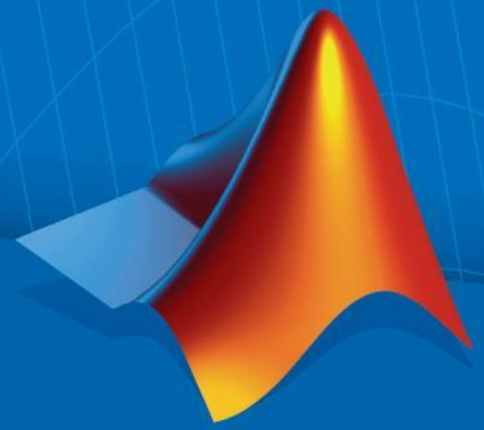
TeraSoft
www.terasoft.com.tw



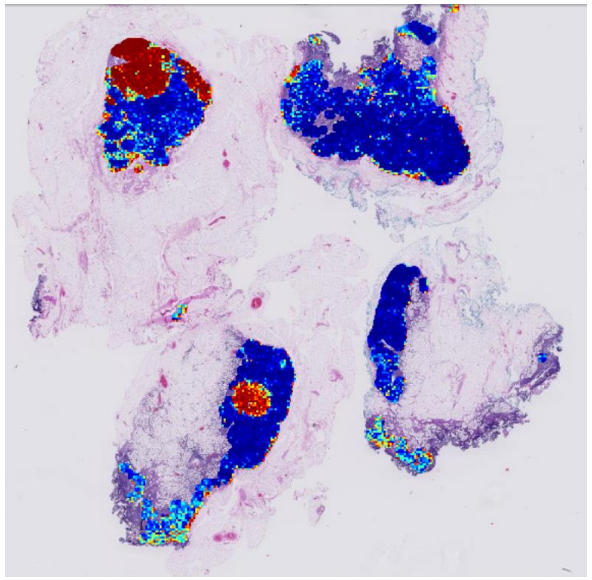
庖丁解魚?剖析深度學習 & MATLAB最新功能

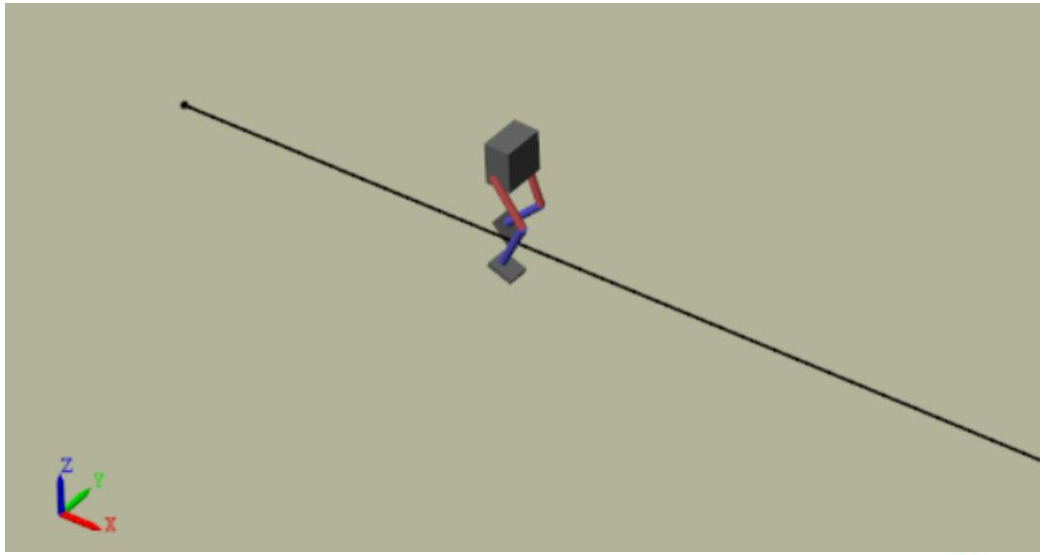
Fred Liu

Application Engineer

MathWorks®
Accelerating the pace of engineering and science

What can MATLAB do?





Outline

Deep learning APP

Deep learning Model

Deep learning Import/Deploy

Signal Application

Audio Application

Text Application

Image Application

Outline

Deep learning APP

Deep learning Model

Deep learning Import/Deploy

Signal Application

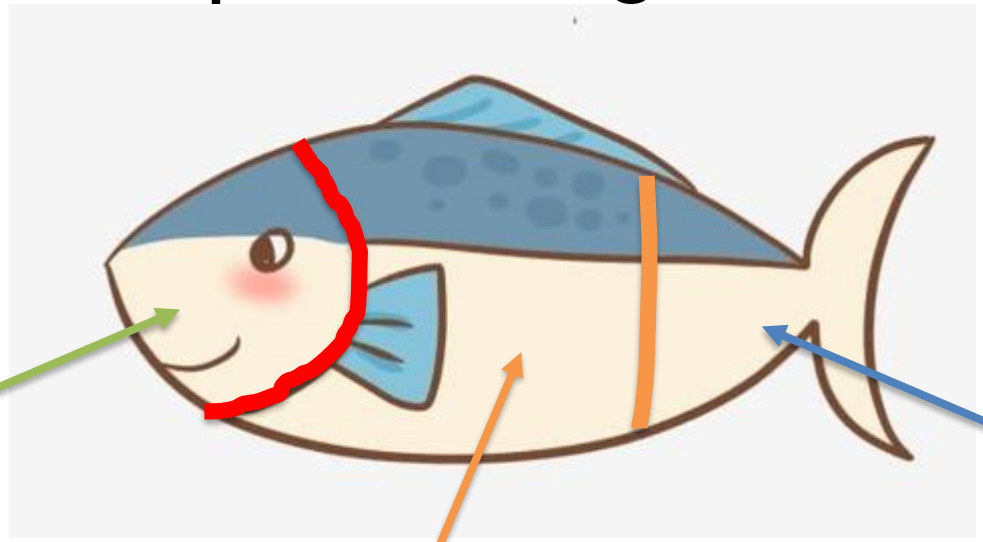
Audio Application

Text Application

Image Application

What is the composition of deep learning?

Deep Learning Model



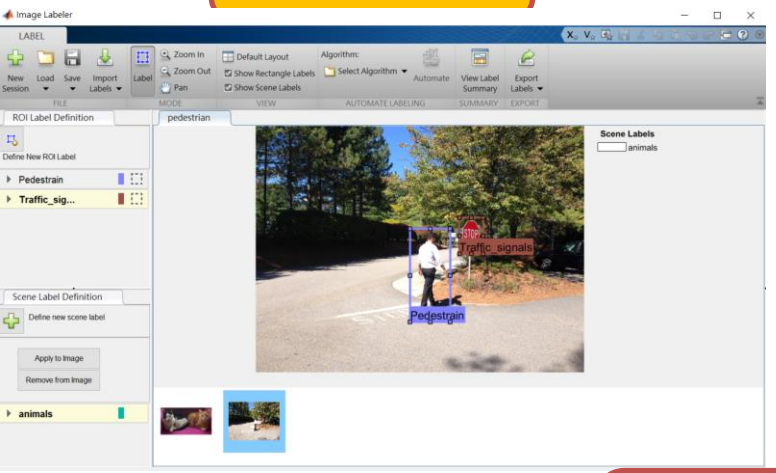
Data

Model

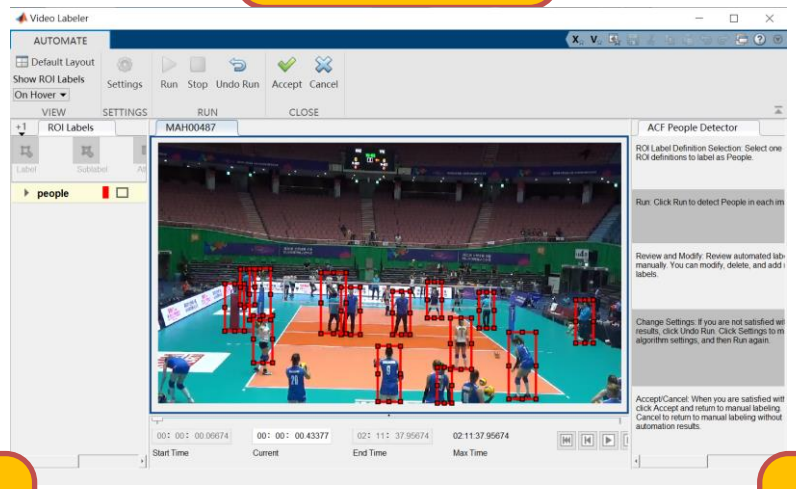
Option (parameter)

Labeler

Image Labeler

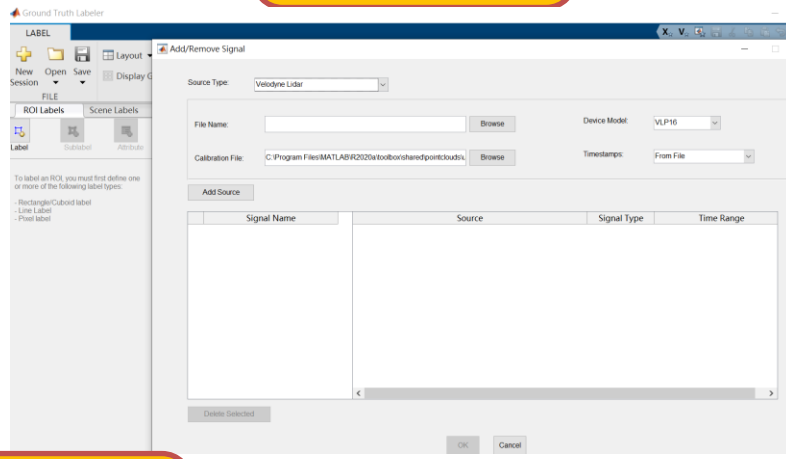


Video Labeler

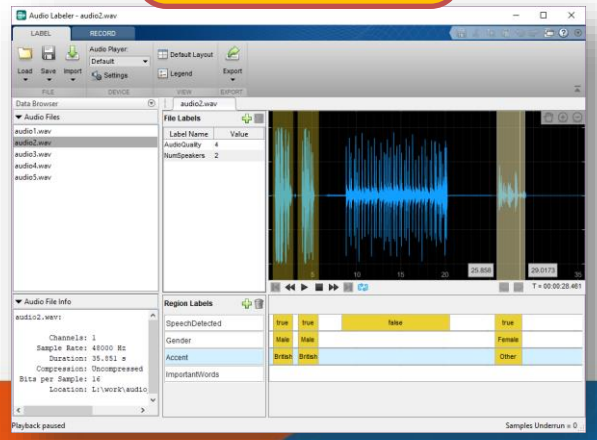


Ground Truth Labeler

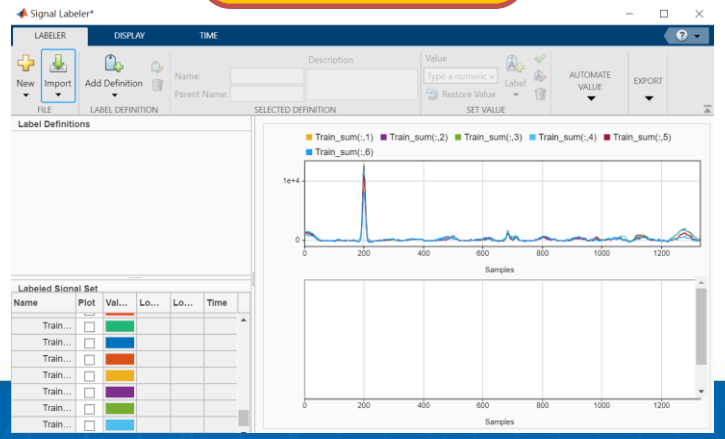
車用!



Audio Labeler

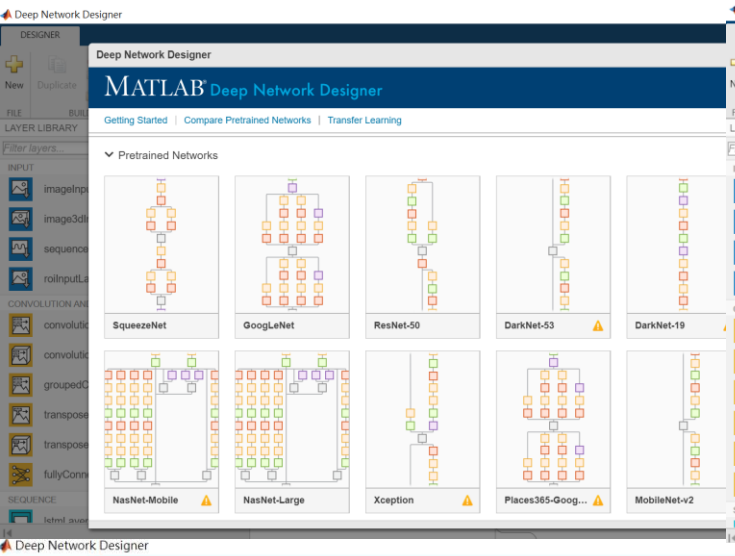


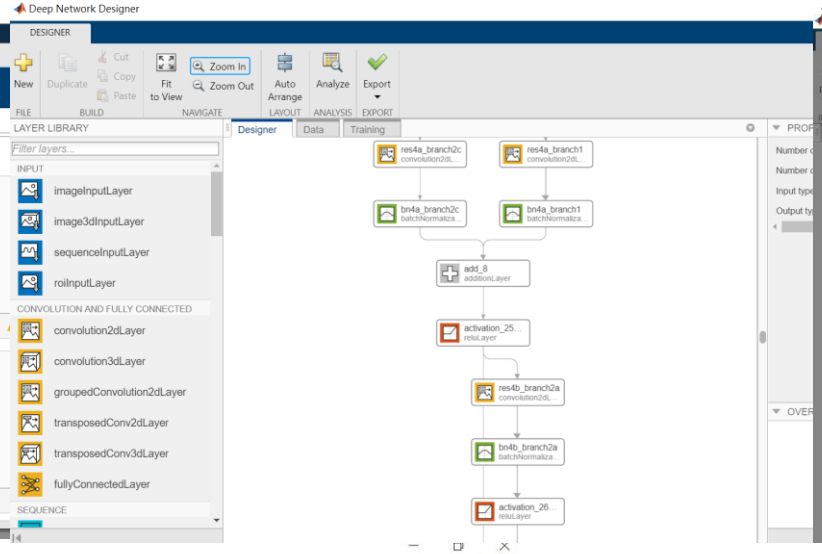
Signal Labeler

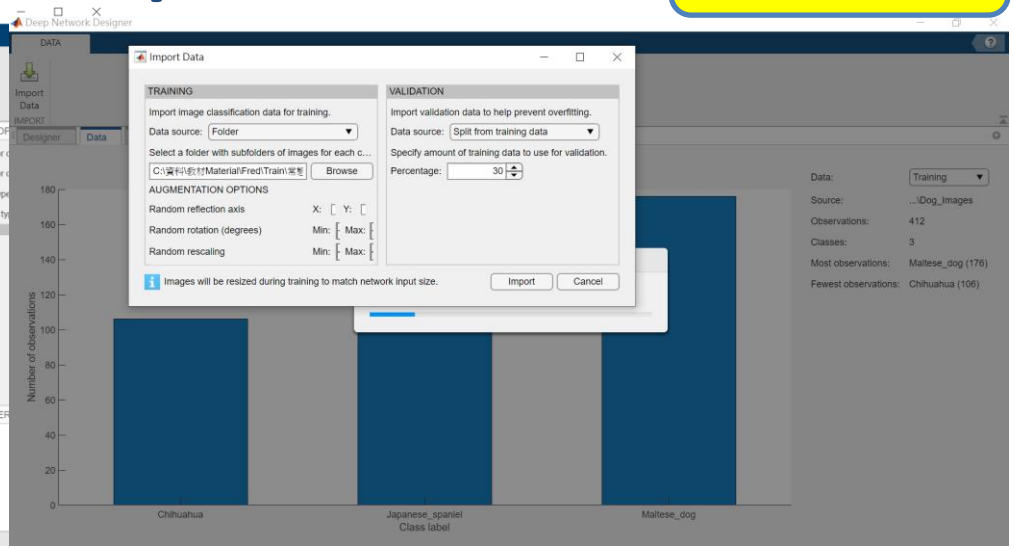


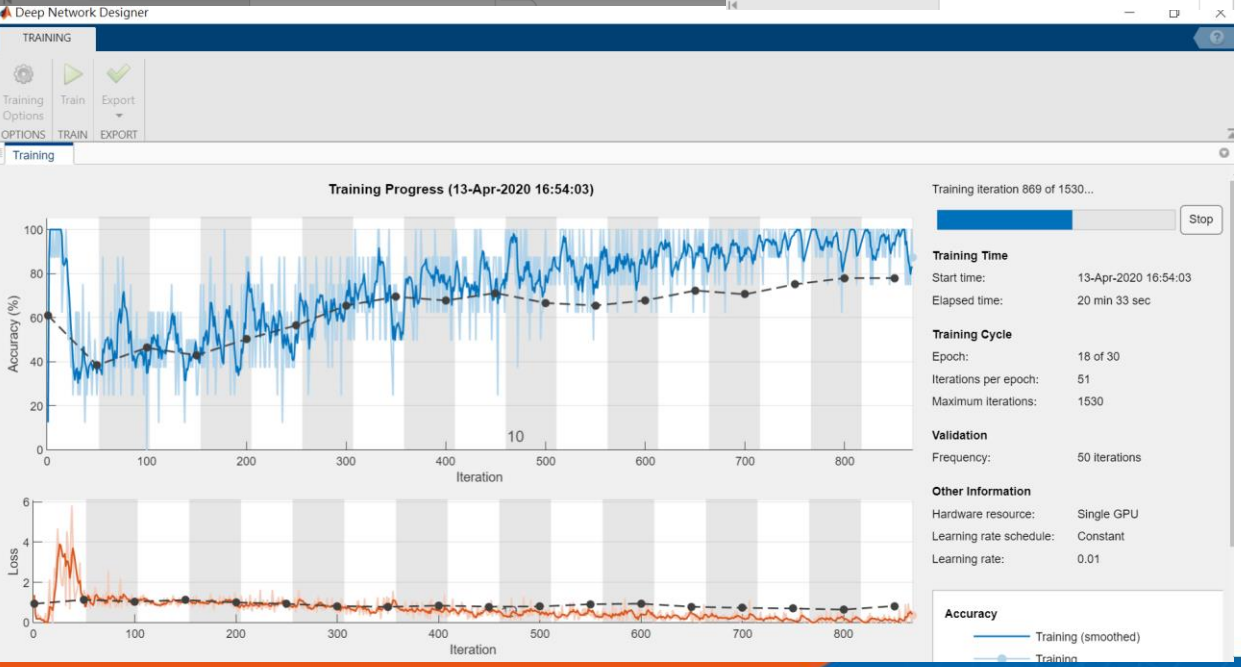
Update : Deep Network Designer(MW)

Model!









功用：建模型，訓練模型

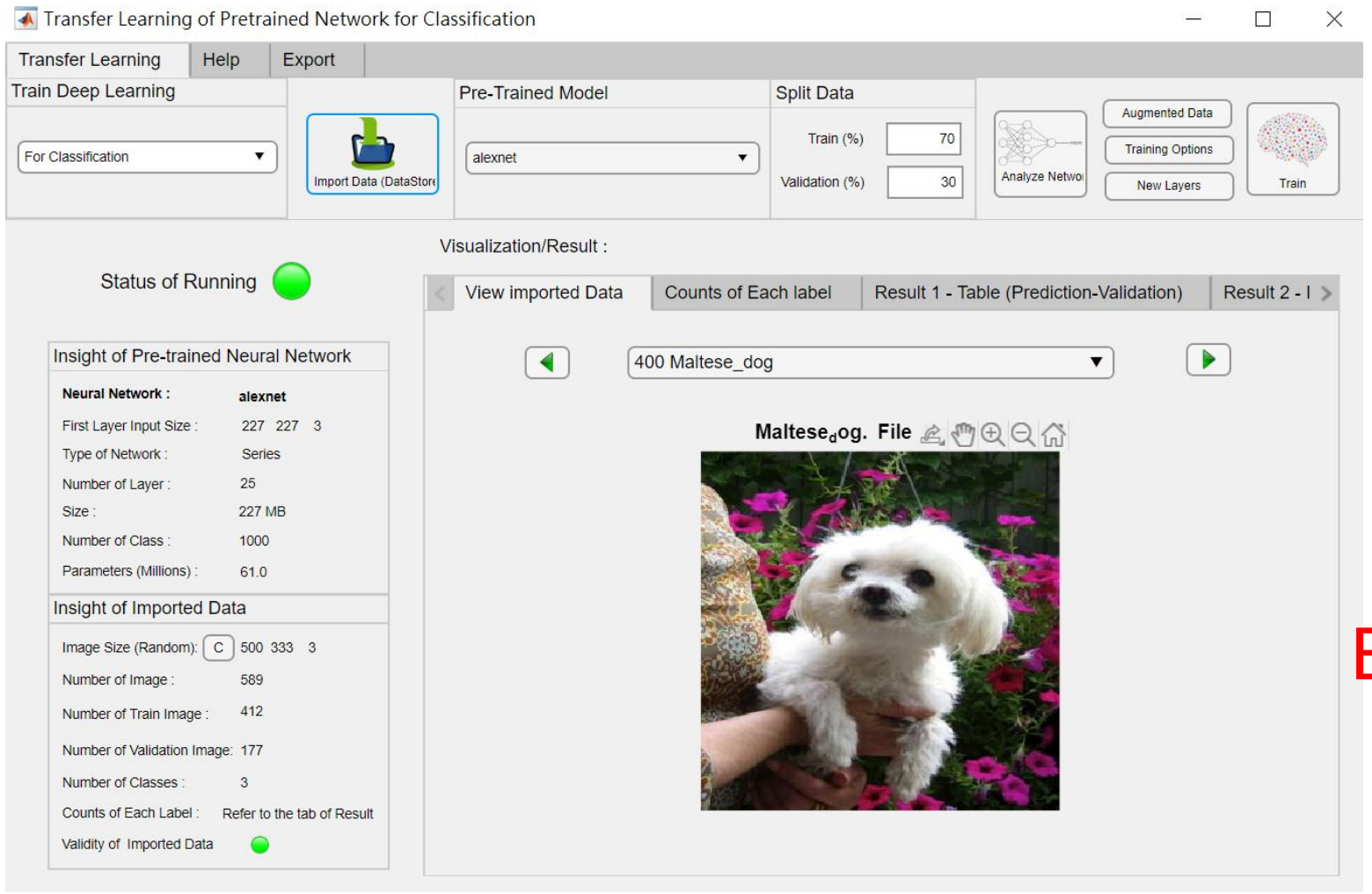
優點：較方便客製化模型，不需要寫程式碼

缺點：步驟較多，不夠可視化

General

Update : Transfer Learning App(Add-On)

Average!



功用：建模型，訓練模型，部屬

優點：快速上手不需要寫程式碼，兩步驟即可完成，支持各種型態模型的轉出

缺點：模型客制化能力較差

Easy to use! Two steps!

Novice

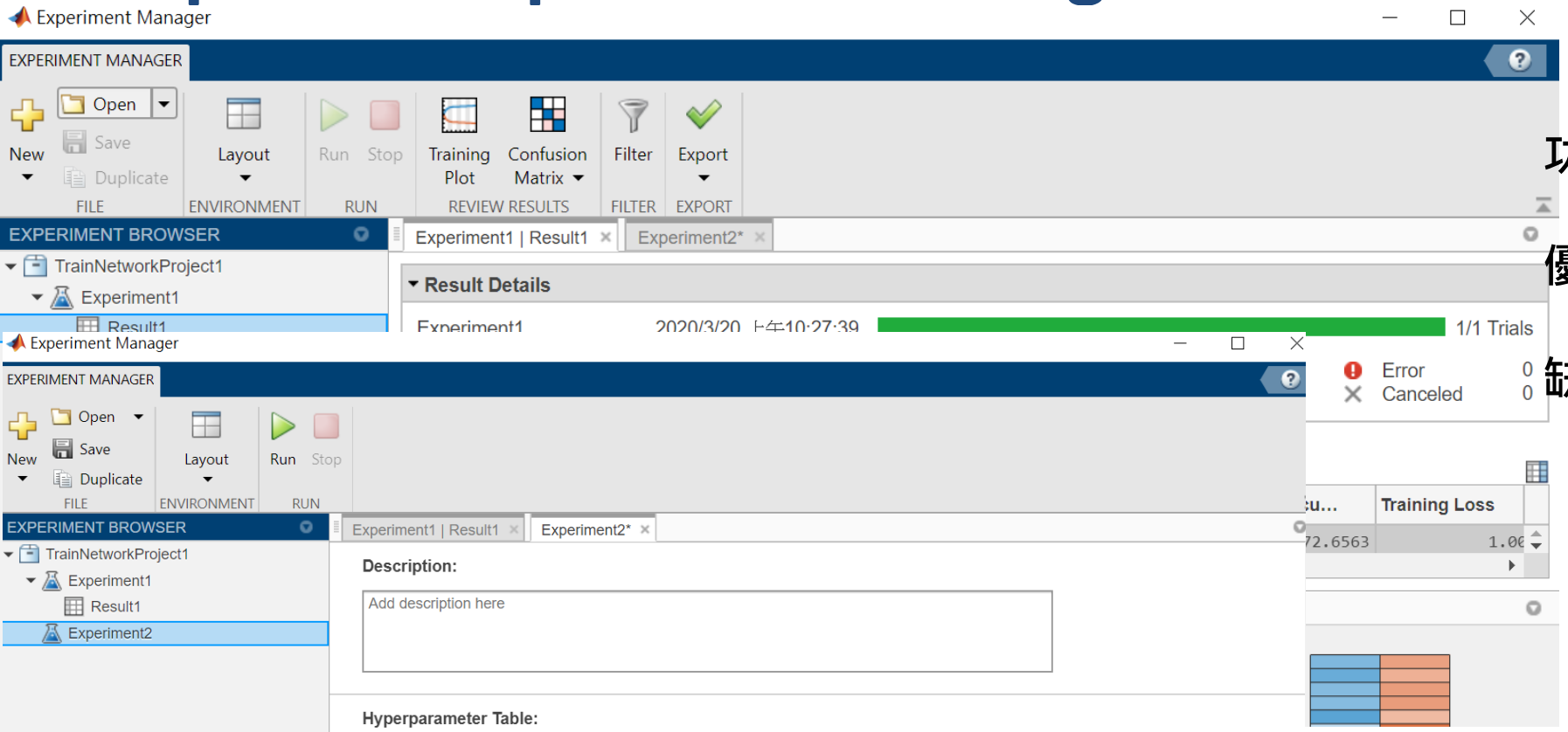
Update : Experiment Manager

Option! (parameter)

功用：訓練最佳參數

優點：更方便來找尋最佳參數

缺點：需要撰寫較多樣本函數



Specify Training Options

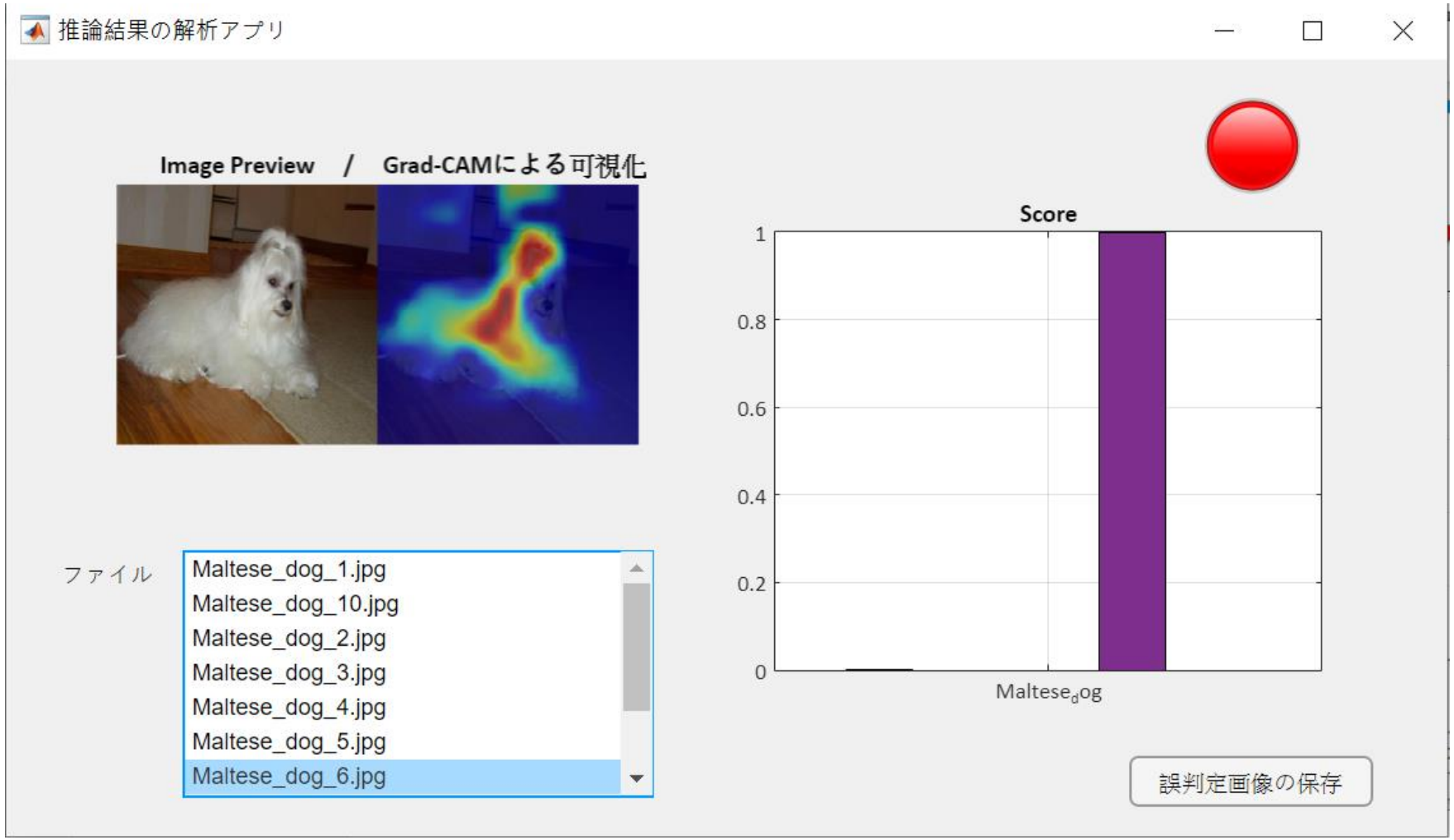
To use the hyperparameters specified in the Experiment Manager, access fields of notation. For example, to use the initial learn rate `myInitialLearnRate` defined in the `trainingOptions` function with the argument value `params.myInitialLearnI`

```
options = trainingOptions('sgdm', ...
    'MaxEpochs',5, ...
    'ValidationData',imdsValidation, ...
    'ValidationFrequency',30, ...
    'InitialLearnRate',params.myInitialLearnRate);
```

Export

Update :ディープラーニング評価キット (深度學習評估套件)

Data!



功用：可視化資料的熱區

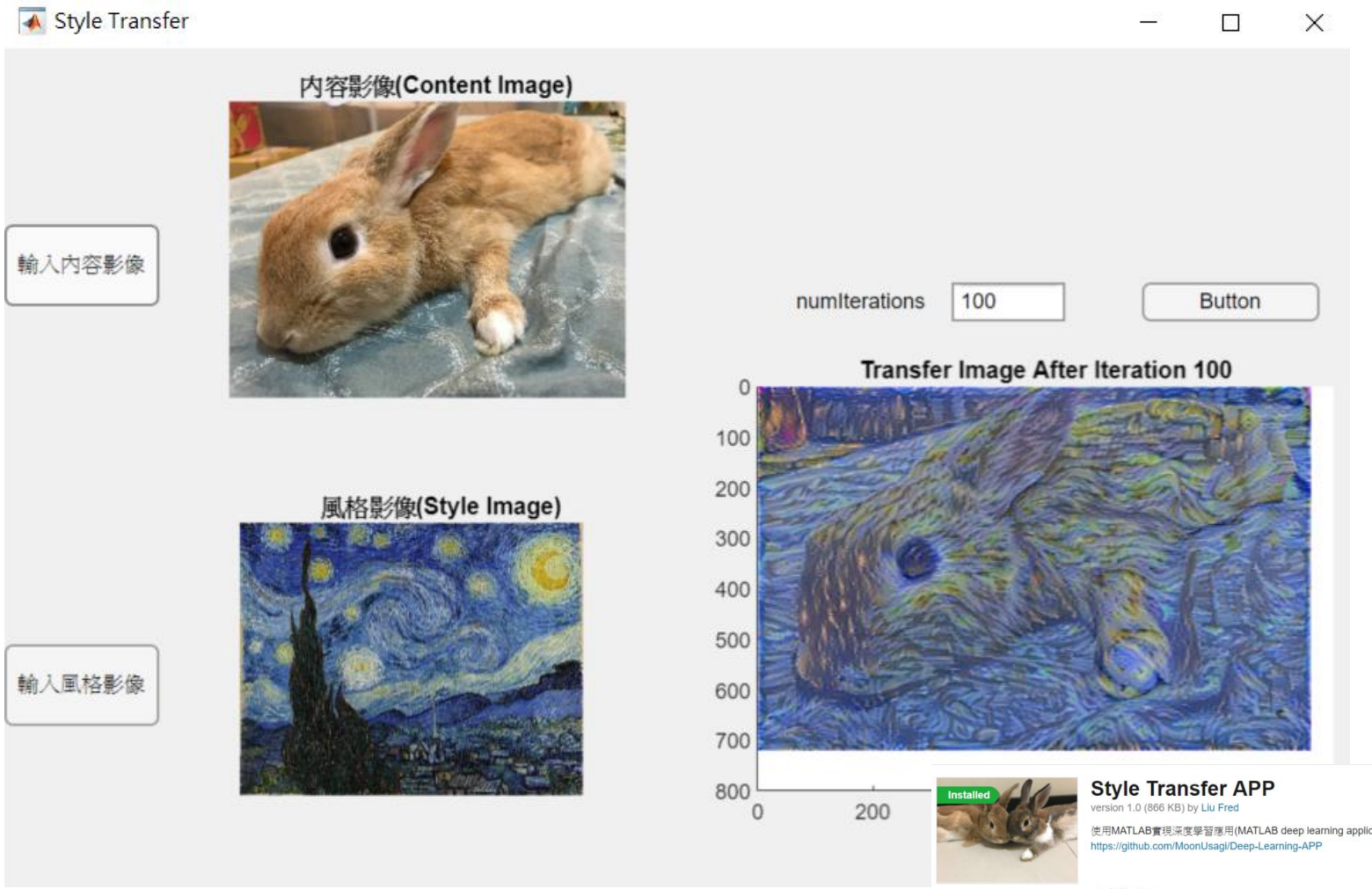
優點：可以更容易的了解模型透過怎樣的特徵學習，與是否認知道正確的特徵。

缺點：app比較不客製化

General

Myself：Style Transfer

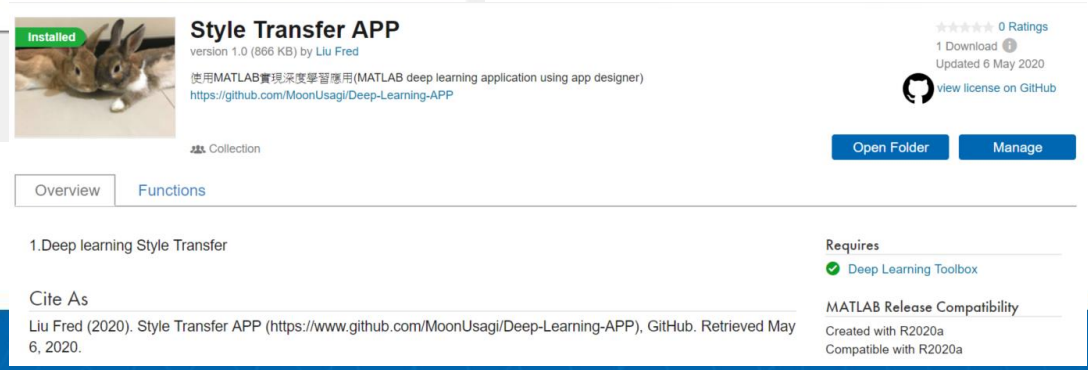
Model!



功用：將內容影像與風格影像作融合

優點：可生產各種風格融合的影像

缺點：無



General

Outline

Deep learning APP

Deep learning Model

Deep learning Import/Deploy

Signal Application

Audio Application

Text Application

Image Application

Doc: Data Sets for Deep Learning

Doc Link

Data Sets for Deep Learning

Use these data sets to get started with deep learning applications.

Image Data Sets

Data Set	Description

Digits	The digits data set consists of 10,000 synthetic grayscale images of handwritten digits. Each image is 28-by-28 pixels and has an associated label denoting which digit the image represents (0–9). Each image has been rotated by a certain angle. When loading the images as arrays, you can also load the rotation angle of the image.
	
	Load the digits data as in-memory numeric arrays using the `digitTrain4DArrayData` and `digitTest4DArrayData` functions.
	`[XTrain,YTrain,anglesTrain] = digitTrain4DArrayData;`
	`[XTest,YTest,anglesTest] = digitTest4DArrayData;`
	
	For examples showing how to process this data for deep learning, see Monitor Deep Learning Training Progress and Train Convolutional Neural Network for Regression.
	
	Load the digits data as an image datastore using the `imageDatastore` function and specify the folder containing the image data.
	`dataFolder = fullfile(toolboxdir('nnet'),'nndemos','nndatasets','DigitDataset');`
	`imds = imageDatastore(dataFolder, ...`
	`'IncludeSubfolders',true, ....`
	`'LabelSource','foldernames');`
	
	For an example showing how to process this data for deep learning, see Create Simple Deep Learning Network for

Flowers	The Flowers data set contains 3670 images of flowers belonging to five classes (*daisy*, *dandelion*, *roses*, *sunflowers*, and *tulips*).
	
	Download and extract the Flowers data set [2] from http://download.tensorflow.org/example_images/flower_photos.tgz. The data set is about 218 MB. Depending on your internet connection, the download process can take some time. Set `downloadFolder` to the location of the data.
	`url = 'http://download.tensorflow.org/example_images/flower_photos.tgz';`
	`downloadFolder = tempdir;`
	`filename = fullfile(downloadFolder,'flower_dataset.tgz');`
	`dataFolder = fullfile(downloadFolder,'flower_photos');`
	`if ~exist(dataFolder,'dir')`
	`fprintf("Downloading Flowers data set (218 MB)... ")`
	`websave(filename,url);`
	`untar(filename,downloadFolder)`
	`fprintf("Done.\n")`
	`end`
	
	Load the data as an image datastore using the `imageDatastore` function and specify the folder containing the image data.
	`imds = imageDatastore(dataFolder, ...`
	`'IncludeSubfolders',true, ...`
	`'LabelSource','foldernames');`
	
	For an example showing how to process this data for deep learning, see Train Generative Adversarial Network (GAN).

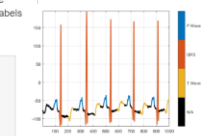
BraTS	The BraTS data set contains MRI scans of brain tumors, namely gliomas, which are the most common primary brain malignancies.
	
	The data set contains 750 4-D volumes, each representing a stack of 3-D images. Each 4-D volume is of size 240-by-240-by-155-by-4, where the first three dimensions correspond to the height, width, and depth of a 3-D volumetric image. The fourth dimension corresponds to different scan modalities. The data set is divided into 484 training volumes with voxel labels and 266 test volumes.
	
	Create a directory to store the BraTS data set [10].
	`dataFolder = fullfile(tempdir,'BraTS');`
	`if ~exist(dataFolder,'dir')`
	`mkdir(dataFolder);`
	`end`
	
	Download the BraTS data from Medical Segmentation Decathlon by clicking the "Download Data" link. Download the "Task01_BrainTumour.tar" file. The data set is about 7 GB. Depending on your internet connection, the download process can take some time.
	
	Extract the TAR file into the directory specified by the `dataFolder` variable. If the extraction is successful, then `dataFolder` contains a directory named `Task01_BrainTumour` that has three subdirectories: `imagesTr`, `imagesTs`, and `labelsTr`.
	
	For an example showing how to process this data for deep learning, see 3-D Brain Tumor Segmentation Using Deep Learning.

Factory Reports	The Factory Reports data set is a table containing approximately 500 reports with various attributes including a plain text description in the variable `Description` and a categorical label in the variable `Category`.
	
	Read the Factory Reports data from the file "`factoryReports.csv`". Extract the text data and the labels from the `Description` and `Category` columns, respectively.
	`filename = "factoryReports.csv";`
	`data = readtable(filename,'TextType','string');`
	`textData = data.Description;`
	`labels = data.Category;`
	
	For an example showing how to process this data for deep learning, see Classify Text Data Using Deep Learning.

Video Data Sets

Data	Description

HMDB: a large human motion database
(Representative example)
	The HMBD51 data set contains about 2 GB of video data for 7000 clips from 51 classes, such as *drink*, *run*, and *pushup*.
	
	Download and extract the HMBD51 data set from HMDB: a large human motion database. The data set is about 2 GB. Depending on your internet connection, the download process can take some time.
	
	After you extract the RAR files, get the file names and the labels of the videos by using the helper function `hmdb51Files`, which used in the example Classify Videos Using Deep Learning. Set `dataFolder` to the location of the data.
	`oldpath = addpath(fullfile(matlabroot,'examples','nnet','main'));`
	`dataFolder = fullfile(tempdir,"hmdb51_org");`
	`[files,labels] = hmdb51Files(dataFolder);`
	
	For an example showing how to process this data for deep learning, see Classify Videos Using Deep Learning.
	
	To restore the path, use the `path` function.
	`path(oldpath);`



patients. To obtain each recording, the examiners placed two electrodes on different locations on a patient's chest, resulting in a two-channel signal. The database provides signal region labels generated by an automated expert system.

Download the PhysioNet ECG Segmentation data set from the https://github.com/mathworks/physionet_ECG_segmentation by downloading the ZIP file `QT_Database-master.zip`. The data set is about 72 MB. Depending on your internet connection, the download process can take some time. Set `downloadFolder` to the location of the data.

```
downloadFolder = tempdir;

url = "https://github.com/mathworks/physionet_ECG_segmentation/raw/master/QT_Database-master.zip";
filename = fullfile(downloadFolder,"QT_Database-master.zip");

dataFolder = fullfile(downloadFolder,"QT_Database-master");

if ~exist(dataFolder,"dir")
    fprintf("Downloading Physionet ECG Segmentation data set (72 MB)... ")
    websave(filename,url);
    unzip(filename,downloadFolder);
    fprintf("Done.\n")
end
```

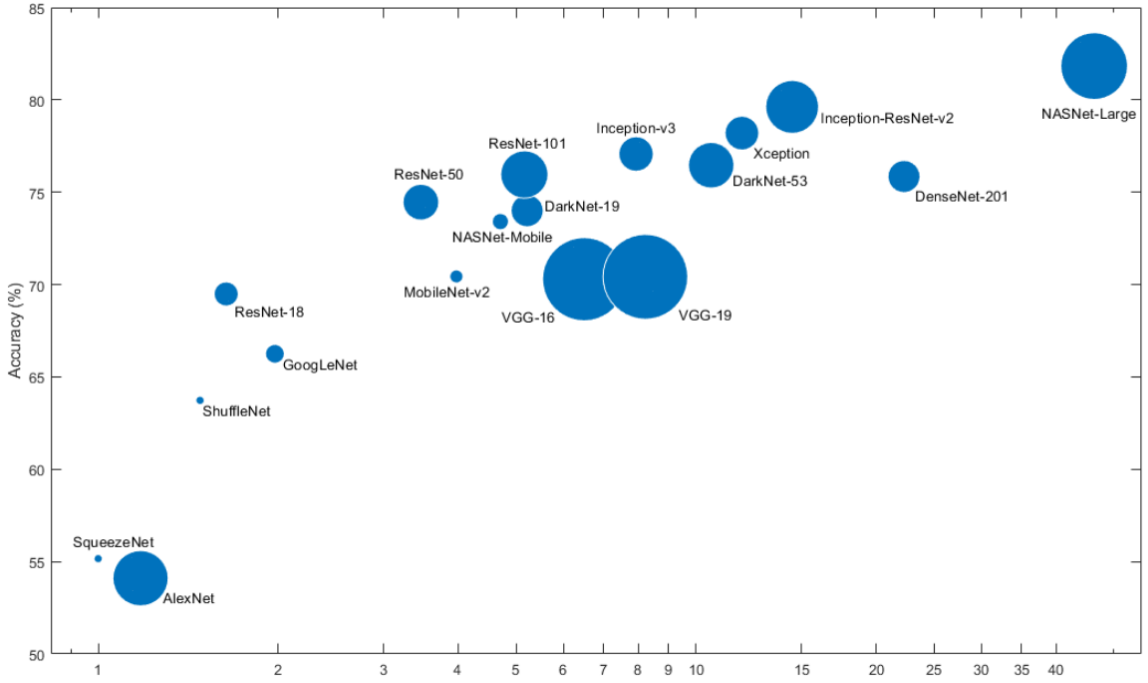
Unzipping creates the folder `QT_Database-master` in your temporary directory. This folder contains the text file `README.md` and the following files:

- `QTData.mat`
- `Modified_physionet_data.txt`
- `License.txt`

Doc: Pretrained Deep Neural Networks

Doc Link

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



Network	Depth	Size	Parameters (Millions)	Image Input Size
squeezenet	18	4.6 MB	1.24	227-by-227
googlenet	22	27 MB	7.0	224-by-224
inceptionv3	48	89 MB	23.9	299-by-299
densenet201	201	77 MB	20.0	224-by-224
mobilenetv2	53	13 MB	3.5	224-by-224
resnet18	18	44 MB	11.7	224-by-224
resnet50	50	96 MB	25.6	224-by-224
resnet101	101	167 MB	44.6	224-by-224
xception	71	85 MB	22.9	299-by-299
inceptionresnetv2	164	209 MB	55.9	299-by-299
shufflenet	50	6.3 MB	1.4	224-by-224
nasnetmobile	*	20 MB	5.3	224-by-224
nasnetlarge	*	360 MB	88.9	331-by-331
darknet19	19	72.5 MB	21.0	256-by-256
darknet53	53	145 MB	41.0	256-by-256
alexnet	8	227 MB	61.0	227-by-227
vgg16	16	515 MB	138	224-by-224
vgg19	19	535 MB	144	224-by-224

You can now load untrained versions of the pretrained networks in Deep Learning Toolbox™. To load an untrained version of a pretrained network as a layer graph, use the corresponding pretrained network function and set the `'Weights'` option to `'none'`. Loading an untrained version of pretrained networks

Computer Vision Tasks in Deep Learning

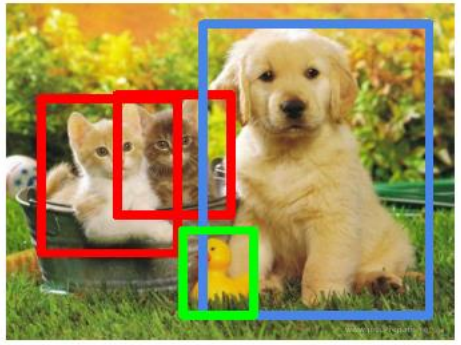
Classification

CAT

Classification + Localization

CAT

Object Detection

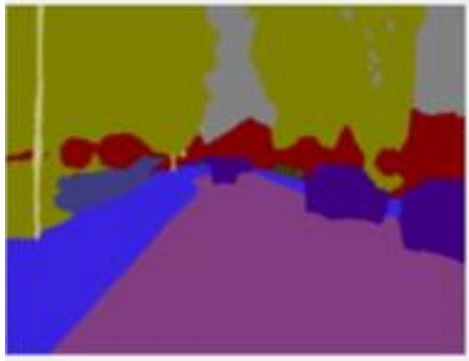


CAT, DOG, DUCK

Instance Segmentation

CAT, DOG, DUCK

Semantic Segmentation



Single object

Multiple object

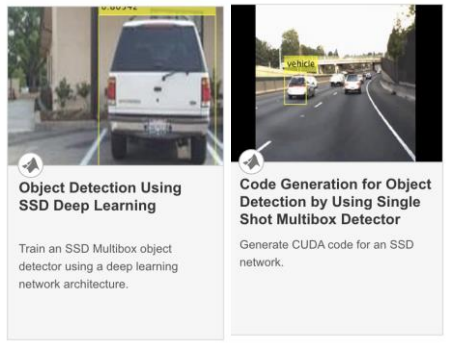
No object , just pixel

New!!

DarkNet19
DarkNet53

New!!

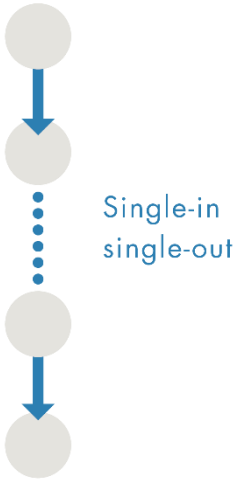
YOLO v3
SSD



Src: http://blog.csdn.net/u011974639

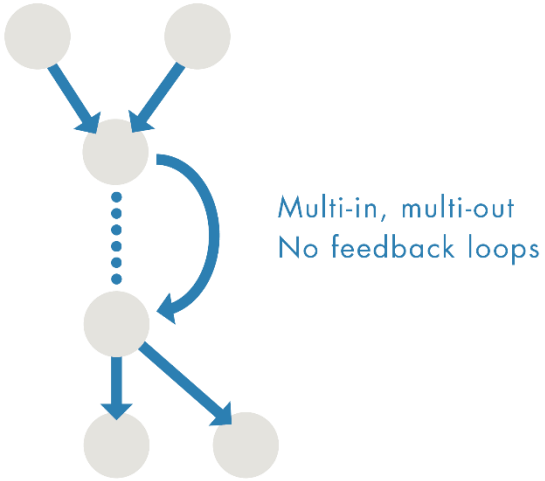
Select Network - Other deep neural networks

SeriesNetwork



Networks: MNIST
Alexnet
VGG(16,19)
Inception-v3
Lane detection
Pedestrian detection

DAGNetwork

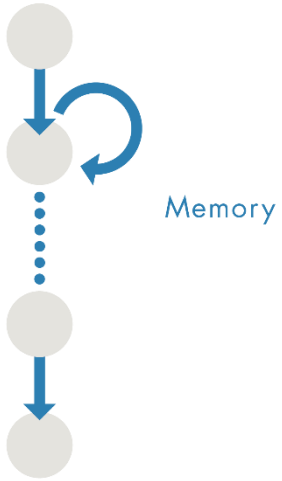


Networks: R-CNN (fast, faster)
GoogLeNet
ResNet(18,50,101)
Inception-ResNet-v2
Densenet201
Squeezenet
SegNet
FCN
DeconvNet

Object detection

Semantic segmentation

Recurrent Network



Networks: LSTM
biLSTM

Importer/Converter

Caffe MODELS

KERAS IMPORTER
Importer for TensorFlow-Keras Models

ONNX Converter
Export to ONNX model format

Caffe Importer
Tensor-Flow-Keras
ONNX Converter

Transfer Learning with Pretrained Models

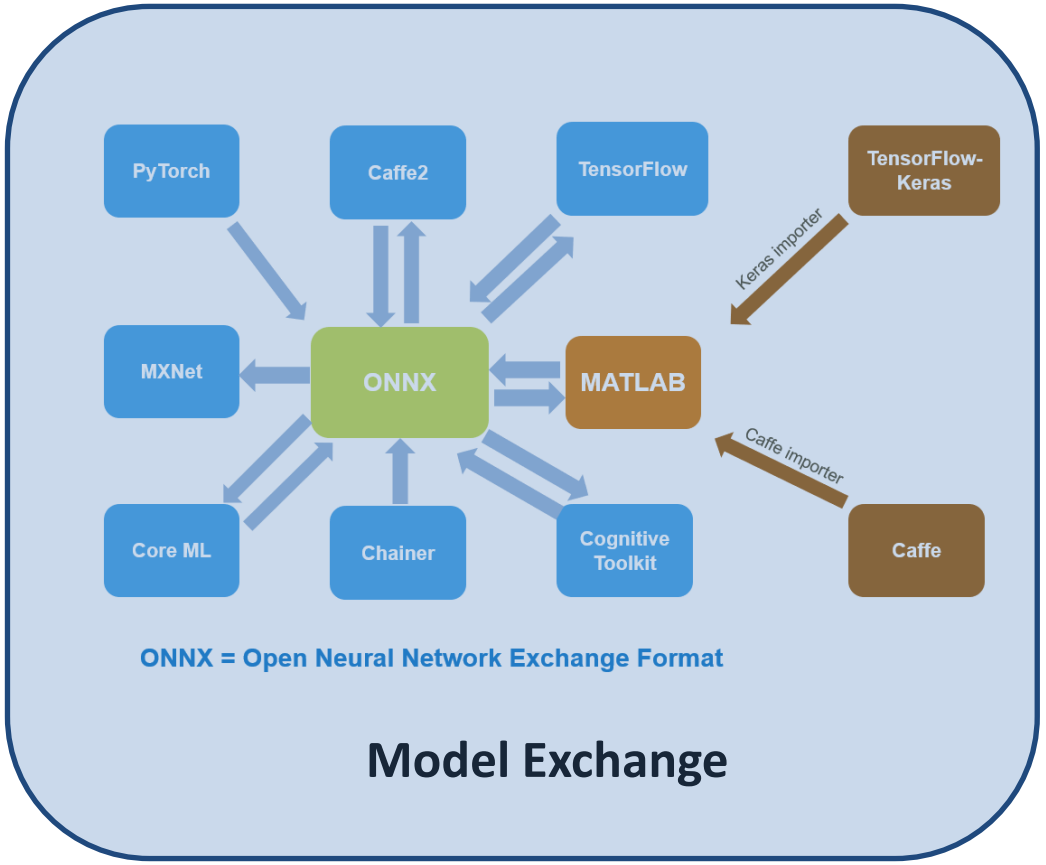
- Inception-v3
- MobileNet-v2
- VGG-16
- Inception-ResNet-v2
- ResNet-18/50/101
- GoogLeNet
- DenseNet-201
- NASNet
- SqueezeNet
- AlexNet
- Places365-GoogLeNet
- Xception

Import & Export Models Between Frameworks

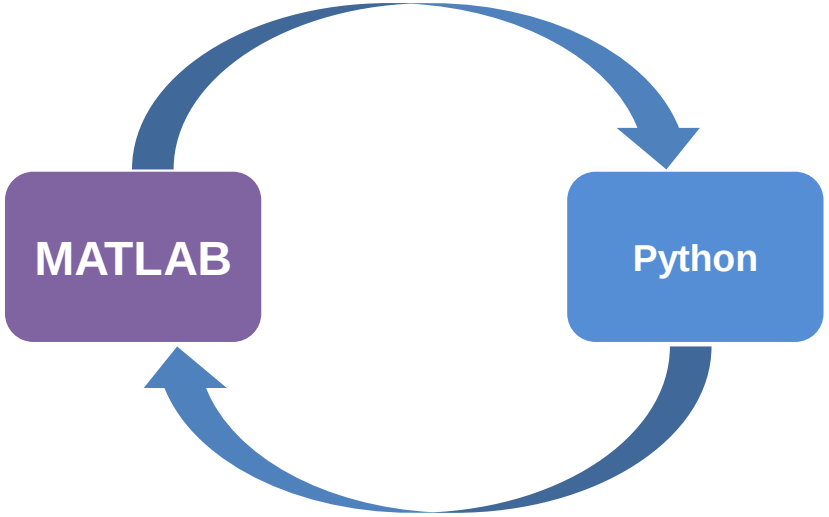
- Keras-Tensorflow Importer
- Caffe Model Importer
- ONNX Model Converter

More comprehensive list here: https://www.mathworks.com/help/deeplearning/ug/pretrained-convolutional-neural-networks.html

Two Ways to Work with TensorFlow and PyTorch

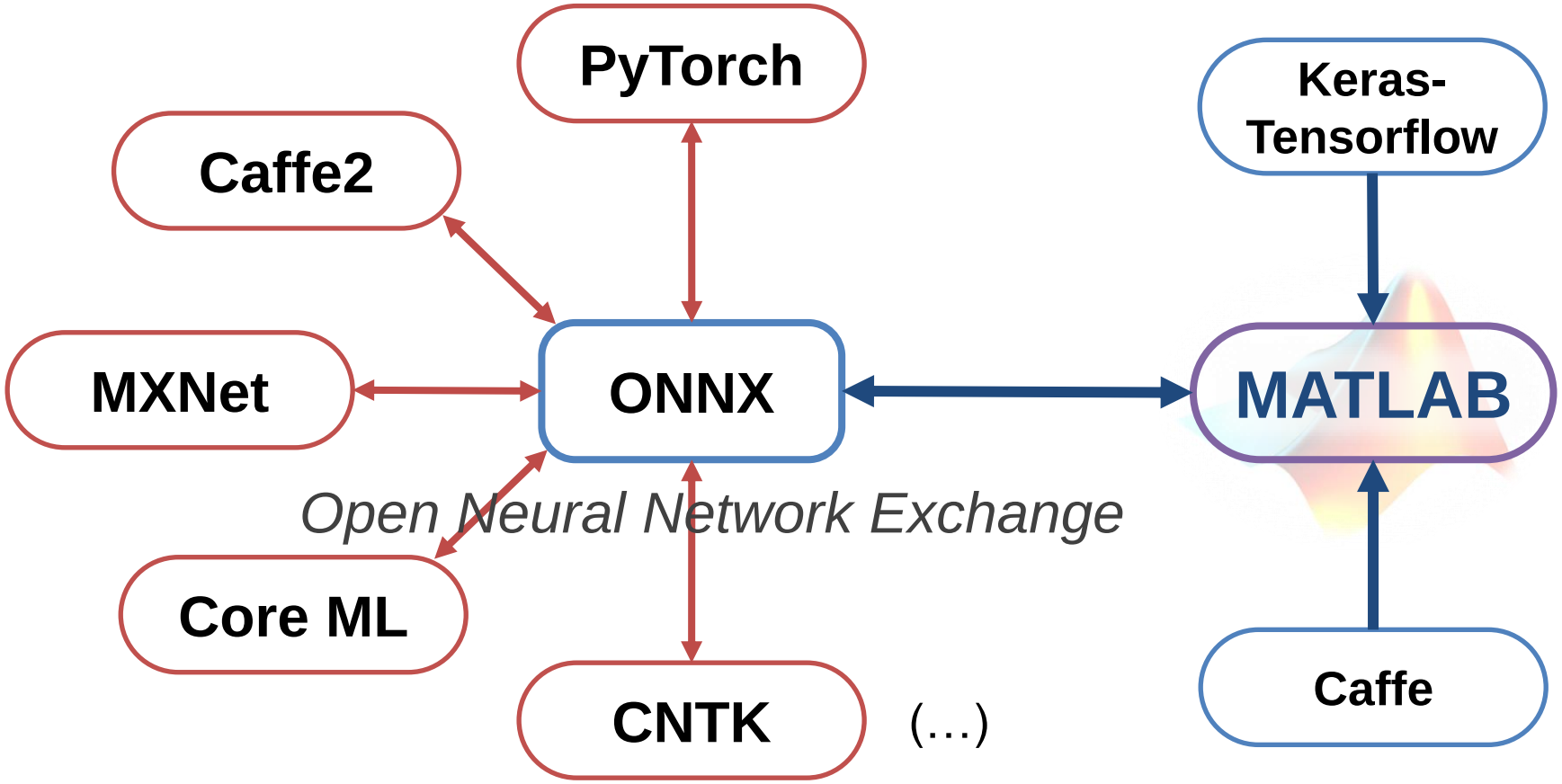


Model Exchange

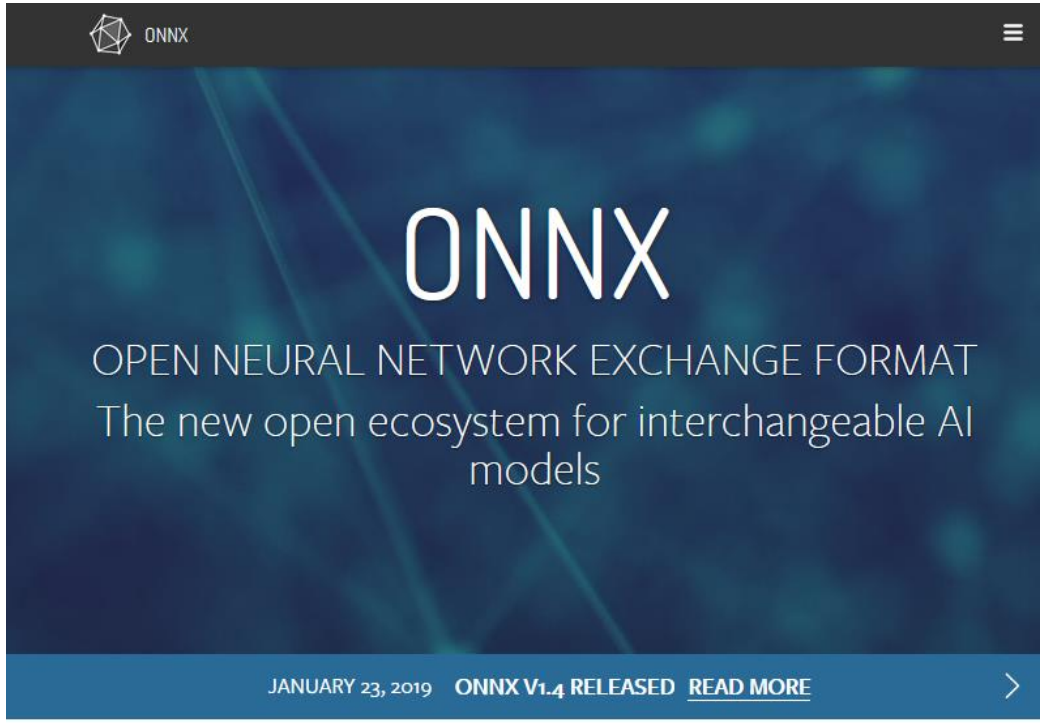


Co-execution (Python and C++)

Model Exchange with MATLAB



ONNX – Industry Standard for Model Exchange



aws Facebook Open Source Microsoft

What is ONNX?

ONNX is a open format to represent deep learning models. With ONNX, AI developers can more easily move models between state-of-the-art tools and choose the combination that is best for them. ONNX is developed and supported by a community of partners.

Frameworks

Caffe2 Chainer Cognitive Toolkit mxnet PyTorch PaddlePaddle MATLAB sas

Converters

K ML learn dmlc XGBoost LibSVM

Runtimes

NVIDIA Qualcomm BITMAIN Tencent vespa Windows

Source: https://onnx.ai/

Outline

Deep learning APP

Deep learning Model

Deep learning Import/Deploy

Signal Application

Audio Application

Text Application

Image Application

Code generation supports a wide range of network architectures

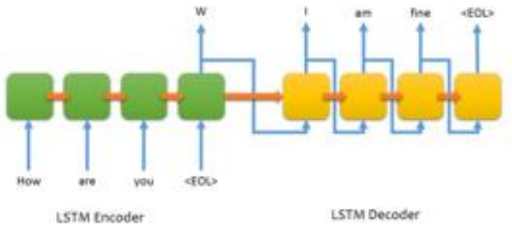
R2019b

CNNs, LSTMs (GPU)
YOLOv2 object detector
Keras, ONNX imported networks

Image Classification, Semantic Segmentation, Object Detection

R2020a

LSTM (ARM target),
Stateful and Bidirectional LSTM
Multi-out networks
SSD

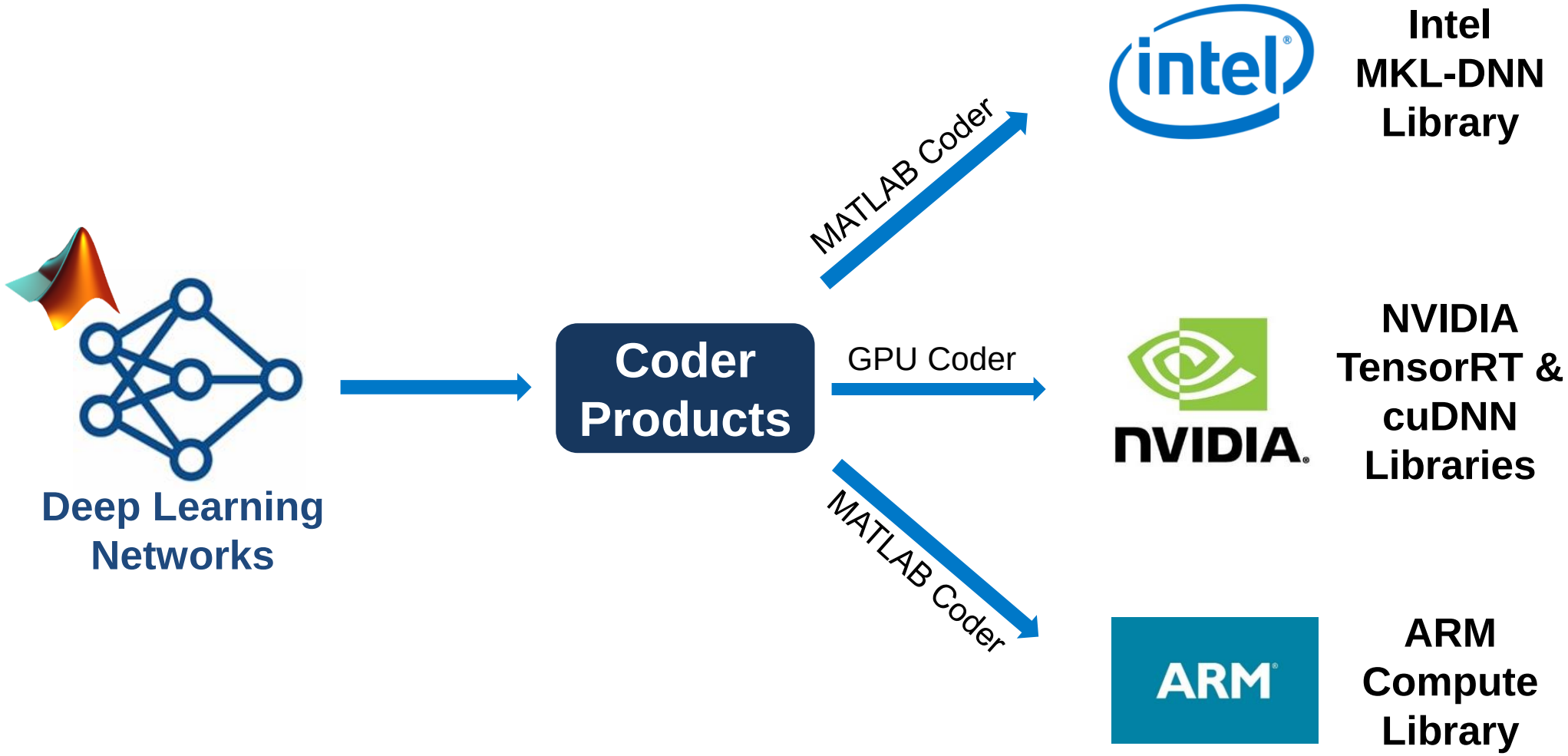


LSTM for time series , text analytics, speech

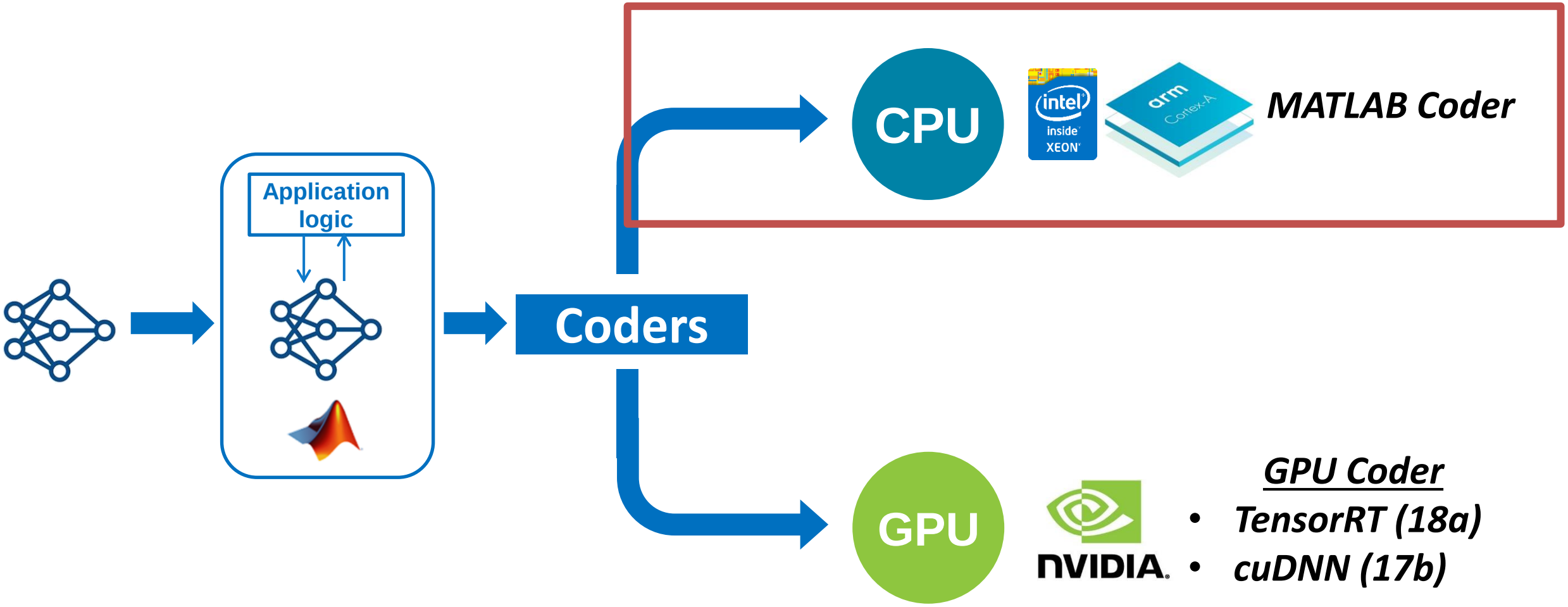
[MATLAB Coder Support Link]

[GPU Coder Support Link]

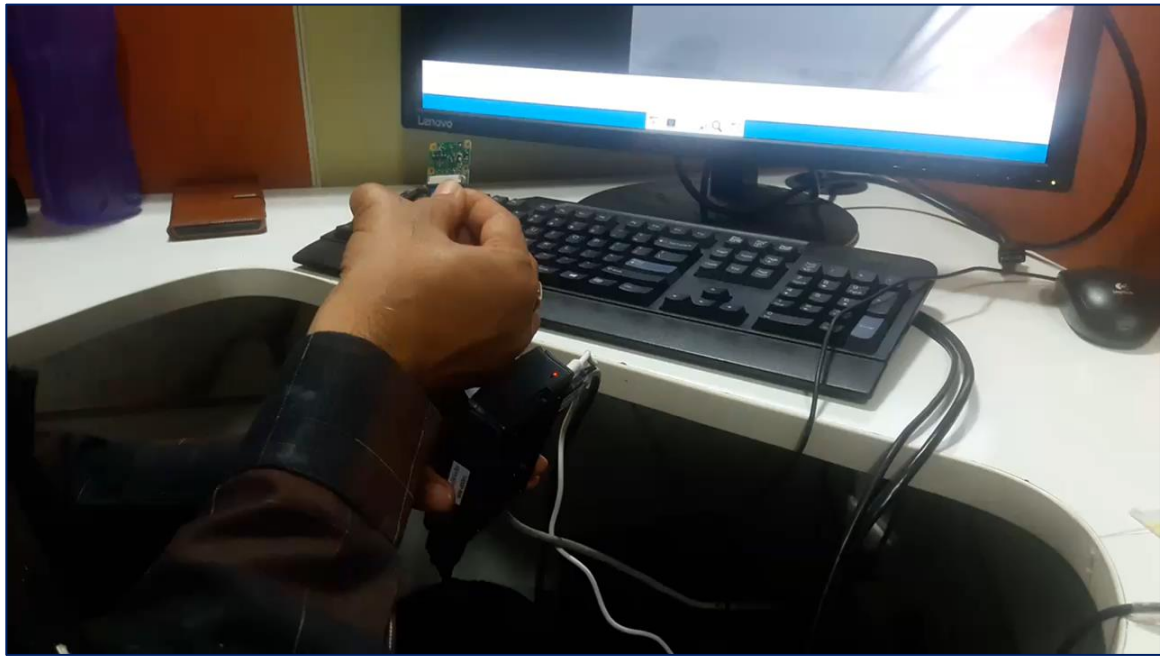
Deploying Deep Learning Models for Inference



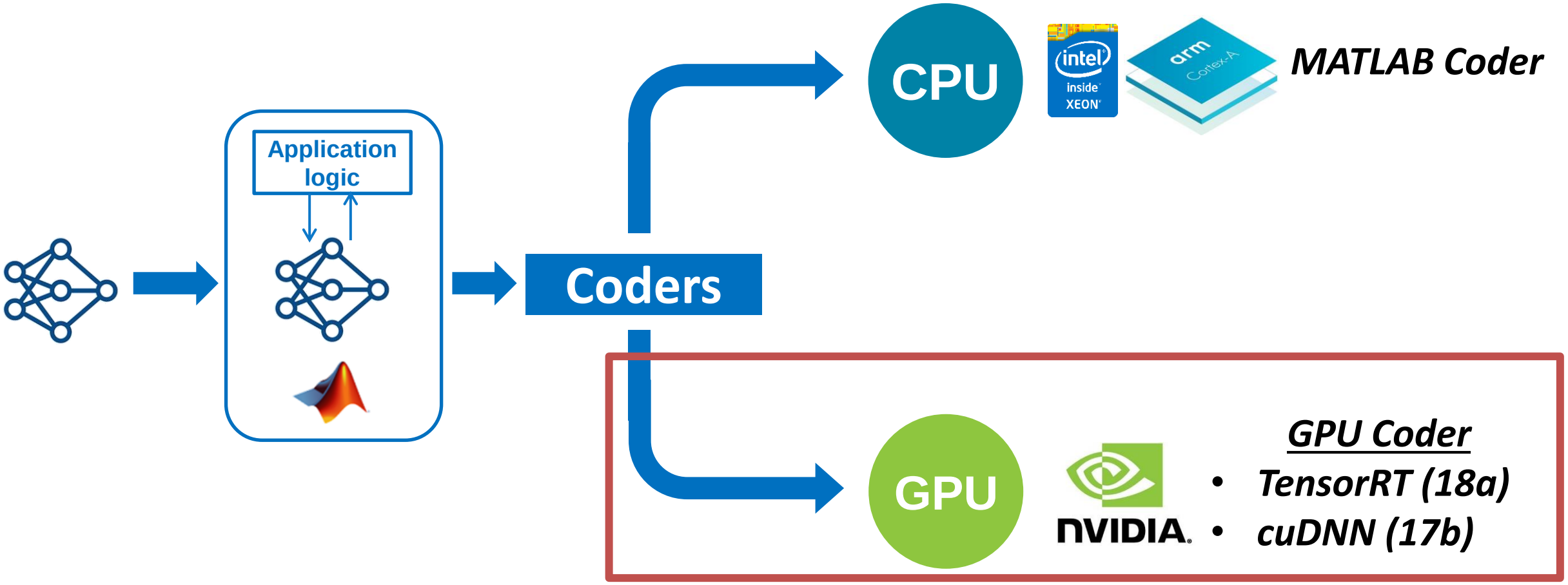
Current Code Generation Support



MATLAB Coder



Current Code Generation Support



GPU Coder
Algorithm Design to Embedded Deployment Workflow

MATLAB algorithm
(functional reference)

GPU Coder

Build type

Call CUDA from MATLAB directly

.mex

Call CUDA from (C++) hand-coded main()

.lib

Call CUDA from (C++) hand-coded main().

Cross-compiled .lib

Desktop GPU

Desktop GPU

Jetson Platform

C++

C++

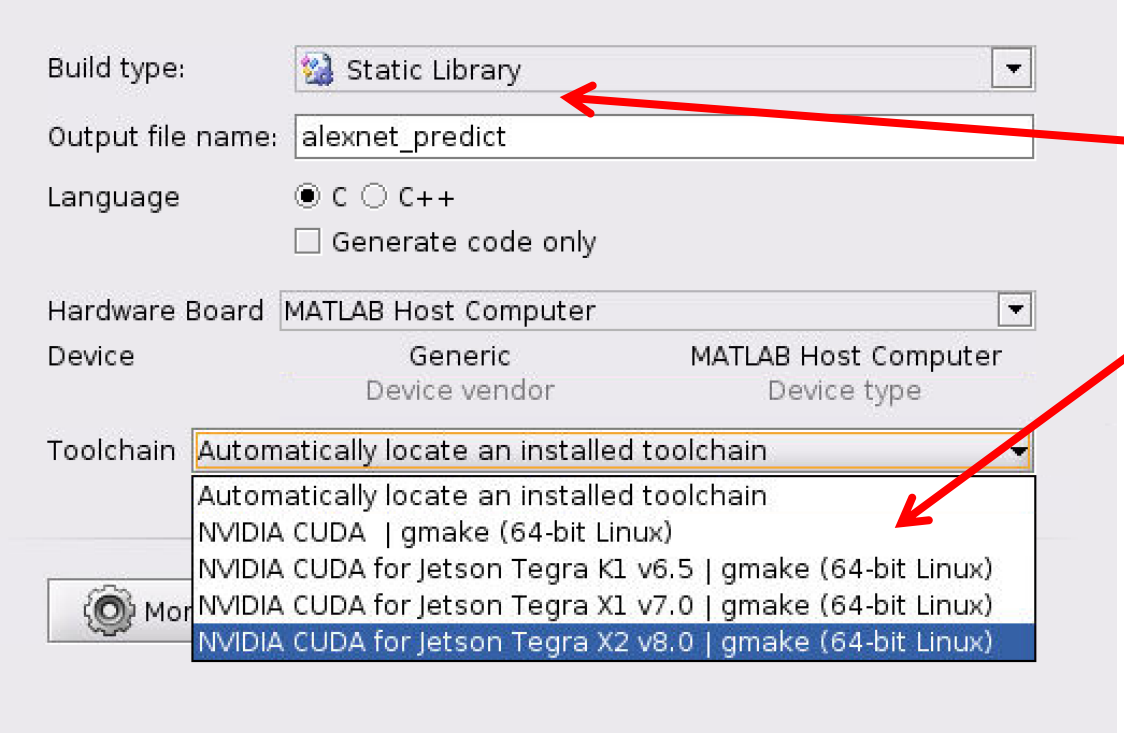
1 Functional test

2 Deployment unit-test

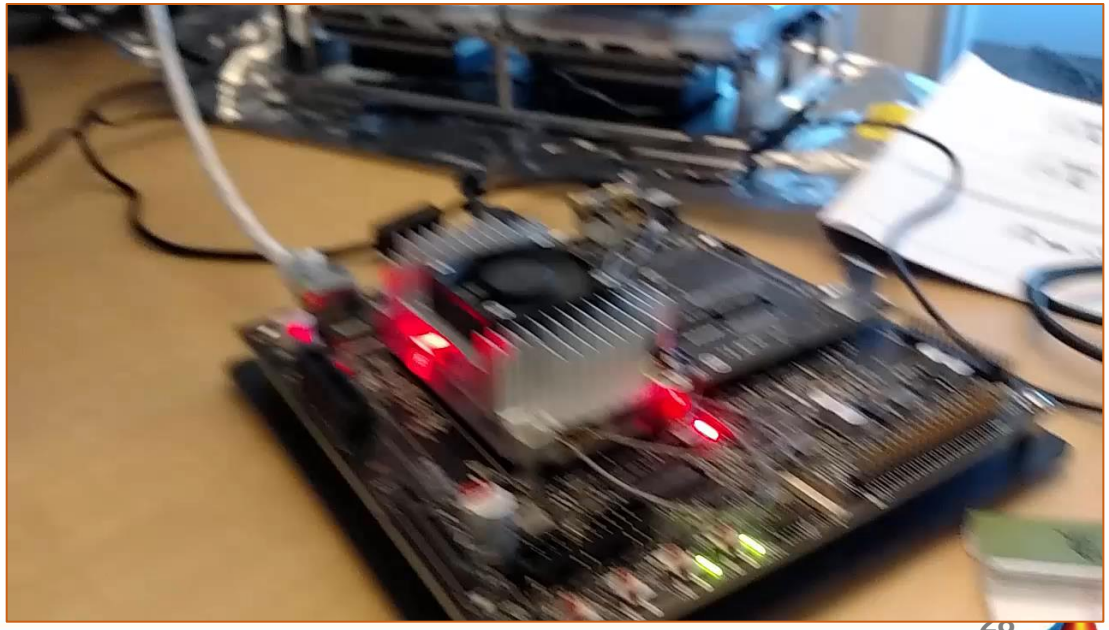
3 Deployment integration-test

4 Real-time test

Deployment to Tegra: Cross-Compiled with ‘lib’

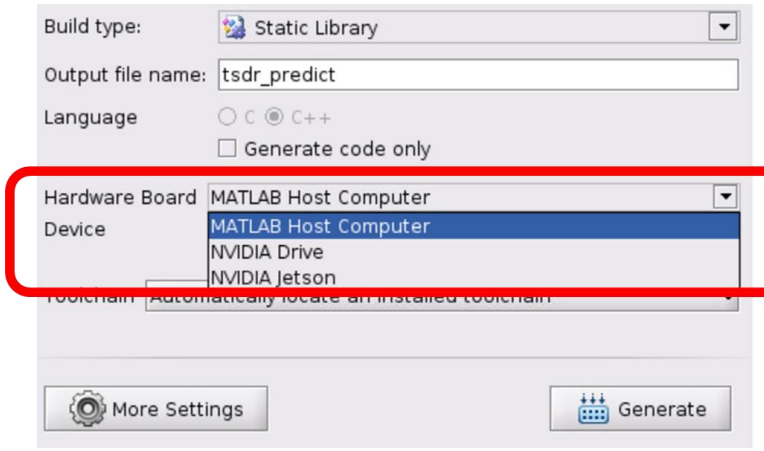


1. Change build-type to ‘lib’
2. Select cross-compile toolchain



Out of Box Targeting for Popular GPU Boards

- **Pain**
  - Time consuming to setup and connect to hardware prototyping boards during system development
- **Solution / Differentiator**
  - GPU Coder Hardware support packages for
    - NVIDIA Jetson
    - NVIDIA Drive
  - Removes manual setup required to setup and target



Simple out-of-box targeting to NVIDIA boards:

Jetson

Drive platform

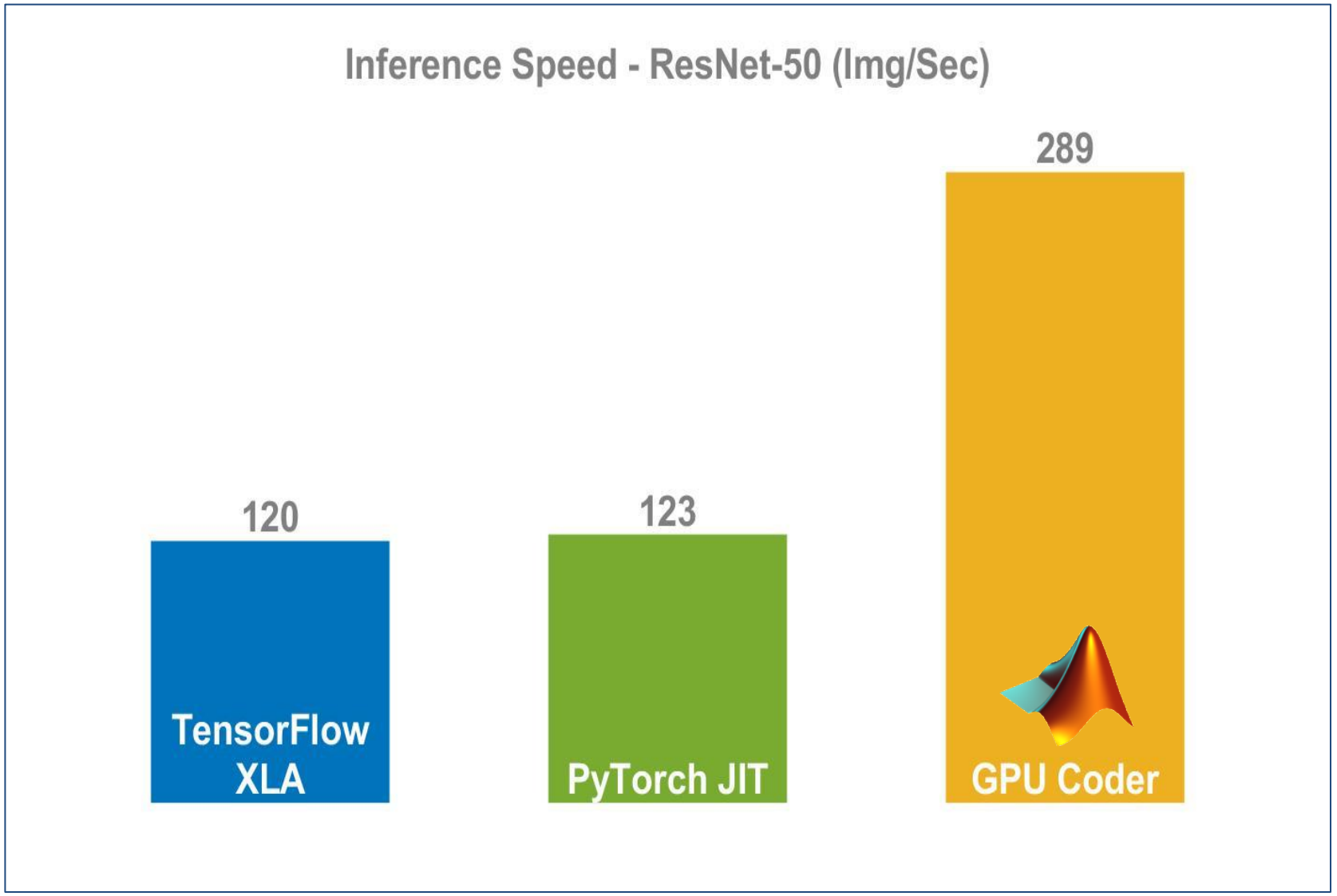
Jetson Tx1 - LogoNet Demo

Semantic Segmentation Speedup

Running in MATLAB

Generated Code from GPU Coder

Deep Learning inference with GPU Coder is best-in-class



Intel® Xeon® CPU 3.6 GHz - Titan V - NVIDIA libraries: CUDA10.0/1 - cuDNN 7.5.0 - Frameworks: TensorFlow 1.13.1, MXNet 1.4.1 PyTorch 1.1

Outline

Deep learning APP

Deep learning Model

Deep learning Import/Deploy

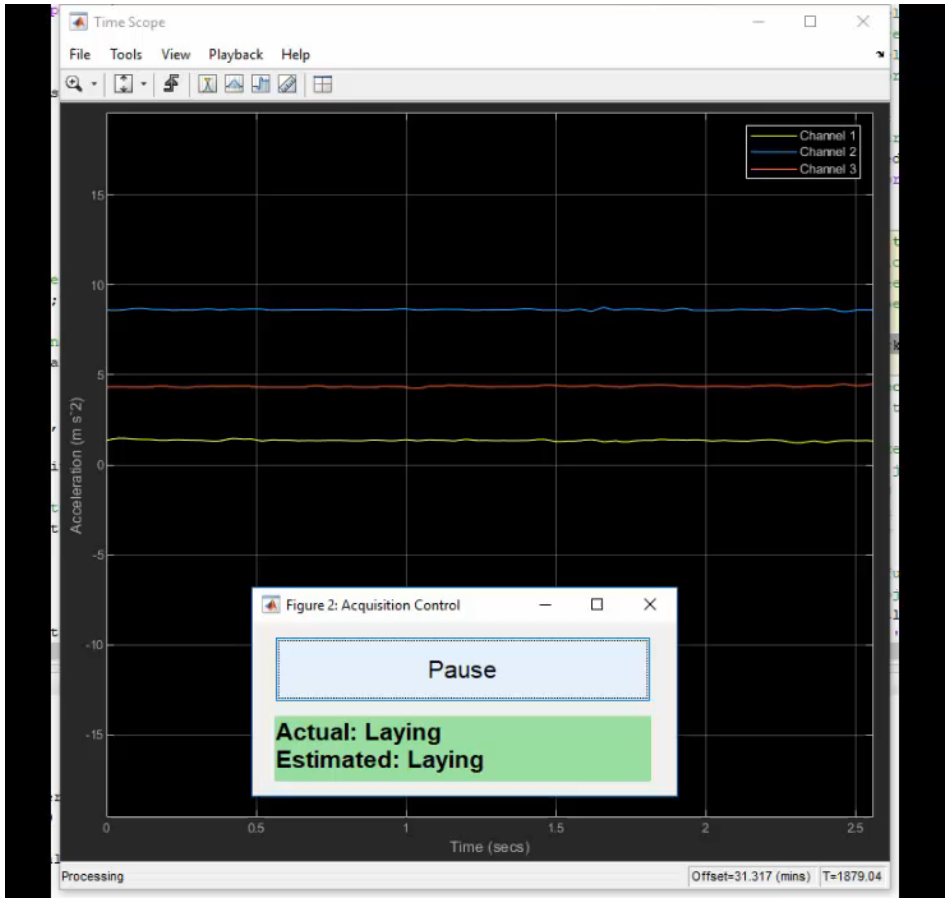
Signal Application

Audio Application

Text Application

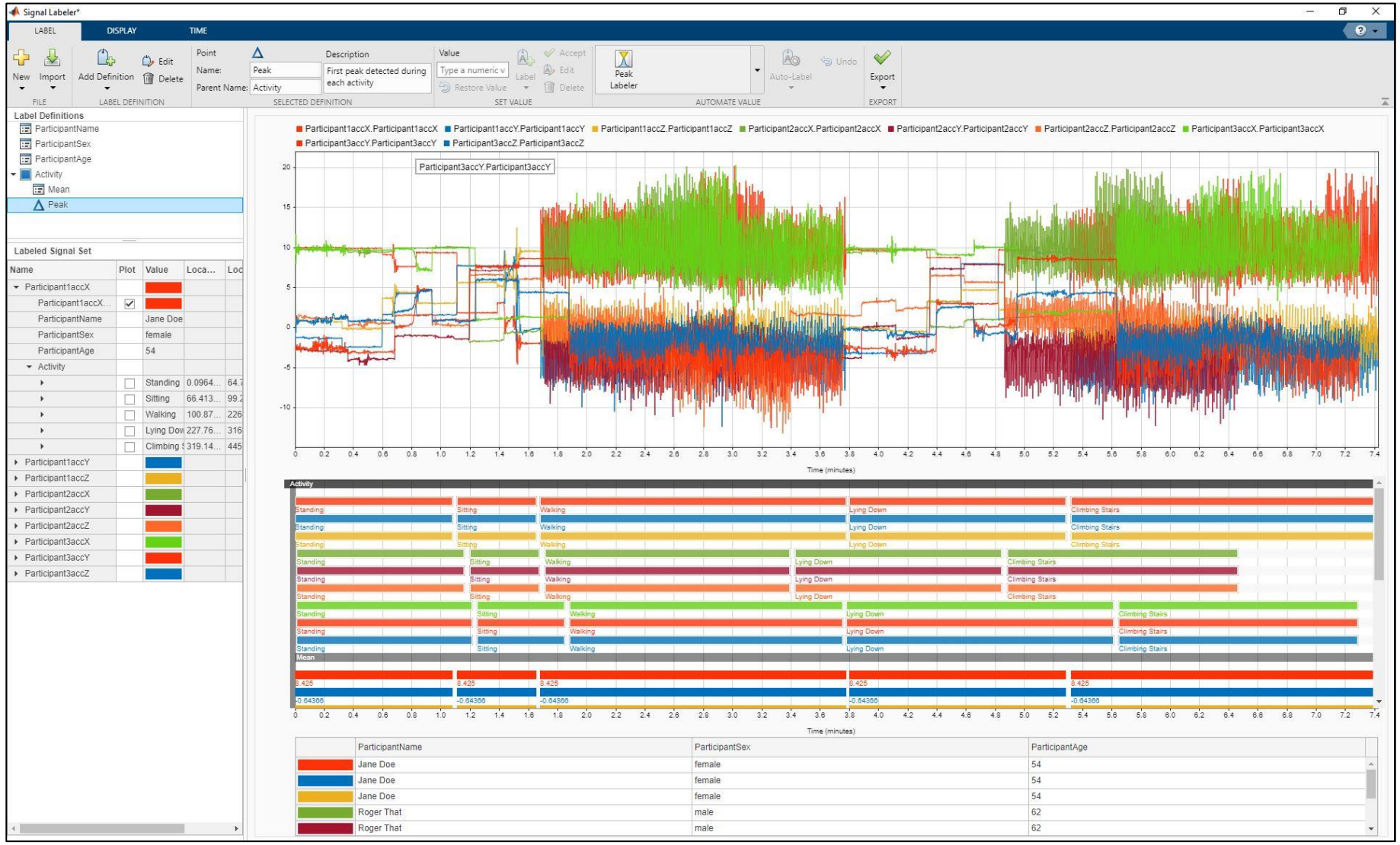
Image Application

Application: Analyzing signal data using deep learning



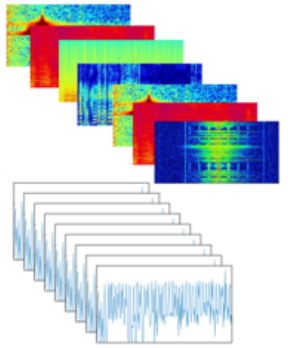
Signal Classification using LSTMs

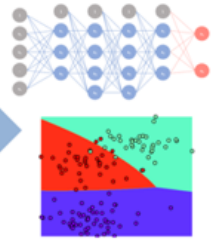
Signal Labeler App



Data Management: Connect Signal Sets with Learners

Split / Normalize / Transform





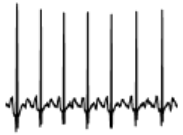
Labeler App
R2019a

Labeled signal dataset
R2018b

signalDatastore
R2020a

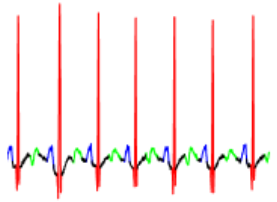
Transformed data and labels

Train model



ECG

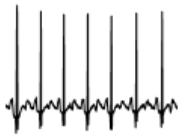
LSTM



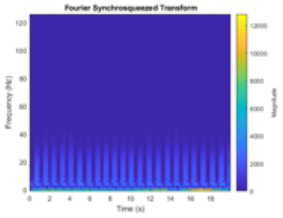
Labeled ECG

Accuracy

72 %

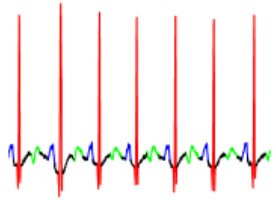


ECG



FSST

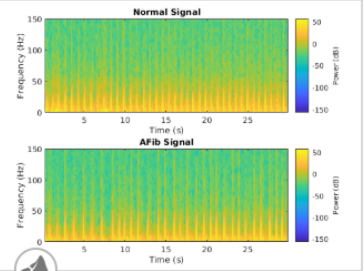
LSTM



Labeled ECG

94 %

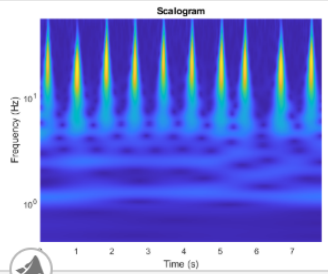
Signal Processing Using Deep Learning



Classify ECG Signals Using Long Short-Term Memory Networks

Classify heartbeat electrocardiogram (ECG) data from the PhysioNet 2017 Challenge using deep learning and signal processing. In particular,

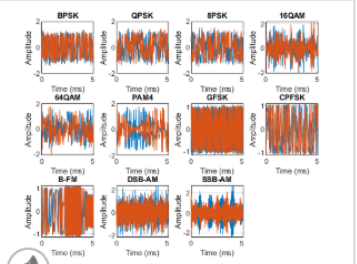
Open Live Script



Classify Time Series Using Wavelet Analysis and Deep Learning

Classify human electrocardiogram (ECG) signals using the continuous wavelet transform (CWT) and a deep convolutional neural network

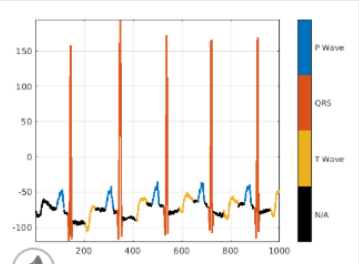
Open Live Script



Modulation Classification with Deep Learning

Use a convolutional neural network (CNN) for modulation classification.

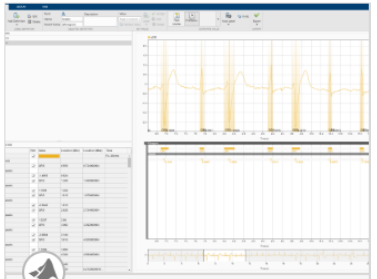
Open Live Script



Waveform Segmentation Using Deep Learning

Segment human electrocardiogram (ECG) signals using recurrent deep learning networks and time-frequency analysis.

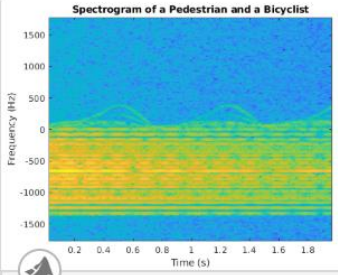
Open Live Script



Label QRS Complexes and R Peaks of ECG Signals Using Deep Network

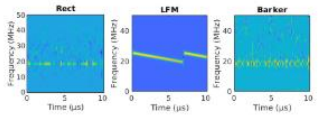
Use custom autolabeling functions in Signal Labeler to label QRS complexes and R peaks of electrocardiogram (ECG) signals.

Open Live Script



Pedestrian and Bicyclist Classification Using Deep Learning

Classify pedestrians and bicyclists based on their micro-Doppler characteristics using a deep learning network and time-frequency



Radar Waveform Classification Using Deep Learning

Classify radar waveform types of generated synthetic data using the Wigner-Ville distribution (WVD) and a deep convolutional neural network

Outline

Deep learning APP

Deep learning Model

Deep learning Import/Deploy

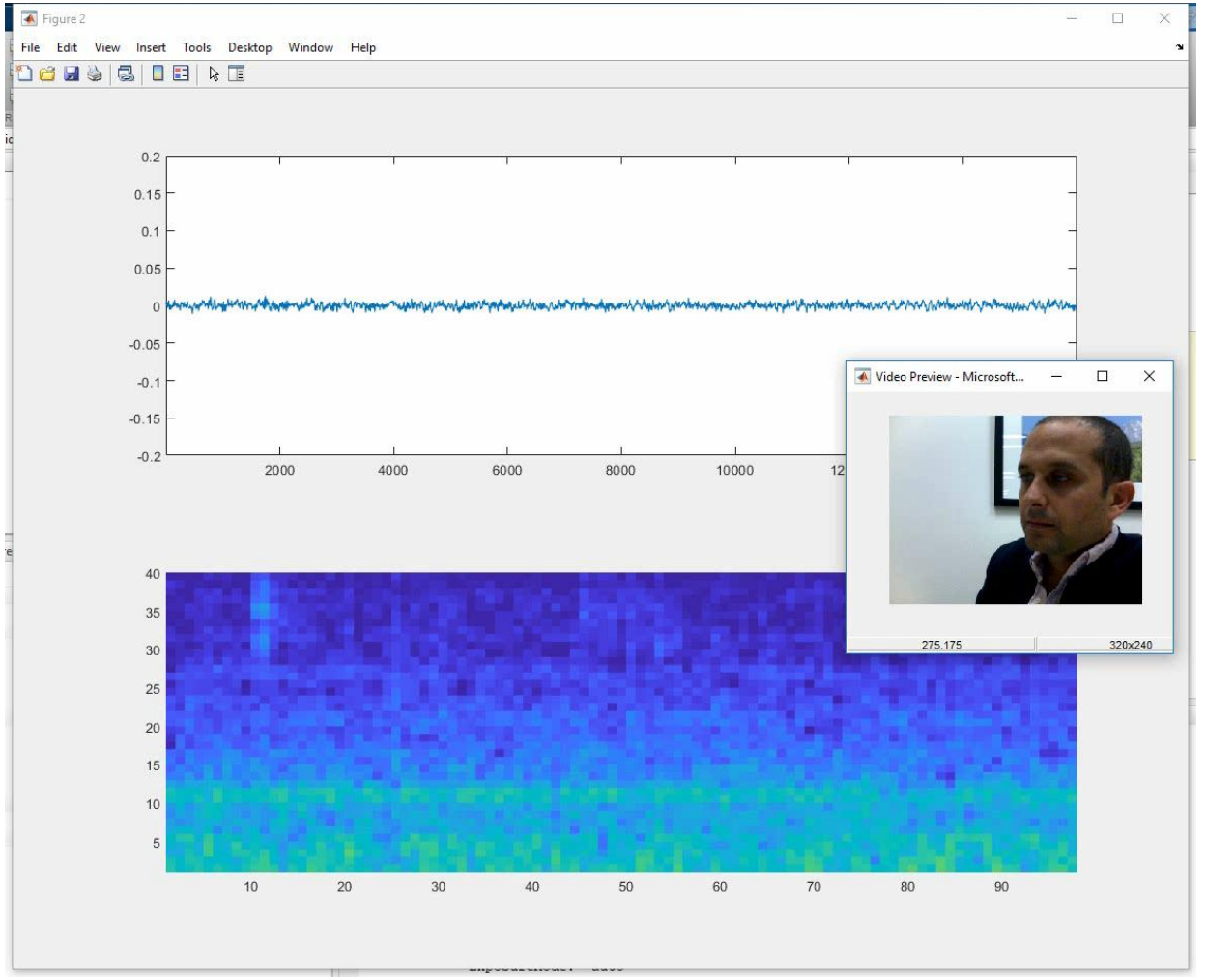
Signal Application

Audio Application

Text Application

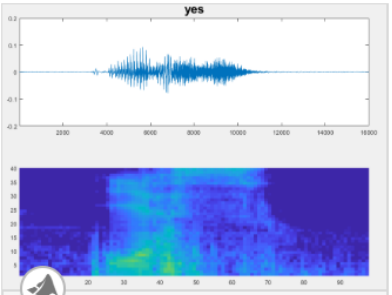
Image Application

Analyzing audio data using deep learning



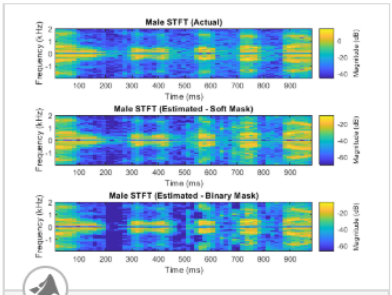
Speech Recognition using CNNs

Audio Processing Using Deep Learning



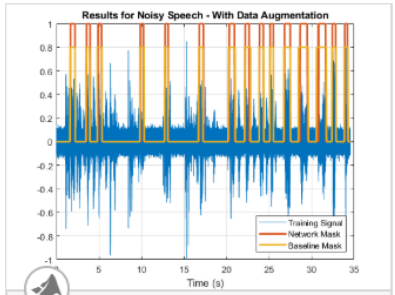
Speech Command Recognition Using Deep Learning

Train a deep learning model that detects the presence of speech commands in audio. The example uses the Speech Commands



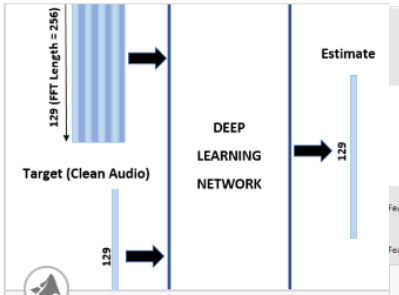
Cocktail Party Source Separation Using Deep Learning Networks

Isolate a speech signal using a deep learning network.



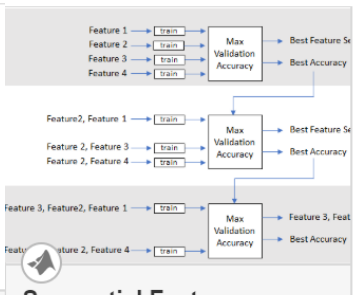
Keyword Spotting in Noise Using MFCC and LSTM Networks

Identify a keyword in noisy speech using a deep learning network. In particular, the example uses a Bidirectional Long Short-Term



Denoise Speech Using Deep Learning Networks

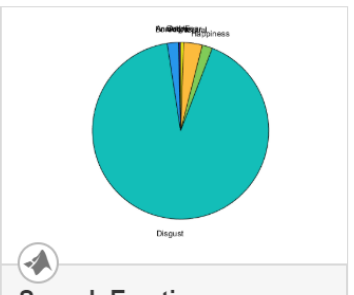
Denoise speech signals using deep learning networks. The example compares two types of networks applied to the same task: fully



Sequential Feature Selection for Audio Features

A typical workflow for feature selection applied to the task of spoken digit recognition.

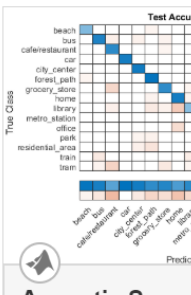
Open Live Script



Speech Emotion Recognition

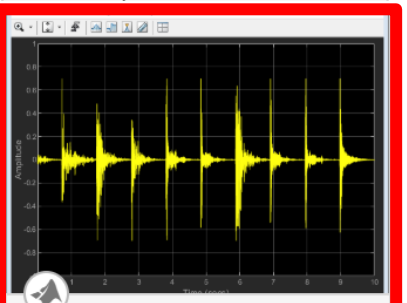
Illustrates a simple speech emotion recognition (SER) system using a BiLSTM network. You begin by downloading the data set and then

Open Live Script



Acoustic Scene Recognition Us Fusion

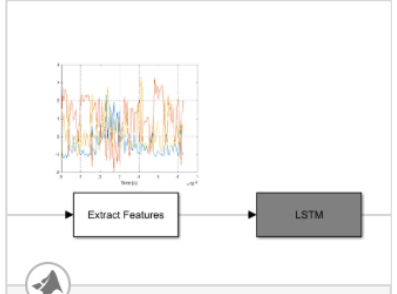
Create a multi-mode system for acoustic recognition. The exa convolutional neural



Train Generative Adversarial Network (GAN) for Sound Synthesis

Train and use a generative adversarial network (GAN) to generate sounds.

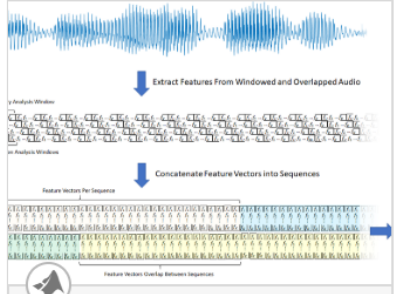
Open Script



Voice Activity Detection in Noise Using Deep Learning

Detect regions of speech in a low signal-to-noise environment using deep learning. The example uses the Speech Commands Dataset to

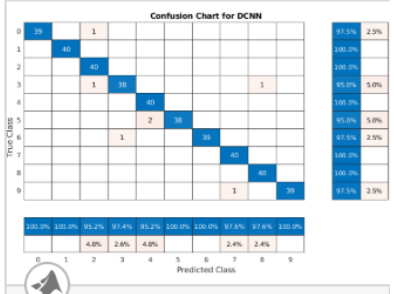
Open Live Script



Classify Gender Using LSTM Networks

Classify the gender of a speaker using deep learning. The example uses a Bidirectional Long Short-Term Memory (BiLSTM) network

Open Live Script



Spoken Digit Recognition with Wavelet Scattering and Deep Learning

Classify spoken digits using both machine and deep learning techniques. In the example, you perform classification using wavelet

Open Live Script

Outline

Deep learning APP

Deep learning Model

Deep learning Import/Deploy

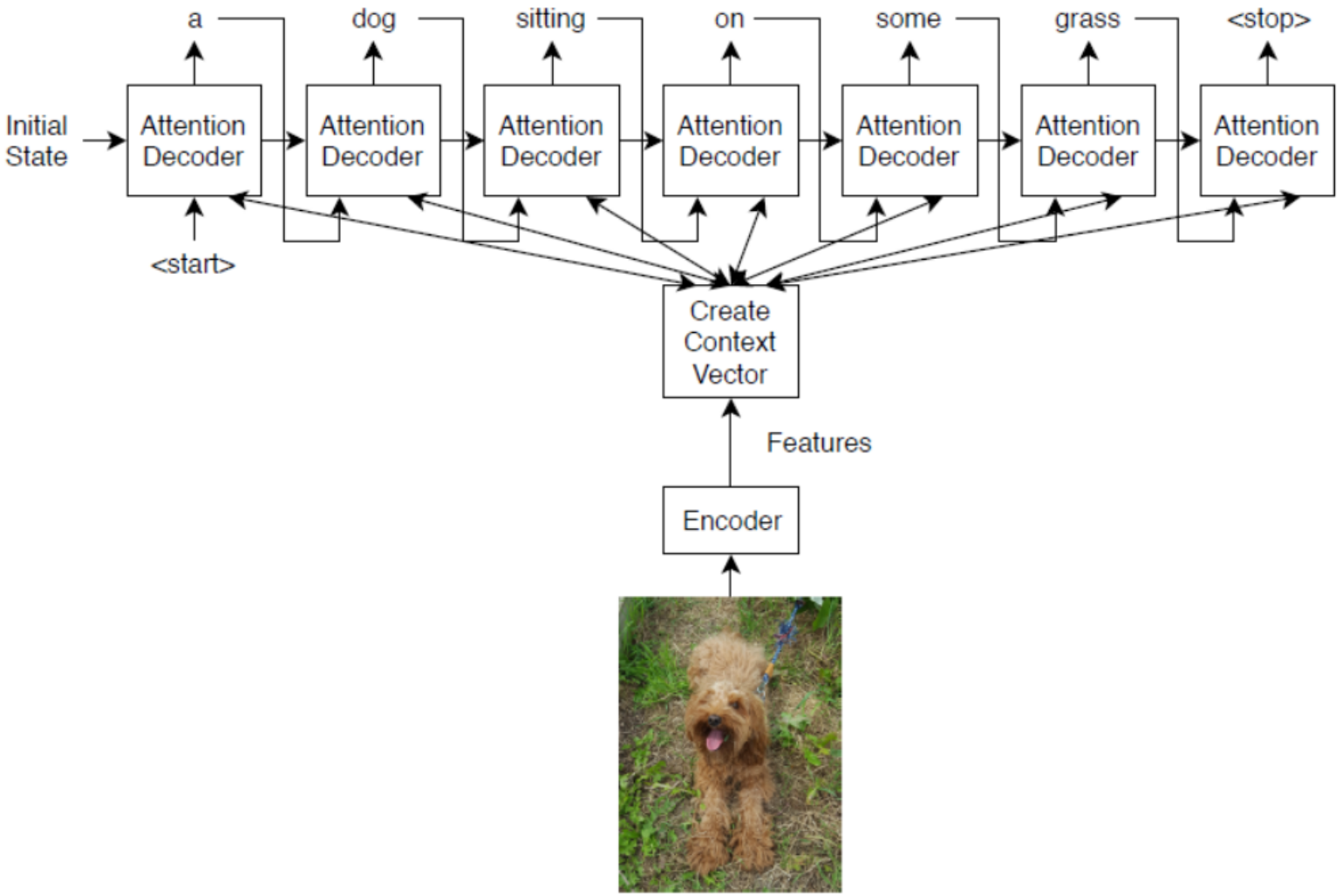
Signal Application

Audio Application

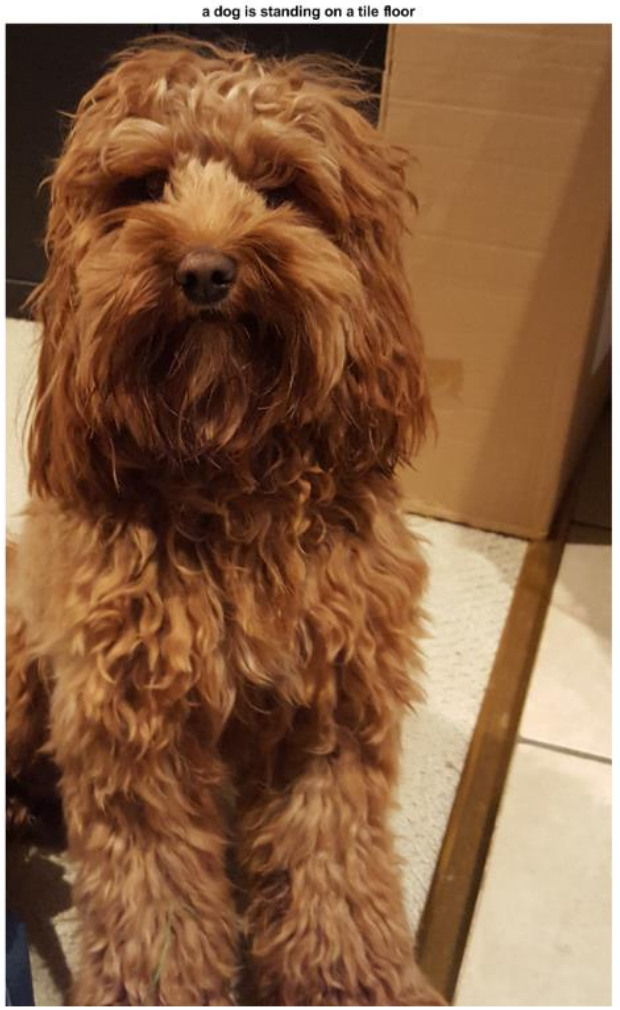
Text Application

Image Application

Image Captioning Using Attention



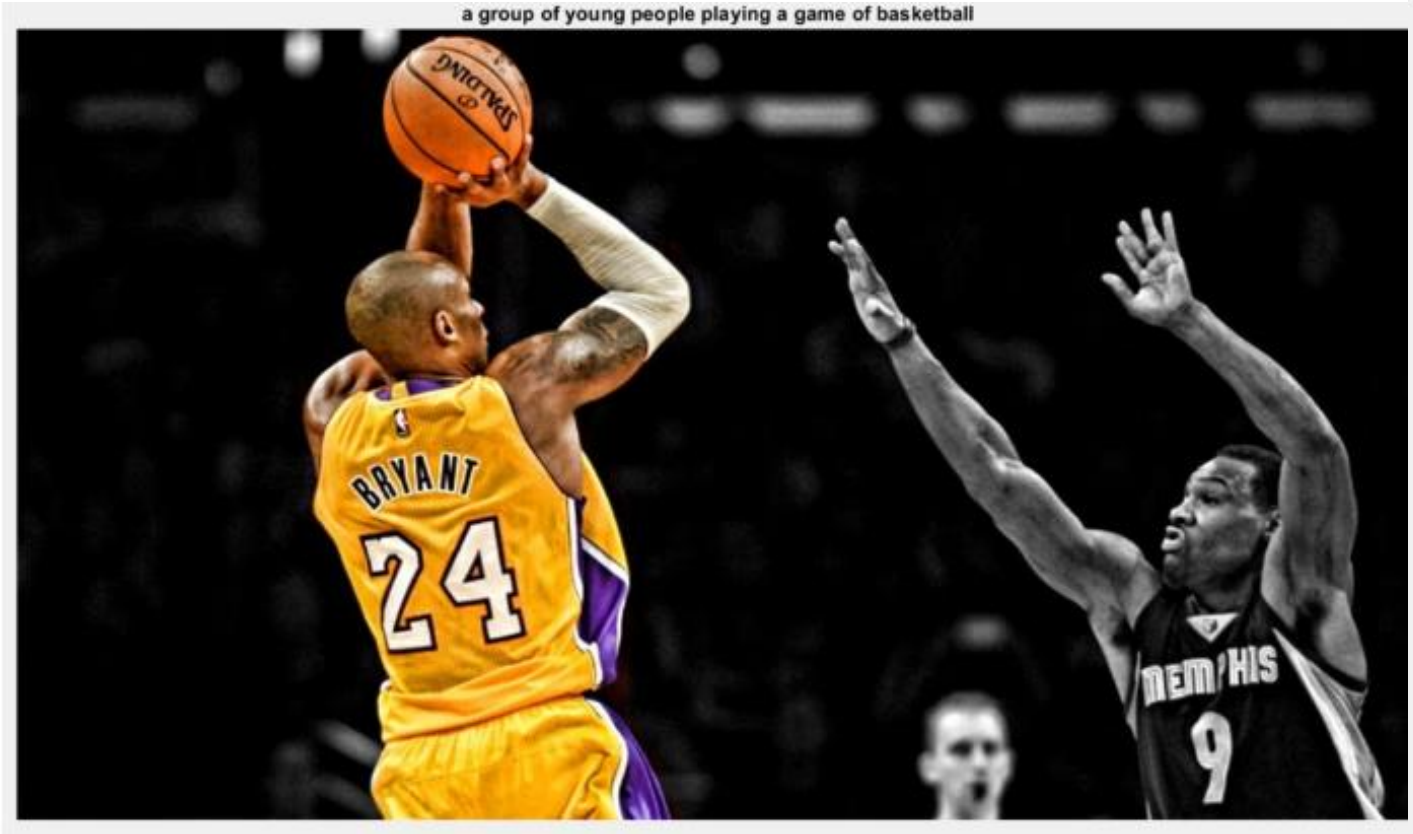
A dog standing on the floor



A baseball player pitching a baseball on a field with a catchers mitt on a field

A group of people sitting around a dinner table Posing for a photo

A group of young people playing a game of basketball



A bunch of train cars lined up a rail guard rail car
And a crew guard at the

a birthday cake shaped like a dump truck

a cat is standing on a book shelf with a bird on the screen and a fish in its

Outline

Deep learning APP

Deep learning Model

Deep learning Import/Deploy

Signal Application

Audio Application

Text Application

Image Application

Video detection and localization using deep learning

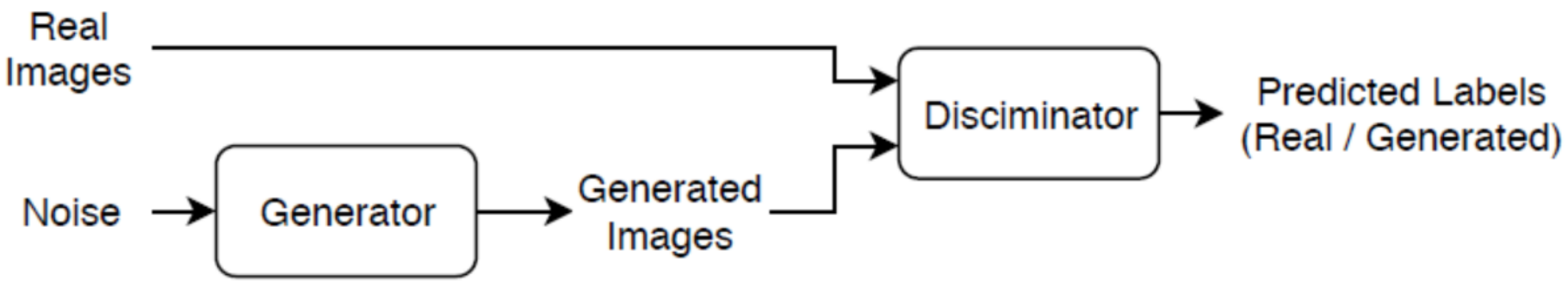
YOLO v2 (You Only Look Once)

Semantic Segmentation using SegNet

GAN & CGAN(Conditional Generative Adversarial Network

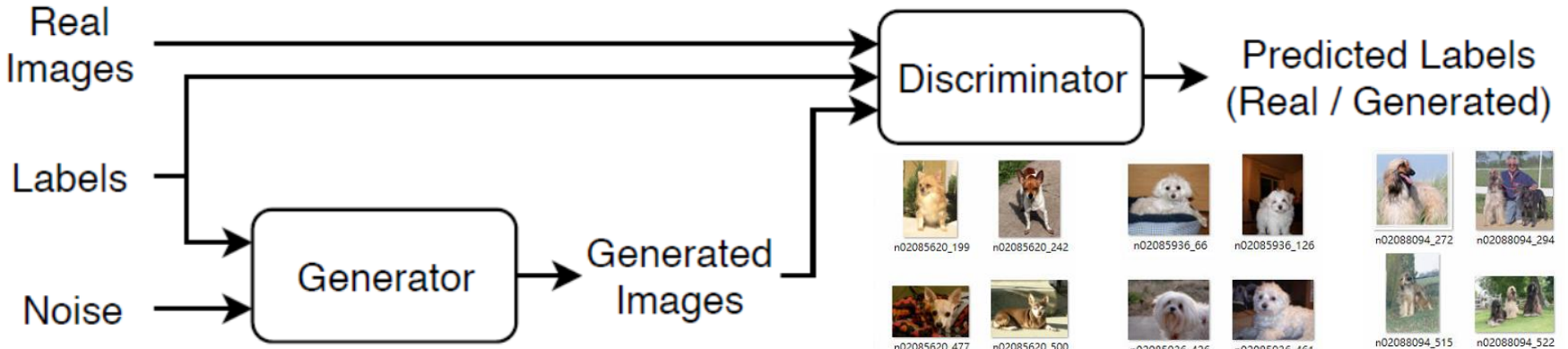
A GAN consists of two networks that train together:

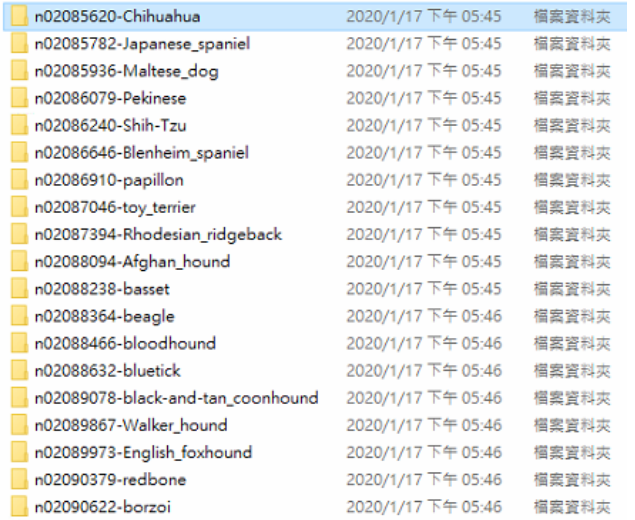
1. Generator — Given a vector of random values as input, this network generates data with the same structure as the training data.
2. Discriminator — Given batches of data containing observations from both the training data, and generated data from the generator, this network attempts to classify the observations as "real" or "generated".



A *conditional* generative adversarial network is a type of GAN that also takes advantage of labels during the training process.

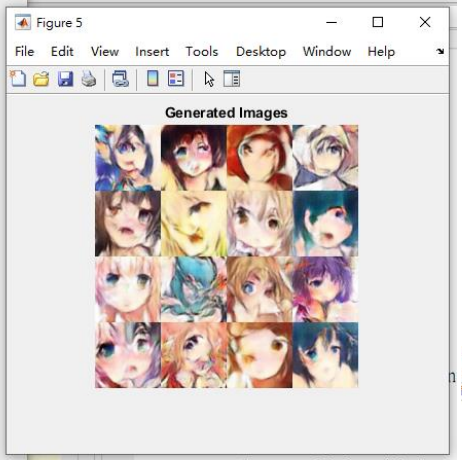
1. The generator - Given a label and random array as input, this network generates data with the same structure as the training data observations corresponding to the same label.
2. The discriminator - Given batches of labeled data containing observations from both the training data and generated data from the generator, this network attempts to classify the observations as "real" or "generated".





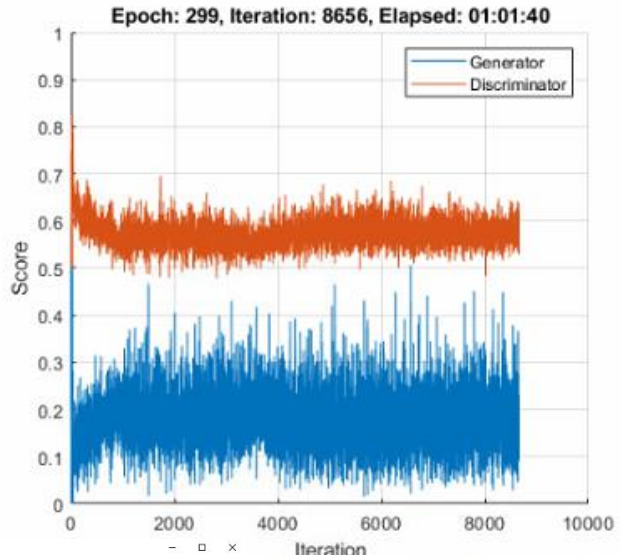
GAN & CGAN(Conditional Generative Adversarial Network

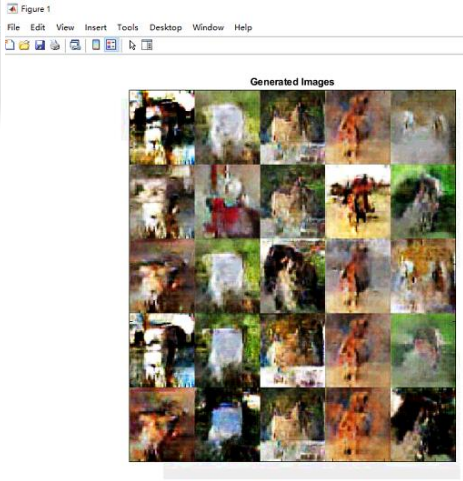
GAN

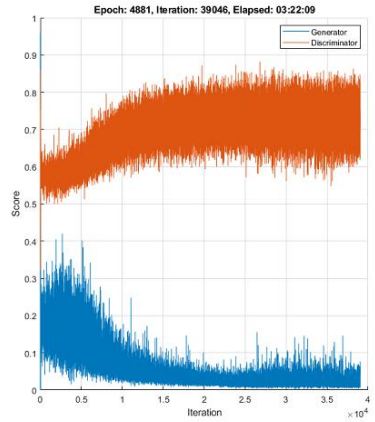


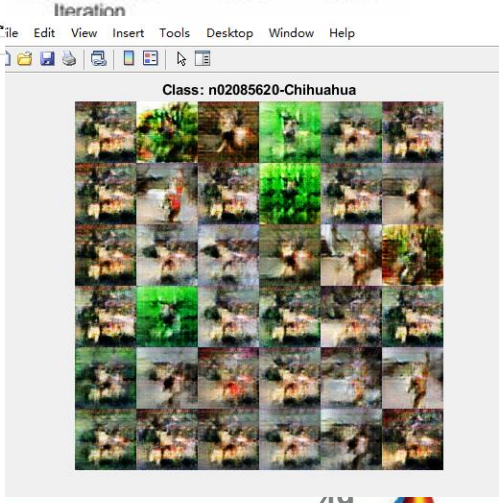
State: [6×3 table]
InputNames: {'in'}

CGAN

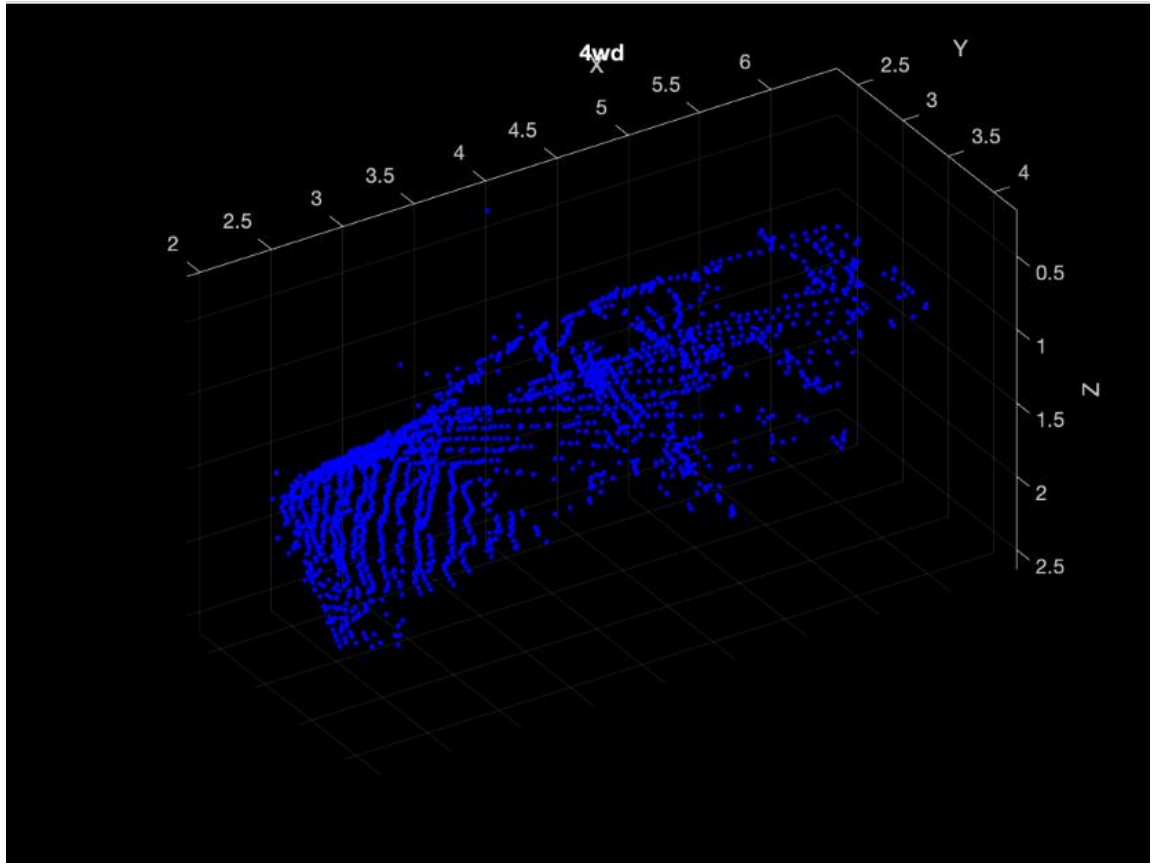


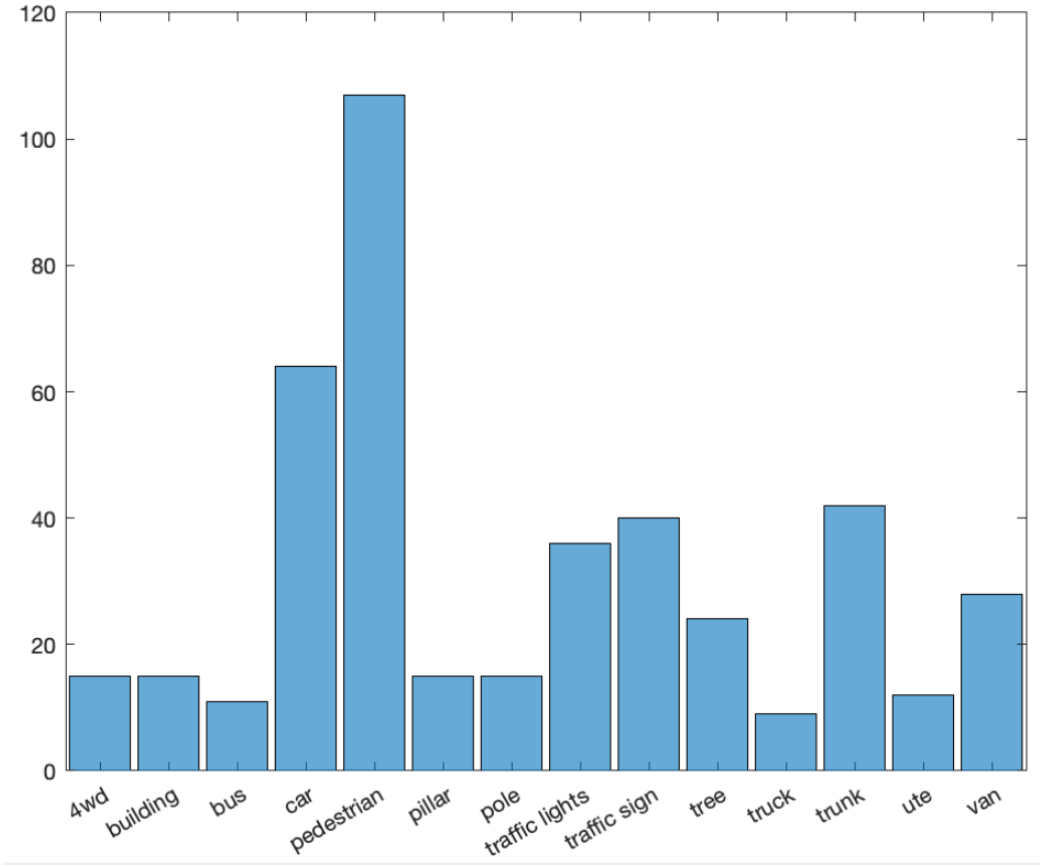






Point Cloud Classification Using PointNet Deep Learning

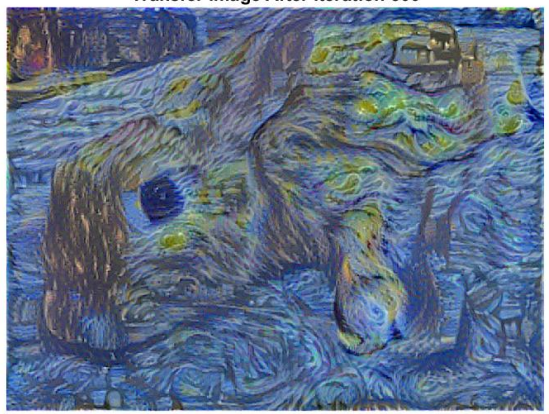




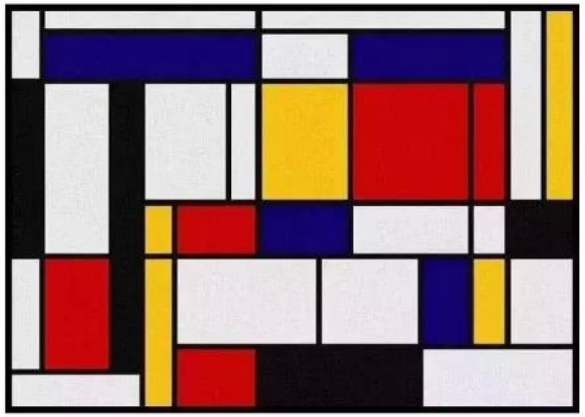
Neural Style Transfer Using Deep Learning

Transfer Image After Iteration 600

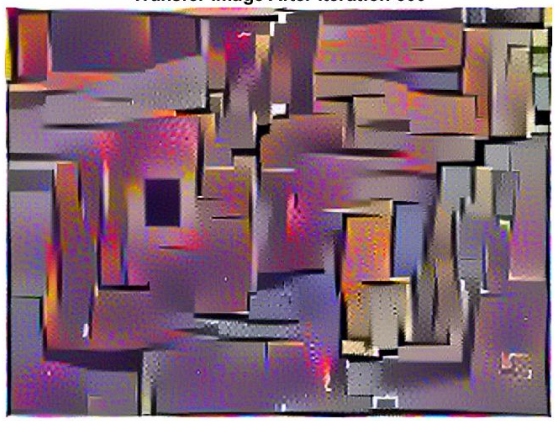
Transfer Image After Iteration 300



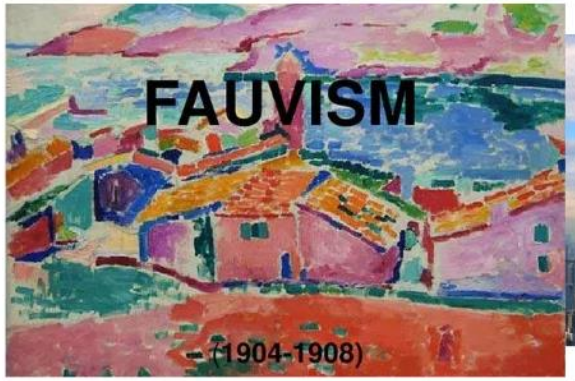
Transfer Image After Iteration 350



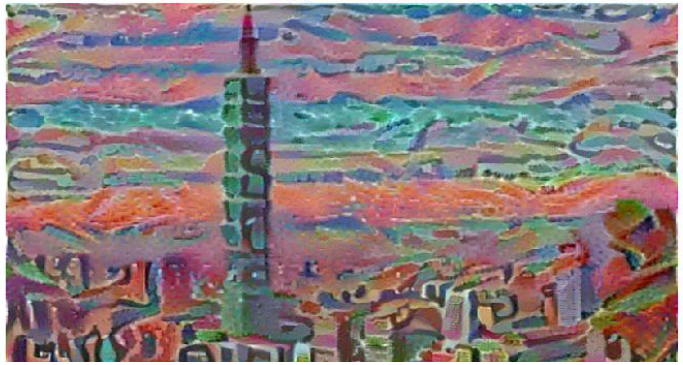
Transfer Image After Iteration 300

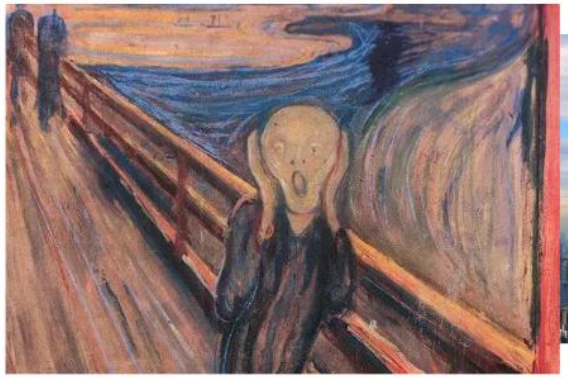


Transfer Image After Iteration 1500



Transfer Image After Iteration 300





Transfer Image After Iteration 2500

Vision SLAM & Barcodes read