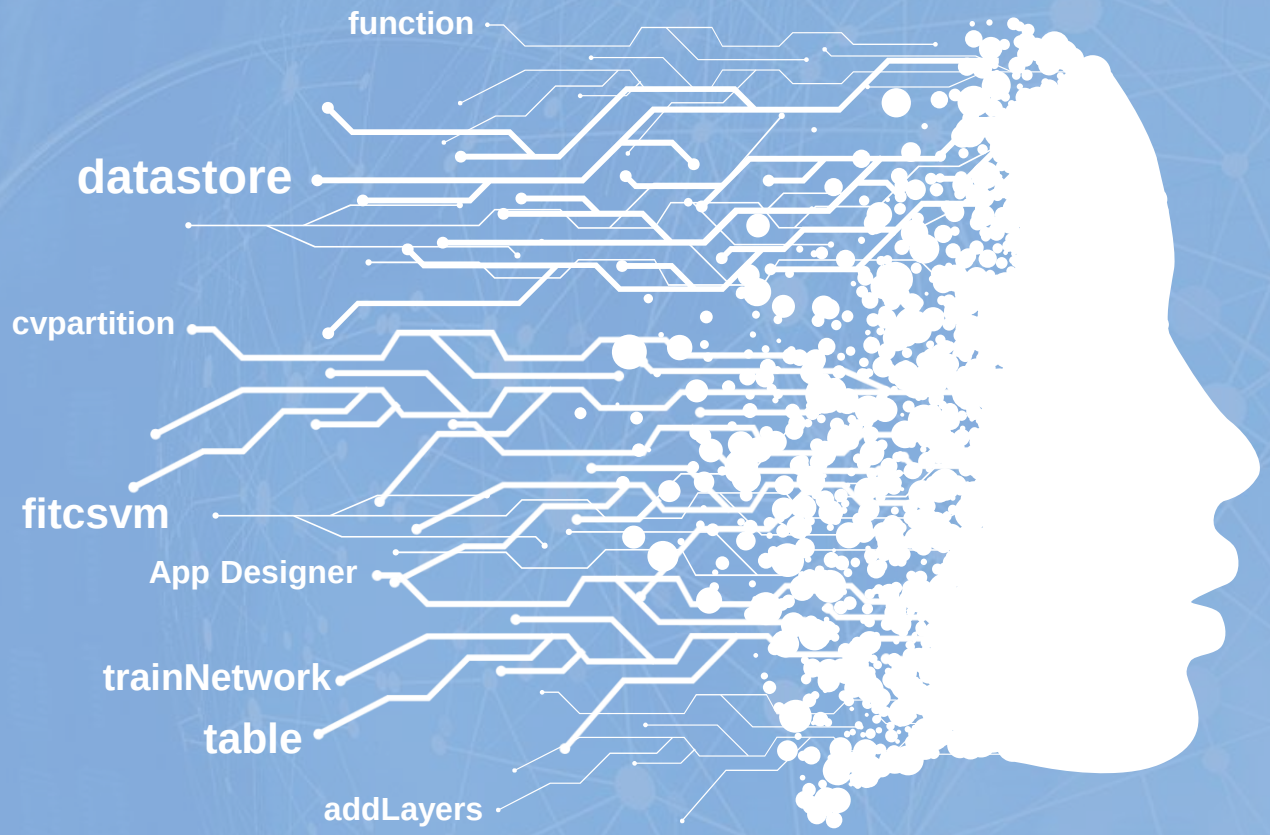
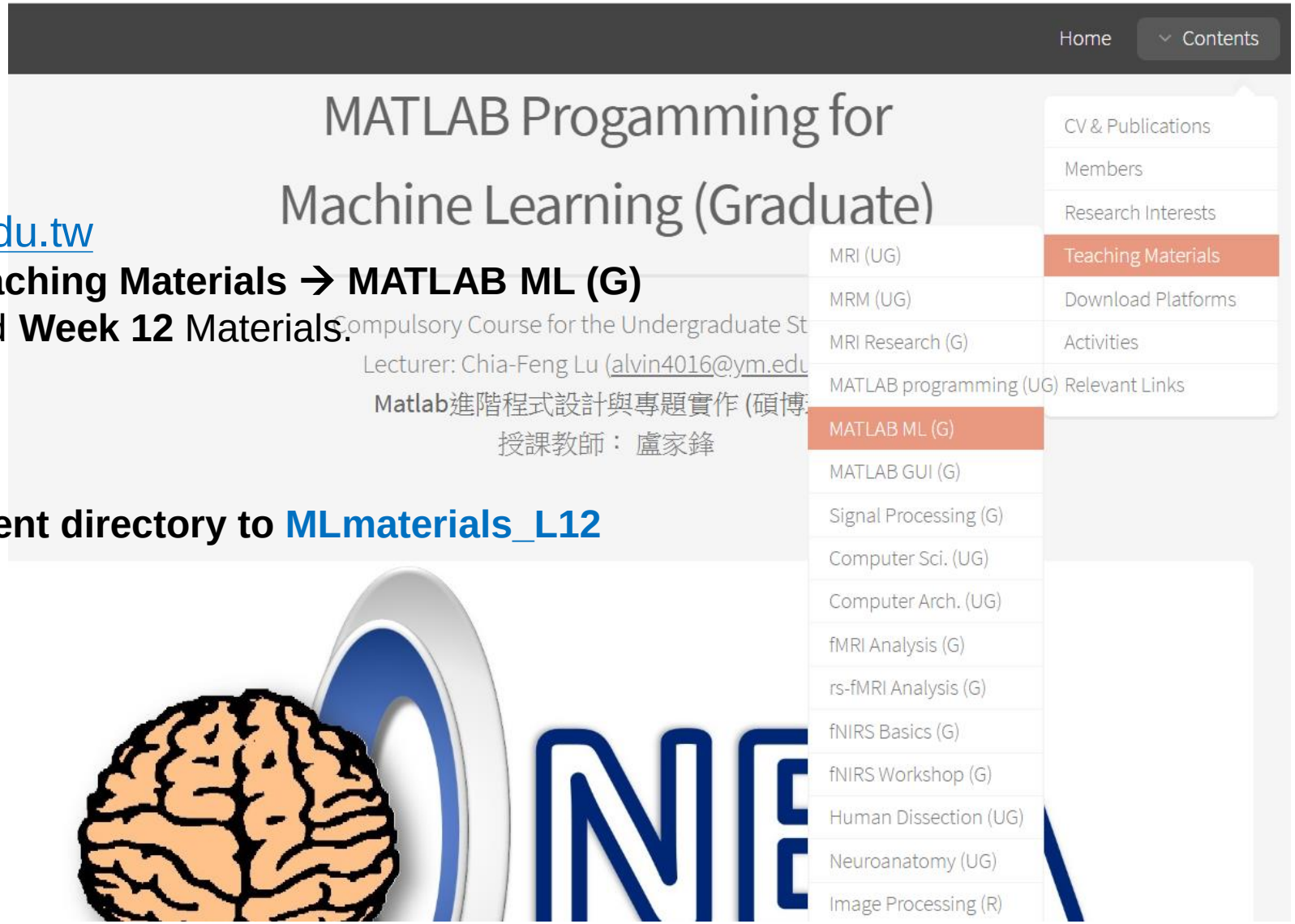


Deep Network Designer

MATLAB進階程式語言與實作

盧家鋒 Chia-Feng Lu, Ph.D.
Department of Biomedical Imaging and Radiological Sciences, NYCU
alvin4016@nycu.edu.tw

Teaching Materials



cflu.lab.nycu.edu.tw

Contents → Teaching Materials → MATLAB ML (G)

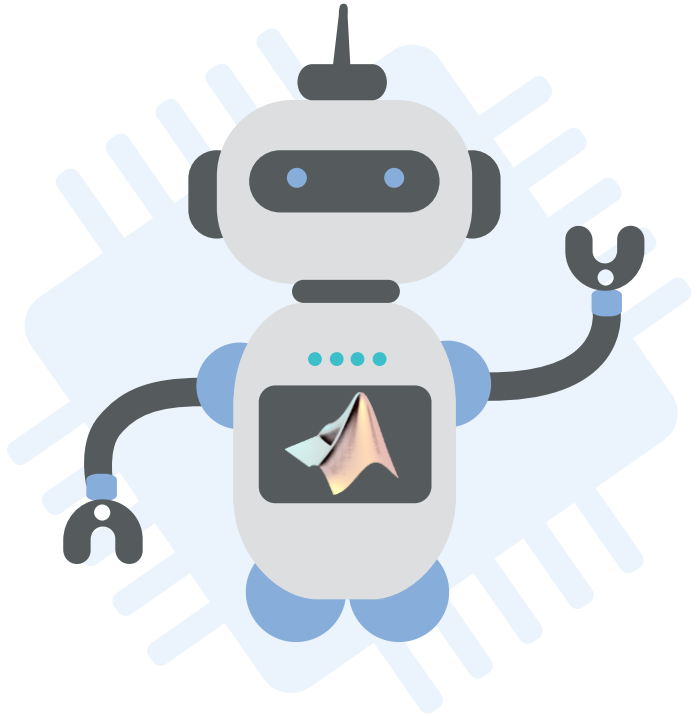
Please download **Week 12** Materials.

Please set current directory to MLmaterials_L12

Contents in this Week

01 Deep Network Designer

Create, import, analyze, train and export



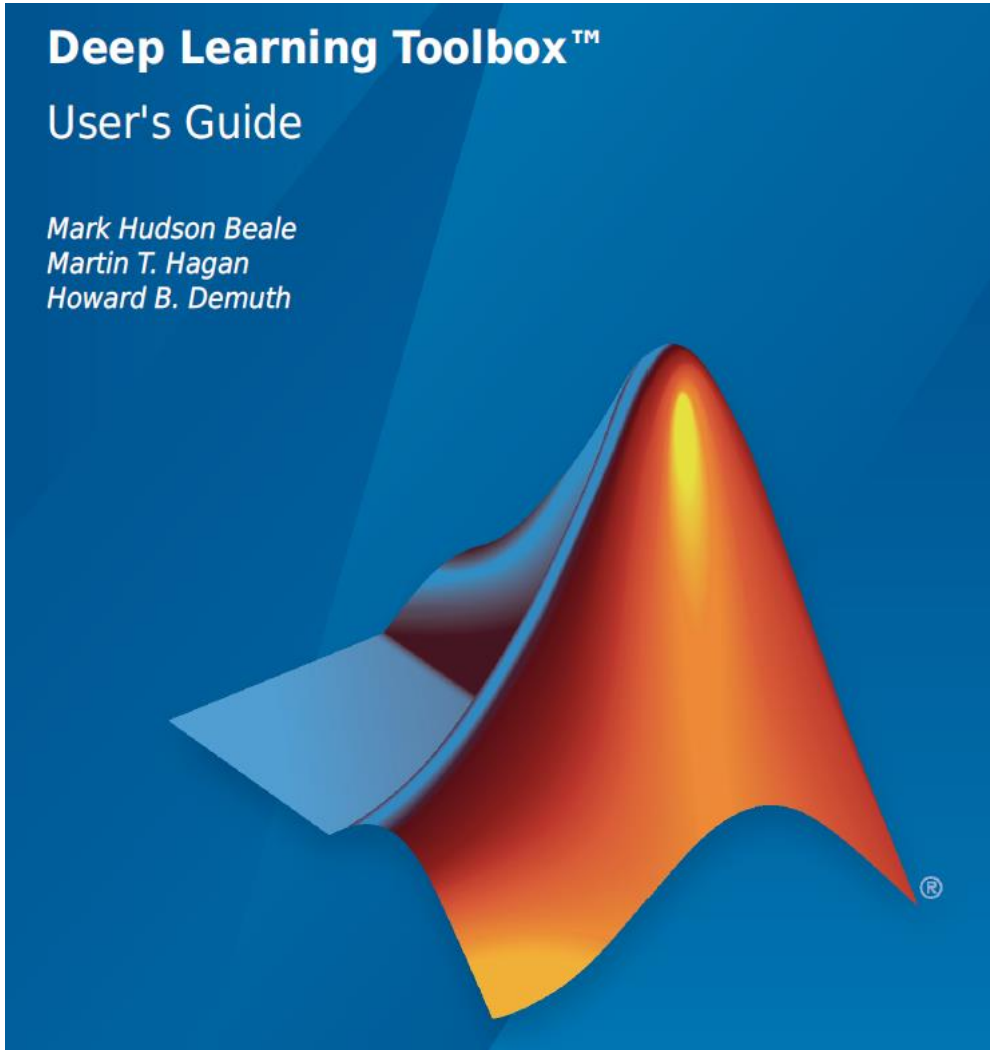
Reference

- MATLAB Deep Learning Toolbox User's Guide (2192 pages)

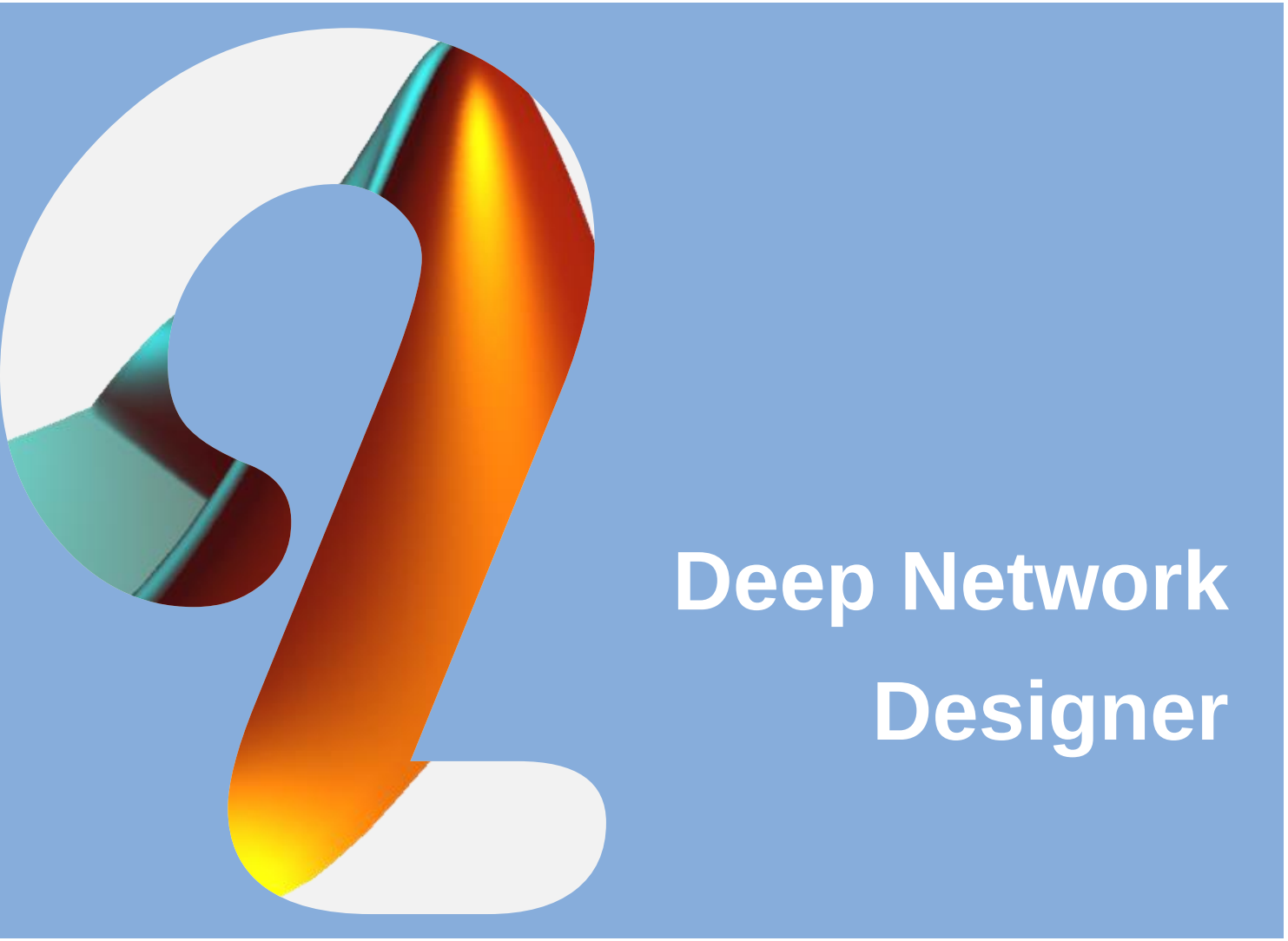
  Mark Hudson Beale

  Martin T. Hagan

  Howard B. Demuth

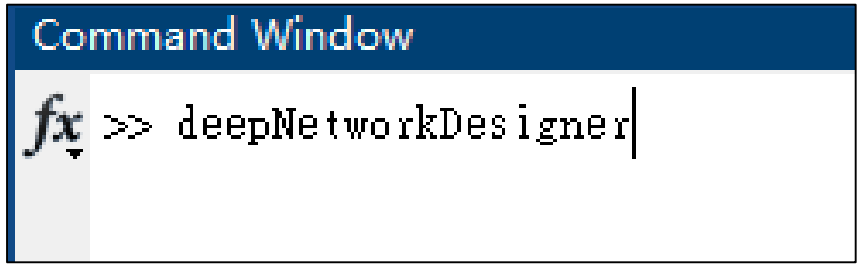


2020b or later version is required!

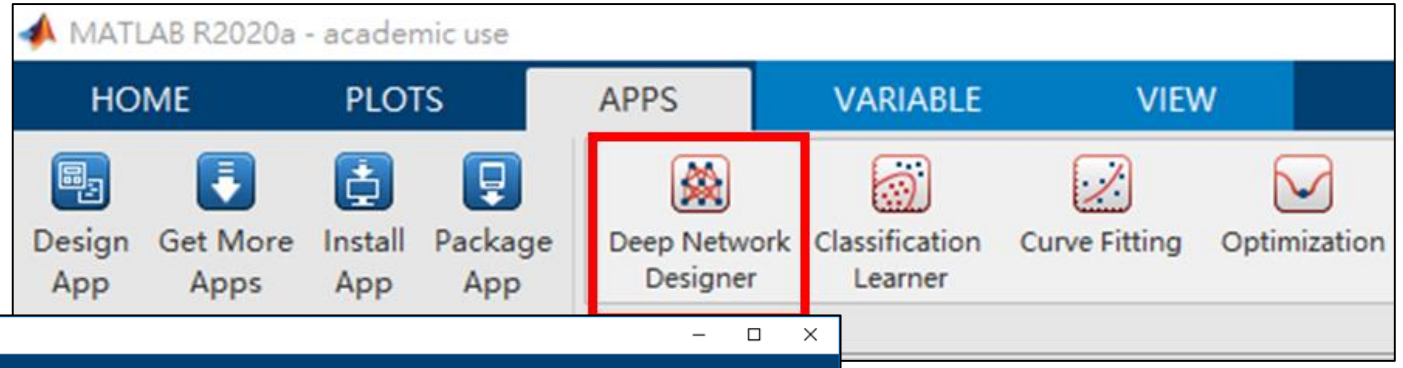


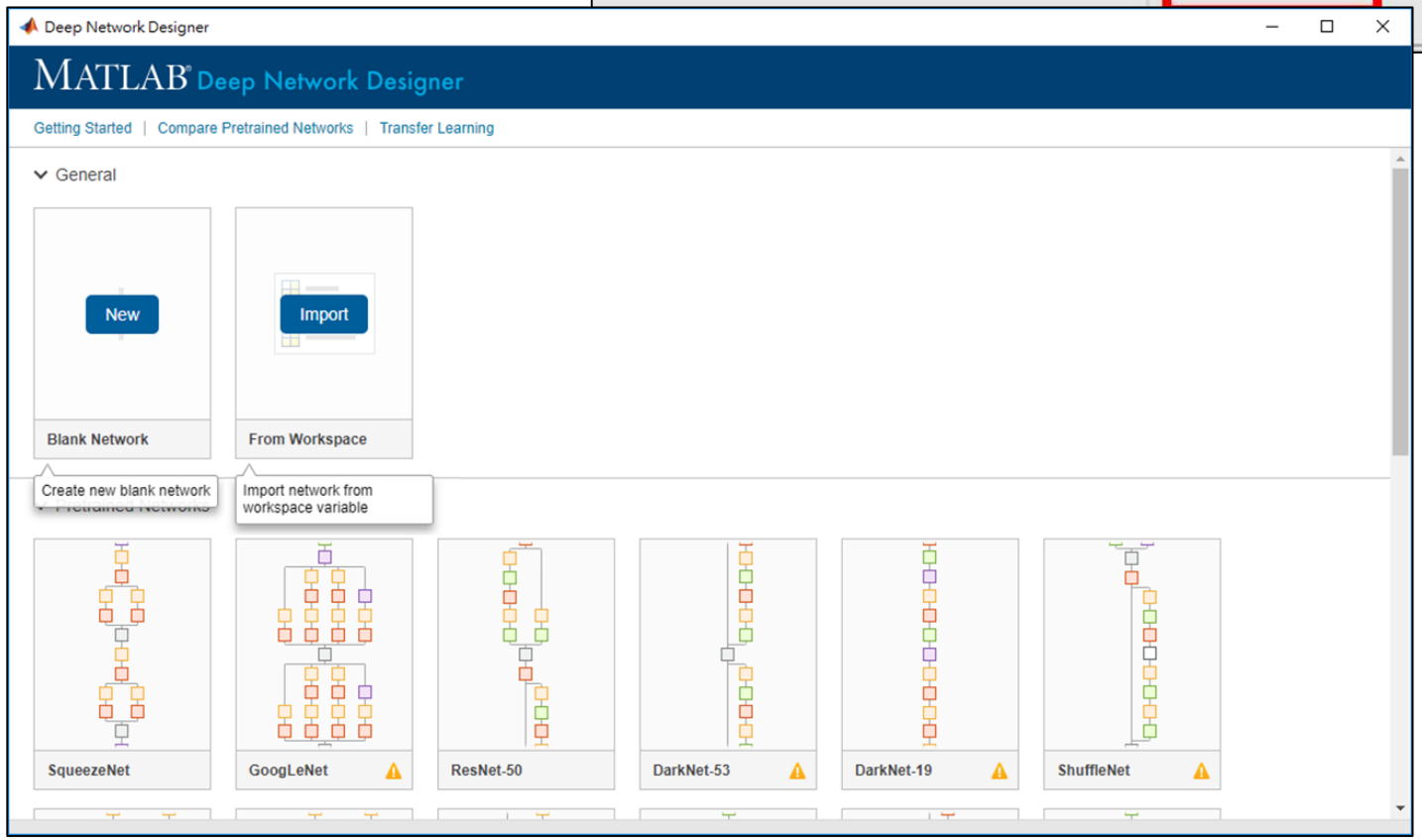
Create, import, analyze, train and export

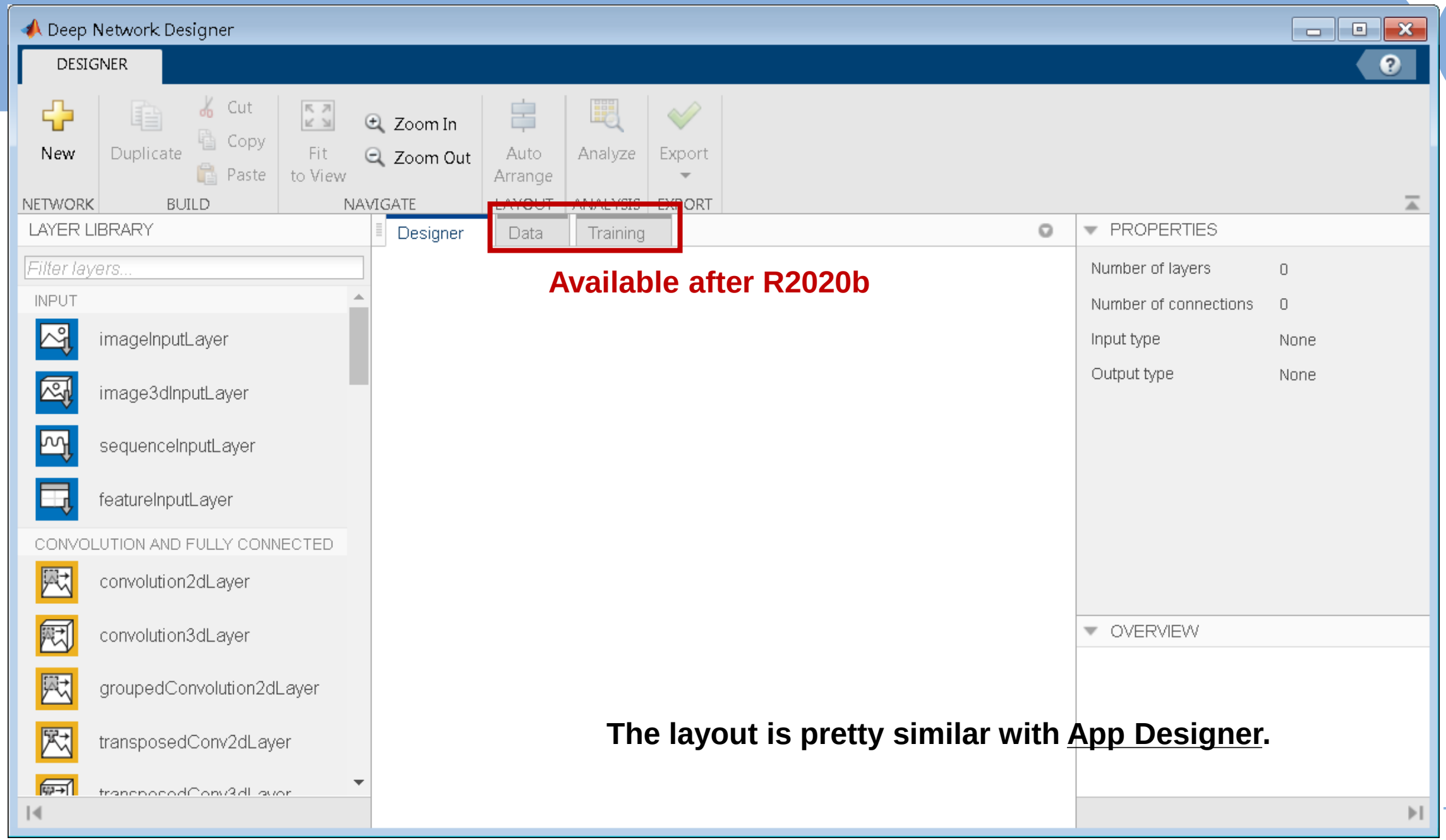
Deep Network Designer



or





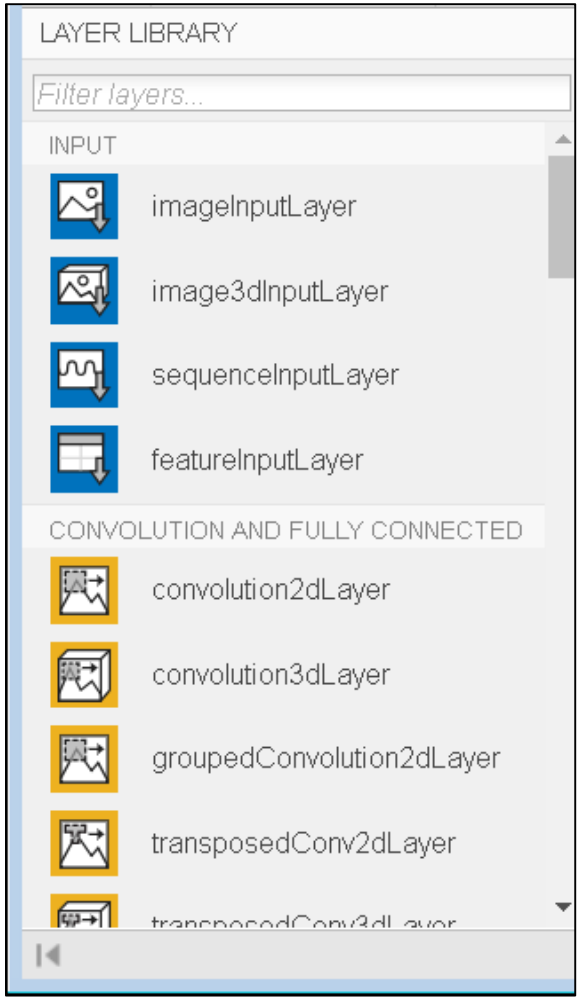


Available after R2020b

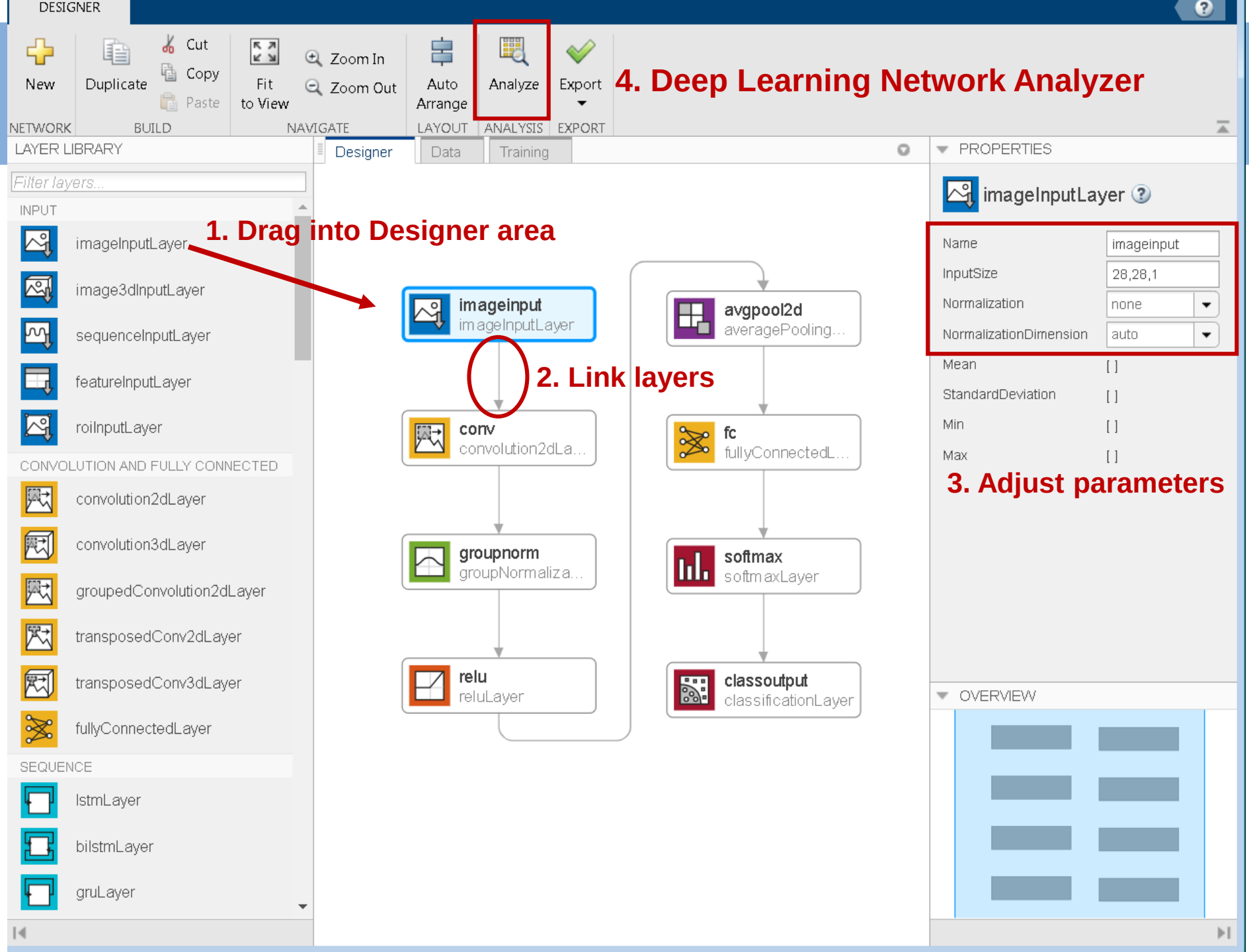
The layout is pretty similar with App Designer.

Layer Library – R2021a

- **Input Layers**
- **Convolution and Fully Connected Layers**
- **Sequence Layers**
- **Activation Layers**
- **Normalization, Dropout, and Cropping Layers**
- **Pooling and Unpooling Layers**
- **Combination Layers**
- **Object Detection Layers**
- **Output Layers**
- **Generative Adversarial Network Layers**

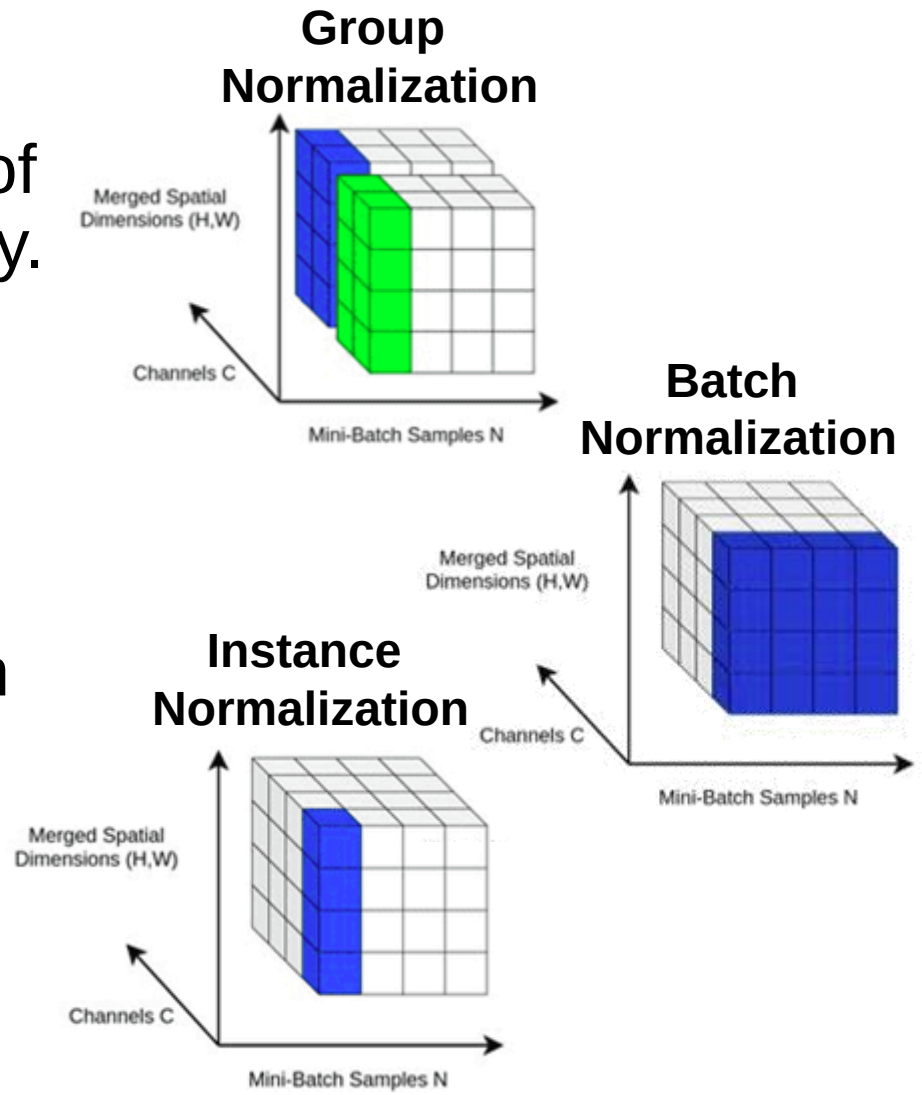


https://www.mathworks.com/help/deeplearning/ug/list-of-deep-learning-layers.html


DESIGNER
New
Duplicate
Cut
Copy
Paste
Fit to View
Zoom In
Zoom Out
Auto Arrange
Analyze
Export
NETWORK
BUILD
NAVIGATE
LAYOUT
ANALYSIS
EXPORT
4. Deep Learning Network Analyzer
LAYER LIBRARY
Filter layers...
INPUT
imageInputLayer
image3dInputLayer
sequenceInputLayer
featureInputLayer
roiInputLayer
CONVOLUTION AND FULLY CONNECTED
convolution2dLayer
convolution3dLayer
groupedConvolution2dLayer
transposedConv2dLayer
transposedConv3dLayer
fullyConnectedLayer
SEQUENCE
lstmLayer
bilstmLayer
gruLayer
Designer
Data
Training
1. Drag into Designer area
imageinput
imageInputLayer
2. Link layers
conv
convolution2dLa...
groupnorm
groupNormaliza...
relu
reluLayer
avgpool2d
averagePooling...
fc
fullyConnectedL...
softmax
softmaxLayer
classoutput
classificationLayer
PROPERTIES
imageInputLayer
Name
imageinput
InputSize
28,28,1
Normalization
none
NormalizationDimension
auto
Mean
[ ]
StandardDeviation
[ ]
Min
[ ]
Max
[ ]
3. Adjust parameters
OVERVIEW

Group Normalization (R2020b or later)

- A group normalization layer normalizes a mini-batch of data across grouped subsets of channels for each observation independently.
- It can speed up training of the convolutional neural network and reduce the sensitivity to network initialization
- Put group normalization layers between convolutional layers and nonlinearities, such as ReLU layers.



Zero mean, unit variance

Deep Learning Network Analyzer

Network from Deep Network Designer

Analysis date: 13-May-2021 20:31:35

8 layers | 0 warnings | 0 errors

ANALYSIS RESULT

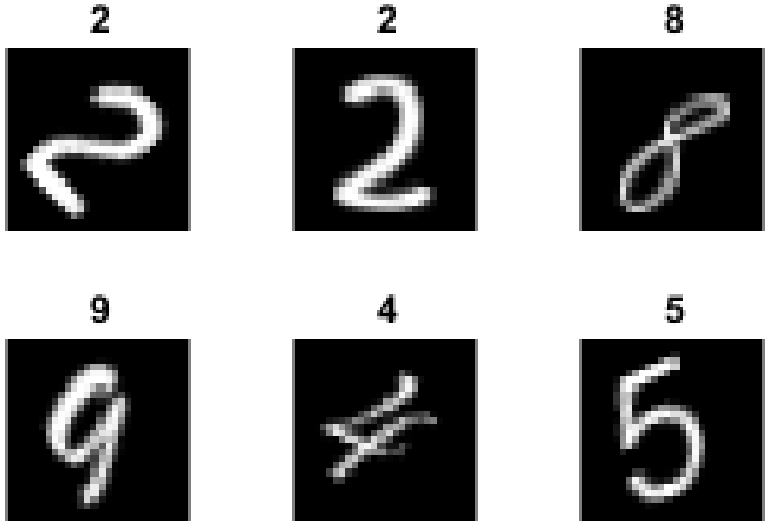
	Name	Type	Activations	Learnables
1	imageinput 28×28×1 images	Image Input	28×28×1	-
2	conv 20 5×5 convolutions with stride [1 …	Convolution	24×24×20	Weights 5×5×1×20 Bias 1×1×20
3	groupnorm Group normalization	Group Normalization	24×24×20	Offset 1×1×20 Scale 1×1×20
4	relu ReLU	ReLU	24×24×20	-
5	avgpool2d 2×2 average pooling with stride [2 …	Average Pooling	12×12×20	-
6	fc 10 fully connected layer	Fully Connected	1×1×10	Weights 10×2880 Bias 10×1
7	softmax softmax	Softmax	1×1×10	-
8	classoutput crossentropyex	Classification Output	-	-

Data Import - datastore

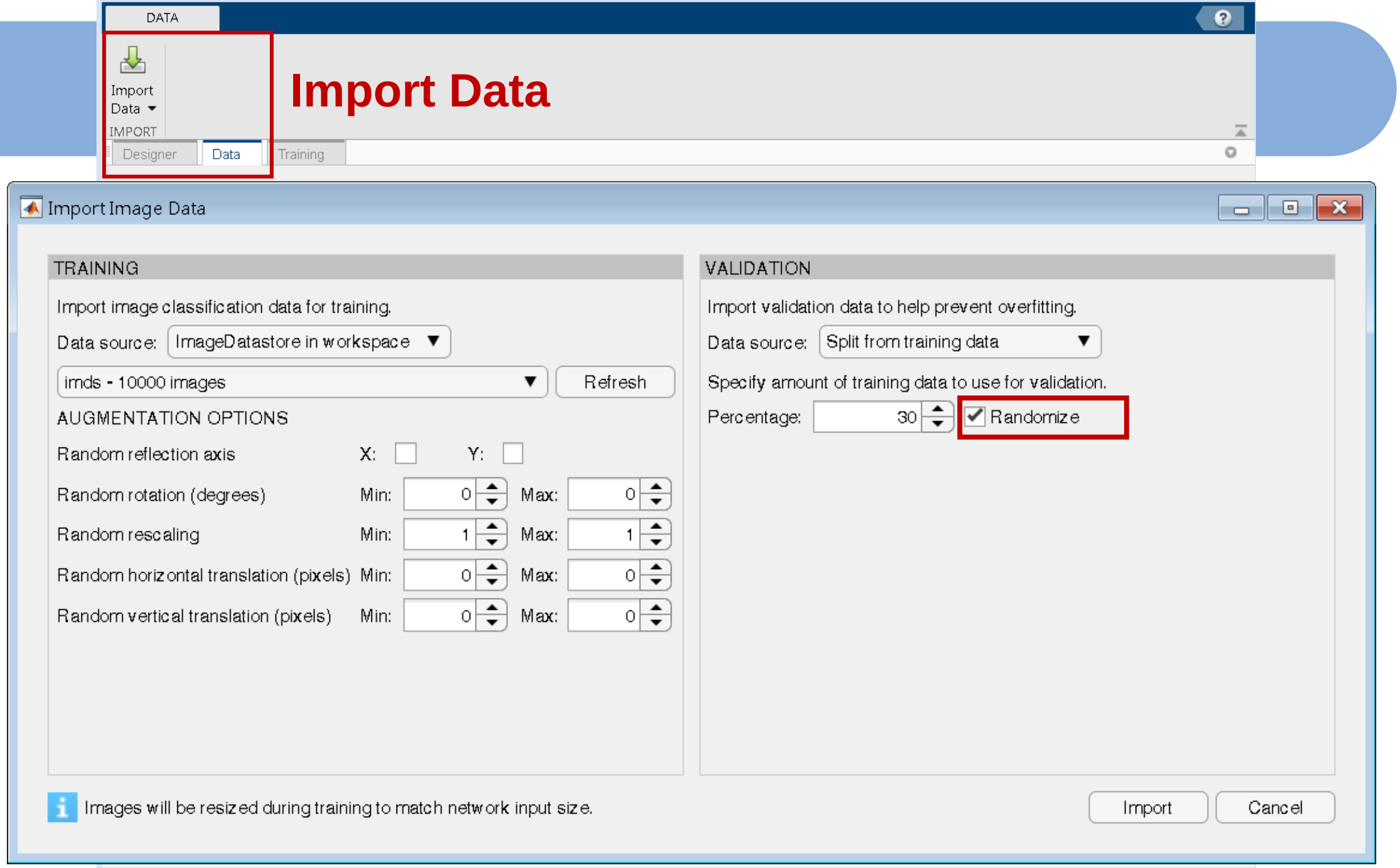
- dataFolder = fullfile(toolboxdir('nnet'),'nndemos','nndatasets','DigitDataset');
- imds = imageDatastore(dataFolder,'IncludeSubfolders',true,...
  'LabelSource','foldernames');

```
imds =
  ImageDatastore with properties:

Files: {
        ' ...\nndatasets\DigitDataset\0\image10000.png';
        ' ...\nndatasets\DigitDataset\0\image9001.png';
        ' ...\nndatasets\DigitDataset\0\image9002.png'
          ... and 9997 more
         }
Folders: {
' ...\MATLAB\R2020b\toolbox\nnet\nndemos\nndatasets\DigitDataset'
             }
Labels: [0; 0; 0 ... and 9997 more categorical]
```



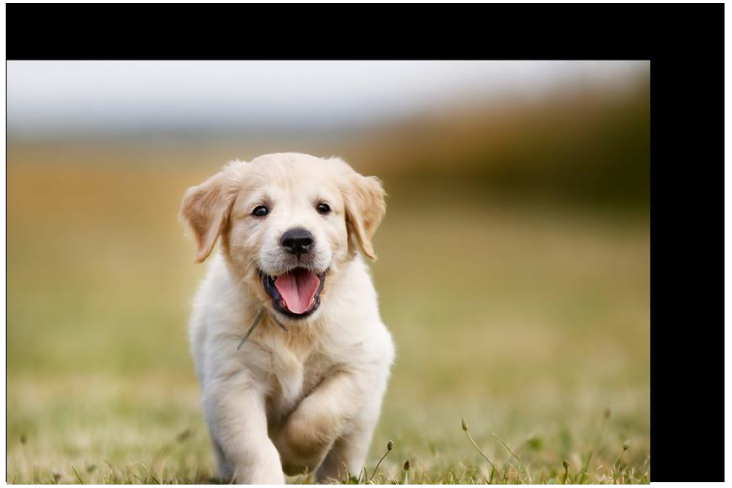
Import Data


DATA
Import Data
IMPORT
Designer
Data
Training
Import Image Data
TRAINING
Import image classification data for training.
Data source: ImageDatastore in workspace
imds - 10000 images
Refresh
AUGMENTATION OPTIONS
Random reflection axis X: Y:
Random rotation (degrees) Min: 0 Max: 0
Random rescaling Min: 1 Max: 1
Random horizontal translation (pixels) Min: 0 Max: 0
Random vertical translation (pixels) Min: 0 Max: 0
VALIDATION
Import validation data to help prevent overfitting.
Data source: Split from training data
Specify amount of training data to use for validation.
Percentage: 30
Randomize
Images will be resized during training to match network input size.
Import
Cancel

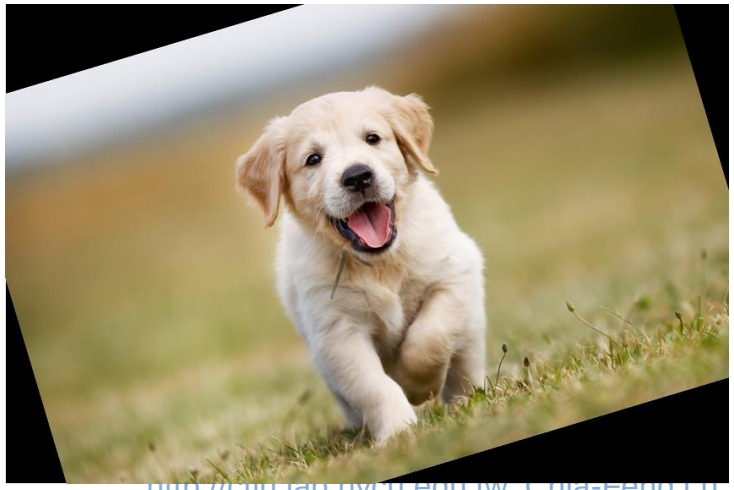
Data Augmentation

Rescale

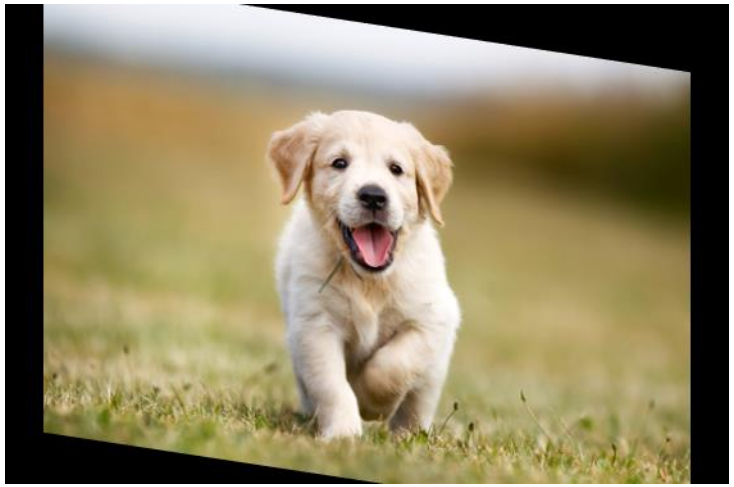
Translation



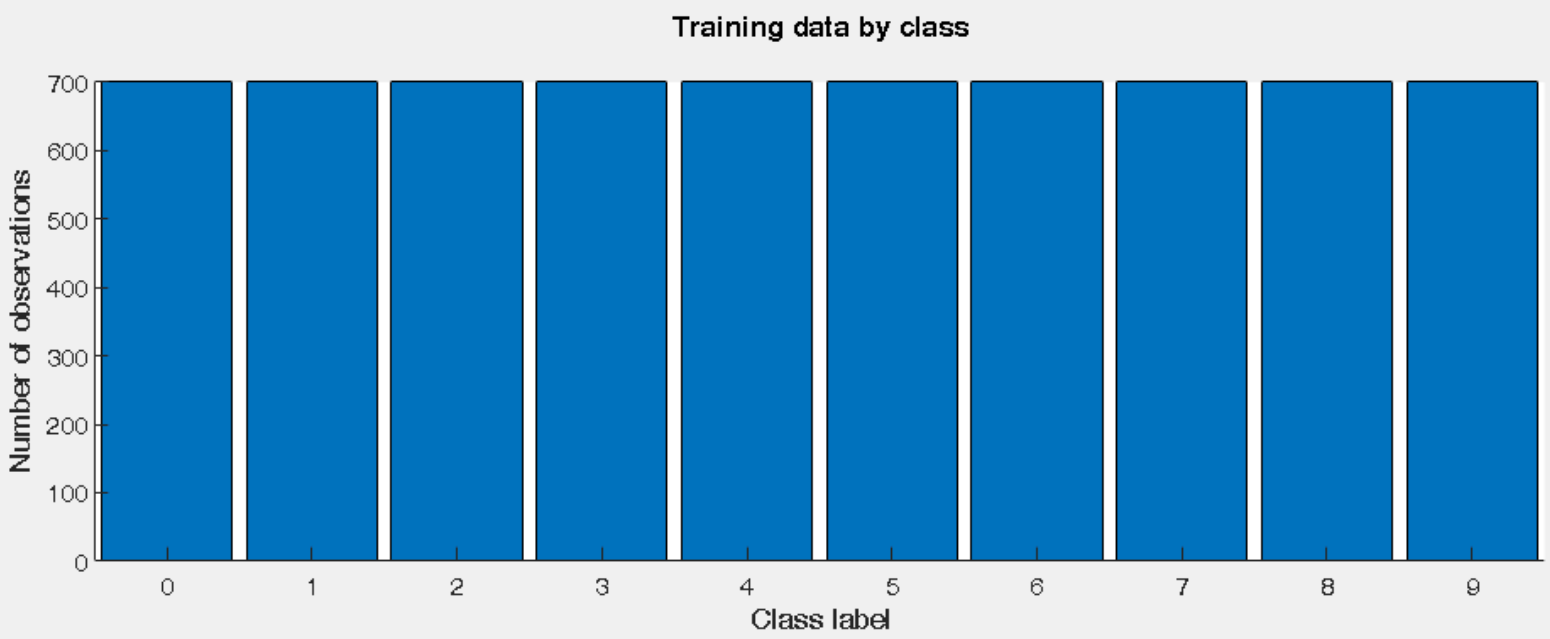
Rotation



Shear



Designer | Data | Training

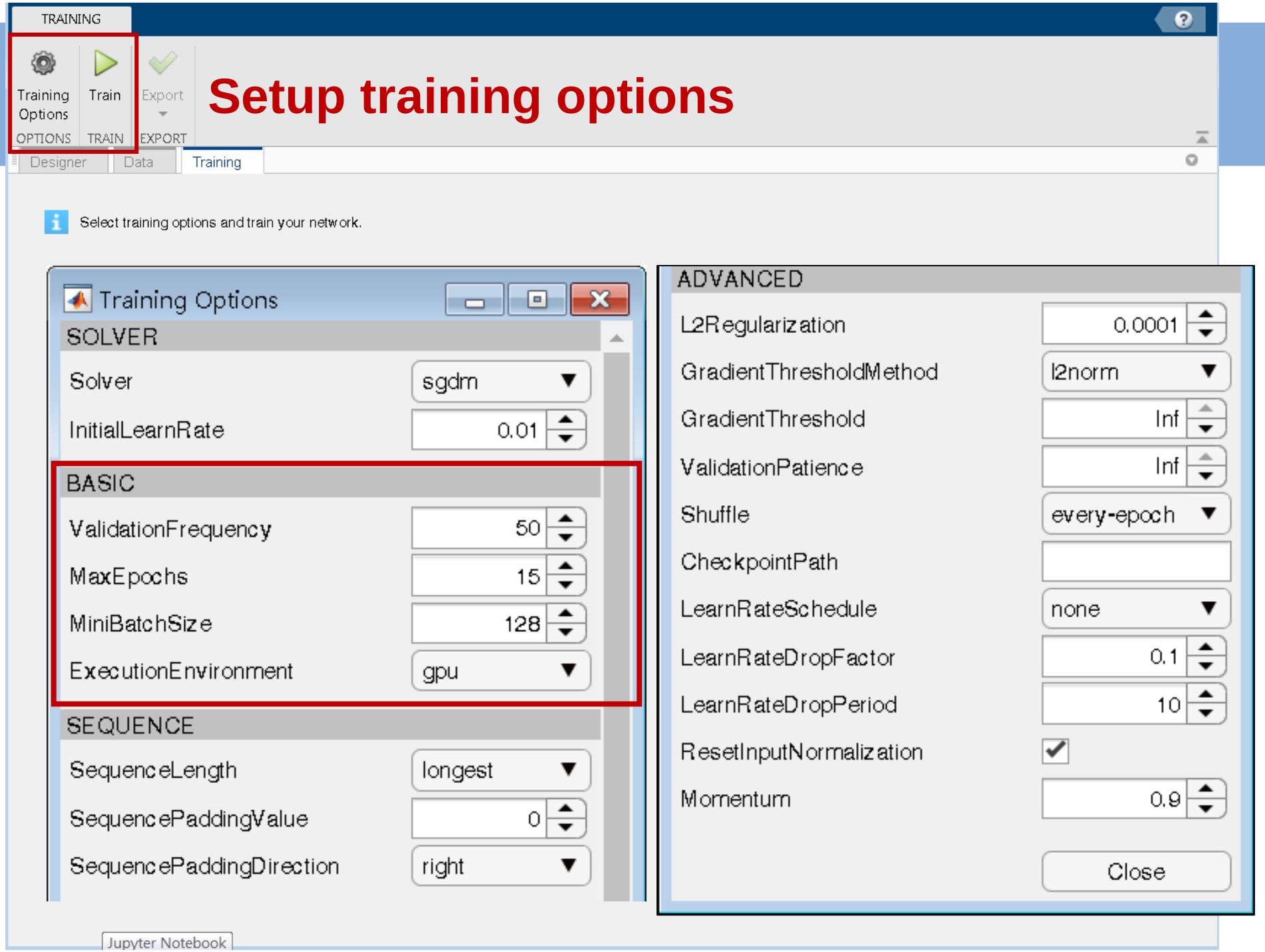


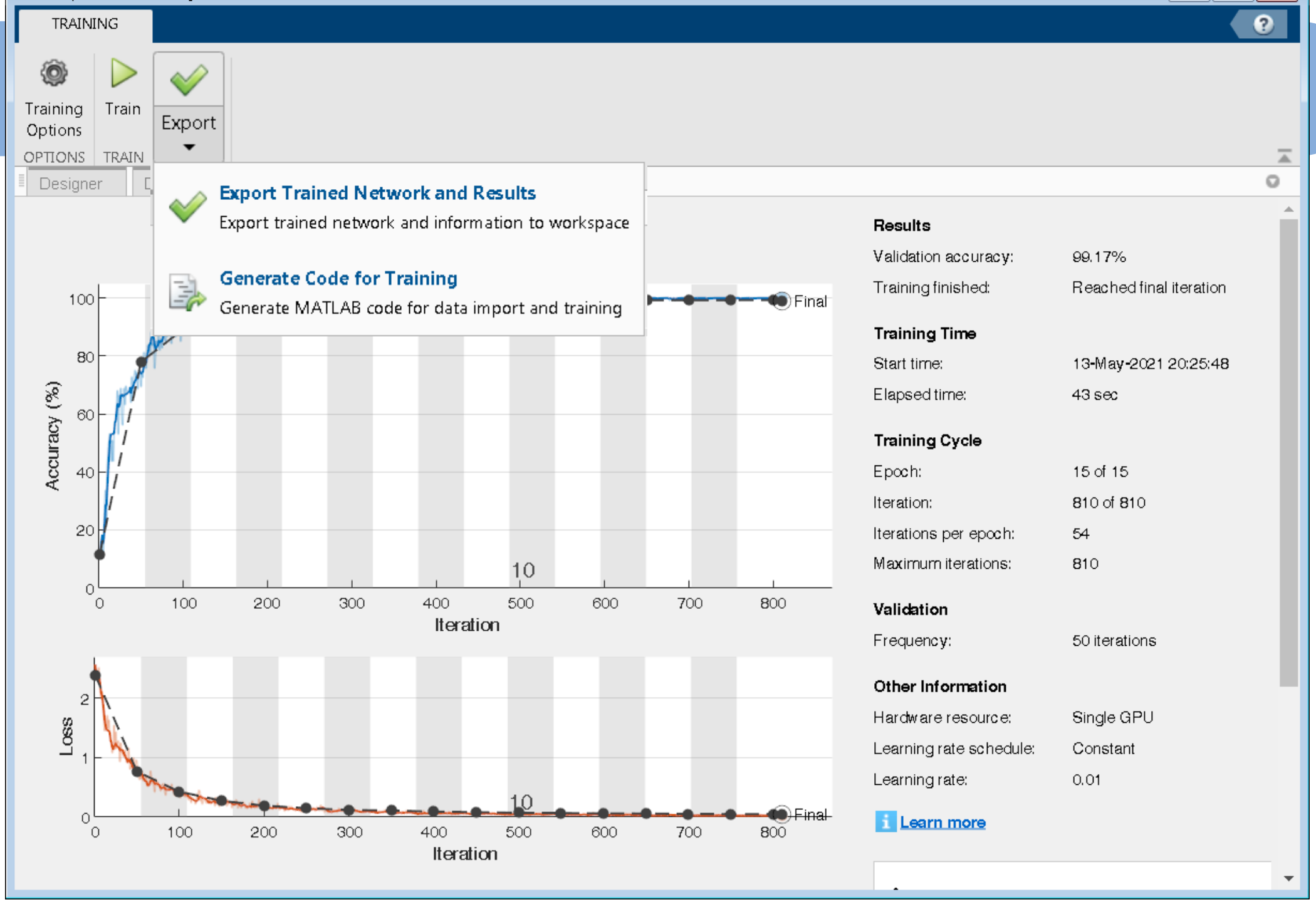
Data:	Training
Source:	imds
Observations:	7000
Classes:	10
Most observations:	0 (700)
Fewest observations:	0 (700)

Show random observations of: <All classes>

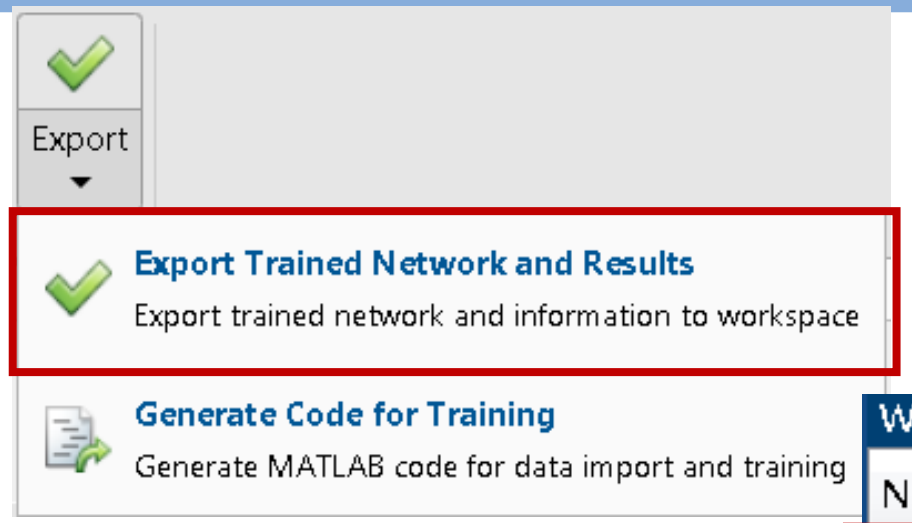
Preview images have 1 channel.

Setup training options

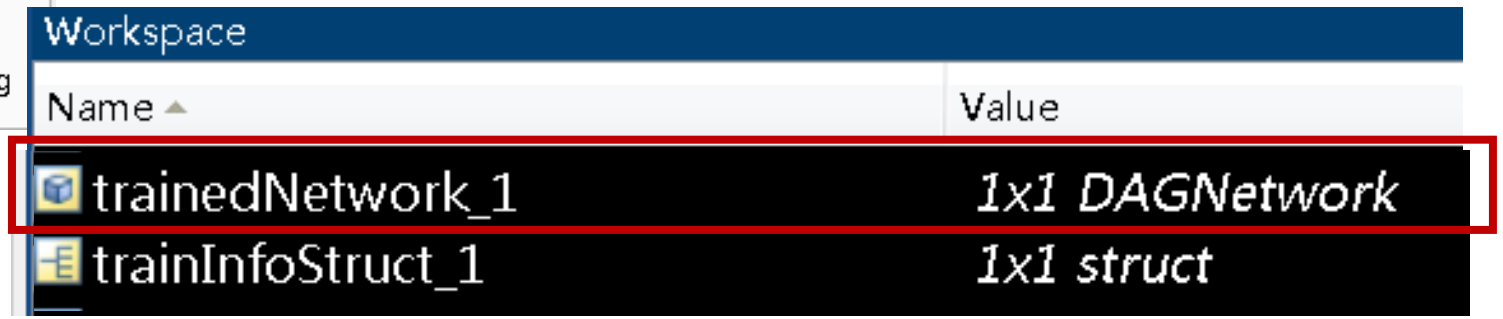



TRAINING
Training Options
Train
Export
OPTIONS
TRAIN
Designer
Export Trained Network and Results
Export trained network and information to workspace
Generate Code for Training
Generate MATLAB code for data import and training
Accuracy (%)
Iteration
Final
Loss
Iteration
Final
Results
Validation accuracy: 99.17%
Training finished: Reached final iteration
Training Time
Start time: 13-May-2021 20:25:48
Elapsed time: 43 sec
Training Cycle
Epoch: 15 of 15
Iteration: 810 of 810
Iterations per epoch: 54
Maximum iterations: 810
Validation
Frequency: 50 iterations
Other Information
Hardware resource: Single GPU
Learning rate schedule: Constant
Learning rate: 0.01
Learn more

Export Trained Model



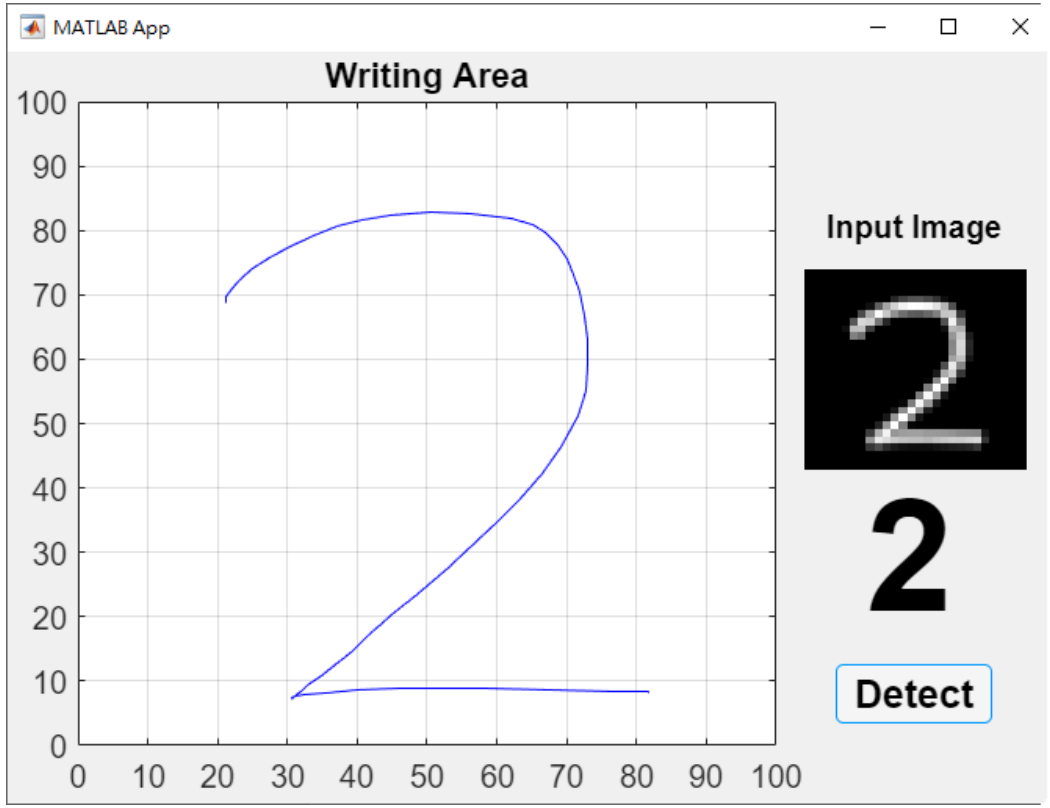
Directed Acyclic Graph (GAG)




```
>> classify(trainedNetwork_1,img)   % Categorical response
>> predict(trainedNetwork_1,img)    % Probability vector
```

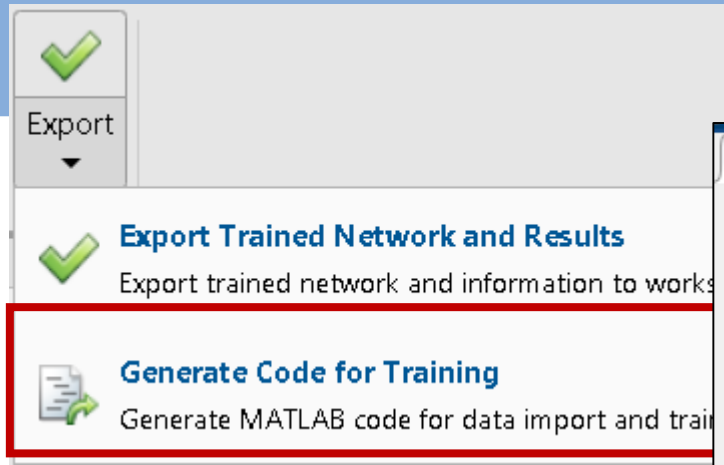
Deploy the Trained Model

- Try the Recognition App
  - **MLmaterials_L12\CNNdigits_App.mlapp**



Test on your own handwritten digits.
→ Much better performance

Generate Code for Training



MATLAB Live Script (*.mlx)

CNNforDigits_CFLu.mlx

Set Training Options

Specify options to use when training.

```
opts = trainingOptions("sgdm",...
    "ExecutionEnvironment","gpu",...
    "InitialLearnRate",0.01,...
    "MaxEpochs",15,...
    "Shuffle","every-epoch",...
    "Plots","training-progress",...
    "ValidationData",augimdsValidation);
```

Create Array of Layers

```
layers = [
    imageInputLayer([28 28 1],"Name","imageinput","Normalization","
    convolution2dLayer([5 5],20,"Name","conv")
    groupNormalizationLayer(1,"Name","groupnorm")
    reluLayer("Name","relu")
    averagePooling2dLayer([2 2],"Name","avgpool2d","Padding","same"
    fullyConnectedLayer(10,"Name","fc")
    softmaxLayer("Name","softmax")
    classificationLayer("Name","classoutput")];
```

plot(lgraph)

```
layers = [
    imageInputLayer([28 28 1],"Name","imageinput","Normalization","none")
    convolution2dLayer([5 5],20,"Name","conv")
    groupNormalizationLayer(1,"Name","groupnorm")
    reluLayer("Name","relu")
    averagePooling2dLayer([2 2],"Name","avgpool2d","Padding","same","Stride",[2 2])
    fullyConnectedLayer(10,"Name","fc")
    softmaxLayer("Name","softmax")
    classificationLayer("Name","classoutput")];

lgraph = layerGraph();
lgraph = addLayers(lgraph,layers);
```

Useful Functions

- deepNetworkDesigner(lgraph)

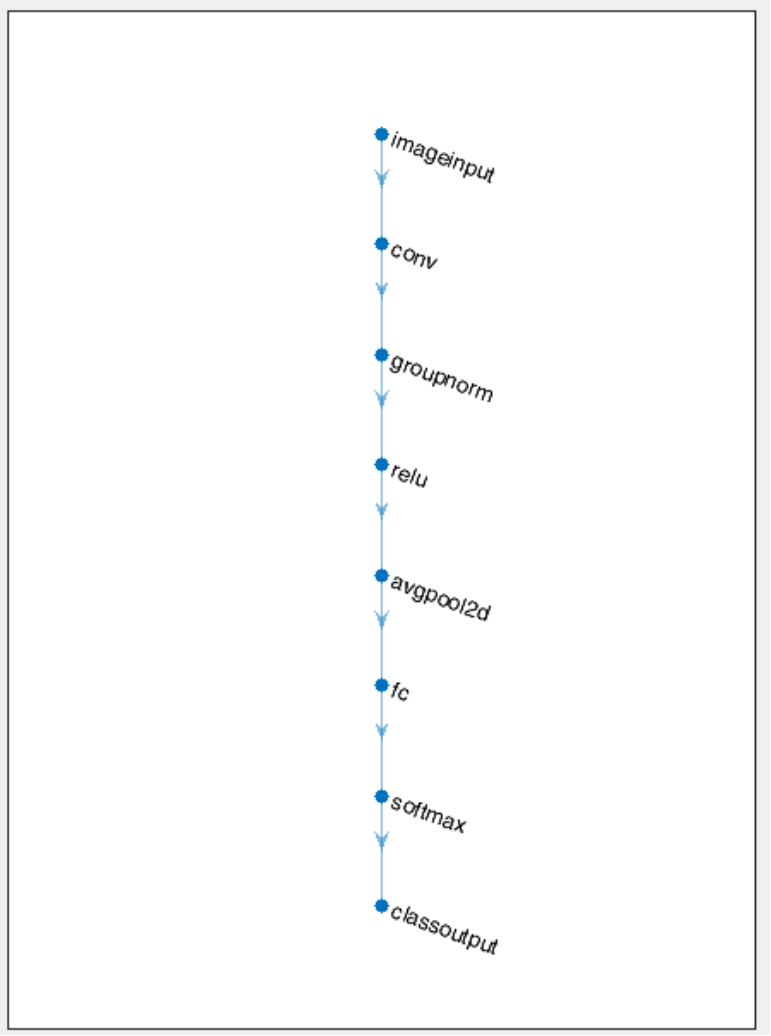
  % lgraph is a Layer or LayerGraph object

- analyzeNetwork(lgraph)

  % lgraph is a Layer or LayerGraph object

- addLayers
- replaceLayer

- lgraph = layerGraph();
- lgraph = addLayers(lgraph,layers);
- plot(lgraph); **% lgraph is a LayerGraph object**



THE END

Contact:
盧家鋒 alvin4016@nycu.edu.tw