

**Inputs**

The autoencoder receives mixed samples (normal + attack). The shape is (X, Y, Z), where X is the number of samples, Y is the window size (10), and Z is the number of features (51).

Inputs

Autoencoder Model

#### Autoencoder Model (evaluation mode)

The autoencoder aims to learn the underlying distribution of normal data by achieving for the lowest possible reconstruction error. Consequently, when an anomaly occurs, the model, having been trained solely on normal patterns, is expected to produce a substantially higher reconstruction error.

Reconstruction Errors (MSE)

#### reconstruction\_errors

The autoencoder produces an output with shape (X, Y, Z). The reconstruction error is derived by first calculating the Mean Squared Error (MSE) for each sub-sample within every sample (first block). Subsequently, for each main sample, a single error value is obtained by averaging the 10 MSEs from its respective sub-samples, following the DAICS methodology (second block).

Reshape (X,Y,Z) to (X)

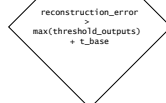
reconstruction\_errors

Threshold Model

#### Threshold Model (evaluation mode)

The threshold model aims to learn the reconstruction error characteristics of normal data from the autoencoder. The core idea is that when anomalies are encountered, the threshold model, given a reconstruction error, will struggle to produce a similar output. This discrepancy arises because anomalous data is theorized to deviate from the distribution of normal data.

threshold\_outputs



#### Classification

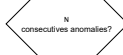
The classification method is directly adopted from DAICS.

- $\max(\text{threshold\_outputs}) \rightarrow$  Calculated each time.
- $t\_base = \mu(\text{reconstruction\_errors}) + \sigma(\text{reconstruction\_errors}) \rightarrow$  Recalculated only when the autoencoder is retrained (see below). **Note:** The system initializes with a  $t\_base$  value derived from the training session.

#### Anomaly Classifications

Controlling false positives is easier than false negatives. In this scenario, an operator can verify the classification, and their feedback can then be used to update the entire system.

Anomaly



#### N

Unlike other values, N is hardcoded. Dynamically determining N could introduce too many dynamic variables into the system, significantly increasing its complexity.

YES

False Positive?

#### Operator

In this system, the operator provides optional feedback used to improve the entire system. While the specific method an operator uses to check for false positives is out of scope, their feedback is crucial: in cases of false positives, it is used to improve the local model and, under certain conditions, also the global one. The ultimate objective is to create a system that minimizes the need for operator intervention.

YES



#### N

Unlike other values, N is hardcoded. Dynamically determining N could introduce too many dynamic variables into the system, significantly increasing its complexity.

Update Local Autoencoder Model

#### Update Local Autoencoder Model

The update procedure mirrors that of DAICS: a single epoch of Stochastic Gradient Descent (SGD). Updating the autoencoder model with each benign sample helps the system remain robust against evolving noise levels and minor shifts.

$\Delta\_autoencoder\_model$

#### $\Delta\_autoencoder\_model$

$\Delta\_autoencoder\_model$  is calculated as the L2 Distance between the local and global autoencoder models. This allows us to capture changes in weights that result, for instance, from data shifts.

L2(local\\_autoencoder\\_model, global\\_autoencoder\\_model)

$\epsilon\_model$

#### $\epsilon\_model$

Unlike other values,  $\epsilon\_model$  is hardcoded. Dynamically determining  $\epsilon\_model$  could introduce too many dynamic variables into the system, significantly increasing its complexity.

Update Global Autoencoder Model

#### Update Global Autoencoder Model

If the system determines a global model update is needed, the current local model is sent to the server for aggregation with the global model. The updated global model is then distributed to all clients.

#### Benign Classifications

Controlling false negatives in benign classifications is extremely challenging, if not impossible. Unlike false positives, where an operator can verify the classification, there isn't a similarly specific method to address false negatives.

Benign

Update Local Threshold Model

#### Update Local Threshold Model

The update procedure mirrors that of DAICS: a single epoch of Stochastic Gradient Descent (SGD). Updating the threshold model with each benign sample helps the system remain robust against evolving noise levels and minor shifts.

$\Delta\_threshold\_model$

#### $\Delta\_threshold\_model$

$\Delta\_threshold\_model$  is calculated as the L2 Distance between the local and global threshold models. This allows us to capture changes in weights that result, for instance, from data shifts.

L2(local\\_threshold\\_model, global\\_threshold\\_model)

$\epsilon\_model$

#### $\epsilon\_model$

Unlike other values,  $\epsilon\_model$  is hardcoded. Dynamically determining  $\epsilon\_model$  could introduce too many dynamic variables into the system, significantly increasing its complexity.

Update Global Threshold Model

#### Update Global Threshold Model

If the system determines a global model update is needed, the current local model is sent to the server for aggregation with the global model. The updated global model is then distributed to all clients.

#### t\_base

Unlike DAICS, where  $t\_base$  is calculated only once (as the threshold model alone assesses concept drift), this solution recalculates  $t\_base$  whenever the autoencoder is retrained.

Update t\_base