtual Network
Server Machine VM
• Cloud Console
• Flex License Manager
VM
MATLAB Production Server Client
https
Worker Auto Scaling Group
Storage Account

Automatic Cluster Creation



Get Public IP for access



Server Machine VM access

- MPS console endpoint: https://xxx.xx.xx.xx

Connect | Start | Restart | Stop | Capture | Delete | Refresh

Advisor (1 of 1): Use availability sets for improved fault tolerance →

Resource group (change) WorkshopMPS	Computer name servermachine
Status Running	Operating system Windows
Location West US	Size Standard D1 (1 vcpus, 3.5 GB memory)
Subscription (change) AEG - Pallavi Kar, Prashant Rao, Amit Doshi	Public IP address 40.118.147.236
Subscription ID a2c5822b-afab-4a1d-896d-c5302aba11e2	Virtual network/subnet vmss1qsca-vnet/vmss1qscasubnet
	DNS name Configure

Tags (change)

Description : Virtual machine running the MATLAB Productio... | provider : D36A3EDC-0566-4EE4-86D3-64F20D2DDA06

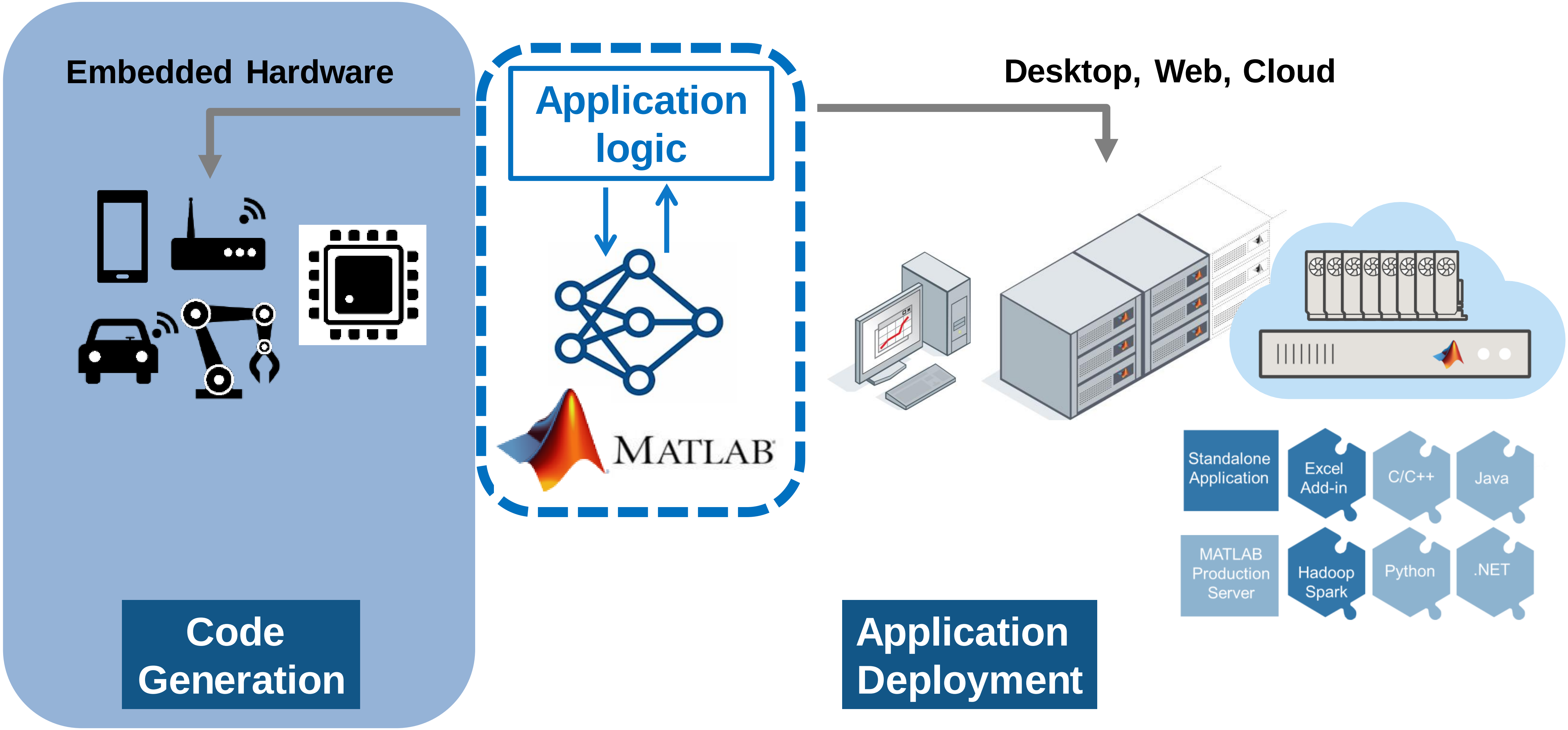
Status:	Running
Number of MATLAB Production Server VMs:	2
Number of MATLAB Production Server Workers per VM:	4
Total Number of Workers:	8
HTTPS Server Endpoint:	https://mpsqscakzlql5xci.westus.cloudapp.azure.com:9910

Additional Information

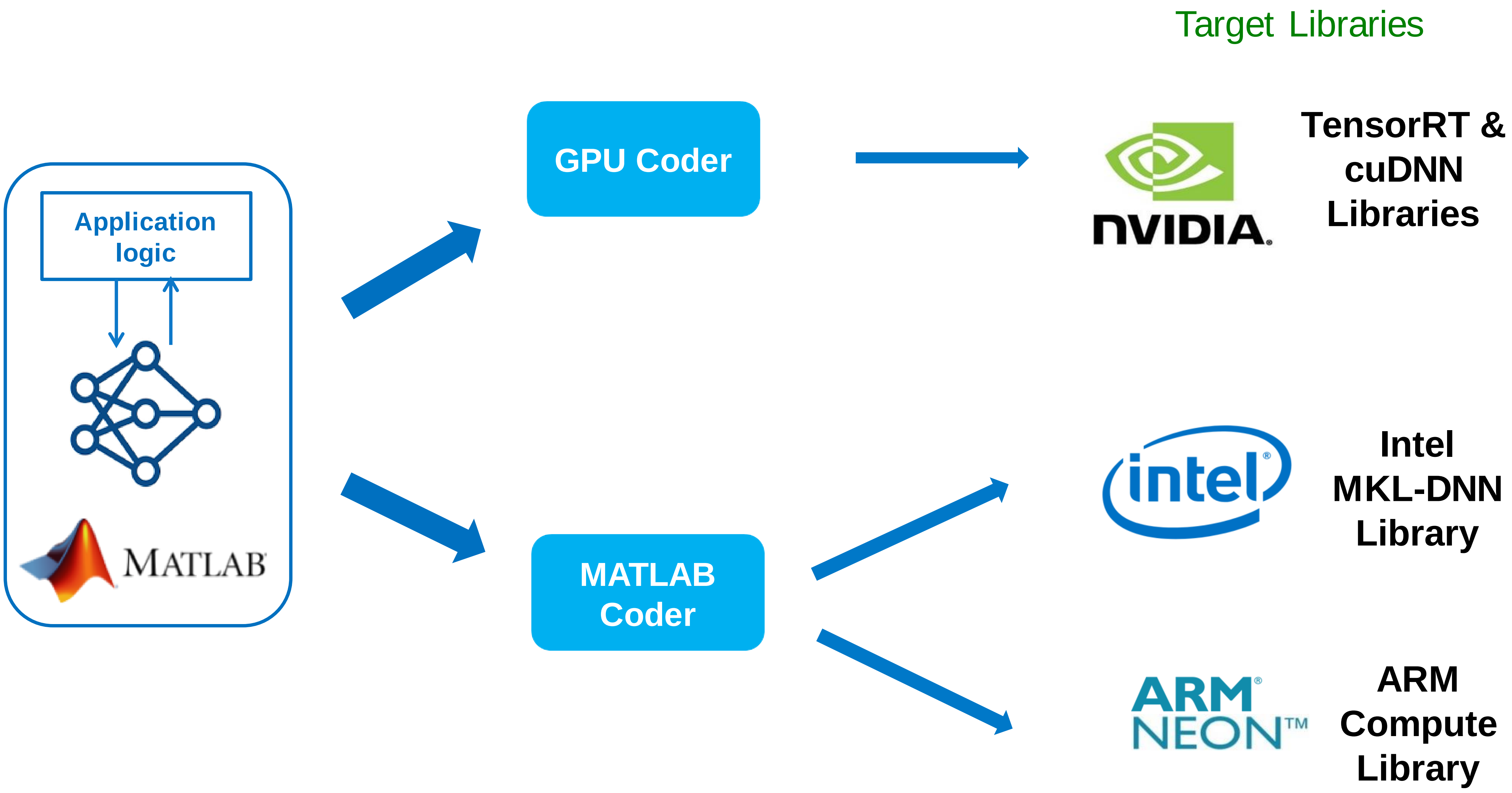
To start using the server:

1. Get a license from the MathWorks License Center and upload it in the Manage License section.
2. Use the HTTPS server endpoint to make requests to the server.

Deploying Deep Learning Application



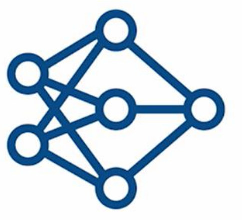
Solution- GPU/MATLAB Coder for Deep Learning Deployment



Deep Learning Deployment Workflows

INFERENCE ENGINE DEPLOYMENT

Trained DNN



cnncodegen

Portable target code

INTEGRATED APPLICATION DEPLOYMENT

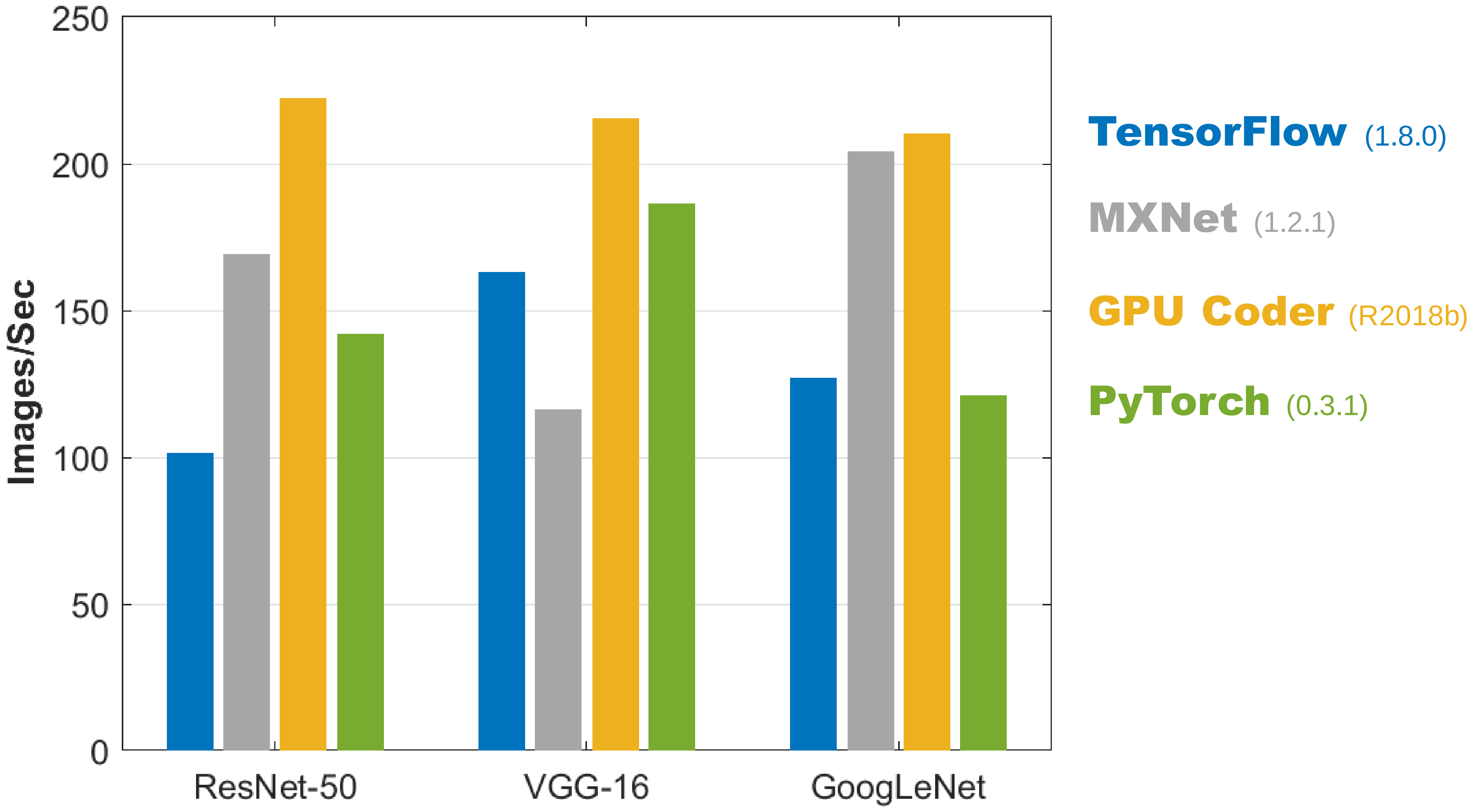
Pre-processing

Post-processing

codegen

Portable target code

Single Image Inference on Titan Xp using cuDNN

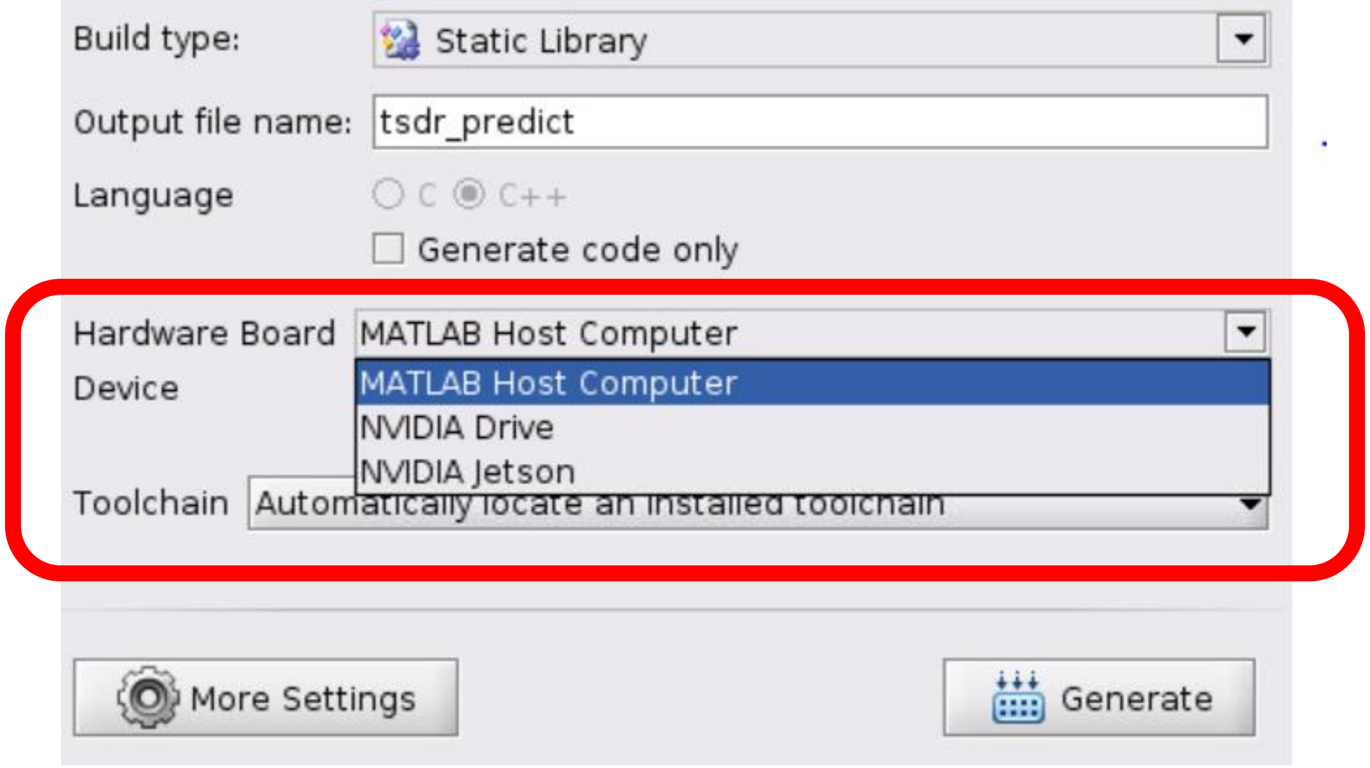


Intel® Xeon® CPU 3.6 GHz - NVIDIA libraries: CUDA9 - cuDNN 7

NVIDIA Hardware Support Package (HSP)

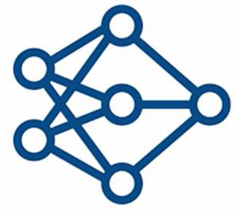
Simple out-of-box targeting for NVIDIA boards:

Jetson **Drive platform**



INFERENCE ENGINE DEPLOYMENT

Trained DNN



cnncodegen

Portable target code

Summary- MATLAB Deep Learning Framework

DEPLOYMENT

Access Data

Design + Train

Intel MKL-DNN Library

NVIDIA TensorRT & cuDNN Libraries

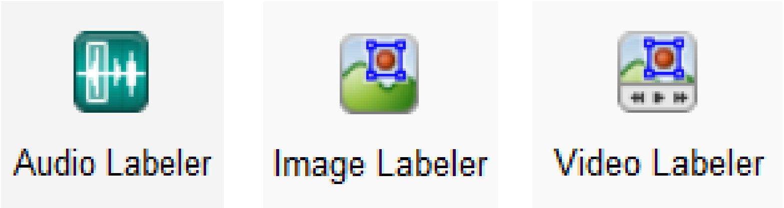
ARM Compute Library

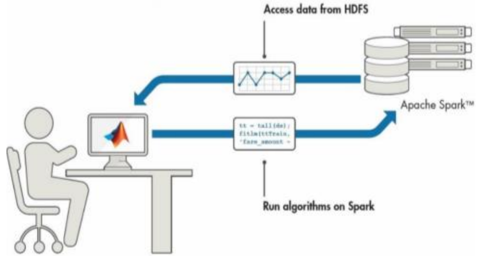
- **Manage** large image sets
- **Automate** image labeling
- **Easy access** to models

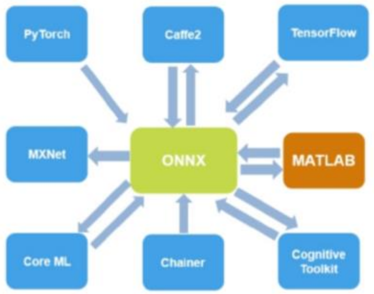
- **Acceleration** with GPU’s
- **Scale** to clusters

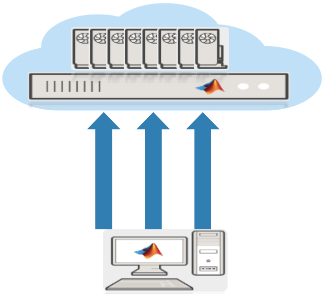
Summary

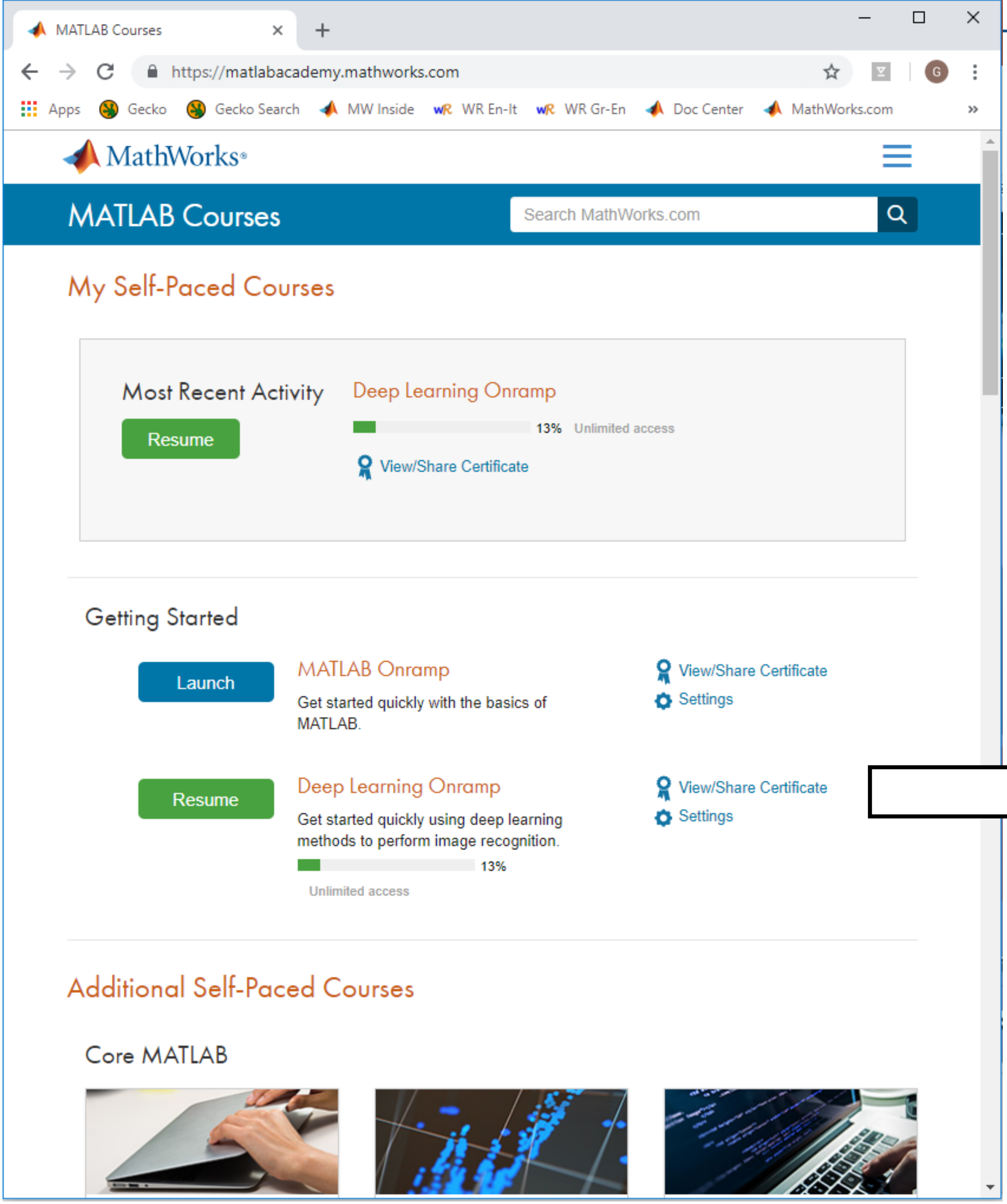
- ✓ Create and validate Deep learning models
- ✓ Automate ground truth labeling
- ✓ Access large amount of data from cluster/cloud
- ✓ Interoperability with Deep learning frameworks
- ✓ Visualization and hyperparameter tuning
- ✓ Seamlessly scale training to GPUs, clusters and cloud
- ✓ Deployment on embedded targets and web services

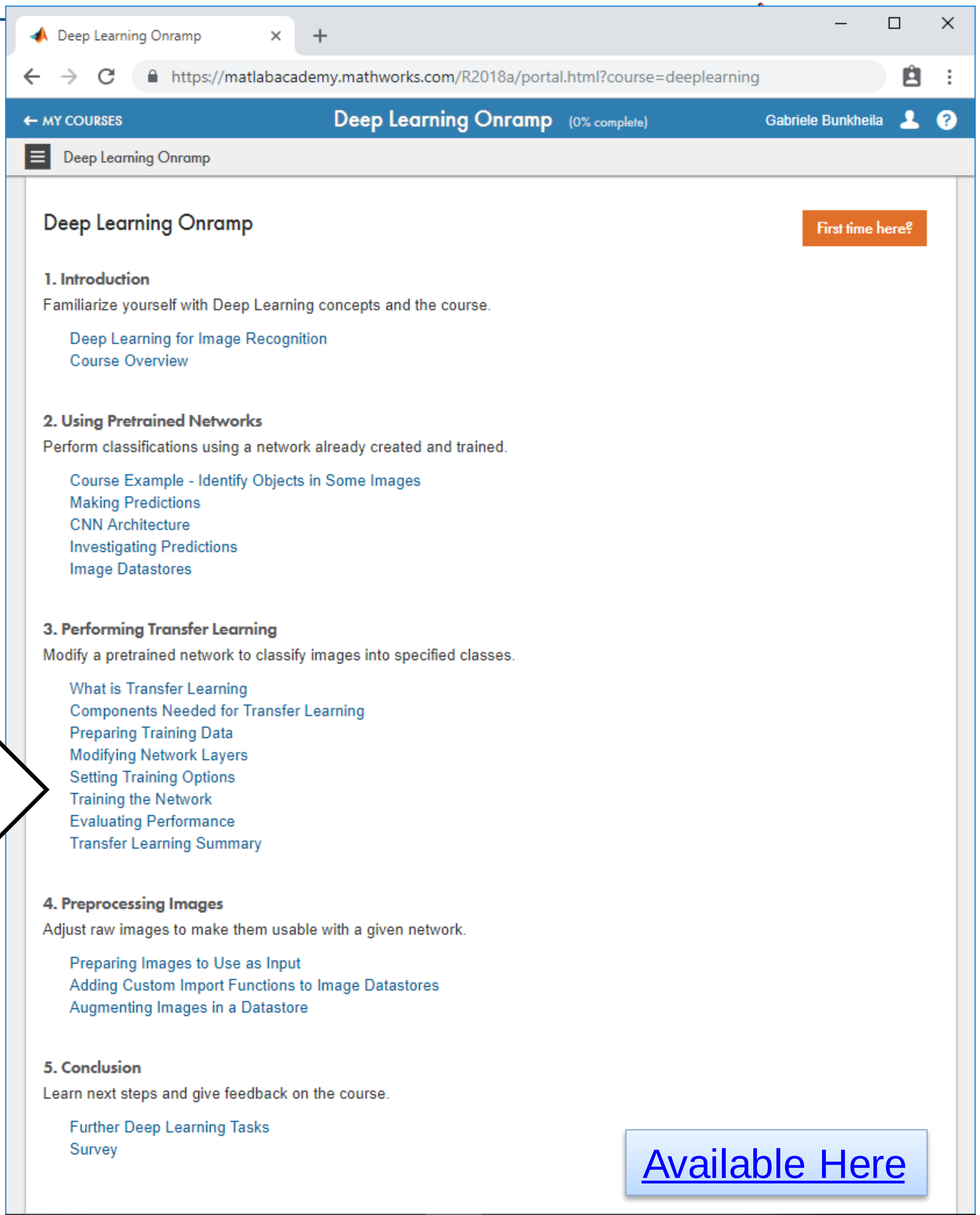












Available Here

Deep Learning with MATLAB

Prerequisites

MATLAB Fundamentals
Deep Learning Onramp

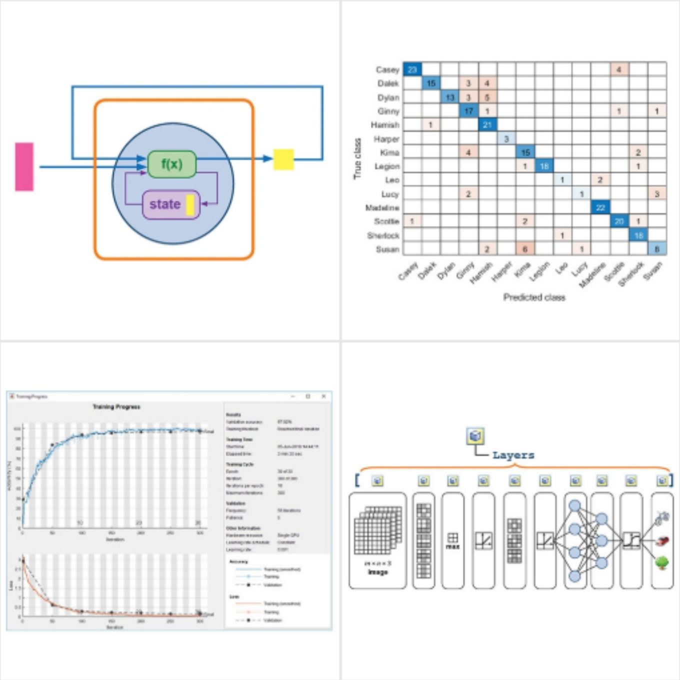
This course is also offered in an online, self-paced format.

Self-paced courses provide active engagement with MATLAB through in-browser, hands-on exercises that you can complete anytime, anywhere, at your own pace.

Watch: The Advantages of Self-Paced Training (1:03)

This two-day course provides a comprehensive introduction to practical deep learning using MATLAB®. Attendees will learn how to create, train, and evaluate different kinds of deep neural networks. Topics include:

- Importing image and sequence data
- Using convolutional neural networks for image classification, regression, and object detection
- Using long short-term memory networks for sequence classification and forecasting
- Modifying common network architectures to solve custom problems
- Improving the performance of a network by modifying training options



See detailed course outline

MathWorks®

Accelerating the pace of engineering and science