MATLAB EXPO

AI기반 가상센서를 이용한 모델기반 설계

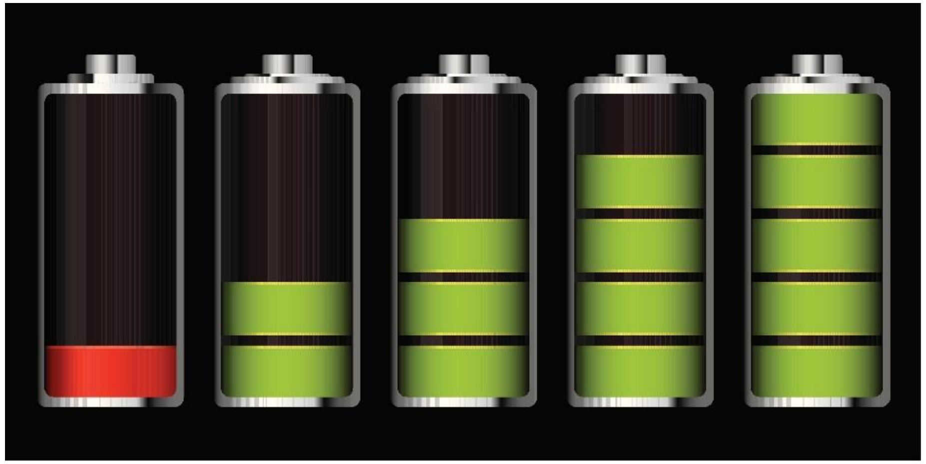
신행재 부장, 매스웍스코리아

MathWorks

Why Virtual Sensors?

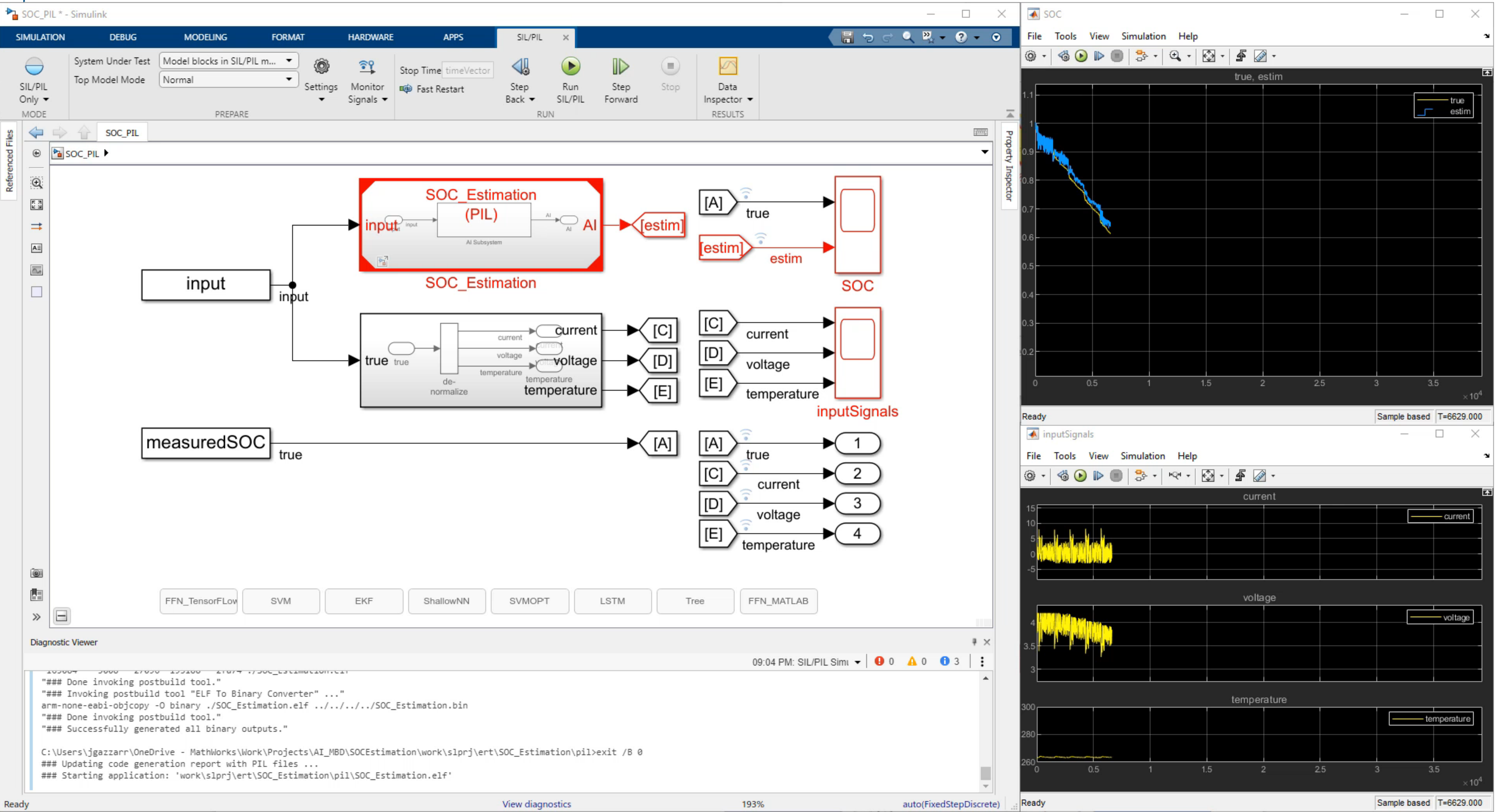
When estimating a quantity that is not measurable

Battery State of Charge (SOC)

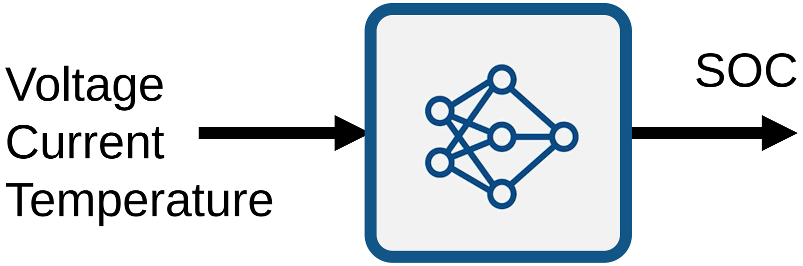


Not directly measurable

We <u>measure</u> voltage, current, temperature and <u>calculate</u> SOC


SOC_PIL * - Simulink
SIMULATION
DEBUG
MODELING
FORMAT
HARDWARE
APPS
SIL/PIL
SIL/PIL Only
MODE
System Under Test
Model blocks in SIL/PIL m...
Top Model Mode
Normal
Settings
Monitor Signals
PREPARE
Stop Time
timeVector
Fast Restart
Step Back
Run SIL/PIL
Step Forward
Stop
RUN
Data Inspector
RESULTS
SOC_PIL
SOC_Estimation
(PIL)
input
AI
AI Subsystem
[estim]
input
true
current
voltage
temperature
de-normalize
[C]
[D]
[E]
measuredSOC
true
[A]
true
estim
SOC
current
voltage
temperature
inputSignals
1
2
3
4
FFN_TensorFLow
SVM
EKF
ShallowNN
SVMOPT
LSTM
Tree
FFN_MATLAB
Diagnostic Viewer
09:04 PM: SIL/PIL Simu
"### Done invoking postbuild tool."
"### Invoking postbuild tool "ELF To Binary Converter" ..."
arm-none-eabi-objcopy -O binary ./SOC_Estimation.elf ../../../../SOC_Estimation.bin
"### Done invoking postbuild tool."
"### Successfully generated all binary outputs."
C:\Users\jgazzarr\OneDrive - MathWorks\Work\Projects\AI_MBD\SOCEstimation\work\slprj\ert\SOC_Estimation\pil>exit /B 0
Updating code generation report with PIL files ...
Starting application: 'work\slprj\ert\SOC_Estimation\pil\SOC_Estimation.elf'
Ready
View diagnostics
193%
auto(FixedStepDiscrete)
SOC
File
Tools
View
Simulation
Help
true, estim
true
estim
Ready
Sample based
T=6629.000
inputSignals
current
voltage
temperature
Ready
Sample based
T=6629.000

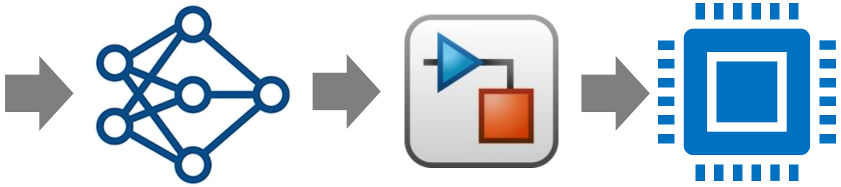
Agenda



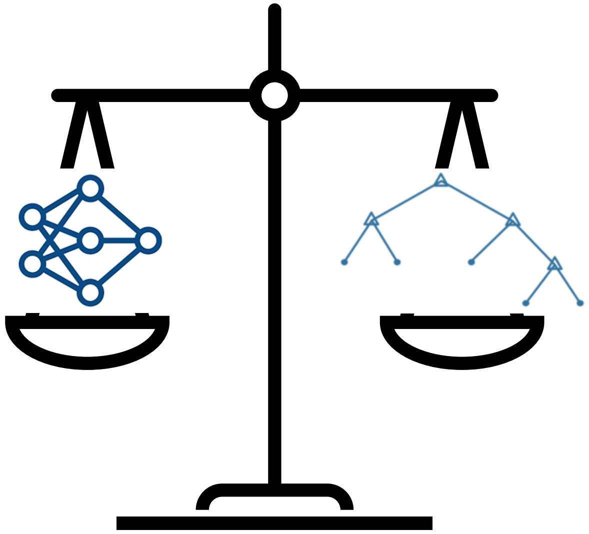
- Develop AI-based virtual sensor for battery SOC estimation

- Workflow - From data acquisition to hardware deployment

Voltage	Current	Temperature
0.7510	0.3851	0.3031
0.7510	0.3852	0.3046
0.7510	0.3852	0.3061
0.7510	0.3852	0.3076
0.7510	0.3852	0.3091



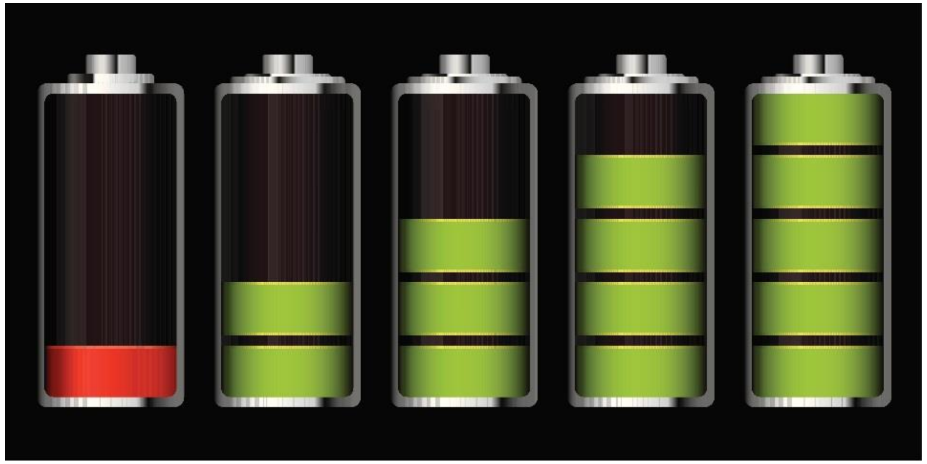
- Compare different AI methods



Battery State of Charge (SOC)

$$SOC(t) = \frac{1}{C}\int_0^t I(p)dp$$

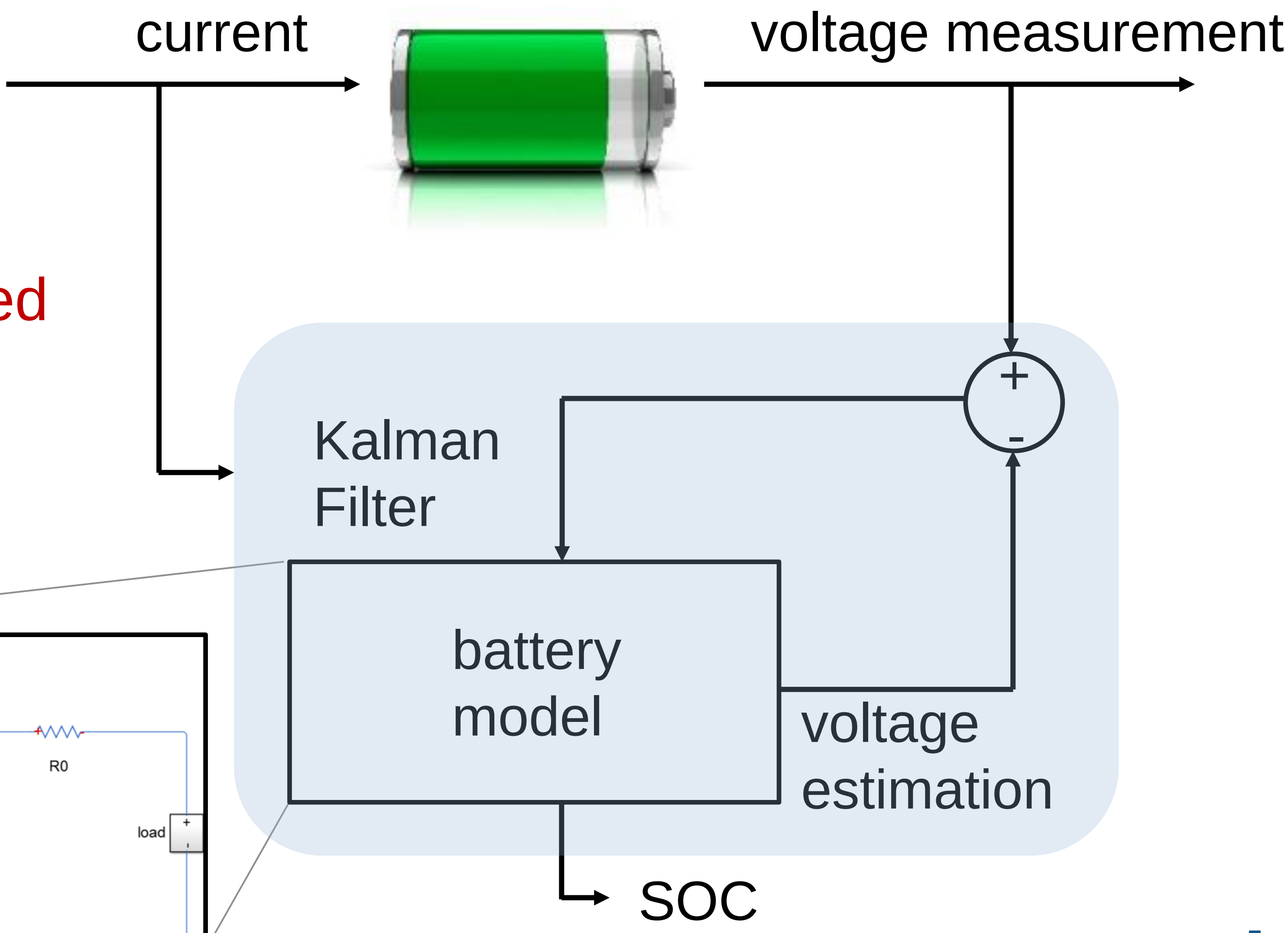
capacity *current*

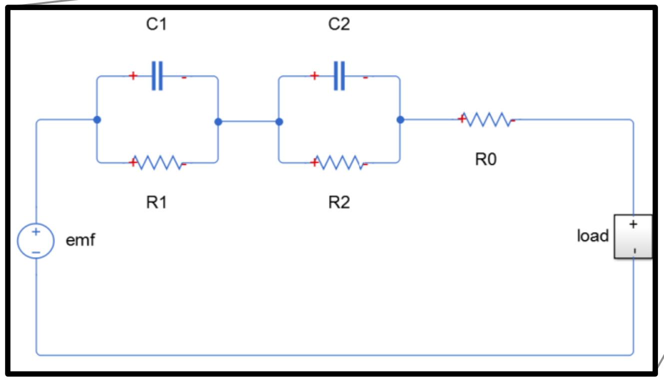


Affected by sensor error

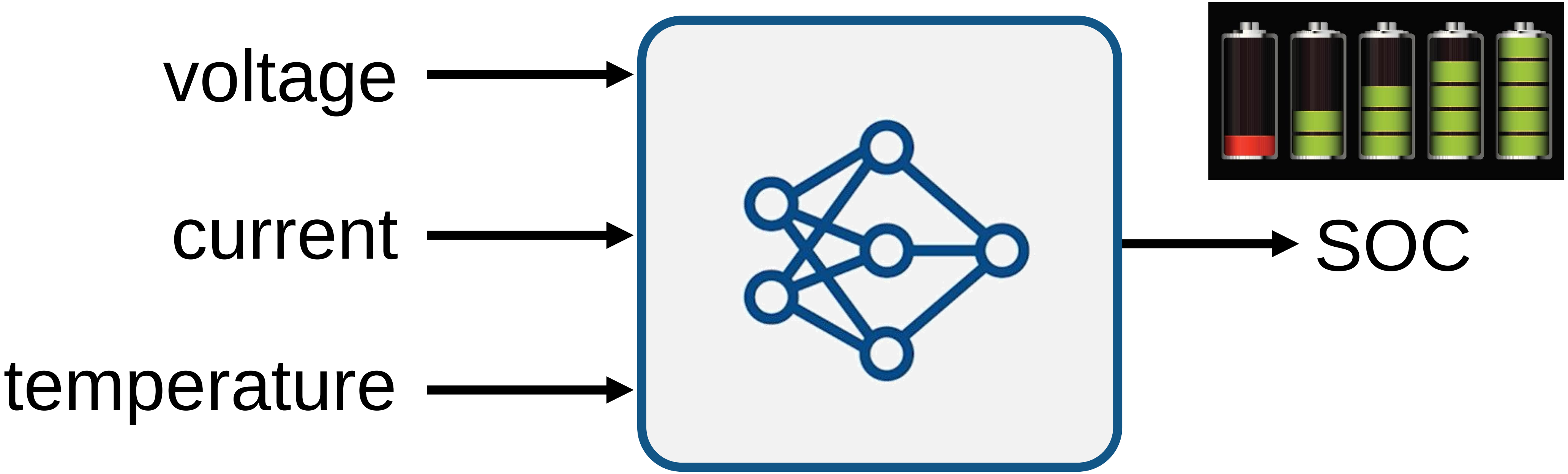
Extended Kalman Filter

- Well established
- Accurate
- Detailed battery model required
  - Operating condition range
- Computationally intensive





How About…



Instead of creating a physics-based model –
Train a Statistical Model

Comparison

Extended Kalman Filter

- Well established
- Accurate
- Detailed battery model required
  - Operating condition range
- Computationally intensive

AI

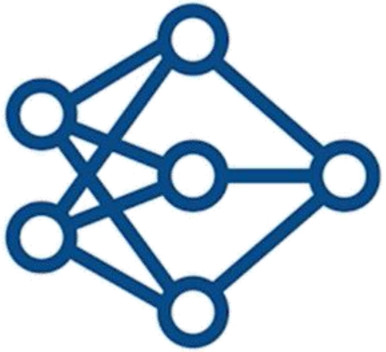
- Training on real data
- Capture very complex data relationships
- No need for battery model
- Interpretability
- Computationally intensive

AI-driven System Design

Data Preparation

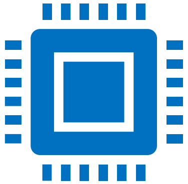
Voltage	Current	Temperature
0.7510	0.3851	0.3031
0.7510	0.3852	0.3046
0.7510	0.3852	0.3061
0.7510	0.3852	0.3076
0.7510	0.3852	0.3091

AI Modeling



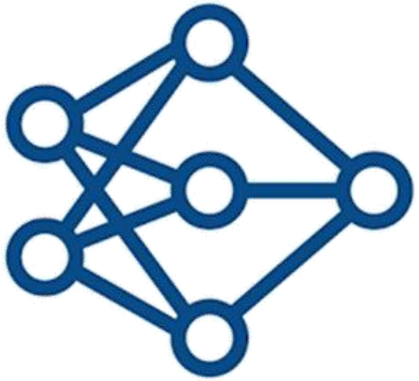
Simulation & Test

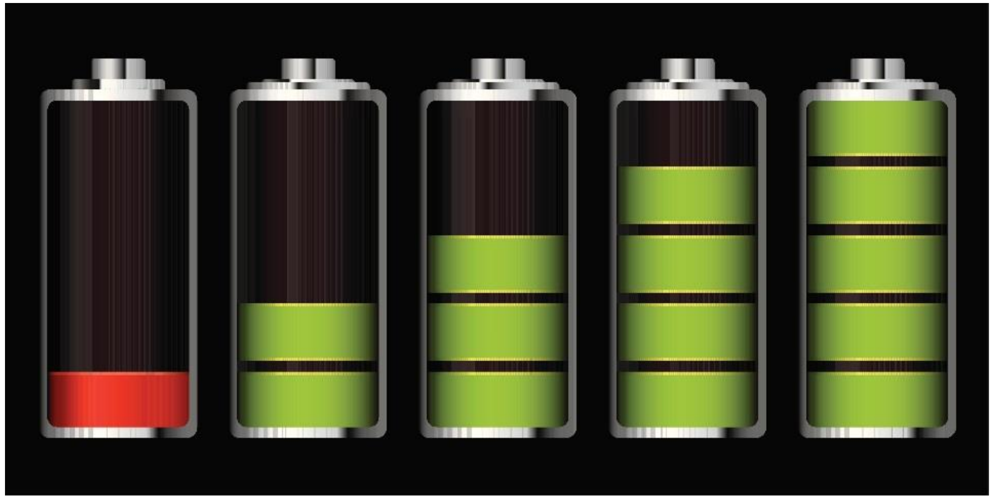
Deployment



Steps involved in creating an AI-based virtual sensor

Back to SOC estimation





2020-01-1181 Published 14 Apr 2020

Robust xEV Battery State-of-Charge Estimator Design Using a Feedforward Deep Neural Network

Carlos Vidal, Phillip Kollmeyer, and Mina Naguib McMaster Automotive Res. Centre

Pawel Malysz and Oliver Gross FCA US LLC

Ali Emadi McMaster University

Citation: Vidal, C., Kollmeyer, P., Naguib, M., Malysz, P. et al., "Robust xEV Battery State-of-Charge Estimator Design Using a Feedforward Deep Neural Network," SAE Technical Paper 2020-01-1181, 2020, doi:10.4271/2020-01-1181.

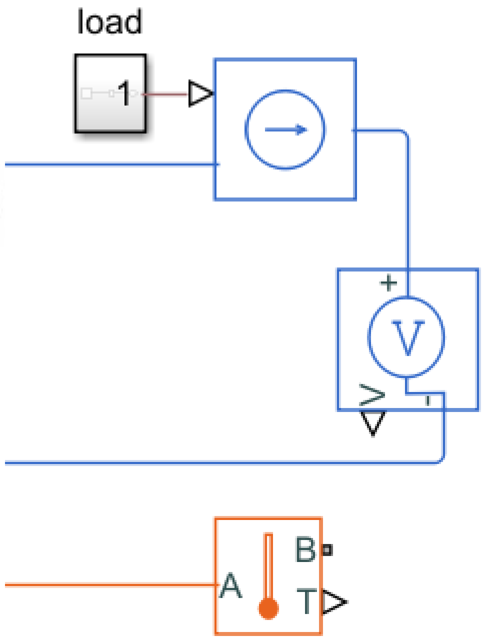
Abstract

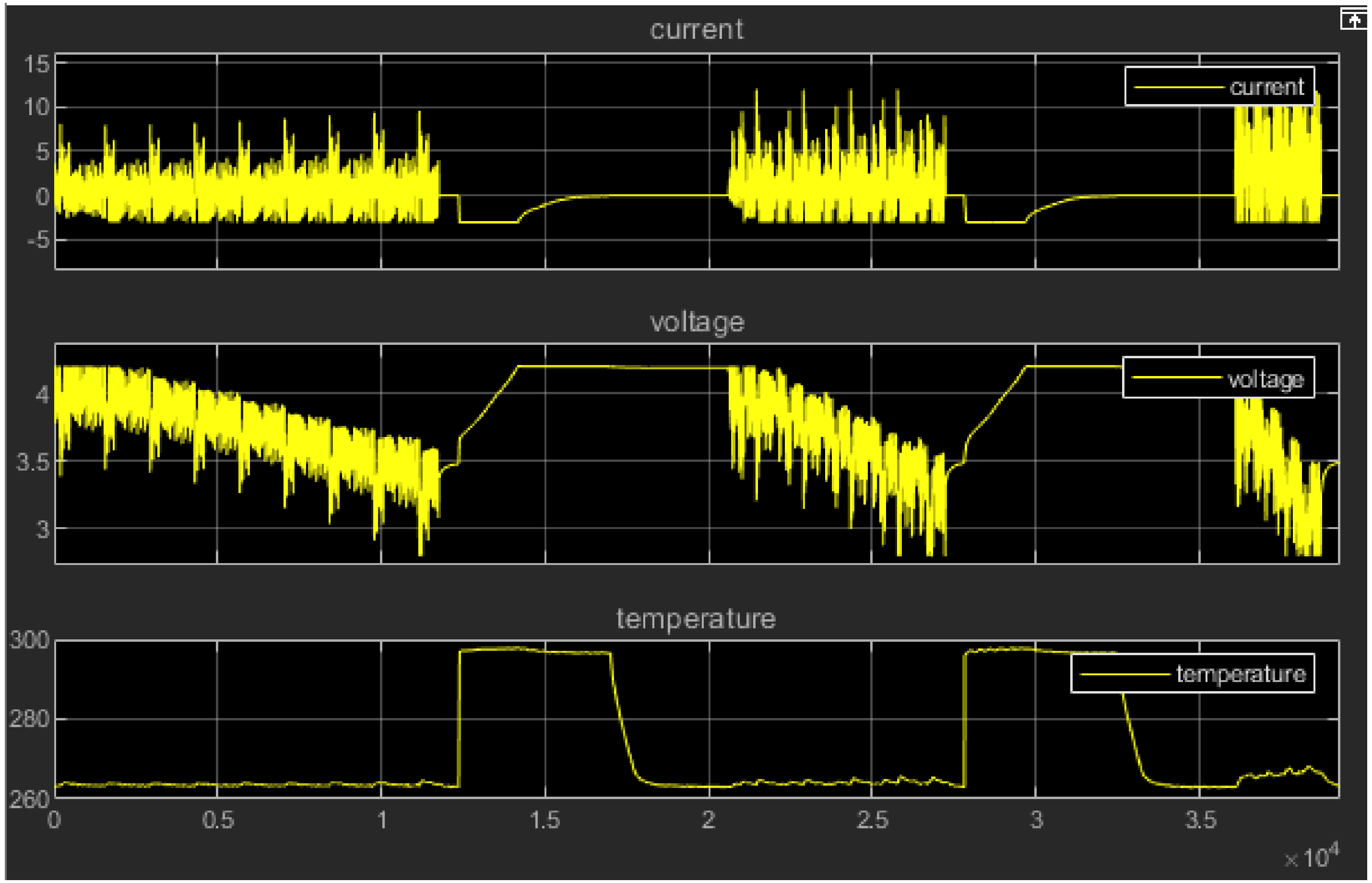
Battery state-of-charge (SOC) is critical information for the vehicle energy management system and must be accurately estimated to ensure reliable and affordable electrified vehicles (xEV). However, due to the nonlinear temperature, health, and SOC dependent behaviour of Li-ion

(FNN) approach. The method includes a description of data acquisition, data preparation, development of an FNN, FNN tuning, and robust validation of the FNN to sensor noise. To develop a robust estimator, the FNN was exposed, during training, to datasets with errors intentionally added to the data, e.g. adding cell voltage variation of ±4mV, cell current

Voltage	Current	Temperature
0.7510	0.3851	0.3031
0.7510	0.3852	0.3046
0.7510	0.3852	0.3061
0.7510	0.3852	0.3076
0.7510	0.3852	0.3091

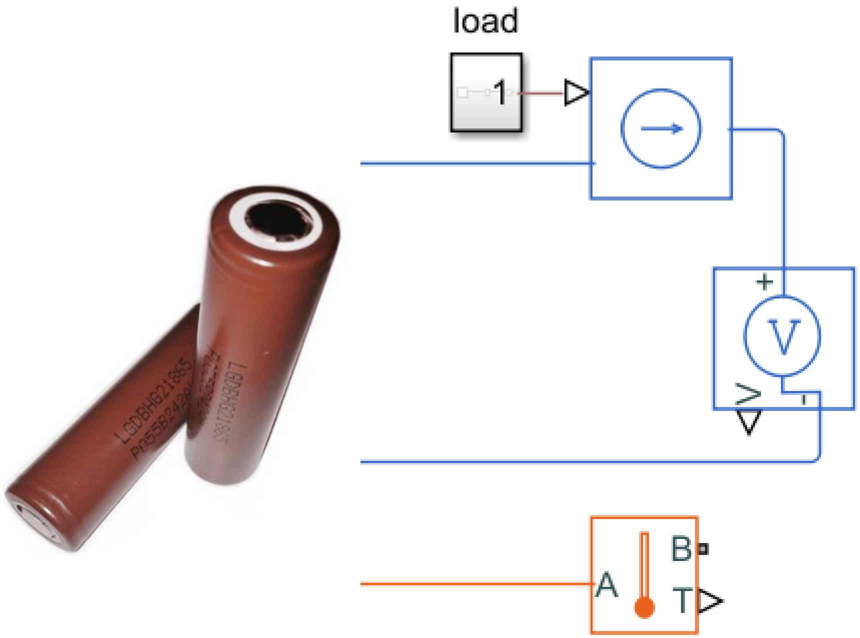
Read data



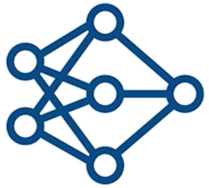


Read data

<u>Data source</u>: McMaster University*



Inputs
Voltage
Current
Temperature
Moving av. voltage
Moving av. current



Outputs
Battery SOC

Predictors

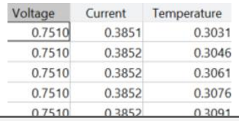
Response

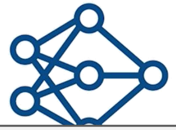
669956x6 table

	1 Voltage	2 Current	3 Temperature	4 Moving Average Voltage	5 Moving Average Current	6 SOC
1	0.7510	0.3851	0.3031	0.7510	0.3851	0.2064
2	0.7510	0.3852	0.3046	0.7510	0.3851	0.2064
3	0.7510	0.3852	0.3061	0.7510	0.3852	0.2064
4	0.7510	0.3852	0.3076	0.7510	0.3852	0.2064
5	0.7510	0.3852	0.3091	0.7510	0.3852	0.2064
6	0.7510	0.3852	0.3106	0.7510	0.3852	0.2064
7	0.7510	0.3852	0.3120	0.7510	0.3852	0.2064
8	0.7510	0.3852	0.3135	0.7510	0.3852	0.2064
9	0.7510	0.3852	0.3150	0.7510	0.3852	0.2064
10	0.7510	0.3852	0.3165	0.7510	0.3852	0.2064

*https://data.mendeley.com/datasets/cp3473x7xv/3

Data Preparation | **AI Modeling** | Simulation & Test | Deployment





Algorithms for AI

Algorithms

Machine learning
Trees, Naïve Bayes, SVM…

Deep learning
CNNs, GANs, LSTM, MIMO…

Reinforcement learning
DQN, A2C, DDPG…

Regression
Linear, nonlinear, trees…

Unsupervised learning
K-means, PCA, GMM…

Predictive maintenance
RUL models, condition indicators…

Bayesian optimization

Pre-built models

Image classification models
AlexNet, GoogLeNet, VGG, SqueezeNet, ShuffleNet, ResNet, DenseNet, Inception…

Reference examples

Object detection
Vehicles, pedestrians, faces…

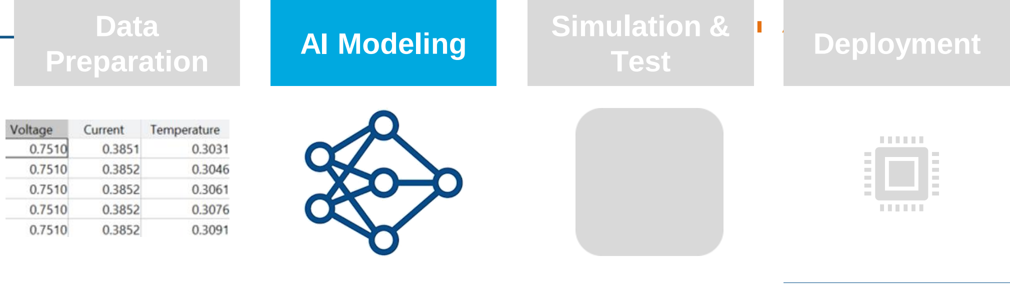
Semantic segmentation
Roadway detection, land cover classification, tumor detection…

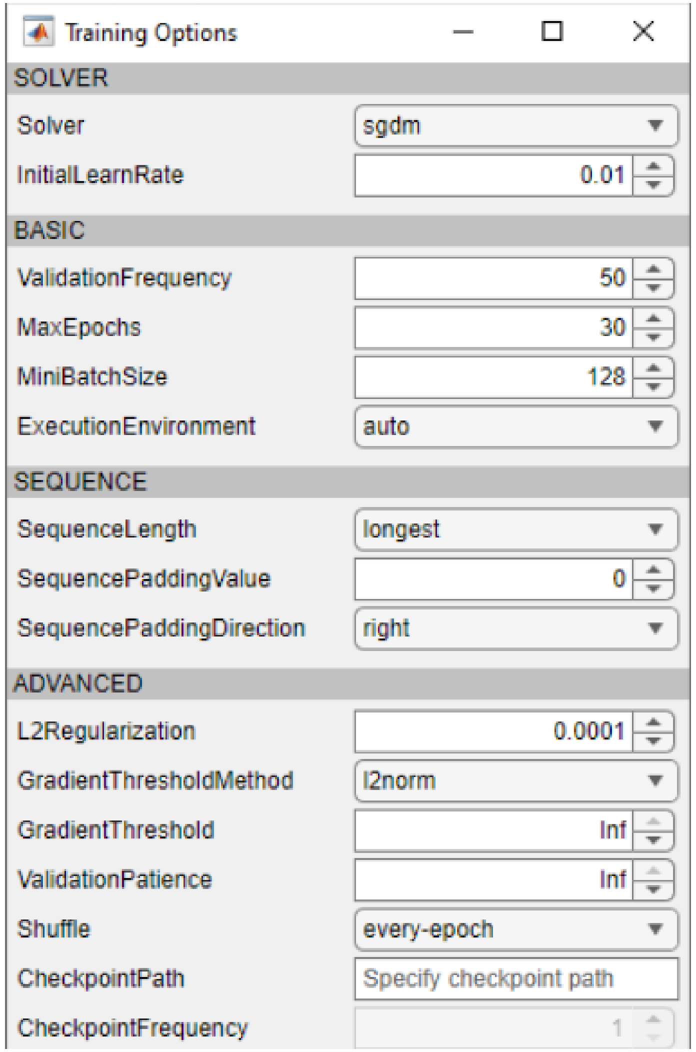
Signal and speech processing
Denoising, music genre recognition, keyword spotting, radar waveform classification…

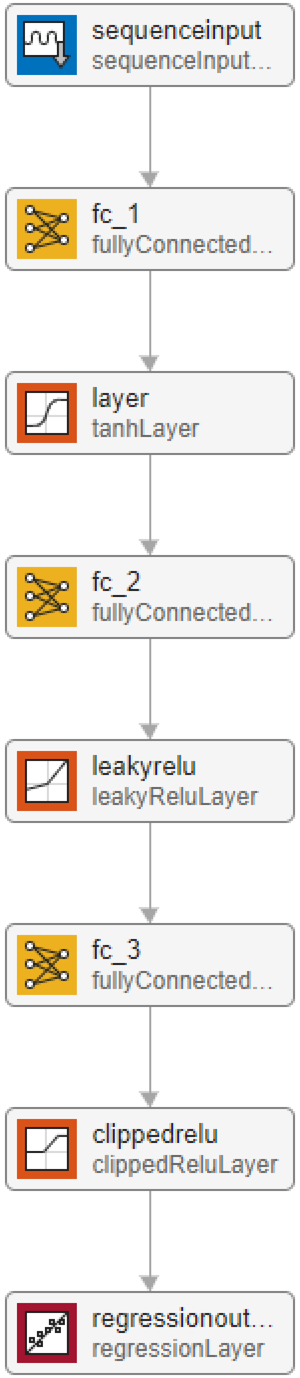
…and more…

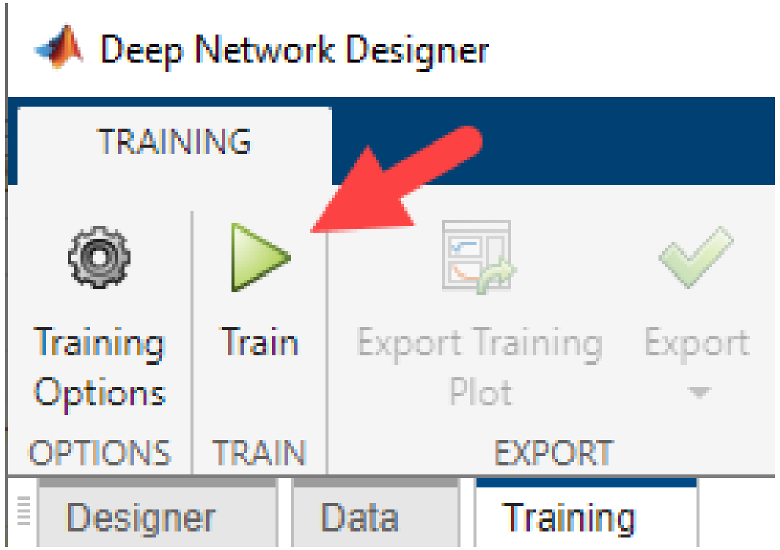
13

- Configure AI model
- Train and Test AI model









AI modeling

FNN_xEV_Li_ion_SOC_EstimatorScript.mlx

xEV Battery State-of-Charge Estimator using a Feedforward Deep Neural Network

Create Datastore datasets - Training and Testing

```
clear
dataFolder = fullfile(pwd, 'LGHG2@n10C_to_25degC');
```

Create train & test datastores

```
trainDataDS = fileDatastore(fullfile(dataFolder,"Train"), 'ReadFcn',@load,'FileExtensions','.mat');
testDataDS = fileDatastore(fullfile(dataFolder,"Test"), 'ReadFcn',@load,'FileExtensions','.mat');

trainData = read(trainDataDS)
```

```
trainData = struct with fields:
    X: [5×669956 double]
    Y: [1×669956 double]
```

```
testData = read(testDataDS)
```

```
testData = struct with fields:
    X: [5×39293 double]
    Y: [1×39293 double]
```

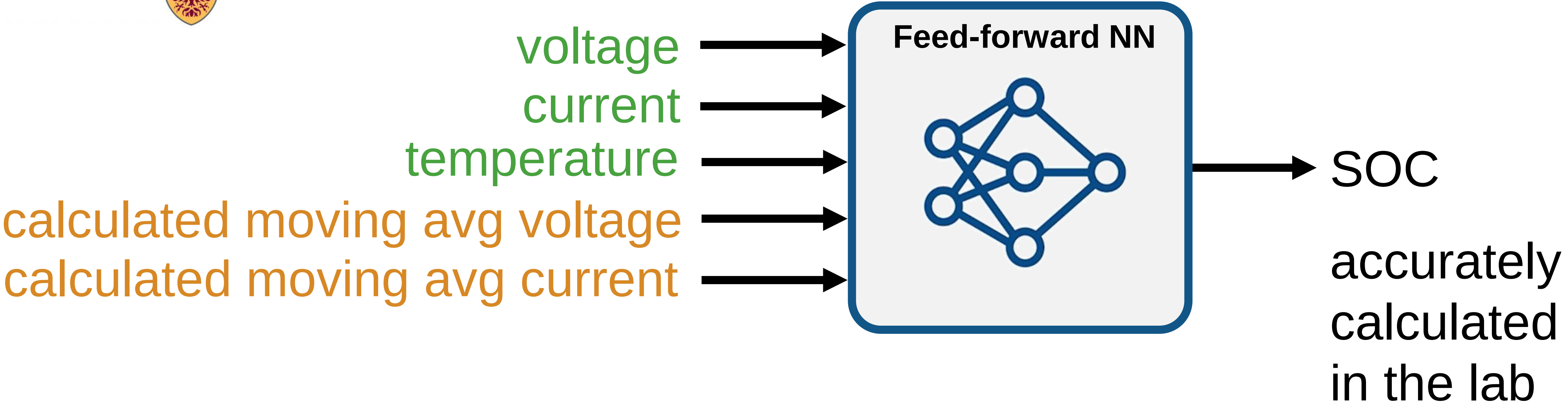
Define NN Architecture

deepNetworkDesigner

```
numFeatures = 5;        % Number of inputs features (V, I, Temp, V_avg, I_avg)
numHiddenUnits = 55;    % Number of hidden units 'N', where each hidden unit for FNN represents a Neuron.
numResponses = 1;       % Number of outputs (SOC)

layers = [...
    sequenceInputLayer(numFeatures,"Normalization","zerocenter")
    fullyConnectedLayer(numHiddenUnits)
    tanhLayer                          % Hyperbolic Tangent
```

McMaster University



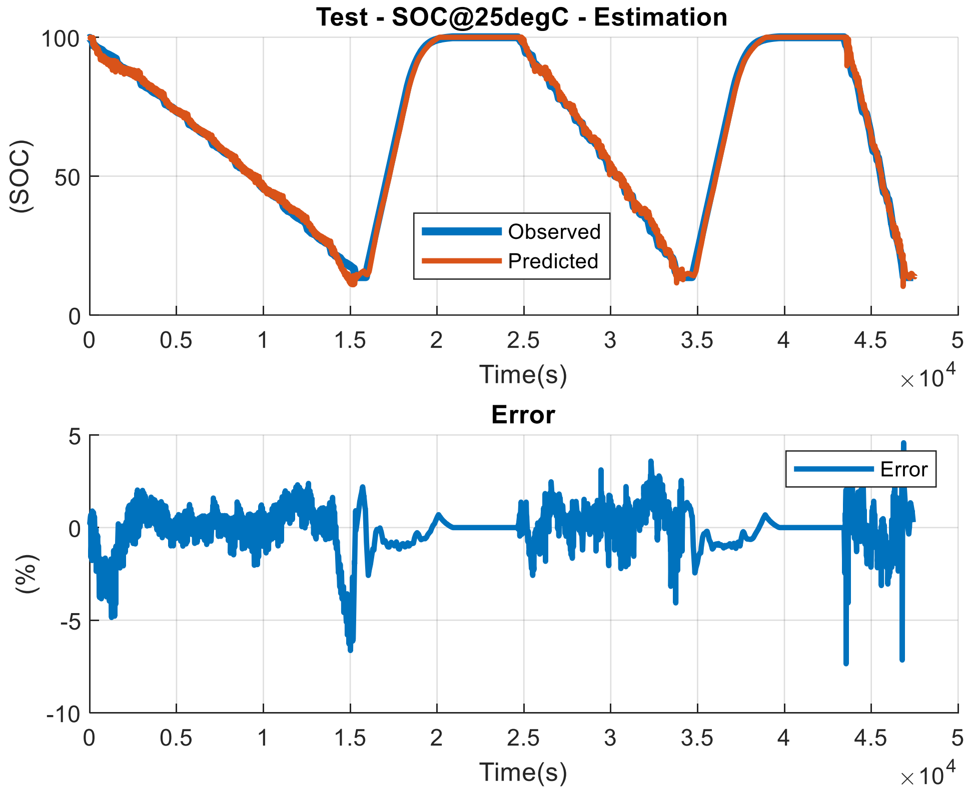
Feed Forward NN is simple – but it has no memory

Moving average added to the input signals

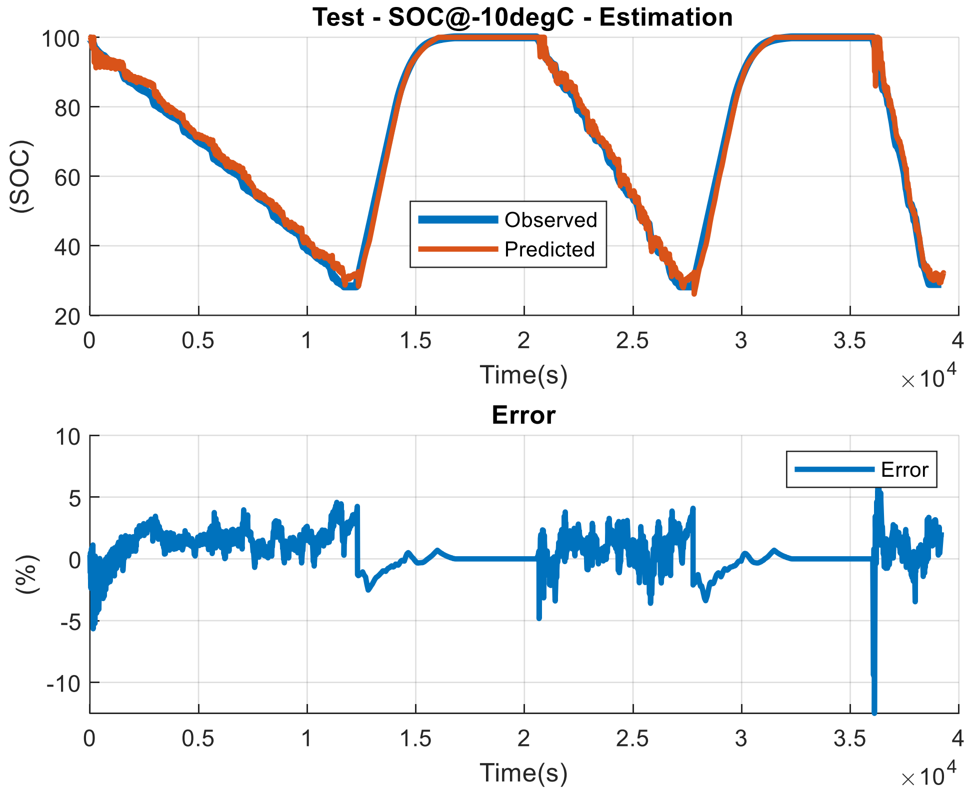
Data source https://data.mendeley.com/datasets/cp3473x7xv/3

Results

25ºC



-10ºC

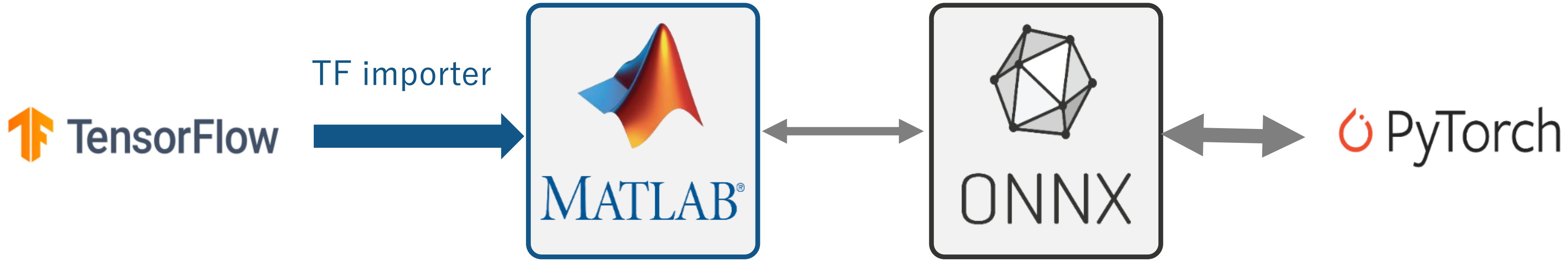


Prediction is good even at low temperatures

prediction

ground truth

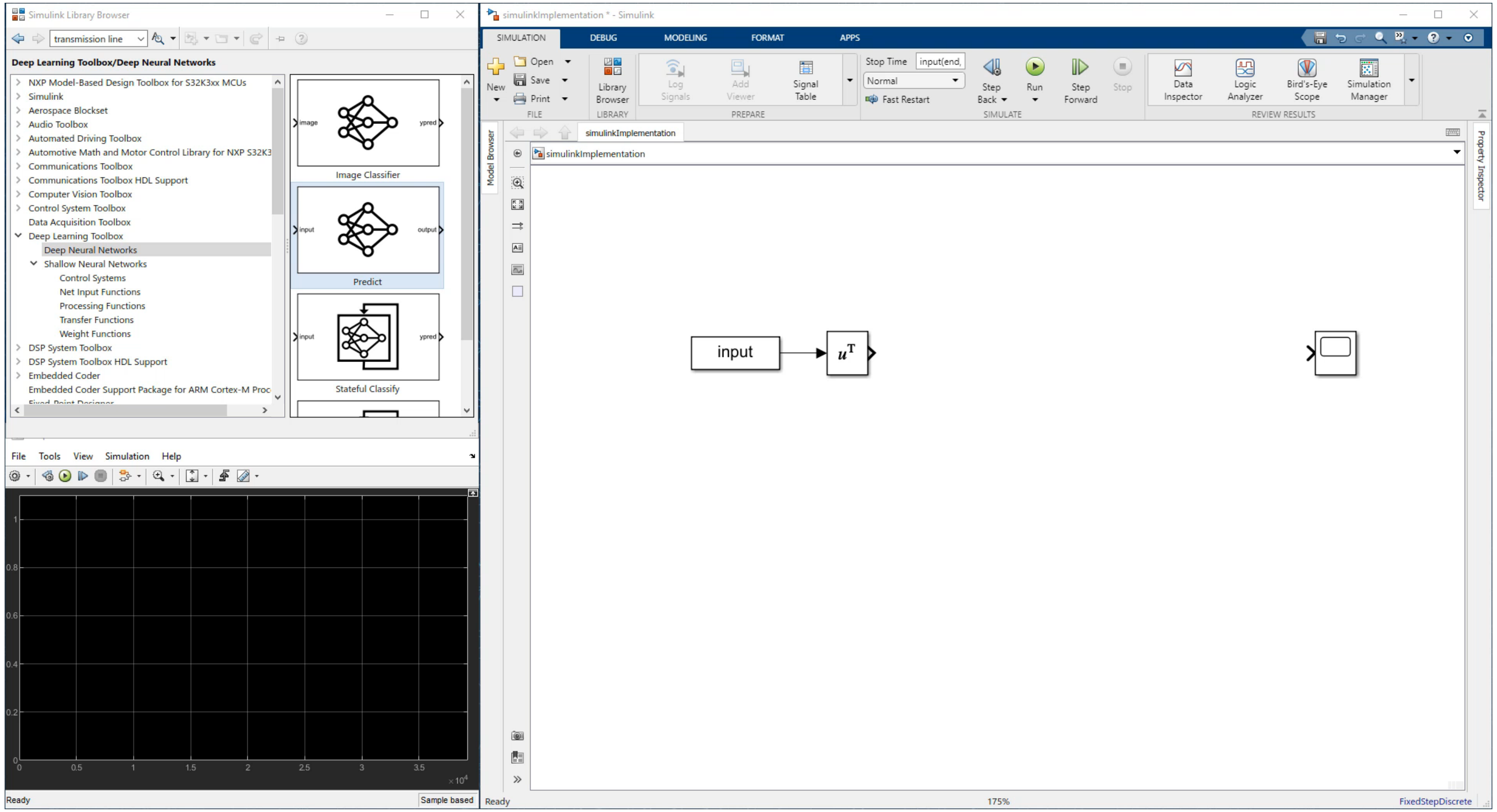
Import Pre-Trained Model



You can also import an AI model trained outside of the MathWorks ecosystem into MATLAB

Voltage	Current	Temperature
0.7510	0.3851	0.3031
0.7510	0.3852	0.3046
0.7510	0.3852	0.3061
0.7510	0.3852	0.3076
0.7510	0.3852	0.3091

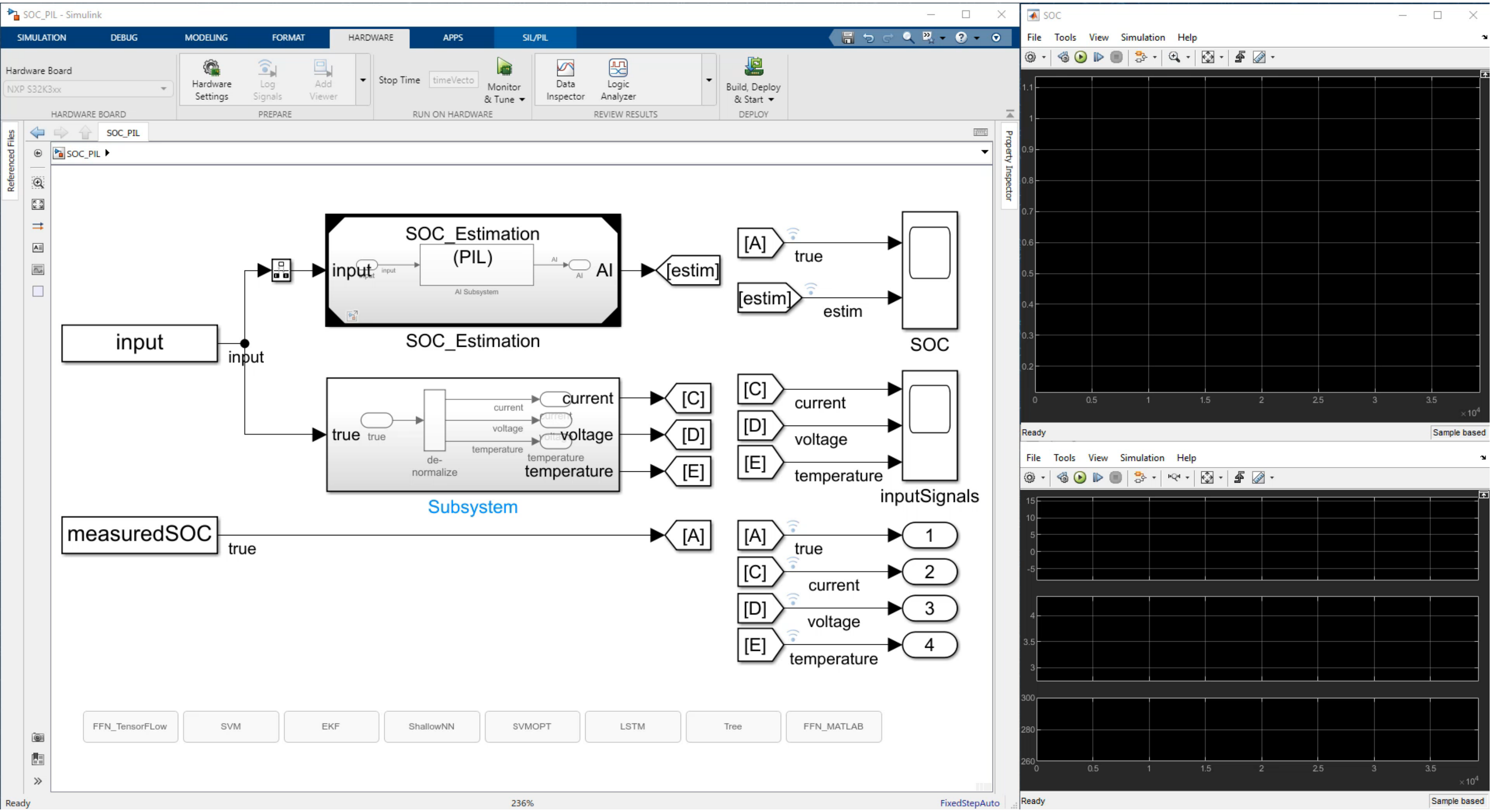
Simulink Integration



Simulink provides blocks with different AI functions We just parameterize them with the AI function name and feed them with signals with the predictors

Data Preparation | AI Modeling | Simulation & Test | Deployment

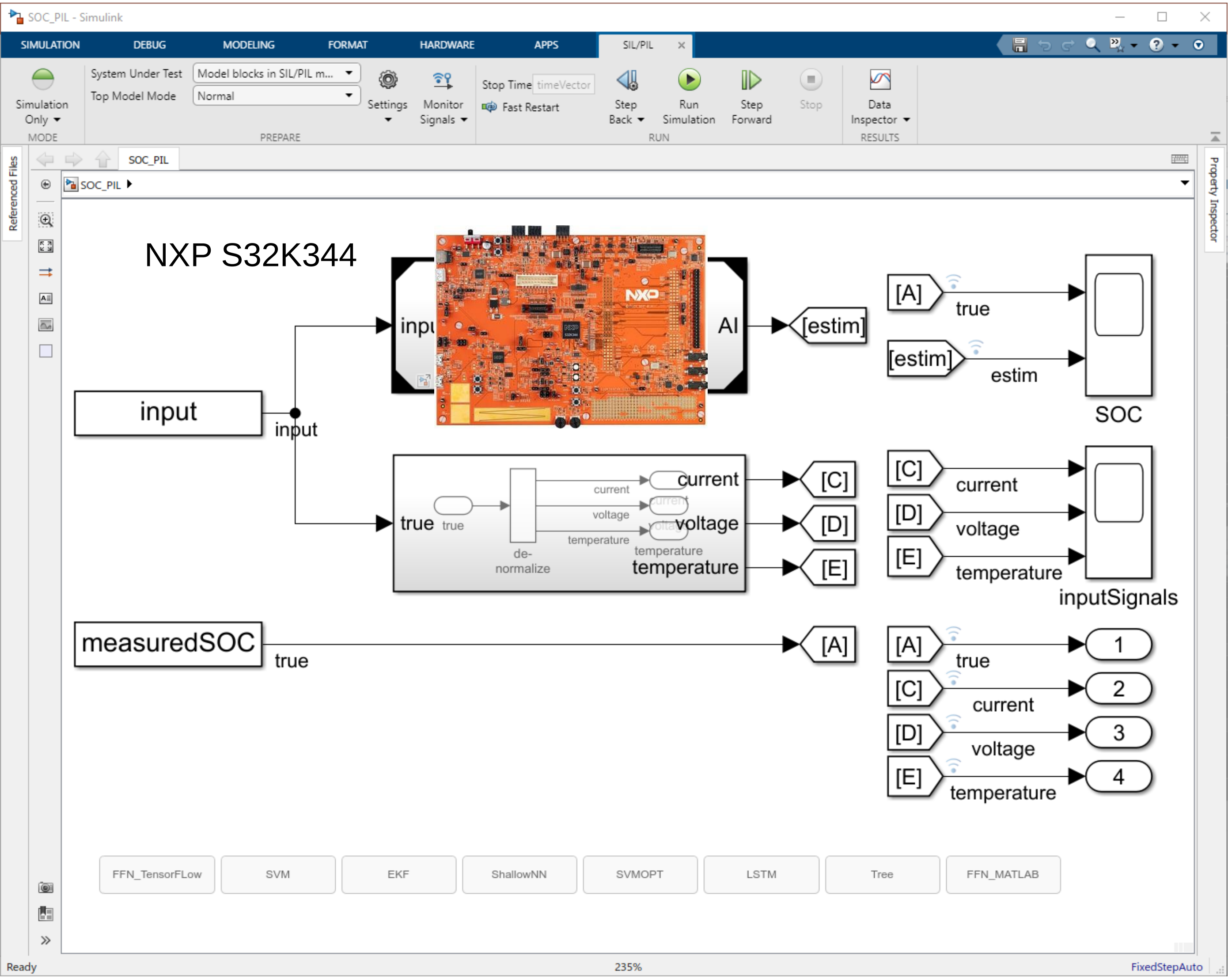
Voltage	Current	Temperature
0.7510	0.3851	0.3031
0.7510	0.3852	0.3046
0.7510	0.3852	0.3061
0.7510	0.3852	0.3076
0.7510	0.3852	0.3091



With Variant Subsystems we can implement several AI functions in the same model and try them one at a time

Processor-in-the-Loop (PIL) Testing on ARM Cortex-M7 Processor

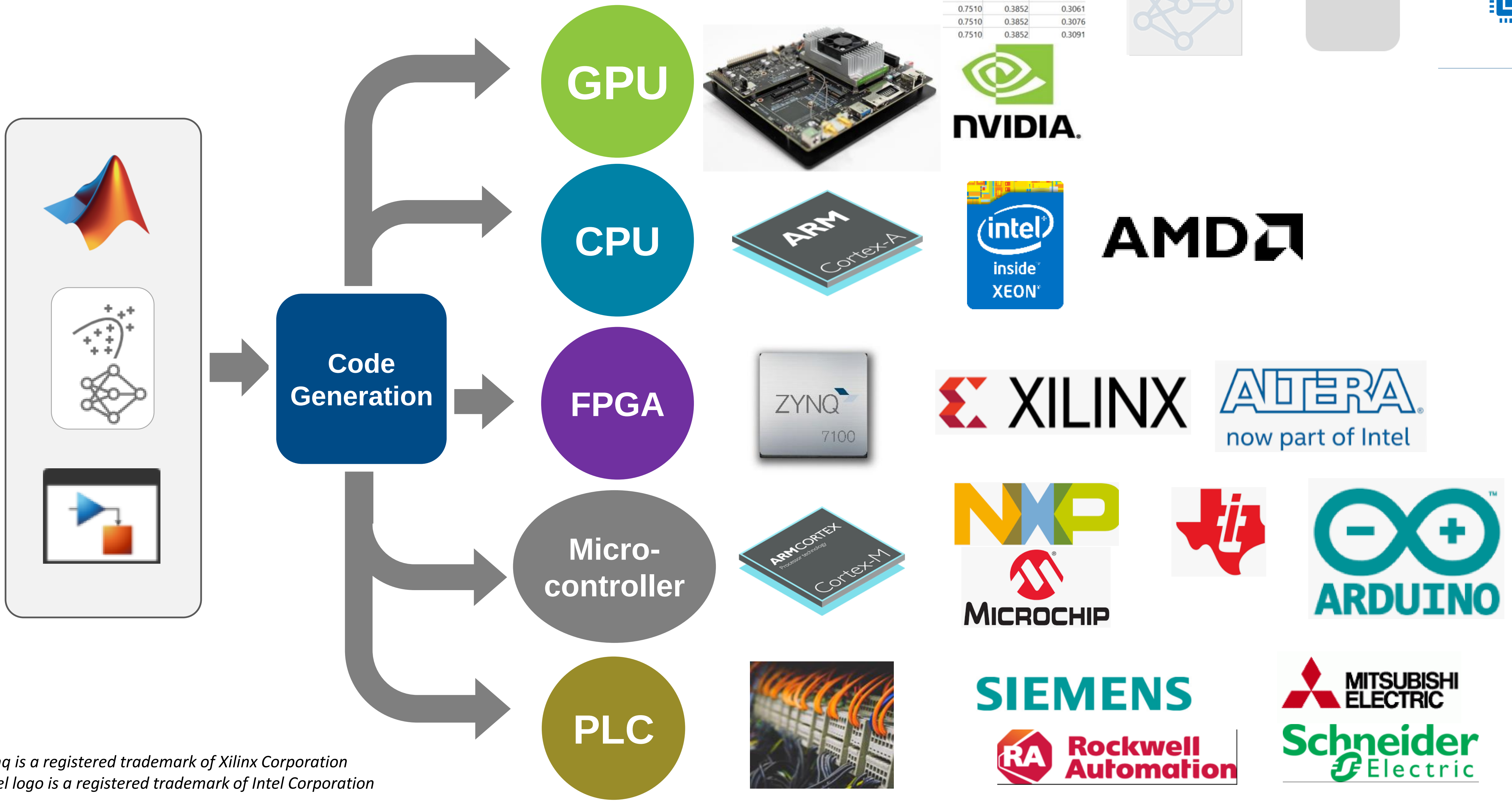
Voltage	Current	Temperature
0.7510	0.3851	0.3031
0.7510	0.3852	0.3046
0.7510	0.3852	0.3061
0.7510	0.3852	0.3076
0.7510	0.3852	0.3091



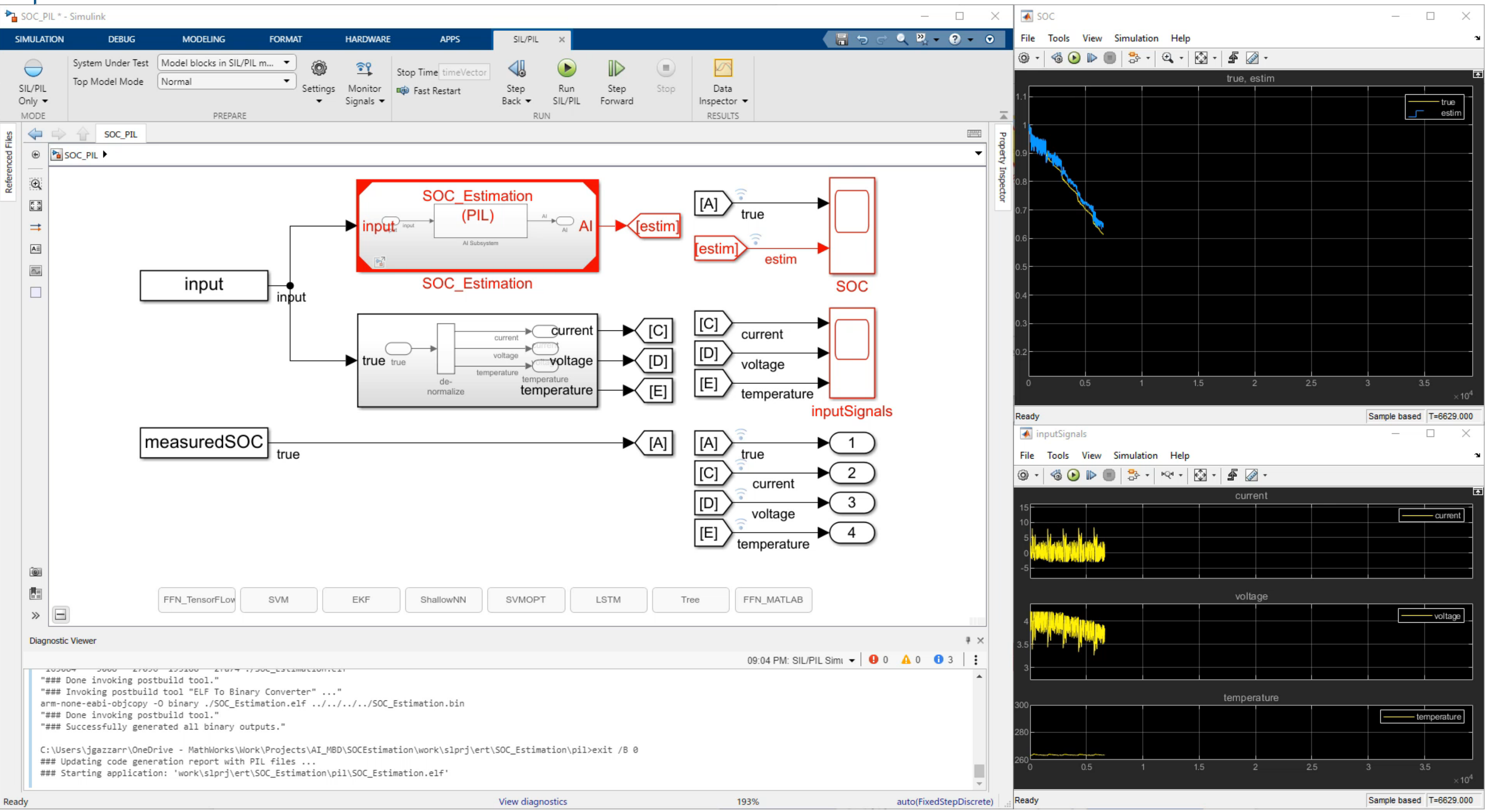
Automatic Library-Free C Code

Any CPU
Inc. ARM Cortex-M

Deploy to target with zero coding errors



Zynq is a registered trademark of Xilinx Corporation
Intel logo is a registered trademark of Intel Corporation



Finally, we can configure the model for Processor-in-the-Loop execution

1- Configure hardware and communication ports

2- Select PIL execution

3- Code is generated for the AI function subsystem, compiled, then downloaded onto the evaluation board

4- The algorithm now runs on target

Tradeoffs and Benchmark

	EKF Extended Kalman Filter	**Tree** Fine Regression Tree	**FFN** 1-hidden layer Feedforward Network	**LSTM** Stacked Long Short-Term Memory Network
Training Speed	N/A	🟢	🟠	🔴
Interpretability	🟢	🟢	🔴	🔴
Inference Speed *	🟠	🟢	🟢	🔴
Model Size *	🟢	🟠	🟢	🔴
Accuracy (RMSE)	🟢	🟠	🟢	🟢

Results are specific to this example

Here is a comparison among AI methods and the EKF benchmark

There is a trade-off among training effort, predictive accuracy, and on-target execution time

* *NXP S32K344 board*

Denso Ten develops workflow process for AI control system development

Challenge

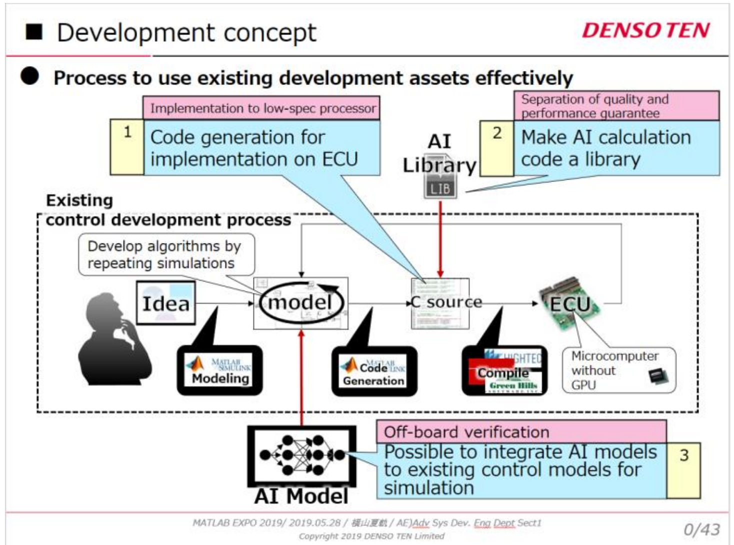
Use AI to improve ECU development efficiency

Solution

Use Deep Learning Toolbox, Embedded Coder, and Simulink Coder in a new workflow for AI/deep learning, ECU simulation and implementation

Results

- Integrated AI model into existing control model
- Used Deep Network Designer for network construction
- Created bidirectional conversion of deep learning model between MATLAB and Simulink
- Accessed original AI library using S-functions



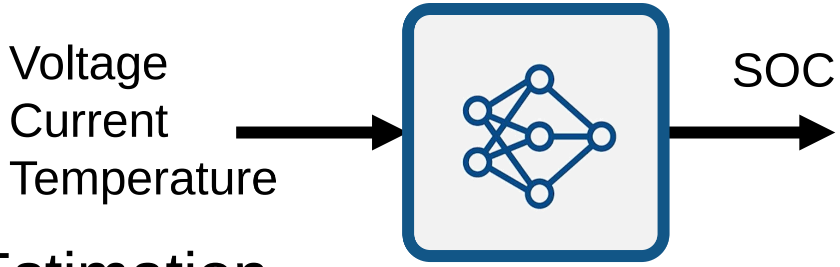
Model-based development workflow.

"A model-based development workflow is essential in order to use AI for control ECUs. Combining the existing control model and the AI model enables us to establish a simulation environment and accelerate product development."

- Natsuki Yokoyama, Denso Ten

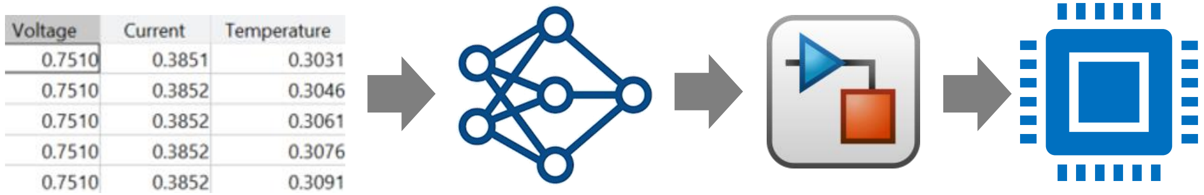
27

Summary



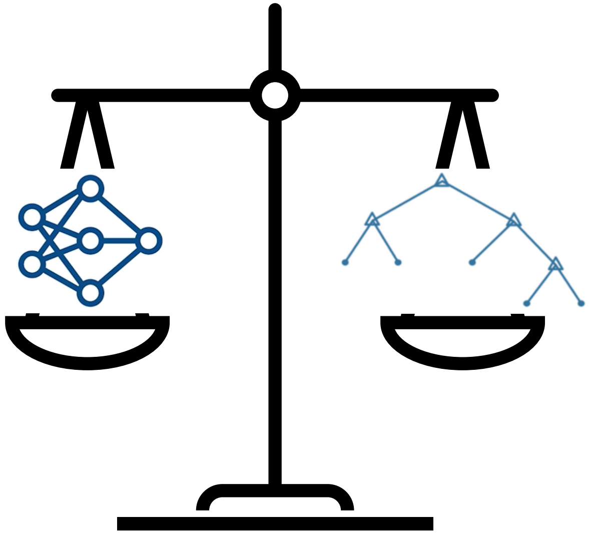
- Develop AI-based Virtual Sensor for Battery SOC Estimation

- Workflow - From Data Acquisition to Hardware Deployment



Voltage	Current	Temperature
0.7510	0.3851	0.3031
0.7510	0.3852	0.3046
0.7510	0.3852	0.3061
0.7510	0.3852	0.3076
0.7510	0.3852	0.3091

- Compare Different AI Methods



MATLAB EXPO

Thank you

MathWorks®