A platform for time-series anomaly detection and evaluation using MATLAB and Simulink
User Guide

July 2022

Contents

1 Introduction

We present a platform for anomaly detection evaluation methods on univariate tine-series for Cyber-Physical Systems (CPS). With this platform, Users are able to evaluate the performance of the detection process according to multiple variables, such as the dataset, preprocessing method, network type and architecture, network and training related hyperparameters, thresholding method and evaluation metrics. The platform also proposes Simulink models usable in the case of online detection, which automatically reshape the data as the models require and flags detected anomalies in real time. Combined with expert knowledge, this tool aims to improve the understanding of the performance of certain models and techniques on specific CPS domains.

2 Used material

2.1 Datasets

Unmanned Aerial Vehicle (UAV)

The UAV dataset was collected from a Simulink-based model adapted from the Simulink Support Package for Parrot Minidrones. Each waypoint contains information about initial position, end position, and global trajectory in the form of (x, y, z) coordinates, where 18 features of sensors and actuators are collected, Each simulation duration is equal to 60 seconds with a sample rate of 100Hz, meaning 6000 samples per simulation.

Faults were injected using the FIBlock in a non-uniform manner to the x-axis of accelerometer and gyroscope sensors during simulation for faulty cases data collection. Injected faults are: Bias/Offset, stuck-at, noise, and package-drop, with each simulation containing a unique fault type.

Autonomous Vehicle System(AVS)

The AVS dataset was also generated from a Simulink model, which design in inspired from MathWorks's adaptive cruise control concept and MOBATSim's

A simulation consists of a normal vehicle moving along a predefined path, which is followed by another autonomous vehicle. For data collection, simulations were run attributing different driving scenarios to two different road types (town and highway).

Simulations were run for 80 seconds with 801-time samples. Fault free data was generated for 24 simulations with different initial conditions, split equally into town and highway driving scenarios. Faulty data was also generated using the FIBlock, by coupling it to the speed sensor of the autonomous vehicle, in order to inject an offset with a magnitude equal to 5, noise of 20% from original data and stuck-at faults. The block was also parameterized to add faults of duration ranging from 0.6s to 5s with a fixed step size of 0.6s at each iteration and spaced at a period of 9s, for a total of 600 generated files.

Secure Water Treatment (SWaT)

The SWaT testbed is one of the few well-labeled datasets reflecting real-world data from a complex Industrial Control System (ICS). The water treatment process is based on six water treatment stages. Data about both physical and cyber state was collected for 11 days, booting the system from its initial state by injecting attacks during the last 4 days of the treatment without interruptions between the phases. Attacks were injected following a certain model for a period going from certain minutes to hours and can happen on multiple stages simultaneously.

The physical properties are defined by several sensors and actuators, whose values were reported in CSV files, indicating also for each timestamp if it involves an attack or not. Attack logs (start/end time, attack points, attack description etc.) of the 36 attacks were gathered into another file.

2.2 Detection techniques

Supervised thresholding

The method selected is the Gaussian method, with some modifications. Basically, it is based on the assumption that the predicted error values are normally distributed. We then determine the mean μ and variance σ and of the non-anomalous validation set of the Deep Neural Network (DNN) with the Maximum Likelihood Estimation. Afterwards, the anomalous validation set is used to determine the threshold value γ by applying the probability density function (PDF) f_X on it and finding the best $\boldsymbol{F_1}$ score using the anomaly labels. We also propose a variant of this method where a threshold is directly computed from the maximized F1 score on VA. In the event that no validation data is used for training, only the second variant is available.

Unsupervised thresholding

The supervised thresholding method adopted is the one from Detecting Spacecraft Anomalies Using LSTMs and Nonparametric Dynamic Thresholding. First, the error vector $\mathbf{r}$ is smoothed using an Exponentially-Weighted Moving Average (EWMA). The threshold γ is then selected from a predefined set of values within a certain range using an algorithm to find the best threshold value. An anomaly pruning method is also introduced where previously found anomalies are removed if a new, more obvious one is found.

3 Usage instructions and guide

In this section, we present the installation instructions and a complete step-by-step guide on how to use the tool.

3.1 Prerequisites

- Matlab/Simulink 2020b or a newer release.
- Matlab's Deep Learning Toolbox and its prerequisites.
- Matlab's Econometrics Toolbox and its prerequisites.
- Matlab's Image Processing Toolbox and its prerequisites.
- Matlab's Predictive Maintenance Toolbox and its prerequisites.
- Matlab's Signal Processing Toolbox and its prerequisites.
- Matlab's Statistics and Machine Learning Toolbox and its prerequisites.

3.2 Installation

1. **Download the `.zip` file here.**
   (a) Extract the `.zip`.
   (b) Add the extracted folder and sub-folders into your Matlab path.
   (c) Install the tool by double-clicking the `myApp.mlappinstall` file.
      i. The tool can also be launched and modified through Matlab's App Designer by opening the `myApp.mlapp` file.

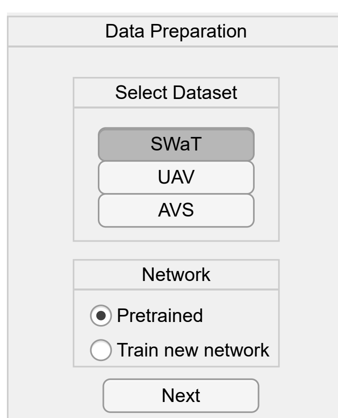


Figure 1: Data selection panel

3.3 Step by step User guide

This part provides a step-by-step detailed guide on how to use the platform and a description of its functionalities.

Dataset selection

The first window that pops up when the application starts is the dataset selection window. The User can select a dataset between the SWaT, UAV and AVS datasets (see Chap 2.1. Moreover, the User can chose to either used a pretrained network or train a new one. However, the usage of a pretrained network is not yet implemented, so it is unusable. The data selection panel is shown on Fig. 1.

Data preparation windows

Depending on the selected dataset, another window appears:
AVS Dataset. If the AVS dataset has been chosen, the User can further decide whether to continue with *highway* or *town* driving scenarios. The fault free simulated data can be previewed with the *Visualize data* button, as shown on Fig. 3.
UAV Dataset. For the UAV dataset, either accelerometer or gyroscope sensor data can be chosen. The simulated data can also be previewed with the *Visualize data* button.
SWaT Dataset. The window showing upon selectin the SWaT dataset is in the form of a table. Each column represents a monitored sensor or actuator. Observations are represented at each row with one-second intervals and can also be visually displayed thanks to *Visualize data* button. Selectable sensors are highlighted in blue. The selected column becomes highlighted in green for the case of a valid selection, and in red if the selected column is not a valid sensor.

Fig. 2 shows all three data preparation windows.

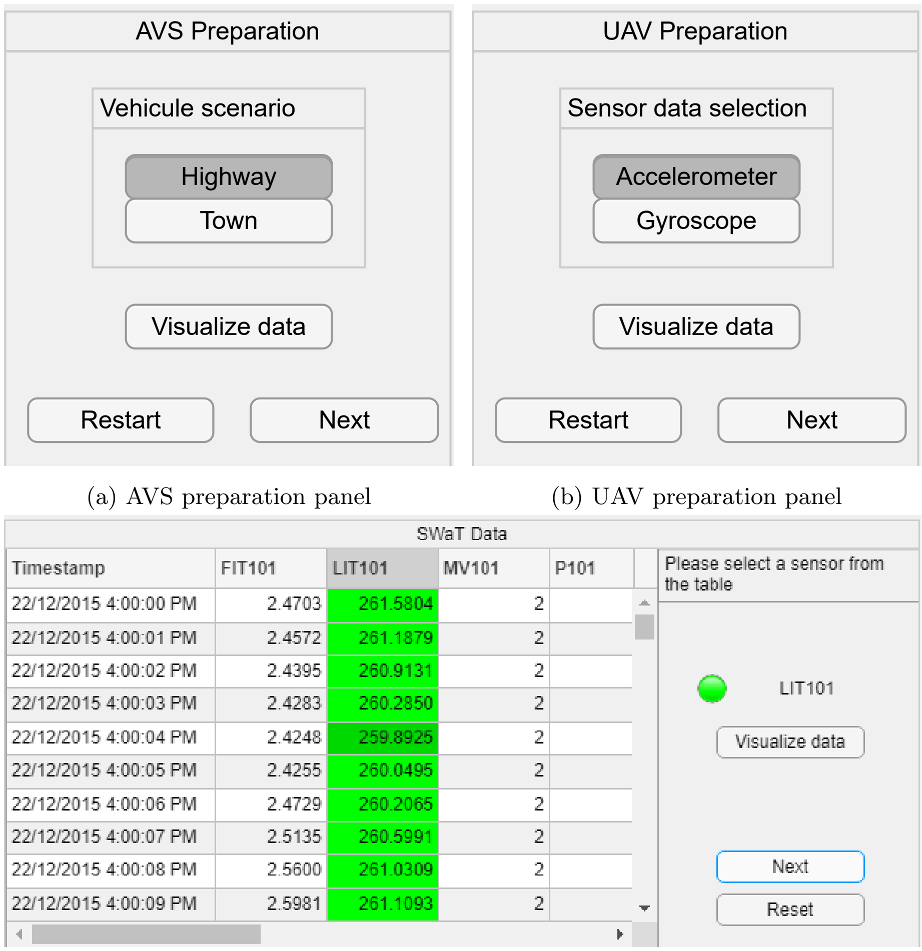
Preprocessing methods

This window is common for all selected datasets and is shown on Fig. 4. Popular preprocessing techniques are available, namely normalization and standardization. For the AVS and UAV datasets, according to the chosen method, a unique mean and standard deviation or minimal and maximal value of all combined simulations is computed. Only nonanomalous data is used for model training. The User also has the possibility to use not preprocessed, raw data.

In addition, the type of DNN can be selected between a recontruction or prediction network.

Network configuration

The next window is the network configuration window (see Fig. 5). Here, you can tune multiple settings.



(a) AVS preparation panel (b) UAV preparation panel

(c) SWaT preparation panel

Figure 2: Data preparation panels

First, the DNN architecture should be chosen. Different options are available depending on the chosen DNN architecture:

1. Prediction network
   - LSTM-based network
   - GRU-based network
   - Hybrid CNN-LSTM-based network
2. Reconstruction network
   - LSTM-based network

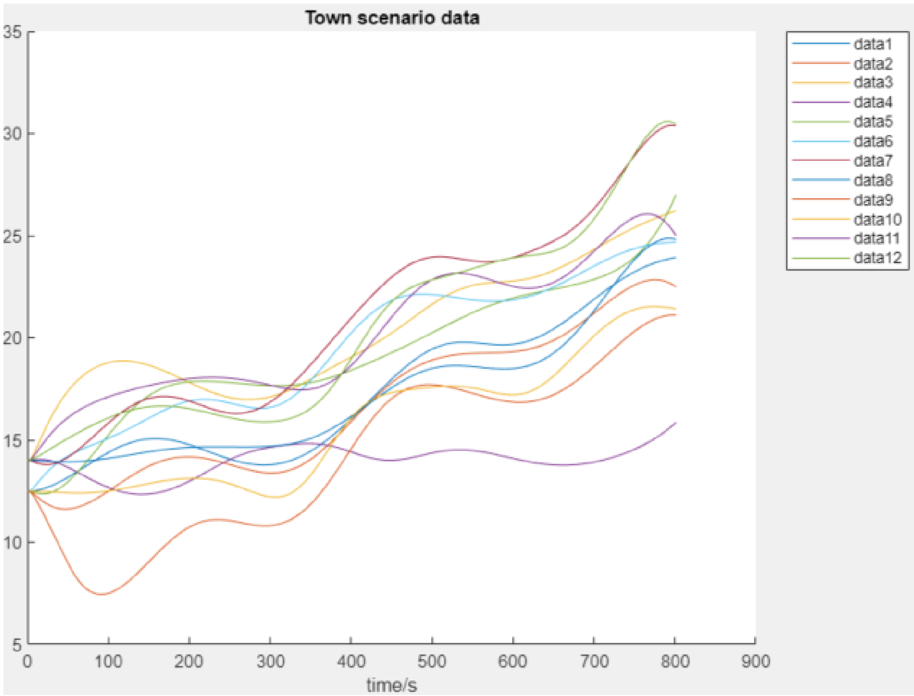


Figure 3: AVS town fault-free data

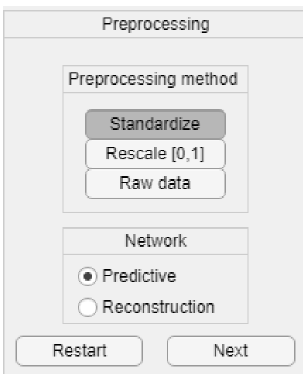


Figure 4: Preprocessing method and DNN type selection panel

- Hybrid CNN-LSTM-based network
- Basic fully connected network

The network architecture can be displayed on a separate window using the *Network prediction* button, Fig. shows an example for the case of an LSTM-based prediction network.

Next, some hyperparameters can be modified, which will also change according to the chosen network architecture. We hereby differentiate between three types of hyperparameters:

(i) **general** hyperparameters
(ii) **training related** hyperparameters
(iii) **network related** hyperparameters.

Fixed hyperparameters are variables that the User is not able to change using the platform. They are predefined and fixed for all datasets and training scenarios.

Fixed hyperparameters Compared to the fixed ones, we consider them as more impactful on the performance of the models. They listed in Table 1. Fig. 5 gives an overview of the network options interface for the GRU predictive model.

Once the hyperparameters have been chosen, the network can be trained with the *Train* button and saved into the MATLAB workspace with the *Save net to workspace* button.

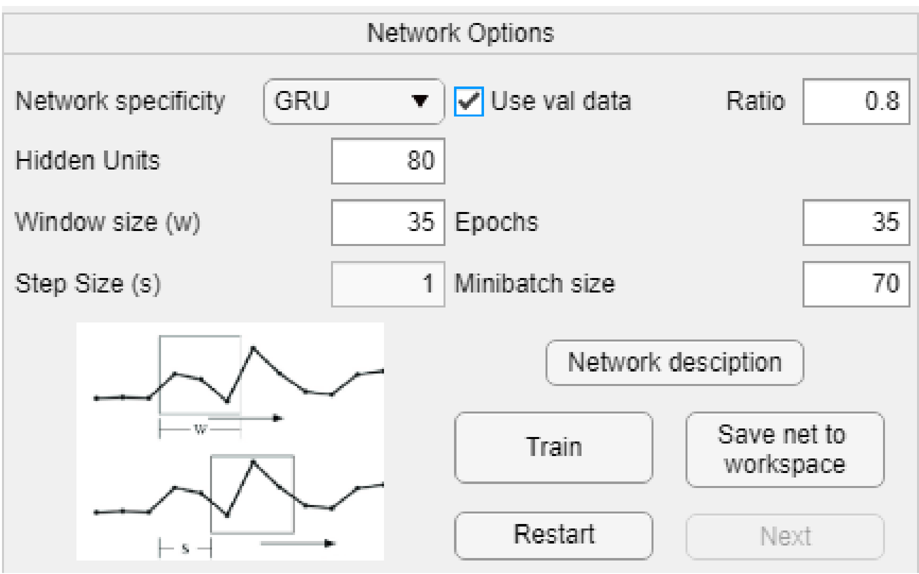


Figure 5: Network options panel

Hyperparameter	Value	Initial value
General		
Preprocessing method	Standardize, rescale, raw data	
$^{V_A}/_{Faultfree}$ ratio	[0,1]	0.8
Window size	[1-Max]	35
Training options		
#Epochs	Free of choice	35
Minibatch size	Free of choice	70
Network related		
Network type	Prediction, reconstruction	
#Hidden units (RNN, GRU)	Free of choice	80
#Neurons	Free of choice	32
#Filters (Conv layer)	Free of choice	32

Table 1: List of User-defined hyperparameters

Faulty data selection

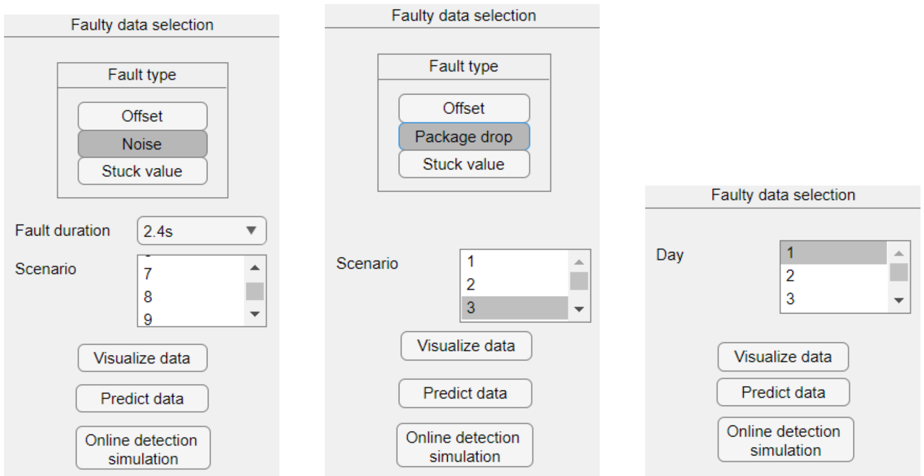
The next window, like the data preparation window, is also dependent of the chosen dataset:
AVS dataset. As shown on Fig. 6a, the *fault type*, *fault duration* and *scenario* can be chosen.
UAV dataset. Here, the *fault type* and *scenario* can be chosen (see Fig. 6b).
SWaT dataset. The User can chose the day of faulty data collection (see Fig. 6c).

Also, each selected faulty data can be visualized on a separate window with the *Visualize data* button.

Offline detection

By clicking the *Predict data* button, a new window opens for the offline detection. On Fig. 7, an example of the detection results for a simulation from the AVS dataset using noise fault of a duration of 1.2s can be found.

- The top half of the window is dedicated for the unsupervised thresholding method and the bottom half for the supervised thresholding.
- The plots on the left side show the faulty data. The labeled anomalies are highlighted in green and the detected anomalies in red.
- The middle graphs display the residual data and the red line is the threshold. All values above the threshold are flagged as anomalous.
- The tables on the right side contain the *precision*, *recall* and $\boldsymbol{F_1}$ scores for the unweighted and weighted segment techniques.



(a) AVS (b) UAV (c) SWaT

Figure 6: Faulty data selection panels

The z_{range}, *anomaly padding* and *window length* parameters of the unsupervised thresholding method can also be changed on the results view on the platform to get a dynamic overview of their influence in the detection process.

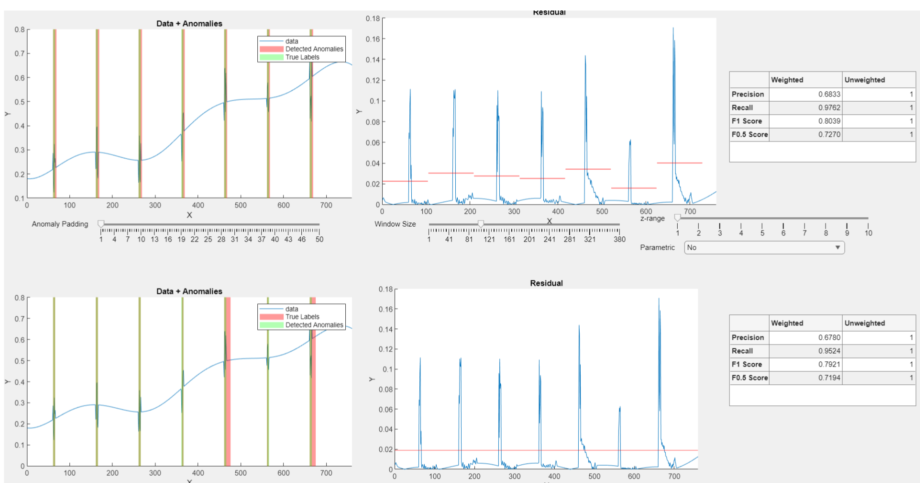


Figure 7: Offline detection window

Online detection simulation

Still on the faulty data selection window, the User can proceed with the online detection simulation with the supervised thresholding mehtod. This will open a new window (see Fig. 8). First, the trained network should be saved to the desired location. Then, the Simulink model can be opened and the simulation can be launched.

Based on the type of the trained network, either the *prediction* (Fig. 10a) or *reconstruction* (Fig. 10b block will be used. The main Simulink model is the same for both network types (Fig. 10). The detection block automatically takes the trained `.mat` network and the *window length* (chosen on the network options window, see Fig. 5) as mask parameters.

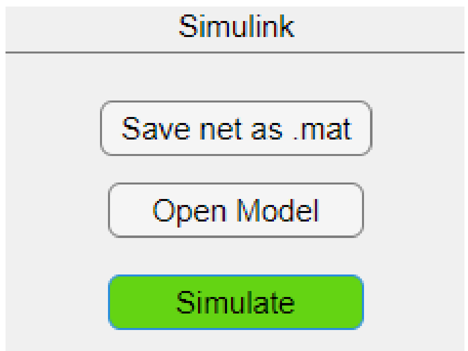


Figure 8: Simulink detection window

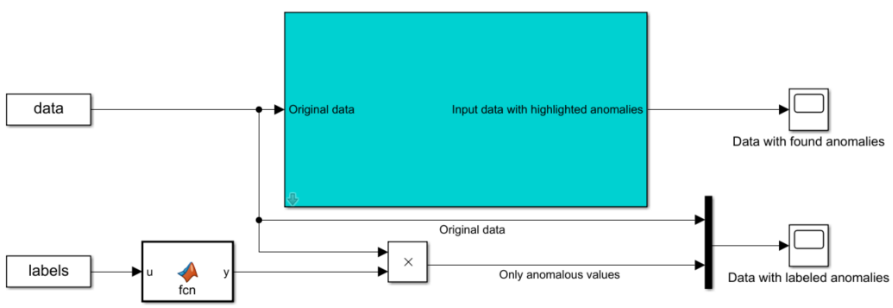
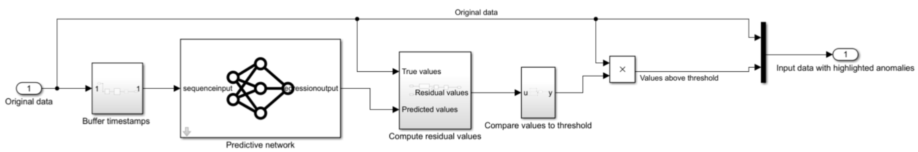
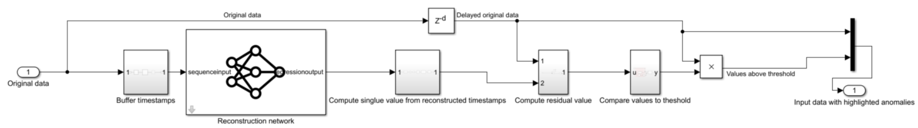


Figure 9: Online detection Simulink model

The *scope* blocks on the main model display the faulty data with highlighted true anomalies and found anomalies. An example is shown on Fig. .

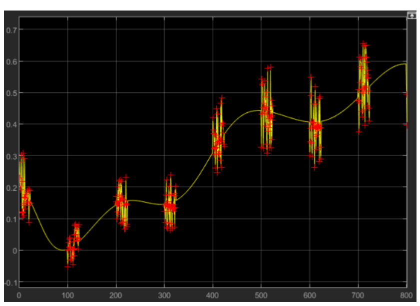


(a) Subsystem for prediction networks

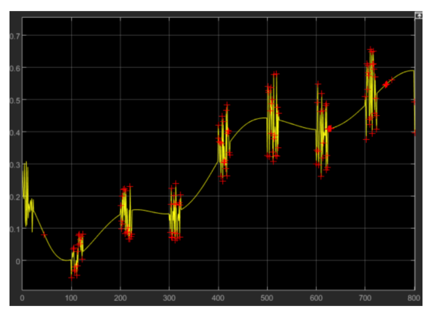


(b) Subsystem for reconstruction networks

Figure 10: Simulink subsystem of the detection block in Fig. 9



(a) Original data with true labels



(b) Original data with detected faults

Figure 11: Example of detection on Noise fault