

MATLAB EXPO

인공지능 모델의 이해와 검증

김종남 부장, 매스웍스코리아

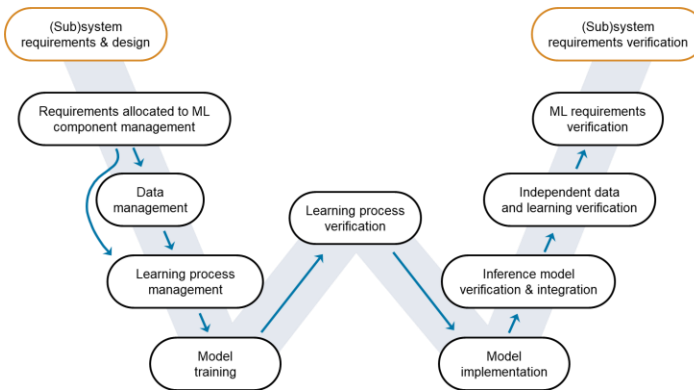


Key Takeaways

MathWorks has capabilities addressing each area of the W-diagram

Library to verify and test robustness of deep learning networks

Our safety-critical certification expertise helps drive new AI standards



Deep Learning Toolbox Verification Library

by MathWorks Deep Learning Toolbox Team **STAFF**

Verify and test robustness of deep learning networks



EUROCAE WG-114 / SAE G-34
Standardization Working Group
“Artificial Intelligence in Aviation”

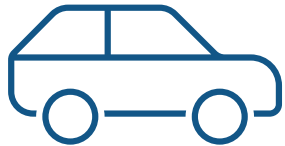
As AI use rises in production, there is a growing need to explain, verify and validate model behavior in safety-critical situations



Challenges in Verification and Validation of AI-enabled Systems



Industries are making progress on verifying AI in systems through whitepapers, standards and planning



Automotive

New WIP [ISO PAS 8800](#)
(Road Vehicles — Safety and artificial intelligence)



Aerospace

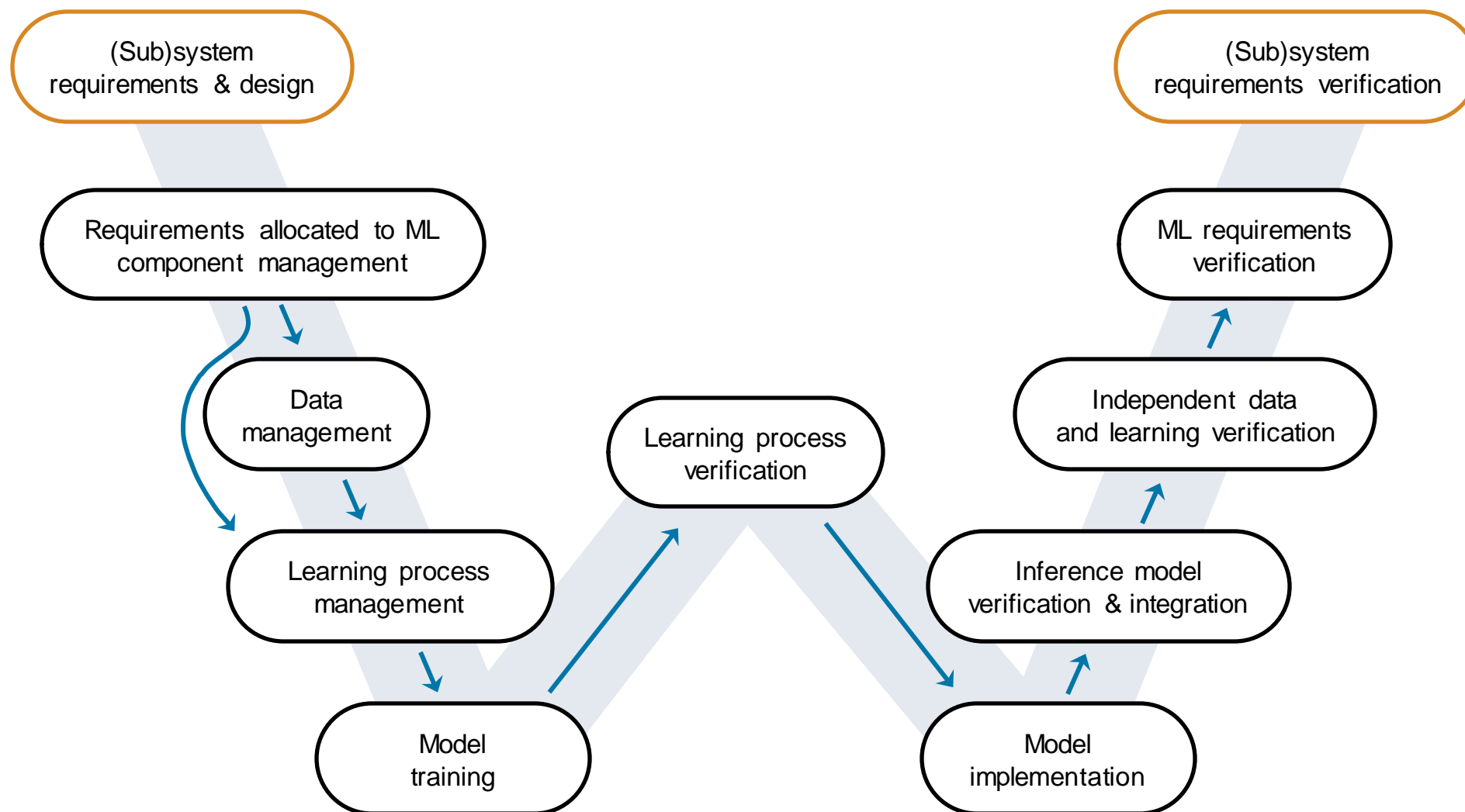
New standard ([AS6983](#)) from [EUROCAE WG-114 / SAE G-34](#) is expected in 2024



Medical Devices

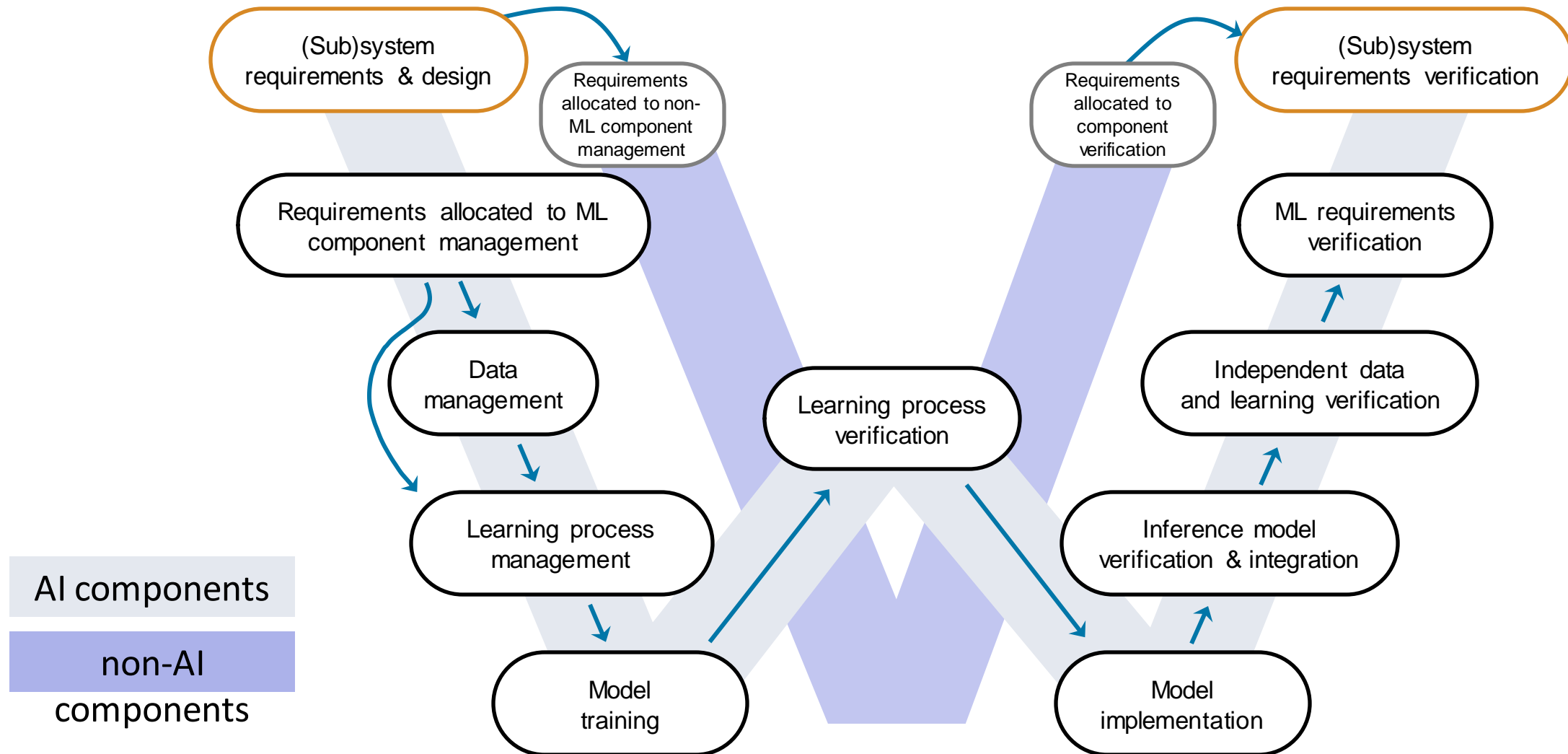
FDA released its first [AI/ML-Based Software as a Medical Device \(SaMD\) Action Plan](#)

W-shaped development process adapting the classical V-shaped cycle to AI applications



Credit: EASA, Daedalean

W-shaped development process can coexist with V-shaped cycle for non-AI components



Task: Verify an image classification network

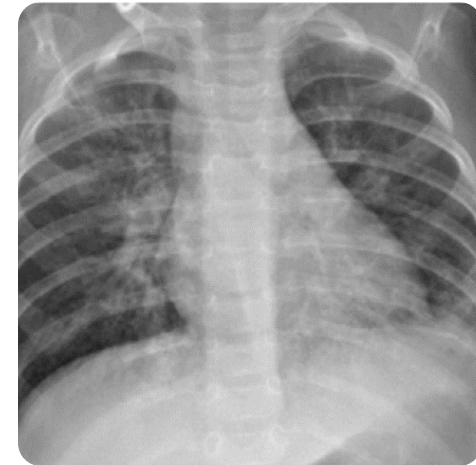
Automotive



Aerospace



Medical



MedMNIST v2 Dataset

MedMNIST v2 - A large-scale lightweight benchmark for 2D and 3D biomedical image classification

Jiancheng Yang, Rui Shi, Donglai Wei, Zequan Liu, Lin Zhao, Bilian Ke, Hanspeter Pfister, Bingbing Ni

¹ Shanghai Jiao Tong University, Shanghai, China

² Boston College, Chestnut Hill, MA

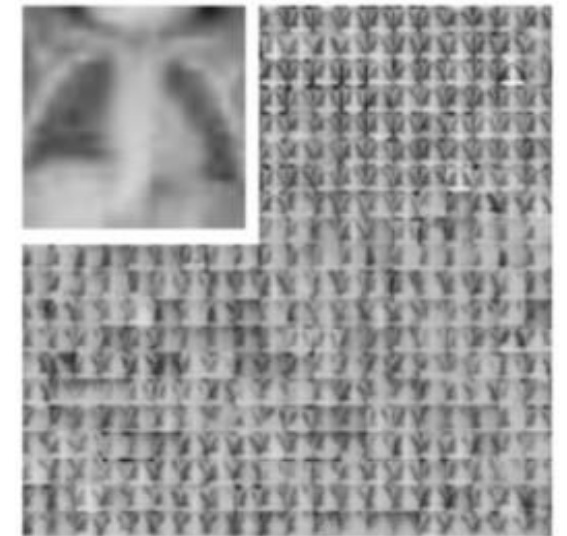
³ RWTH Aachen University, Aachen, Germany

⁴ Fudan Institute of Metabolic Diseases, Zhongshan Hospital, Fudan University, Shanghai, China

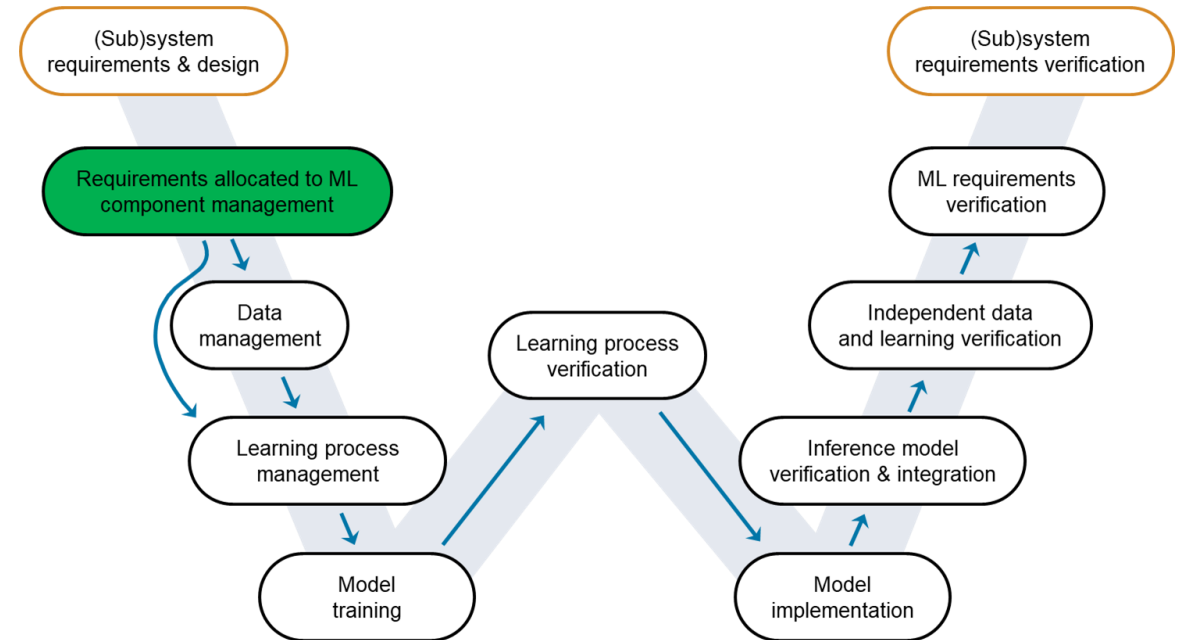
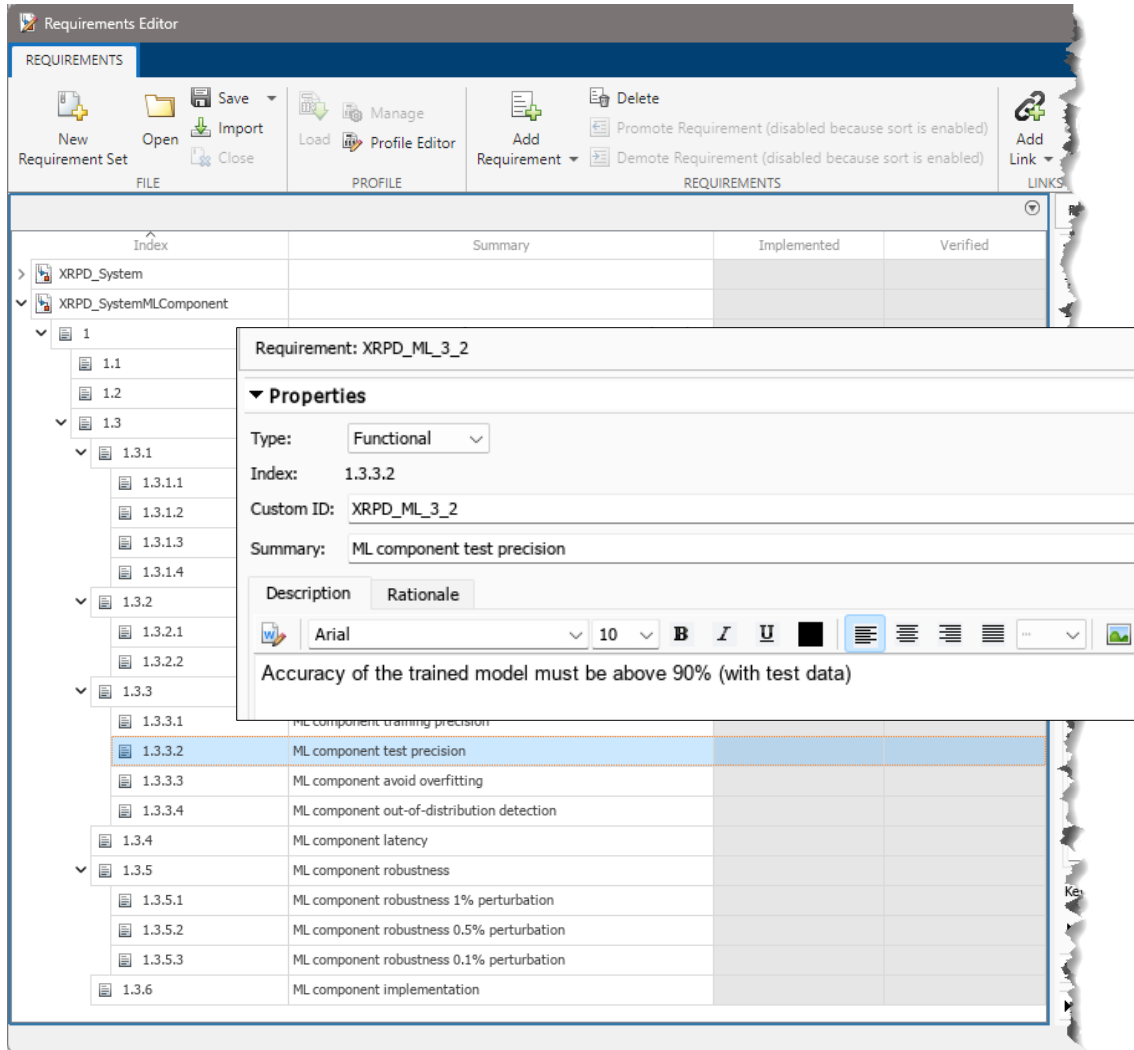
⁵ Shanghai General Hospital, Shanghai Jiao Tong University School of Medicine, Shanghai, China

⁶ Harvard University, Cambridge, MA

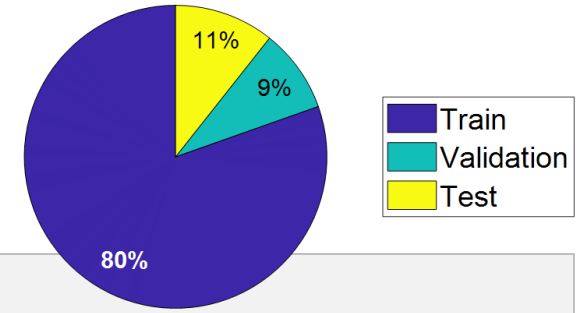
PneumoniaMNIST



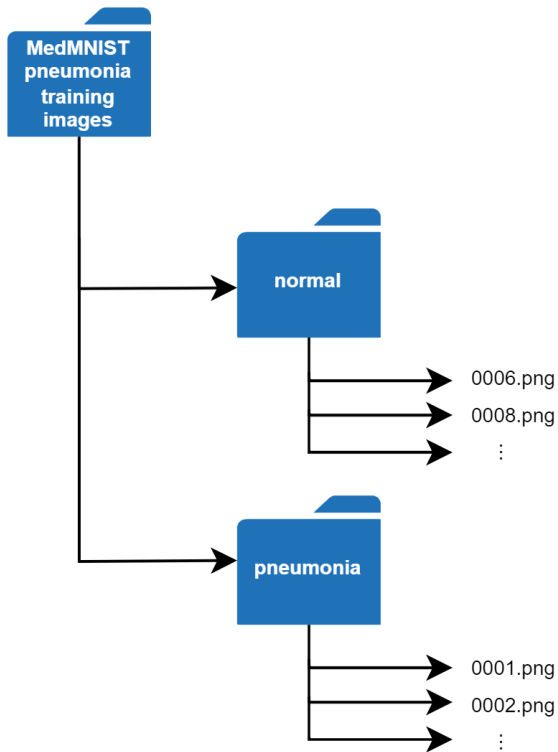
Start by collecting requirements allocated to the ML component



Conveniently manage large collections of data



```
trainingDataFolder = "pneumiamnist\Train";
imdsTrain = imageDatastore(trainingDataFolder, IncludeSubfolders=true, LabelSource="foldernames");
```

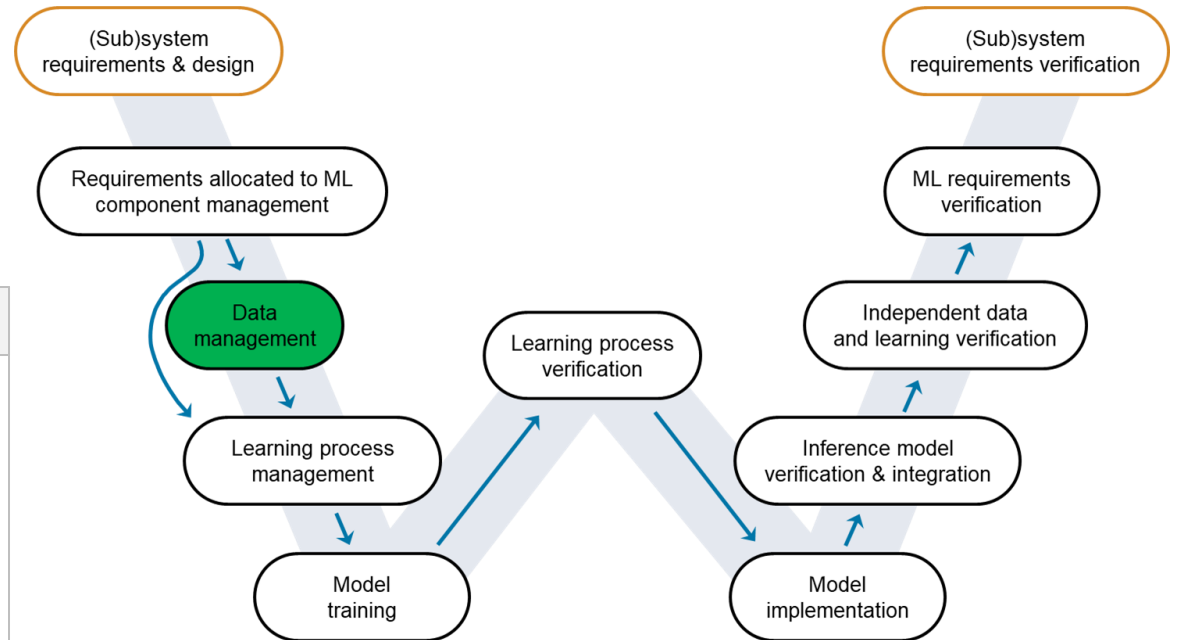


```
countEachLabel(imdsTrain)
```

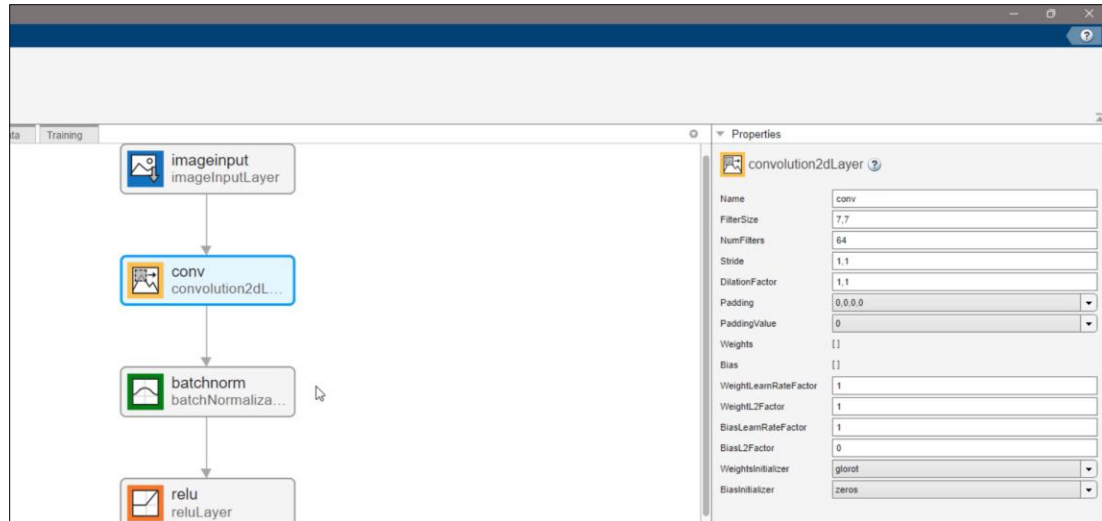
ans =

2x2 [table](#)

Label	Count
normal	1214
pneumonia	3494



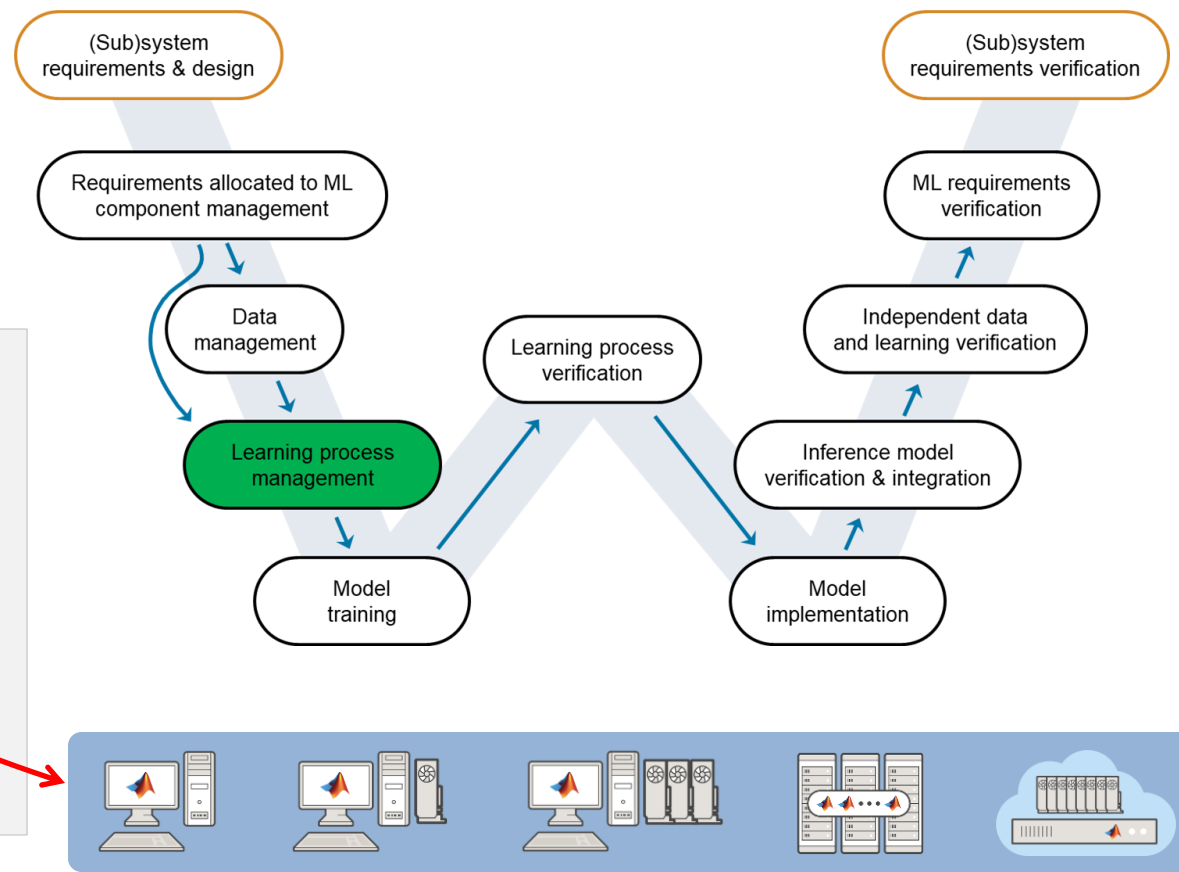
Visually creating networks enables faster design



```

numClasses = numel(classNames);
layers = [
    imageInputLayer(imageSize,Normalization="none")
    convolution2dLayer(7,64,Padding=0)
    batchNormalizationLayer()
    reluLayer()
    dropoutLayer(0.5)
    averagePooling2dLayer(2,Stride=2)
    convolution2dLayer(7,128,Padding=0)
    batchNormalizationLayer()
    reluLayer()
    dropoutLayer(0.5)
    averagePooling2dLayer(2,Stride=2)
    fullyConnectedLayer(numClasses)
    softmaxLayer
    classificationLayer(Classes=classNames,ClassWeights=classWeights)];

options = trainingOptions("adam", ...
    ExecutionEnvironment="auto", ...
    InitialLearnRate=0.001, ...
    MaxEpochs=50, ...
    MiniBatchSize=256, ...
    Shuffle="every-epoch", ...
    LearnRateSchedule="piecewise", ...
    LearnRateDropPeriod=30, ...
    LearnRateDropFactor=0.1, ...
    Plots="training-progress", ...
    ValidationData={XVal,TVal}, ...
    ValidationPatience=10, ...
    OutputNetwork="best-validation-loss");
    
```

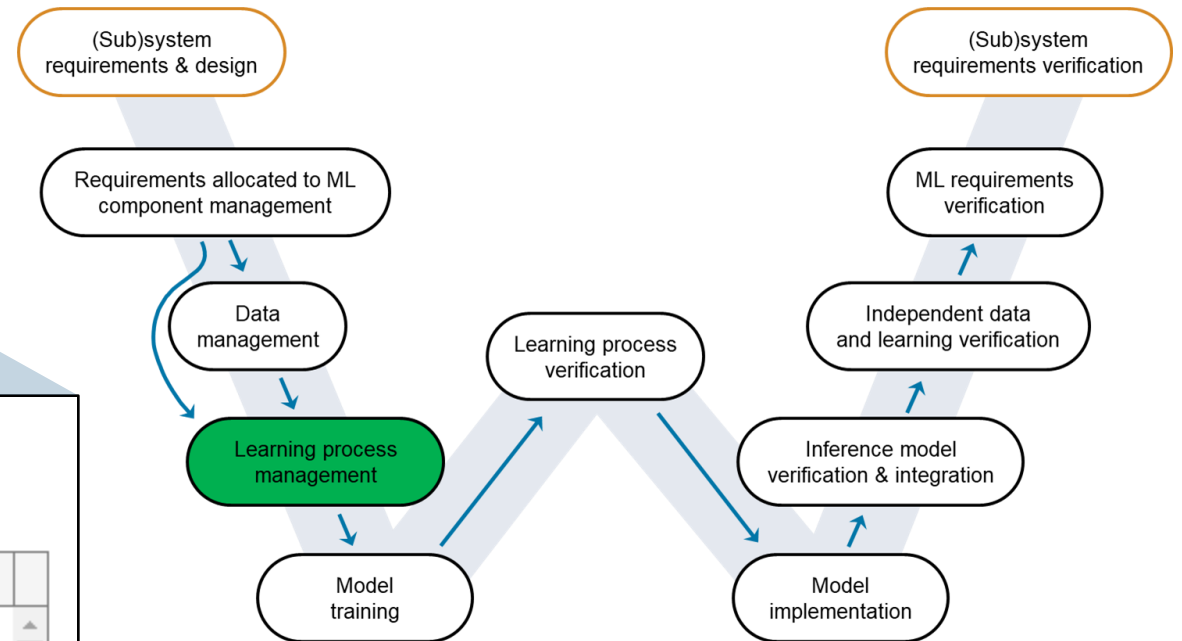


Find optimal paraments and audit experiments for reproducibility

The screenshot shows the MATLAB Experiment Manager interface. The main window displays the configuration for an experiment named "Experiment_pneumonia_CNN". The description is "Image Classification by Parameter Sweeping of Hyperparameters". The hyperparameters are configured using an "Exhaustive Sweep" strategy. The hyperparameters table is as follows:

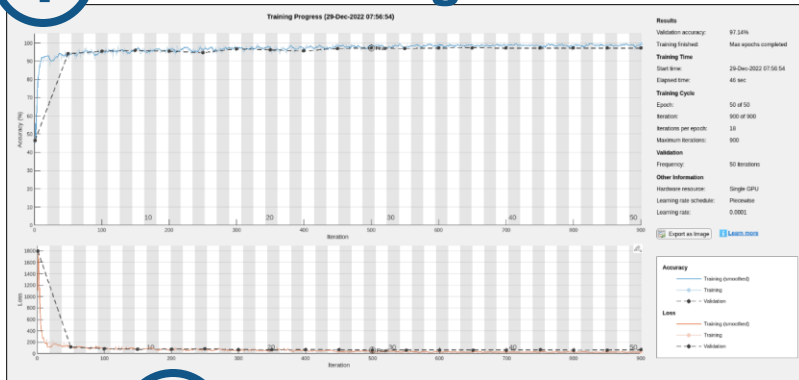
Name	Values
solver	["adam"]
filterSize	[5 7]
numFilters1	[16 32]
numFilters2	[32 64]

The setup function is "W3_W5_Experiment_pneumonia_CNN". A detailed view of the hyperparameters section is shown in a separate box below the main screenshot.



An iterative approach towards building an accurate and robust model

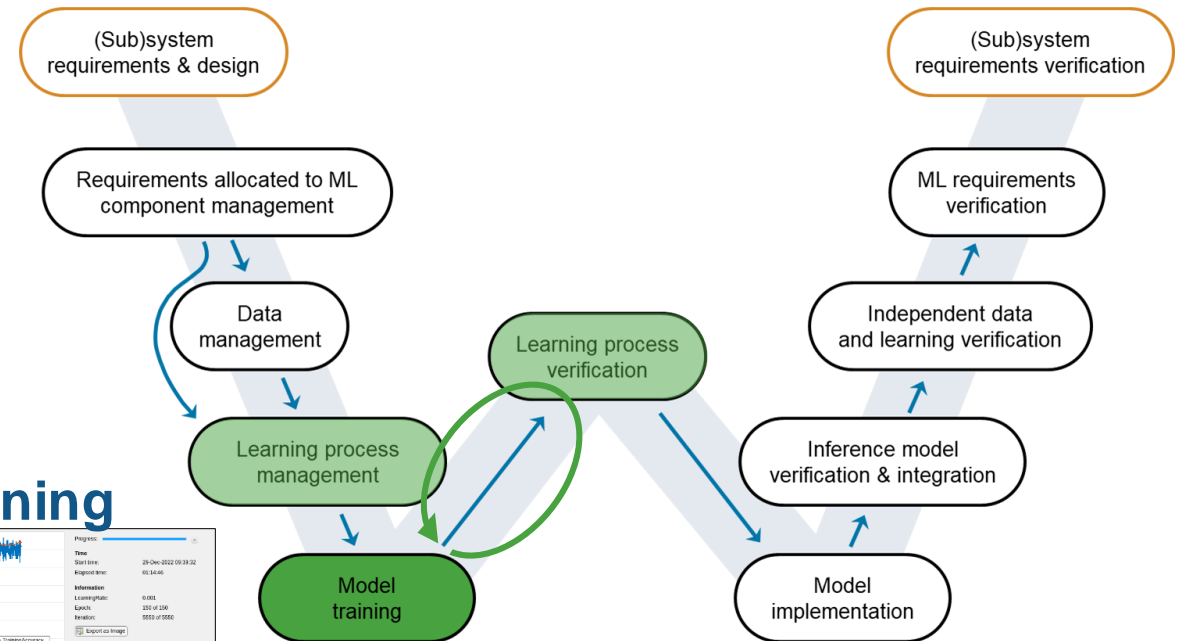
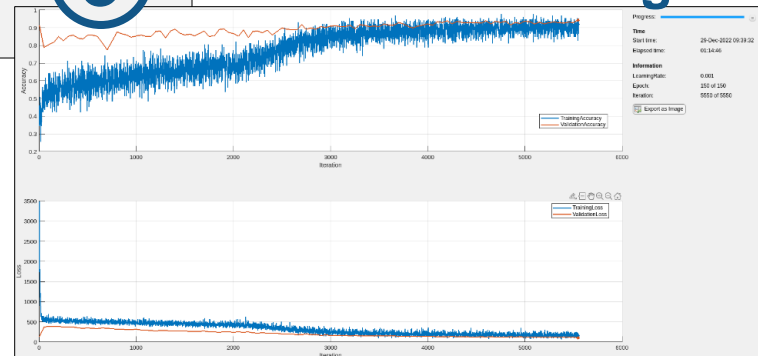
1 Initial training



2 Data-augmented training

```
imageAugmenter = imageDataAugmenter(...
    FillValue=mean(XTrain(:)), ...
    RandXReflection=true, ...
    RandXTranslation=[-2,2], ...
    RandYTranslation=[-2,2], ...
    RandRotation=[-10,10],...
    RandScale=[1,1.25], ...
    RandXShear=[-5,5], ...
    RandYShear=[-5,5]);
```

3 Adversarial training



Adversarial Example

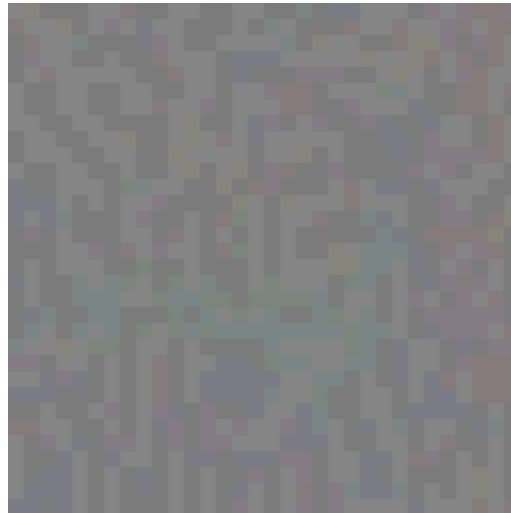
cat



Original image

+

δ



Perturbation

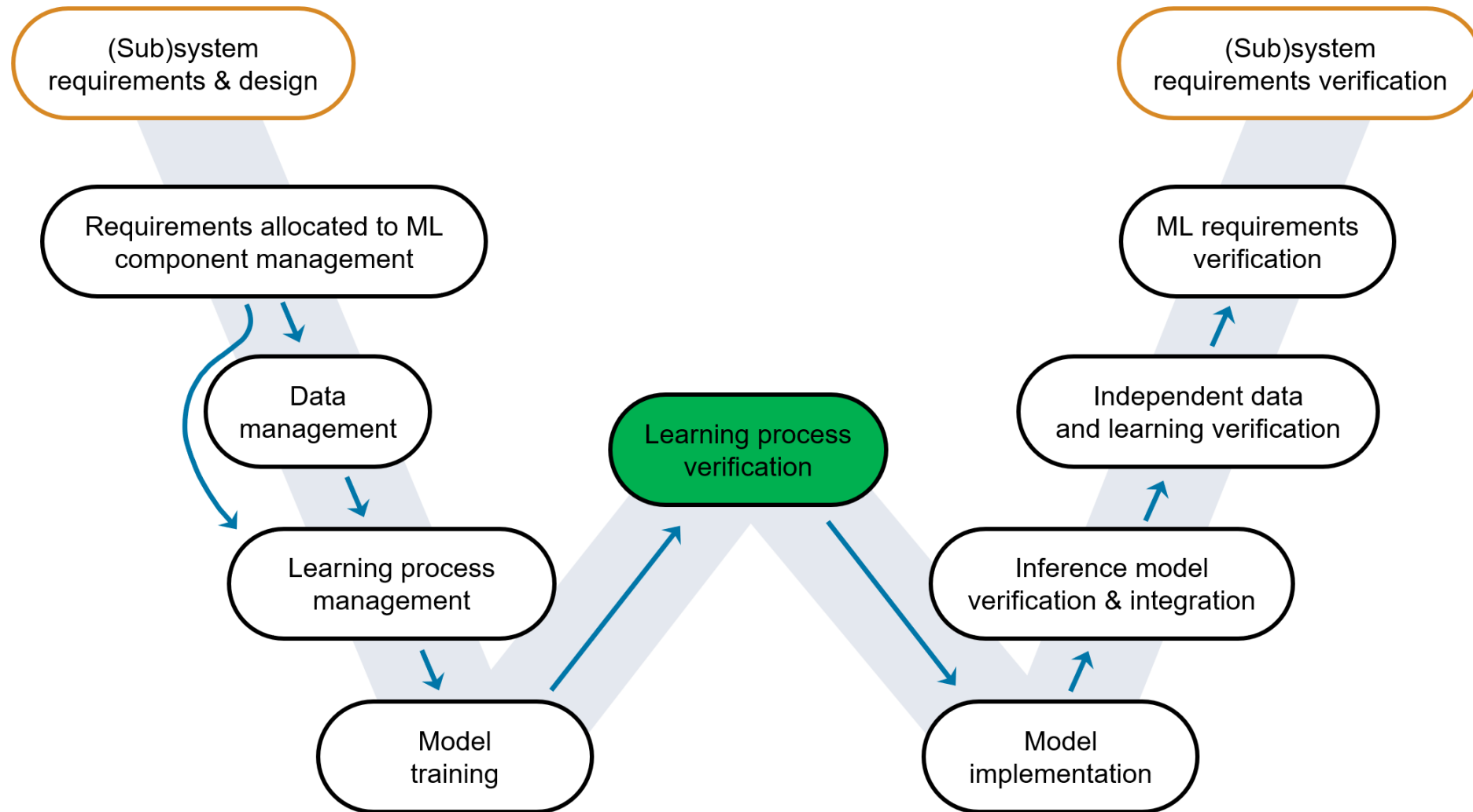
=

cat or deer



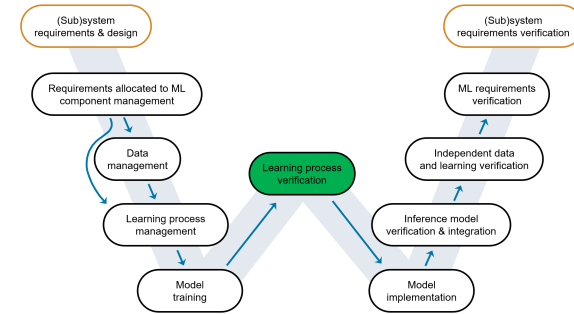
Adversarial image

Learning process verification



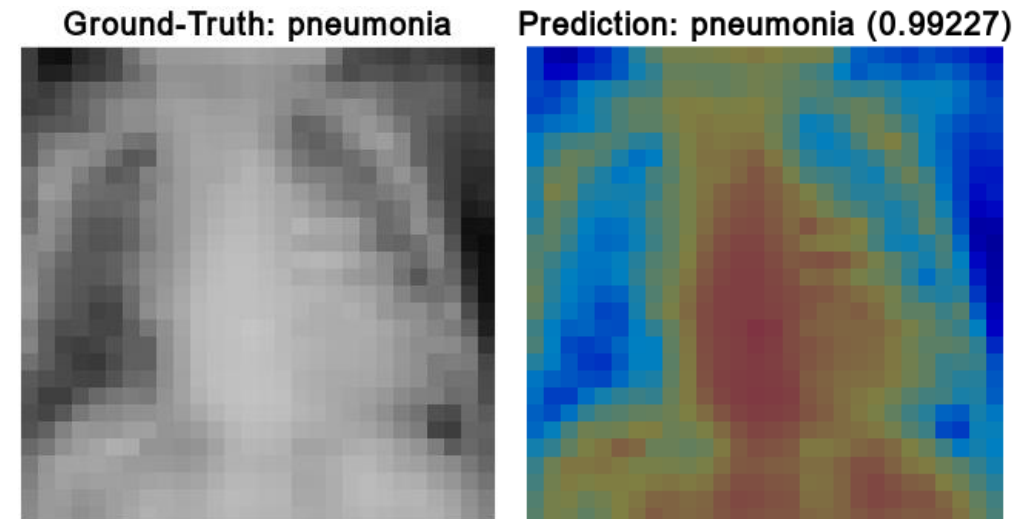
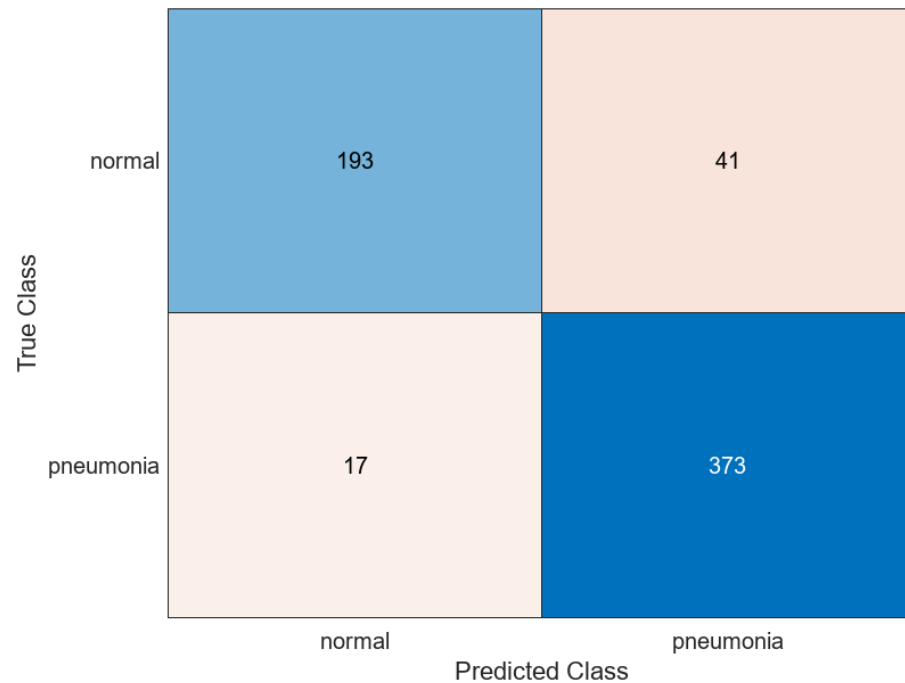
Testing and understanding model performance with an independent test set

Accuracy: 90.71%



```
confusionchart(T,Y)
```

```
scoreMap = gradCAM(net,X,label)
```



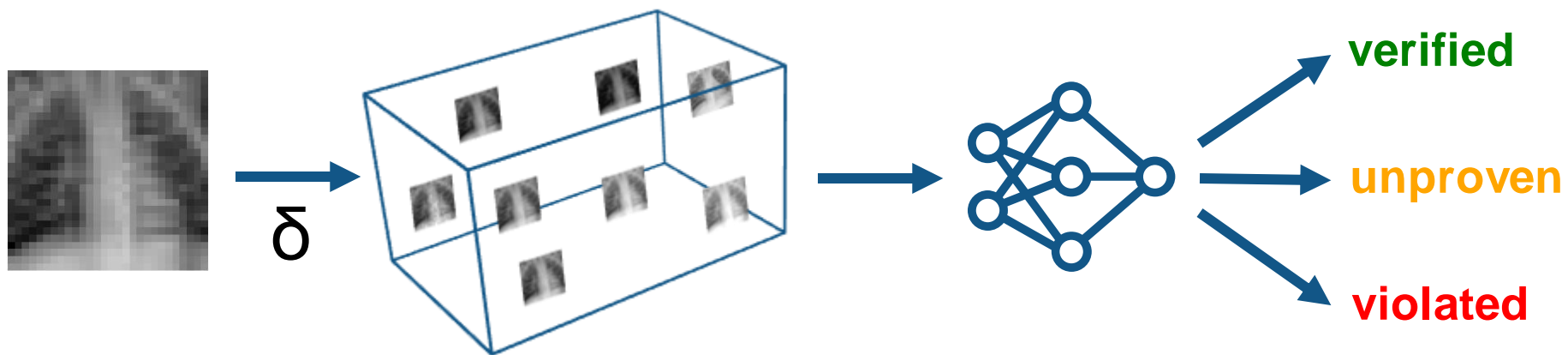
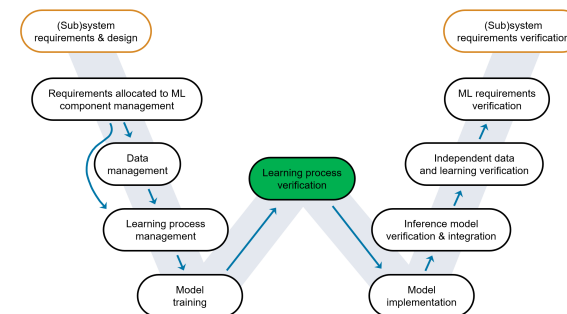
Verify robustness of deep learning networks



Deep Learning Toolbox Verification Library

by MathWorks Deep Learning Toolbox Team **STAFF**

Verify and test robustness of deep learning networks



Formal Verification

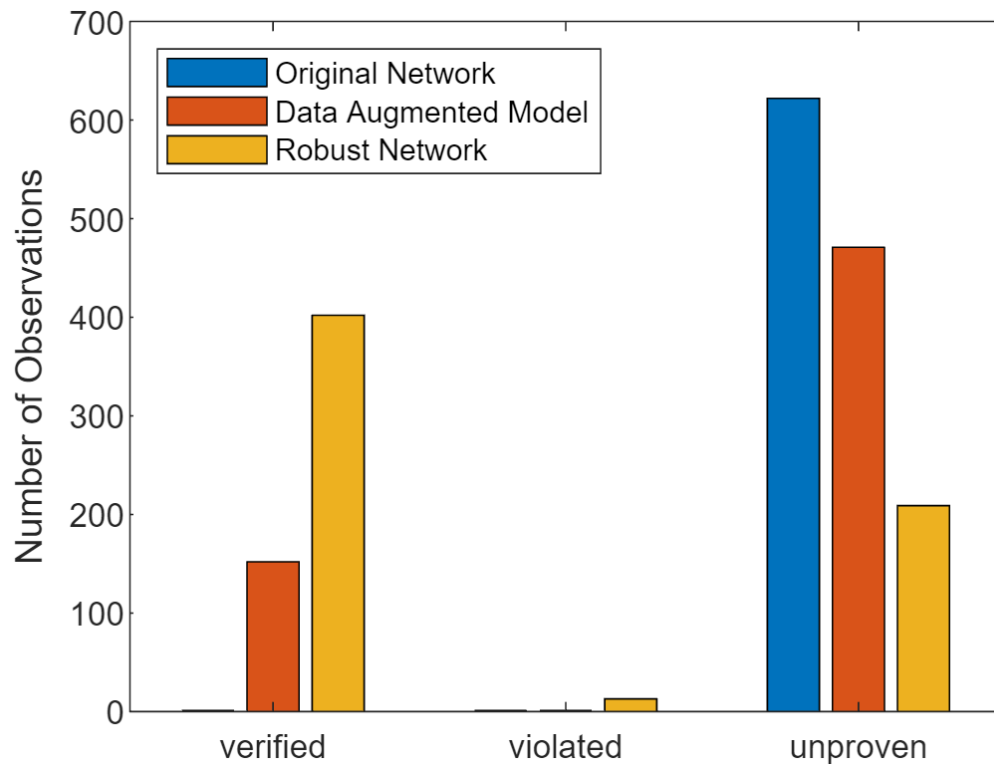
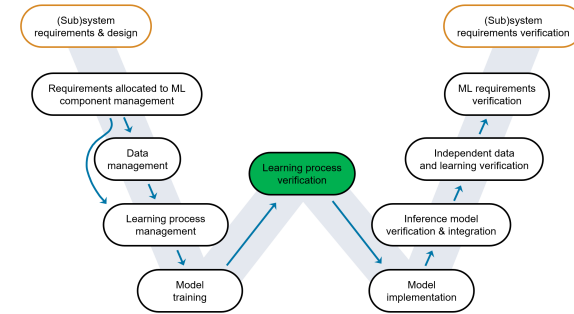
Verify robustness of deep learning networks



Deep Learning Toolbox Verification Library

by MathWorks Deep Learning Toolbox Team **STAFF**

Verify and test robustness of deep learning networks



```

perturbation = 0.01;
XLower = XTest - perturbation;
XUpper = XTest + perturbation;
XLower = dlarray(XLower, "SSCB");
XUpper = dlarray(XUpper, "SSCB");
result = verifyNetworkRobustness(net, ...
    XLower, XUpper, TTest);
    
```

summary(result)	
verified	402
violated	13
unproven	209

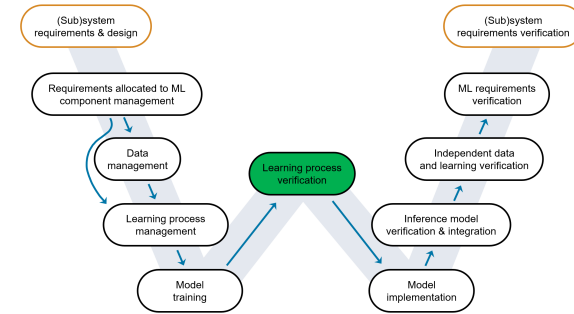
Identify unknown examples to the model and reject or transfer to a human for safe handling



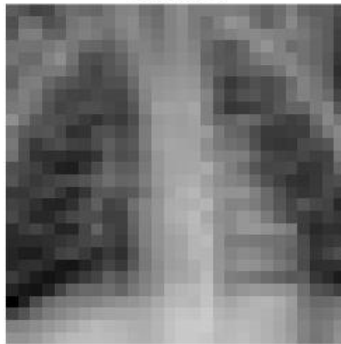
Deep Learning Toolbox Verification Library

by MathWorks Deep Learning Toolbox Team **STAFF**

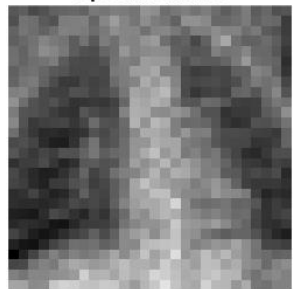
Verify and test robustness of deep learning networks



Original

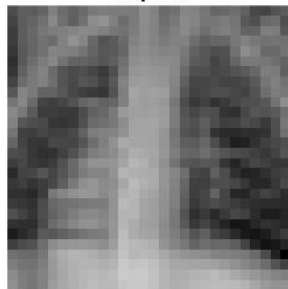


SpeckleNoise



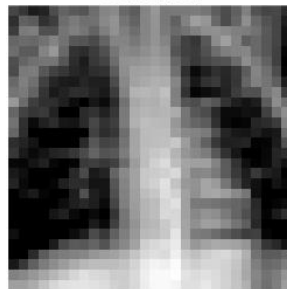
In-distribution

FlipLR

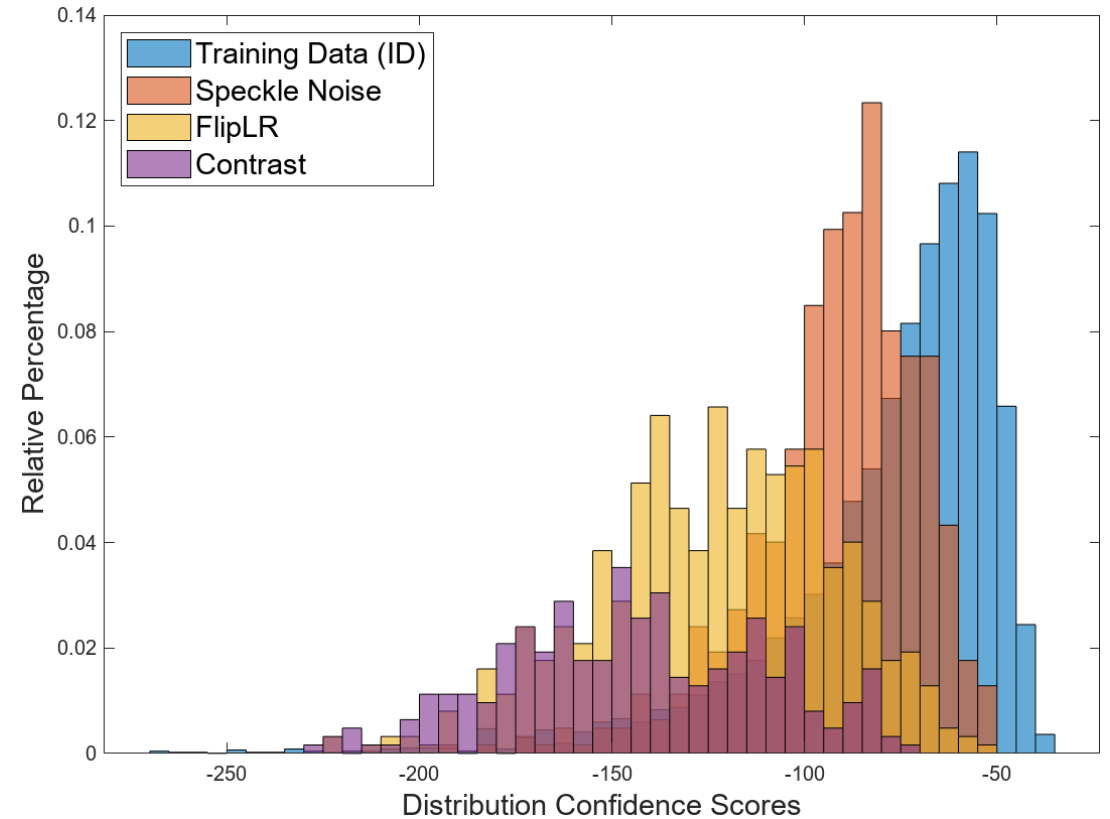


Out-of-distribution

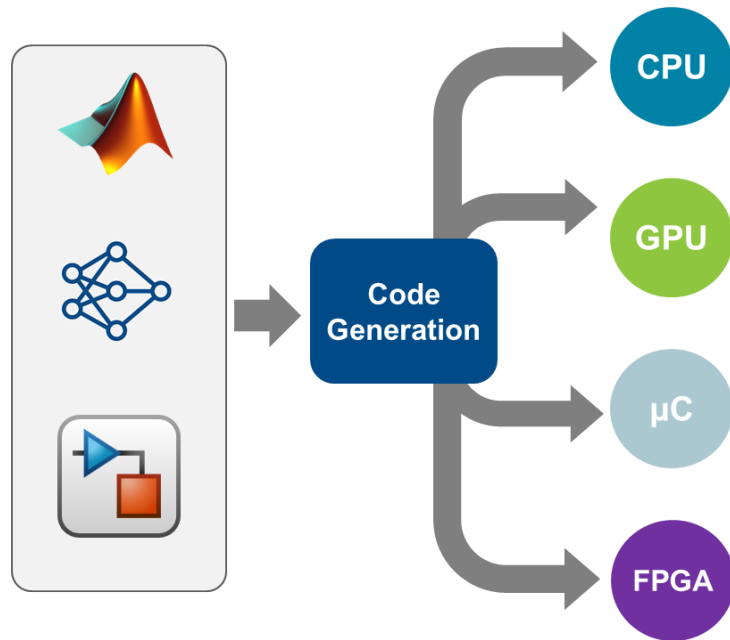
Contrast



Out-of-distribution

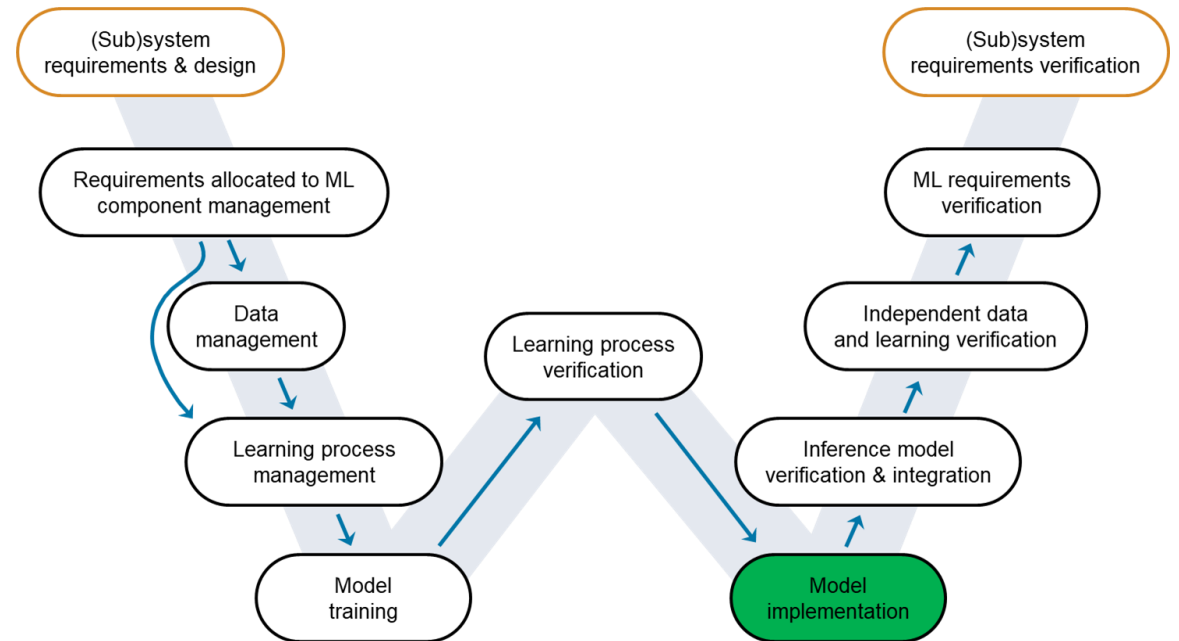


Deploy to target with zero coding errors

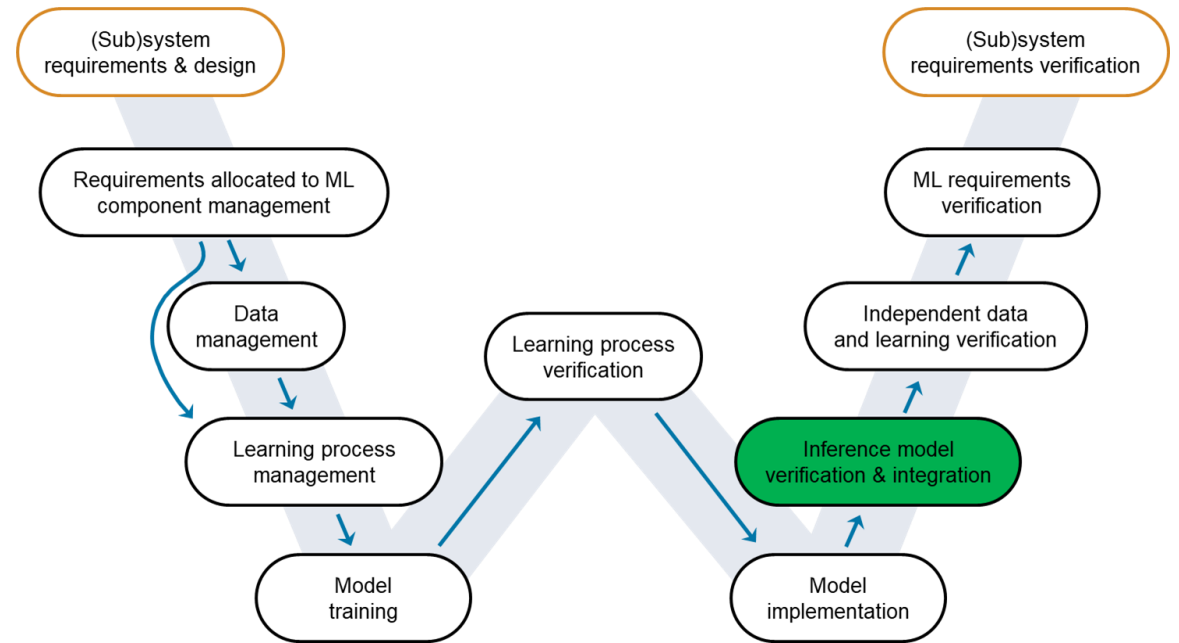
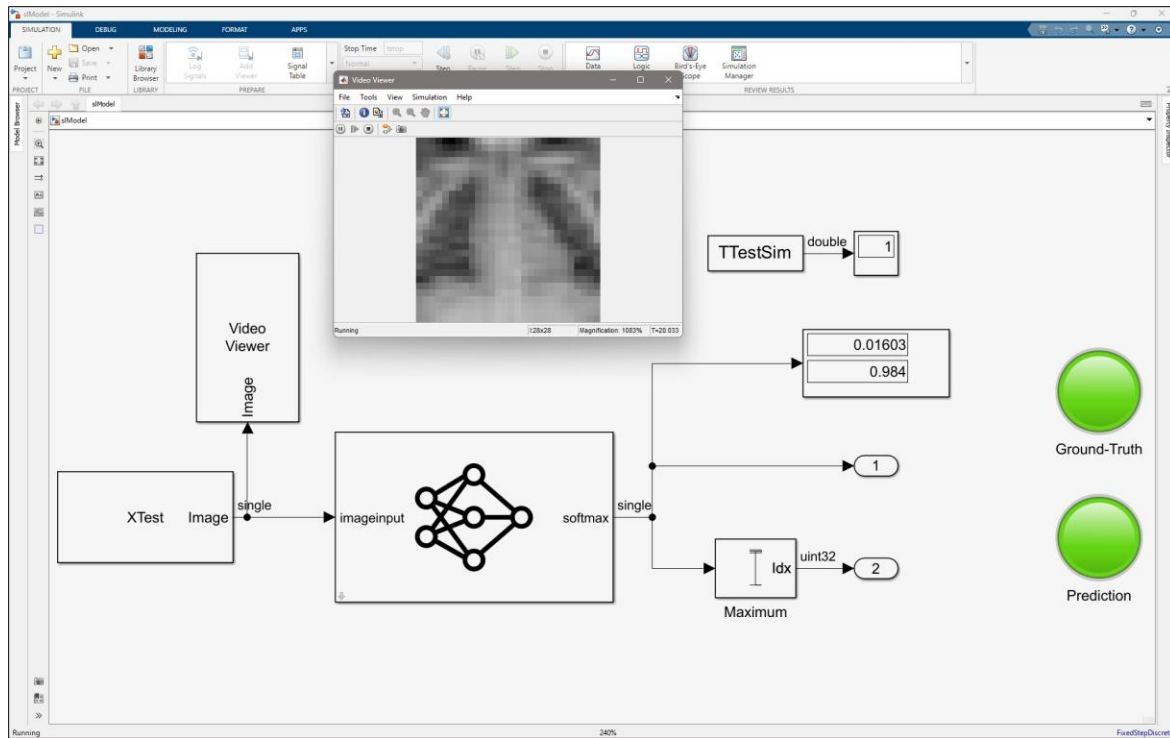


`analyzeNetworkForCodegen(net)`

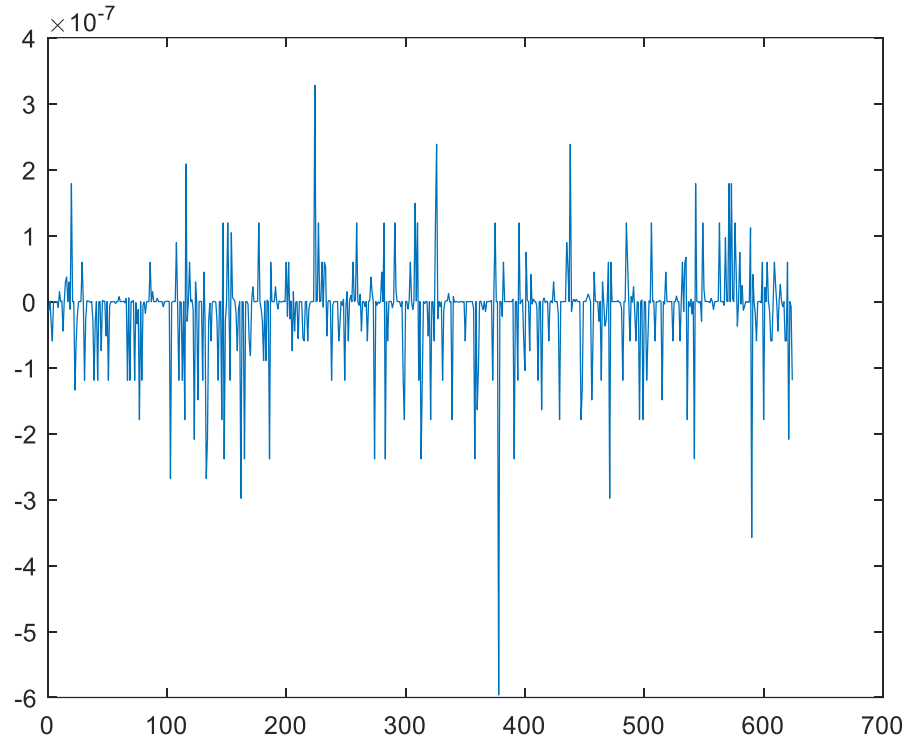
	Supported
none	"Yes"
arm-compute	"Yes"
mkldnn	"Yes"
cuda	"Yes"
tensorrt	"Yes"



Integrate your AI model in Simulink for system-level simulation and test

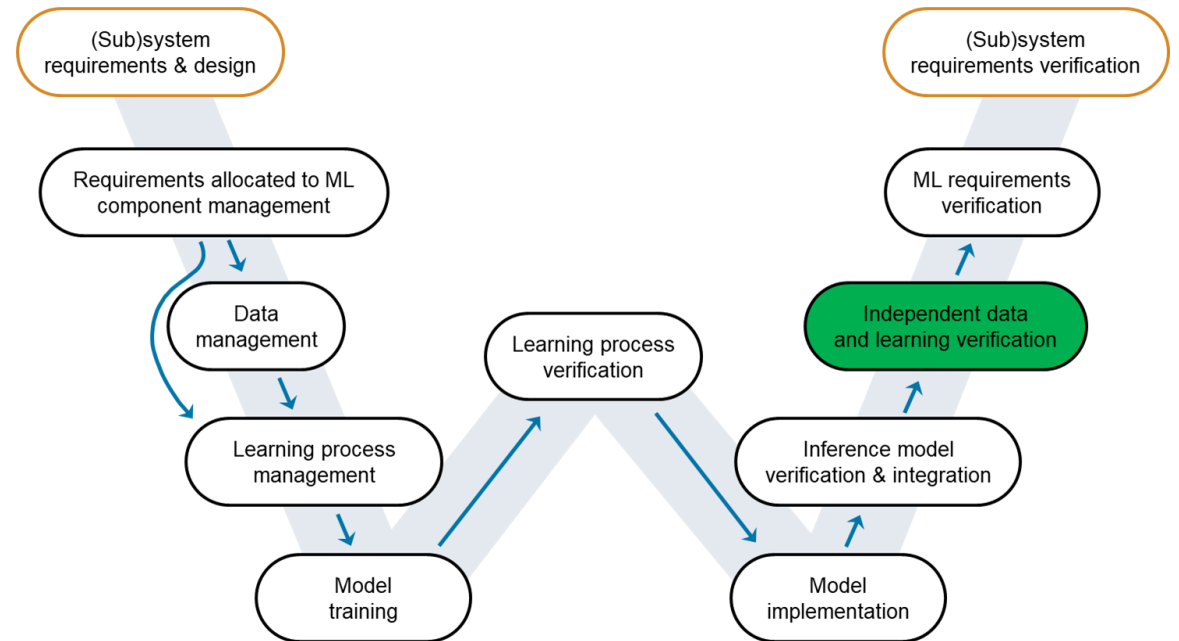


No differences between development and inference models



```
max(abs(differences))
```

```
ans = single  
5.9605e-07
```



Verifying requirements have been fully tested

MATLAB Test Manager: All Tests in Current Project

16 Total Tests
 13 Passed
 0 Failed
 0 Incomplete
 0 Not Run

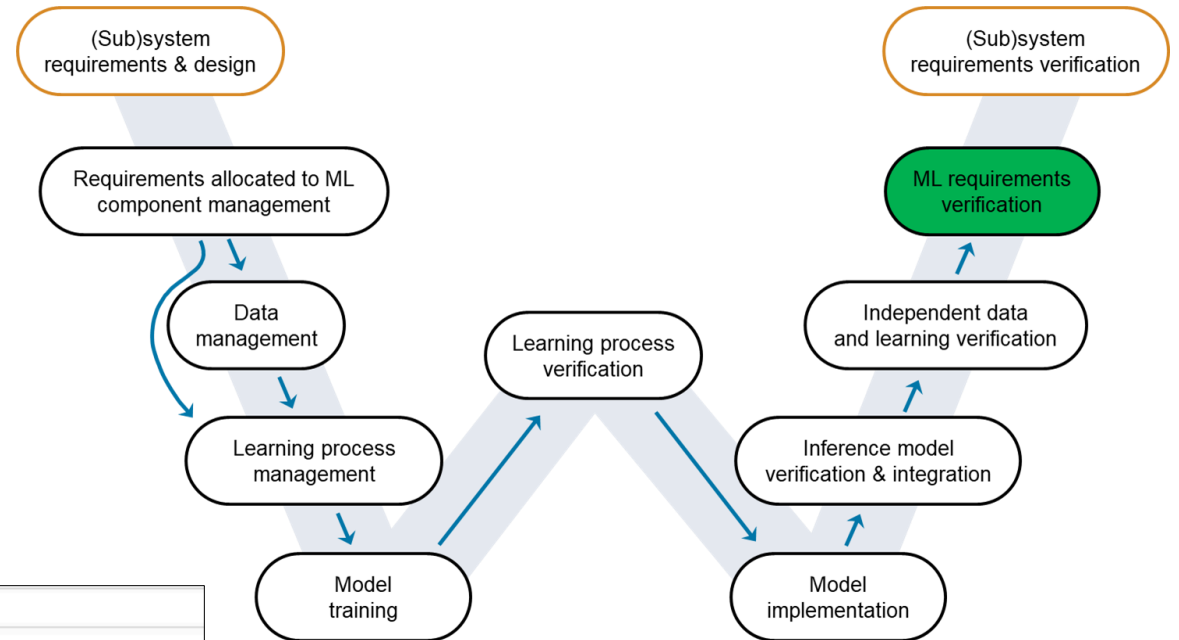
Running tests... 13/16

Requirements Editor

Index	Summary	Implemented	Verified
1	ML component requirement for X-Ray Pneumonia Detector (XRPD)	Implemented	Verified
1.1	Introduction	Implemented	Verified
1.2	ML component description	Implemented	Verified
1.3	ML component requirements	Implemented	Verified
1.3.1	ML component input	Implemented	Verified
1.3.1.1	ML component input should be 28x28x1	Implemented	Verified
1.3.1.2	ML component input data (training) should be 28x28x1	Implemented	Verified
1.3.1.3	ML component input data (validation) should be 28x28x1	Implemented	Verified
1.3.1.4	ML component input data (test) should be 28x28x1	Implemented	Verified
1.3.2	ML component output	Implemented	Verified
1.3.2.1	ML component output should be 2	Implemented	Verified
1.3.2.2	ML component output labels should be 'normal' or 'pneumonia'	Implemented	Verified
1.3.3	ML component accuracy	Implemented	Verified
1.3.3.1	ML component training precision	Implemented	Verified
1.3.3.2	ML component test precision	Implemented	Verified
1.3.3.3	ML component avoid overfitting	Implemented	Verified
1.3.3.4	ML component out-of-distribution detection	Implemented	Verified
1.3.4	ML component latency	Implemented	Verified
1.3.5	ML component robustness	Implemented	Verified
1.3.5.1	ML component robustness 1% perturbation	Implemented	Verified
1.3.5.2	ML component robustness 0.5% perturbation	Implemented	Verified
1.3.5.3	ML component robustness 0.1% perturbation	Implemented	Verified
1.3.6	ML component implementation	Implemented	Verified

Links

- Implemented by: [738897.723.1 in evaluateModelAccuracy.m](#)
- Refines: [XRPD_ML_3 ML component accuracy](#)
- Verified by: [738897.723.2 in MLComponent_Accuracy.m](#) ✓

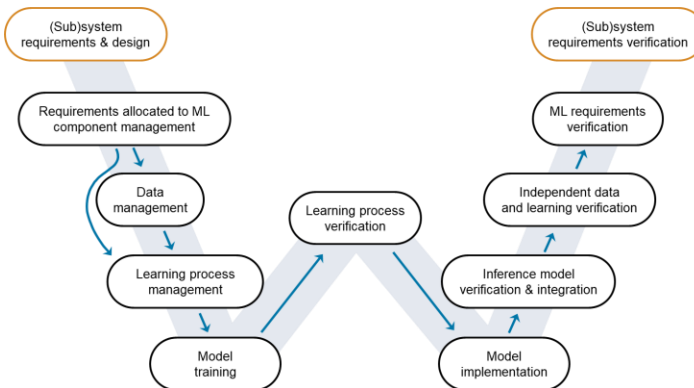


Key Takeaways

MathWorks has capabilities addressing each area of the W-diagram

Library to verify and test robustness of deep learning networks

Our safety-critical certification expertise helps drive new AI standards



Deep Learning Toolbox Verification Library

by MathWorks Deep Learning Toolbox Team **STAFF**

Verify and test robustness of deep learning networks



EUROCAE WG-114 / SAE G-34
Standardization Working Group
“Artificial Intelligence in Aviation”

MATLAB EXPO

Thank you



© 2023 The MathWorks, Inc. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See [mathworks.com/trademarks](https://www.mathworks.com/trademarks) for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.