Time Series Anomaly Detection for Cyber-physical Systems via Neural System Identification and Bayesian Filtering

Cheng Feng
cheng.feng@siemens.com
Siemens AG
Beijing, China

Pengwei Tian
pengwei.tian@siemens.com
Siemens AG
Beijing, China

ABSTRACT

Recent advances in AIoT technologies have led to an increasing popularity of utilizing machine learning algorithms to detect operational failures for cyber-physical systems (CPS). In its basic form, an anomaly detection module monitors the sensor measurements and actuator states from the physical plant, and detects anomalies in these measurements to identify abnormal operation status. Nevertheless, building effective anomaly detection models for CPS is rather challenging as the model has to accurately detect anomalies in presence of highly complicated system dynamics and unknown amount of sensor noise. In this work, we propose a novel time series anomaly detection method called Neural System Identification and Bayesian Filtering (NSIBF) in which a specially crafted neural network architecture is posed for system identification, i.e., capturing the dynamics of CPS in a dynamical state-space model; then a Bayesian filtering algorithm is naturally applied on top of the "identified" state-space model for robust anomaly detection by tracking the uncertainty of the hidden state of the system recursively over time. We provide qualitative as well as quantitative experiments with the proposed method on a synthetic and three real-world CPS datasets, showing that NSIBF compares favorably to the state-of-the-art methods with considerable improvements on anomaly detection in CPS.

CCS CONCEPTS

• **Computer systems organization** → **Embedded and cyber-physical systems**; • **Computing methodologies** → **Neural networks**; **Anomaly detection**; • **Mathematics of computing** → *Bayesian computation*.

KEYWORDS

Time series; Anomaly detection; Cyber-physical systems; Neural networks; System identification; Bayesian filtering

1 INTRODUCTION

Cyber-physical systems (CPS) are commonly used for the monitoring and controlling of industrial processes in important infrastructure assets such as power plants, gas pipelines and water treatment facilities. The active monitoring of the sensor readings and actuator states in such systems for early detection of unexpected system behavior is highly critical as it allows for timely mitigating actions – such as fault checking, predictive maintenance and system shutdown – to be taken such that potential economical and environmental loss caused by system failures can be avoided. However, with the increasing complexity of modern CPS, traditional anomaly detection mechanisms based on manually crafted evolution rules in CPS becomes insufficient. As a result, utilizing machine learning algorithms to build data-driven anomaly detection frameworks in CPS has become a clear tendency thanks to the recent advances in AIoT (AI+IoT) techniques.

Detecting anomalous behavior on a set of correlated time series signals has been an active research area in the machine learning community for a long time [9, 10]. Many approaches have been proposed for this multivariate time series anomaly detection task. For example, the traditional unsupervised anomaly detection algorithms such as one-class SVM [44] and isolation forest [40] can be applied on the CPS data with time-series patterns ignored or implicitly captured. A more commonly used method is the residual-error based anomaly detection. Specifically, the residual-error based anomaly detection either relies on a predictive model such as recurrent neural networks [22] to predict future sensor measurements, or a reconstruction model such as autoencoders [37] to compress sensor measurements to lower dimensional embeddings and reconstruct them afterwards. Then the predicted or reconstructed measurements are compared with observed measurements to generate a residual error. An anomaly is detected if the residual error exceeds a predefined threshold. In practice, a robust threshold on residual errors of sensor measurements in CPS is however hardly definable because of the uncertainties from various sources such as sensor noise caused by imperfect manufacturing and process noise caused by imperfect control. As a result, although such methods have shown superior performance on many conventional time series anomaly detection tasks [7, 11, 29, 43, 47, 56], their performance may deteriorate on the CPS data where the signals contain unstable noises over time. Density-based anomaly detection can mitigate this issue by modelling the likelihood of observed measurements. For example, state estimation algorithms [15] that utilize state-space models and the Kalman filter [13] to track the uncertainty of the

true system states under noisy measurements are commonly used for CPS due to their robustness to sensor and process noises. However, as CPS become more ubiquitous and commoditized with the trend of IoT, the requirement of heavy domain knowledge and lack of generality often restrict their applicability in practice. Recent work on density-based anomaly detection based on deep learning and variational methods [5, 53, 55, 57] show promising result on several time series tasks, however, their performance on CPS data needs to be further studied.

In this work, we propose a novel density-based time series anomaly detection framework for CPS called Neural System Identification and Bayesian Filtering (NSIBF). Specifically, we leverage an end-to-end trained neural network with a specialized architecture to capture the dynamics of CPS via a state-space model. In the detection phase, a Bayesian filtering algorithm is naturally applied on top of the "identified" state-space model to track the uncertainty of the hidden states of the system and estimate the likelihood of observed sensor measurements over time. NSIBF is a combination of neural networks and the traditional state estimation method for CPS. As a result, NSIBF benefits from both: the generality and ability to capture highly nonlinear system dynamics; and the robustness to process and sensor noises in CPS. Extensive qualitative and quantitative experiments on a synthetic and three real-world CPS datasets show that our method achieves superior performance compared with the state-of-the-art time series anomaly detection methods. We summarize the main contributions of our paper as follows:

- We propose a novel time series anomaly detection method based on neural network-based system identification and Bayesian filtering that is robust to process and sensor noises in CPS.
- We studied the performance of various start-of-the-art time series anomaly detection methods on real-world CPS datasets.
- Through both qualitative and quantitative experiments, we show that our proposed method consistently outperforms the start-of-the-art methods on anomaly detection for CPS.
- We publish our data and code in a GitHub repository[1] for better reproducibility of our experimental results.

The rest of this paper is structured as follows. We briefly introduce the commonly used anomaly detection method for CPS based on state estimation in the next section. In Section 3 our proposed method NSIBF is presented. Section 4 and 5 present the qualitative and quantitative experiments, respectively. Related work is discussed in Section 6. Lastly, Section 7 draws final conclusions.

2 BACKGROUND

In a typical CPS, the physical processes are directly controlled by a network consisting of three types of field devices called sensors, actuators and Programmable Logical Controllers (PLCs) [38]. Specifically, the sensors are devices which convert physical parameters into electronic measurements; actuators are devices which convert control commands into physical state changes (e.g., open or close a valve); based on measurements received from sensors, PLCs send control commands to actuators to activate physical state changes.
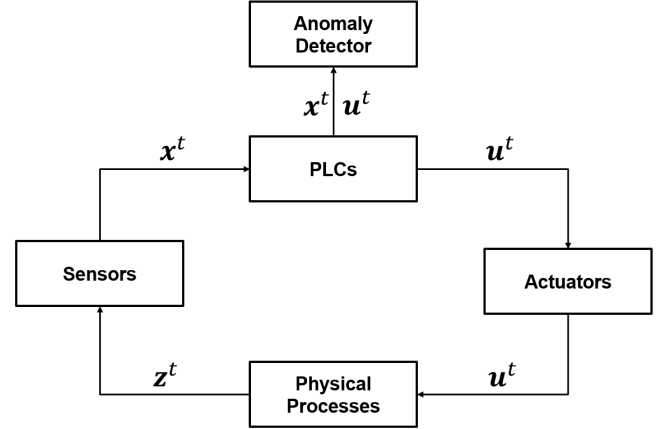
[1]https://github.com/NSIBF/NSIBF

Figure 1: Anomaly detection architecture in CPS.

To protect CPS from system failures, the physical evolution of system states are monitored via the sensor measurements and actuator states at discrete time steps. A commonly used anomaly detection architecture for CPS is illustrated in Figure 1. Concretely, let $\mathbf{x}^t$ be the measurements from the sensors at time t, $\mathbf{u}^t$ be the states of actuators at time t, the dynamics of CPS is commonly represented by a state space model as follows:

$$
\begin{align}
\mathbf{z}^t &= F\mathbf{z}^{t-1} + B\mathbf{u}^{t-1} + \boldsymbol{\epsilon}^t \tag{1}\\
\mathbf{x}^t &= H\mathbf{z}^t + \boldsymbol{\varepsilon}^t \tag{2}
\end{align}
$$

where $\mathbf{z}^t$ is the hidden state variables (e.g., the real temperature of liquids), F, B, H are the state transition, control and measurement matrices which define the dynamics of the system, $\boldsymbol{\epsilon}^t$ and $\boldsymbol{\varepsilon}^t$ are vectors of noise for state transition and sensor measurements with a random process with zero mean. On top of the state space model, Bayesian filters such as Kalman filter [27] and its variants [13] are commonly used to estimate the joint probability distribution of the state variables recursively over time. To detect unexpected behaviors, we need to use $\mathbf{x}^{t-1}$ and $\mathbf{u}^{t-1}$ to obtain the prior state estimation for $\mathbf{z}^t$, and then predict the corresponding distribution for $\mathbf{x}^t$. An alert is then raised when the residual error between observed measurements and the predicted mean above a threshold or the likelihood of observed measurements below a threshold [19].

Despite many successful applications of the above state estimation technique in the past decades, there are some key limitations which restrict its applicability for anomaly detection of CPS in the IoT era. First of all, the step to building up the mathematical models of the process dynamics for establishing the state-space model is called "system identification" [41]. This step is generally very challenging in practice as it requires heavy prior knowledge of the system dynamics. Furthermore, for security, privacy and other technical issues, it is often impossible for anomaly detection modules to access the control logic of the system [23]. In these cases, the anomaly detection model shall be able to detect unexpected system behaviors with only a partial part of the signals being accessible. Most importantly, with the growing scale and complexity of modern CPS, traditional state space models defined by domain

knowledge may not be able to capture the complex dynamics of such systems any more.

3 NEURAL SYSTEM IDENTIFICATION AND BAYESIAN FILTERING

In this section our proposed system identification method for CPS based on a specialized neural network architecture, and the corresponding Bayesian filtering method for robust time series anomaly detection are presented.

3.1 Neural System Identification

Without loss of generality, let $\mathbf{x}^t$ and $\mathbf{u}^t$ be the vectors for the monitored sensor measurements and actuator states of a CPS at a discrete time point t, we propose the neural network architecture in Figure 2 to capture the dynamics of the system. Specifically, our proposed architecture consists of three subnets which are called the g net, f net and h net:

- The g net, parameterized by ω, is a feed-forward network which takes the sensor measurements at time point $t-1$ as input, and encodes the measurements into a lower dimensional hidden state vector $\mathbf{z}^{t-1}$.
- The f net, parameterized by θ, takes the sensor measurements and actuator states in a sliding window of length l as input, and encodes them into a hidden vector $\mathbf{h}^{t-1}$ using recurrent layers such as LSTM [22]. The vector $\mathbf{h}^{t-1}$ provides the time-series context for hidden state transition. Specifically, together with $\mathbf{h}^{t-1}$, the f net also takes the hidden state vector $\mathbf{z}^{t-1}$ as input, and then leverages dense layers to predict the hidden state vector at the next time point $\mathbf{z}^t$.
- The h net, parameterized by φ, is a feed-forward network which takes a hidden state vector as input, and decodes the hidden state vector to the corresponding sensor measurements. Note that the two h nets in the architecture are identical meaning that they share the same parameters and weights.

Overall, we denote the whole network model as $F_{\omega,\theta,\varphi}$. The network has two inputs: a vector $\mathbf{x}^{t-1}$ which is the sensor measurements at time point $t-1$, and a sequence $(\mathbf{x}, \mathbf{u})^{t-l:t-1}$ which contains the sensor measurements and actuator states in the sliding window of the past l time points. There are also two outputs for the network: $\widetilde{\mathbf{x}}^{t-1}$ is the reconstructed vector of sensor measurements at time point $t-1$, $\tilde{\mathbf{x}}^t$ is the predicted sensor measurements at time point t.

$F_{\omega,\theta,\varphi}$ is trained in an end-to-end fashion. For anomaly detection purpose, the model shall be trained with CPS datasets with only normal data. Specifically, consider a CPS dataset with only normal data, we split the dataset into a training set D_{tr} and a validation set D_{val}. Assume there are T time points in D_{tr}, the loss function of the model is calculated as follows:

$$
\begin{aligned}
L(\omega,\theta,\varphi) \quad = \quad & \sum_{t=l}^{T} w_1 ||\mathbf{x}^{t-1} - \widetilde{\mathbf{x}}^{t-1}||_2^2 + w_2 ||\mathbf{x}^t - \widetilde{\mathbf{x}}^t||_2^2 \\
& + w_3 ||\mathbf{z}^t - \mathbf{z}^{t-1}||_2^2
\end{aligned}
\tag{3}
$$

in which the first two terms of L are the reconstruction and prediction errors of sensor measurements, the third term is a factor to make consecutive hidden state vectors be close to each other

for capturing the temporal smoothness exists among system states. w_1, w_2, w_3 are hyperparameters representing the weights of the three terms.

After the model has been trained using stochastic gradient descent algorithms such as Adam [30], we represent the dynamics of the CPS by the following dynamical state-space model:

$$
\mathbf{z}^t \quad = \quad f_\theta\big(\mathbf{z}^{t-1}, (\mathbf{x}, \mathbf{u})^{t-l:t-1}\big) + \mathbf{e}_f^t
\tag{4}
$$

$$
\mathbf{x}^t \quad = \quad h_\varphi(\mathbf{z}^t) + \mathbf{e}_h^t
\tag{5}
$$

where $\mathbf{e}_f^t$ and $\mathbf{e}_h^t$ are vectors of the hidden state prediction error and the sensor measurement reconstruction error at time point t respectively. Furthermore, we assume $\mathbf{e}_f^t$ and $\mathbf{e}_h^t$ follow multivariate Gaussian distribution with zero means and covariance matrices Q and R respectively. Specifically, Q is the covariance matrix for hidden state prediction errors empirically evaluated based on the time points in D_{val}, and the hidden state prediction error at each time point in D_{val} is calculated as follows:

$$
\mathbf{e}_f^t = g_\omega(\mathbf{x}^t) - f_\theta\big(g_\omega(\mathbf{x}^{t-1}), (\mathbf{x}, \mathbf{u})^{t-l:t-1}\big)
\tag{6}
$$

R is the covariance matrix for sensor measurement reconstruction errors which is also empirically computed based on the time points of D_{val}, and the sensor measurement reconstruction error at each time point is calculated as follows:

$$
\mathbf{e}_h^t = \mathbf{x}^t - h_\varphi\big(g_\omega(\mathbf{x}^t)\big)
\tag{7}
$$

Notably the above dynamical state-space model is similar to the one as illustrated in Section 2. The differences are that we replace the state transition and control matrices by the f net; the measurement matrix by the h net; and the process and sensor measurement noises by hidden state prediction and sensor measurement reconstruction errors, respectively.

3.2 Bayesian Filtering and Anomaly Detection

On top of the dynamical state-space model constructed by neural system identification, in the detection phase we keep track of the probability distribution of the hidden states of the CPS via a Bayesian filtering method, and anomalies are detected by estimating the likelihood of observed sensor measurements over time. Our method is inspired by the unscented Kalman filters [26] which are commonly used for Bayesian state estimation for nonlinear state-space models. Compared with other nonlinear filtering methods such as particle filters [6, 31] and extended Kalman filters [49, 51], unscented Kalman filters often achieve a better balance between accuracy and efficiency [36].

Concretely, assume the hidden states of the system follow a multivariate Gaussian distribution, let m denote the dimension of the hidden states, we use $\bar{\mathbf{z}}^t \in \mathbb{R}^m$ and $P^t \in \mathbb{R}^{m \times m}$ to denote the mean vector and the covariance matrix of the distribution at time t. We initialize $\bar{\mathbf{z}}^0 = g_\omega(\mathbf{x}^0)$, $P^0 = \epsilon I_m$, where ϵ is a value close to zero. Then, we estimate $\bar{\mathbf{z}}^t$ and P^t, and conduct anomaly detection following the two steps as described below recursively over time.

3.2.1 The prediction step. In this step, we predict the mean and covariance of hidden state distribution at time t according to known $\bar{\mathbf{z}}^{t-1}$ and P^{t-1}. Concretely, we first generate a set of sigma points $Z \in \mathbb{R}^{n \times m}$ and their corresponding weights $\mathbf{w}^m \in \mathbb{R}^n$ and $\mathbf{w}^c \in \mathbb{R}^n$ for mean and covariance functions respectively to efficiently
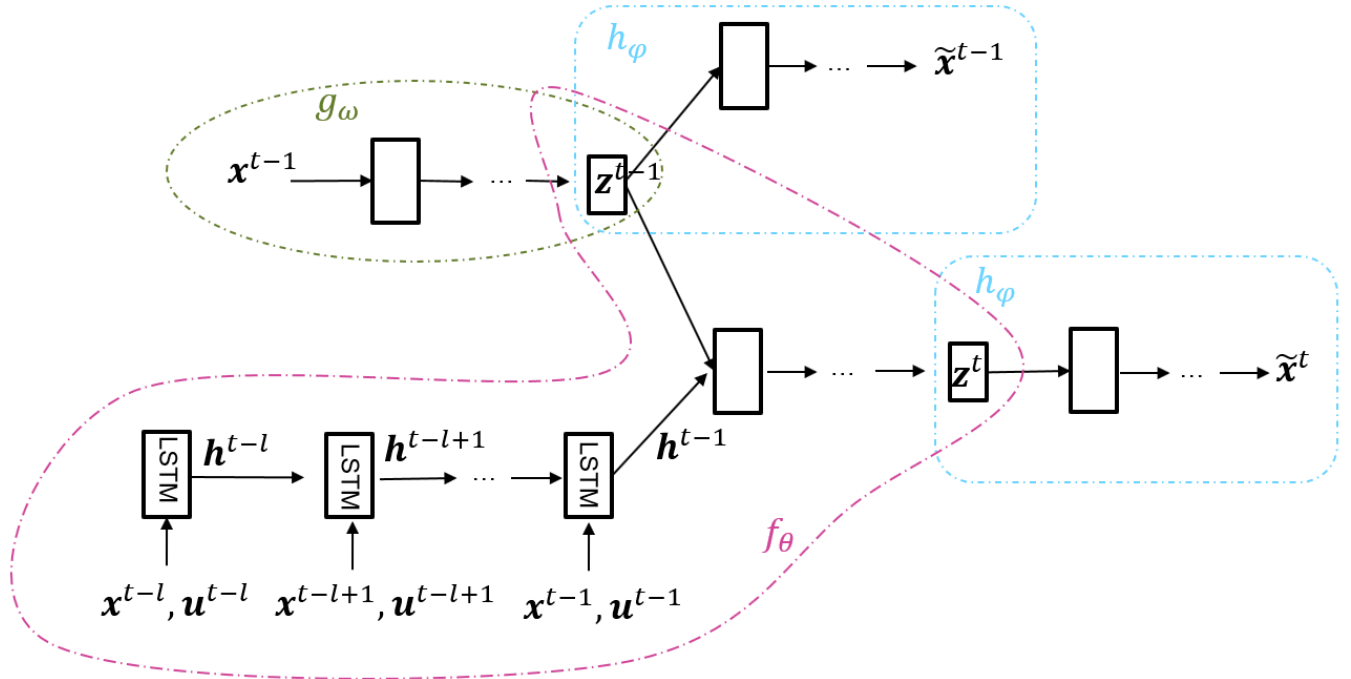
Figure 2: Proposed neural network architecture for system identification in CPS.

represent the hidden state distribution at time $t - 1$, where n is the number of generated sigma points. These sigma points are smartly selected using the Julier sigma function as introduced in [25] such that:

$$Z, \mathbf{w}^m, \mathbf{w}^c = \text{Sigma Function}(\bar{\mathbf{z}}^{t-1}, P^{t-1}) \quad (8)$$

The details of the sigma function is omitted here as it is not the focus of the paper. Then, we pass the selected sigma points through the f net to generate a set of predicted state vectors at time t, such that

$$Y = f_\theta(Z, (\mathbf{x}, \mathbf{u})^{t-l:t-1}) \quad (9)$$

After that, we predict the mean and covariance of the state distribution at time t by the unscented transform function as follows:

$$\hat{\mathbf{z}}^t = \sum_i \mathbf{w}_i^m Y_i \quad (10)$$

$$\hat{P}^t = \sum_i \mathbf{w}_i^c (Y_i - \hat{\mathbf{z}}^t)(Y_i - \hat{\mathbf{z}}^t)^T + Q \quad (11)$$

$\hat{\mathbf{z}}^t$ and $\hat{P}^t$ are also treated as the prior mean and covariance of the hidden state distribution at time t.

3.2.2 The update and anomaly detection step. In this step, given the observed sensor measurements, we conduct anomaly detection and update the posterior mean and covariance for the hidden state distribution at time t. Concretely, we firstly decode the set of predicted state vectors Y at time t to their corresponding sensor measurements using the h net:

$$X = h_\varphi(Y) \quad (12)$$

Again, we predict mean and covariance for the sensor measurements at time t through the unscented transform function:

$$\mu = \sum_i \mathbf{w}_i^m X_i \quad (13)$$

$$\Sigma = \sum_i \mathbf{w}_i^c (X_i - \mu)(X_i - \mu)^T + R \quad (14)$$

Then, we derive an anomaly score for the observed measurements $\mathbf{x}^t$ by computing the c distance [45] between $\mathbf{x}^t$ and its predicted distribution as follows:

$$\text{anomaly score} = \sqrt{(\mathbf{x}^t - \mu)^T \Sigma^{-1} (\mathbf{x}^t - \mu)} \quad (15)$$

A larger anomaly score indicates it is more likely for $\mathbf{x}^t$ to be an anomaly. In practice, an anomaly threshold can be tuned by methods such as setting an acceptable false alarm rate or mean time between false alarms.

To obtain the posterior mean and covariance for the state distribution at time t, we calculate the Kalman gain K and then update $\bar{\mathbf{z}}^t$ and P^t by fusing the prior estimation and the observed measurements under the Kalman framework:

$$K = [\sum_i \mathbf{w}_i^c (Y_i - \hat{\mathbf{z}}^t)(X_i - \mu)^T]\Sigma^{-1} \quad (16)$$

$$\bar{\mathbf{z}}^t = \hat{\mathbf{z}}^t + K(\mathbf{x}^t - \mu) \quad (17)$$

$$P^t = \hat{P}^t - K\Sigma K^T \quad (18)$$

3.3 Algorithm and Implementation

To sum up, we give our method for time series anomaly detection for CPS in Algorithm 1. The method is implemented using the TensorFlow [1] and FilterPy [35] open source libraries.

It is worth noting that although our method is described for point-wise detection, in practice we can also conduct sequence-level detection by stacking the sensor measurements in several consecutive time points into a vector form as the input for g net. In this way the $\mathbf{x}^t$ in our proposed method becomes the stacked sensor measurements in consecutive time points. As a result, some temporal features of the measurements are also encoded in the learned hidden states and the anomalies are detected in the sequence level.

Algorithm 1 Neural system identification and Bayesian filtering for anomaly detection

---

Require: A CPS dataset without anomalies
 1: **The training phase:**
 2: Split the dataset into a training set D_{tr} and a validation set D_{val}.
 3: Train the neural network $F_{\omega,\theta,\varphi}$ on D_{tr}.
 4: Estimate the covariance matrices Q and R using D_{val}.
 5: **The detection phase:**
 6: Initialize $\bar{\mathbf{z}}^0 = g_\omega(\mathbf{x}^0)$, $P^0 = \epsilon I_m$
 7: **loop**
 8:   Predict $\hat{\mathbf{z}}^t$ and $\hat{P}^t$, the prior mean and covariance of the state distribution at time t by Equations (8-11).
 9:   Predict $\boldsymbol{\mu}$ and Σ, the mean and covariance of the sensor measurements at time t by Equations (12-14).
10:   Obtain $\mathbf{x}^t$ and $\mathbf{u}^t$, the sensor measurements and actuator states at the current time point.
11:   Compute the anomaly score at time t by Equation (15).
12:   Update $\bar{\mathbf{z}}^t$ and P^t, the posterior mean and covariance of hidden states at time t by Equations (16-18).
13: **end loop**

---

4 QUALITATIVE EXPERIMENTS

In our qualitative experiment, we test whether our method benefits from the Bayesian filtering approach for anomaly detection compared with merely residual error-based detection. For this, we consider a simple illustrative CPS example which simulates a noisy sine wave as follows:

$$t\%30 = 0 \quad \longrightarrow \quad u^t = 9 - u^t \tag{19}$$
$$z^t = \sin(t/u^t) + \epsilon^t \tag{20}$$
$$x^t = 2 * z^t + \varepsilon^t \tag{21}$$

where $t = 1, 2, ..., N$, z^t is the hidden state of the system, u^t is the state of the actuator to control the frequency of the sinusoid, x^t is the observed sensor measurement; $\epsilon^t \sim \mathcal{N}(0, 0.1^2)$ is the process noise, $\varepsilon^t \sim \mathcal{N}(0, 0.2^2)$ is the measurement noise. With $u^0 = 3$, we simulate the system for 10000 time points to generate the training data. Furthermore, we simulate the system for another 10000 time points to generate the testing data in which anomalies are injected. Specifically, anomalies are injected consecutively for 100 time points

in every 1000 time points. During the anomalous periods, we set $\epsilon^t \sim \mathcal{N}(0, 0.6^2)$.

In our experiment, we use the data in the first 7500 time points of the training data to train our proposed neural network for system identification, the data in the remaining 2500 time points are used as the validation set. Furthermore, we conduct sequence-level anomaly detection by which we stack the sensor measurements in 31 consecutive time points (to cover at least one cycle of actuator state change) into a vector and then compress them to a two dimensional hidden state using the g net. Then, the f net with $l = 62$ (each sliding window consists of 62 time points) is used to predict the hidden state for the next 31 time points. Lastly, the h net is used to reconstruct the sensor measurements in the current 31 time points as well as to decode the measurements in the next 31 time points.

As can be noticed, using our trained neural network for system identification $F_{\omega,\theta,\varphi}$, we can have three methods for anomaly detection: 1) The first method is to leverage Bayesian filtering as described in Section 3, we denote this method as NSIBF; 2) The second method is to directly use the residual errors between the reconstructed measurements generated by $F_{\omega,\theta,\varphi}$ and the observed measurements to detect anomalies, we denote this method as NSIBF-RECON; 3) The third method is to directly use the residual errors between the predicted measurements generated by $F_{\omega,\theta,\varphi}$ and the observed measurements to detect anomalies, we denote this method as NSIBF-PRED. We use the above three methods to detect the anomalies in the testing data. We illustrate the observed measurements and the generated measurements by the three methods in one anomalous period in Figure 3a. Furthermore, the anomaly scores generated by NSIBF as well as the residual errors generated by NSIBF-RECON and NSIBF-PRED for the whole testing phase are shown in Figure 3b. The corresponding receiver operating characteristic (ROC) curves for the anomaly detection performances of the three methods are given in Figure 3c. As can be seen from Figure 3b and 3c, it is harder to detect the injected anomalies by using the residual errors. However, using the anomaly scores generated by NSIBF, we can easily detect those anomalies. We believe there are two reasons: first, density-based method is more capable than residual error-based method to detect anomalies in noisy measurements; second, the reconstructed and predicted measurements by NSIBF-RECON and NSIBF-PRED are more likely to overfit the noises. But NSIBF with a underlying mechanism to track the uncertainty of the hidden states by fusing the state prediction and the observed measurements via Bayesian filtering is more robust to such noises. Thus we can see that measurements generated by NSIBF can still have a relatively stable pattern during the anomalous period as illustrated in Figure 3a.

5 QUANTITATIVE EXPERIMENTS

In our quantitative experiments, we apply our method for anomaly detection in three real-world CPS datasets and compare the results with various state-of-the-art anomaly detection methods.

5.1 Datasets

We consider the following three real-world CPS datasets:
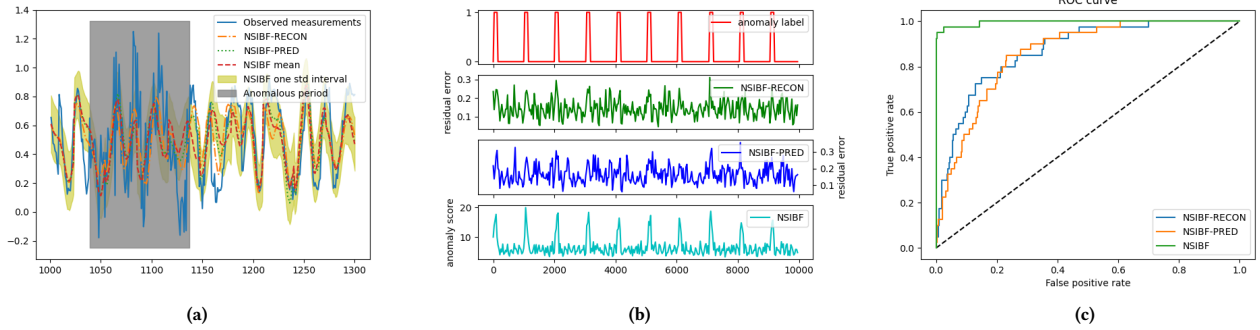
Figure 3: (a): the observed measurements and generated measurements by NSIBF-RECON, NSIBF-PRED and NSIBF in one anomalous period; (b): the trace of anomaly scores generated by NSIBF and residual errors generated by NSIBF-RECON and NSIBF-PRED; (c): the ROC curve for NSIBF-RECON, NSIBF-PRED and NSIBF for anomaly detection of the simulated CPS example in the qualitative experiment.

Table 1: The percentage of anomalies within the test data and the details of benchmark CPS datasets.

	Train size	Test size	Num. sensors	Num. actuators	Anomalies (%)
PUMP	76901	143401	44	0	10.05
WADI	241921	15701	67	26	7.09
SWAT	99360	89984	25	26	11.99

- **PUMP**: This dataset is collected from a water pump system of a small town. The data is collected every minute for five months.
- **WADI**: This dataset is collected from a fully operational physical testbed that represents a scaled-down version of a real urban water distribution system [2]. The data is collected every second for 16 days. In our experiments, we downsampled it to a data point in every five seconds and the data in the last day is ignored as they have different distributions with previous 15 days due to change of operational mode.
- **SWAT**: This dataset is collected from a CPS testbed that is a scaled down water treatment plant producing filtered water [20]. The data is collected every second for 11 days. In our experiments, we downsampled it to a data point in every five seconds.

More details of the benchmark datasets are given in Table 1. Note that all the datasets are labelled by domain experts and anomalies only exist in the test data.

5.2 Baseline Anomaly Detection Methods

We select several state-of-the-art anomaly detection methods as baselines.

- **Isolation Forest**: Isolation Forest [40] is a popular anomaly detection algorithm that works on the principle of isolating anomalies using tree-based structures.

- **Sparse-AE**: Sparse-AE [46] is a residual error-based anomaly detection algorithm using an autoencoder with sparse hidden embeddings as a reconstruction model.
- **EncDec-AD**: EncDec-AD [43] is a residual error-based time series anomaly detection algorithm using an LSTM-based encoder-decoder as a reconstruction model.
- **LSTM-PRED**: LSTM-PRED [21] is a residual error-based time series anomaly detection algorithm using an LSTM-based regressor as a predictive model.
- **DAGMM**: DAGMM [57] is a density-based anomaly detection algorithm using a deep generative model that assumes the Gaussian mixture prior in the latent space to estimate the likelihood of input samples.
- **OmniAnomaly**: OmniAnomaly [53] is a density-based time series anomaly detection algorithm that employs a deep generative model with stochastic variable connection and planar normalizing flow to describe non-Gaussian distributions of latent space, and uses the reconstruction probabilities of input samples as anomaly scores.
- **USAD**: USAD [7] is a residual error-based time series anomaly detection algorithm that employs an autoencoder as a reconstruction model. Particularly, the autoencoder is adversarial trained to amplify the reconstruction errors for anomalies.

Note that since Isolation Forest, Sparse-AE and DAGMM are not anomaly detection methods for time series, we stack sensor measurements and actuator states in multiple consecutive time points to form a vector as the inputs for these models such that they can also capture the temporal dependence of signals in the benchmark datasets. Specifically, we stack signals of five time points for the PUMP dataset, and 12 time points for the WADI and SWAT datasets. With respect to NSIBF, we also stack the sensor measurements in five consecutive time points for the PUMP dataset and 12 time points for the WADI and SWAT datasets into vectors as the input for g net. The weights w_1, w_2, w_3 in the loss function of Equation (3) are set to 0.45, 0.45 and 0.1 for all the experiments (including the

Table 2: Comparison of the performance metrics of anomaly detection methods on benchmark CPS datasets.

MODEL	PUMP			WADI			SWAT		
	PRE	REC	F1	PRE	REC	F1	PRE	REC	F1
ISOLATION FOREST	0.977	0.582	0.729	0.826	0.772	0.798	0.975	0.754	0.850
SPARSE-AE	0.798	0.737	0.767	0.769	0.771	0.770	0.999	0.666	0.799
ENCDEC-AD	0.438	0.796	0.565	0.589	0.887	0.708	0.945	0.620	0.748
LSTM-PRED	0.925	0.581	0.714	0.620	0.876	0.726	0.996	0.686	0.812
DAGMM	0.931	0.798	0.860	0.886	0.772	0.825	0.946	0.747	0.835
OMNIANOMALY	0.937	0.840	**0.886**	0.846	0.893	**0.869**	0.979	0.757	**0.854**
USAD	0.984	0.582	0.732	0.806	0.879	0.841	0.987	0.740	0.846
NSIBF-RECON	0.859	0.798	0.827	0.814	0.495	0.615	0.999	0.667	0.800
NSIBF-PRED	0.897	0.798	0.845	0.685	0.876	0.769	0.982	0.697	0.815
NSIBF	0.936	0.889	**0.912**	0.915	0.887	**0.901**	0.982	0.863	**0.919**

(a)　　　　　　　　　　　　　　(b)　　　　　　　　　　　　　　(c)
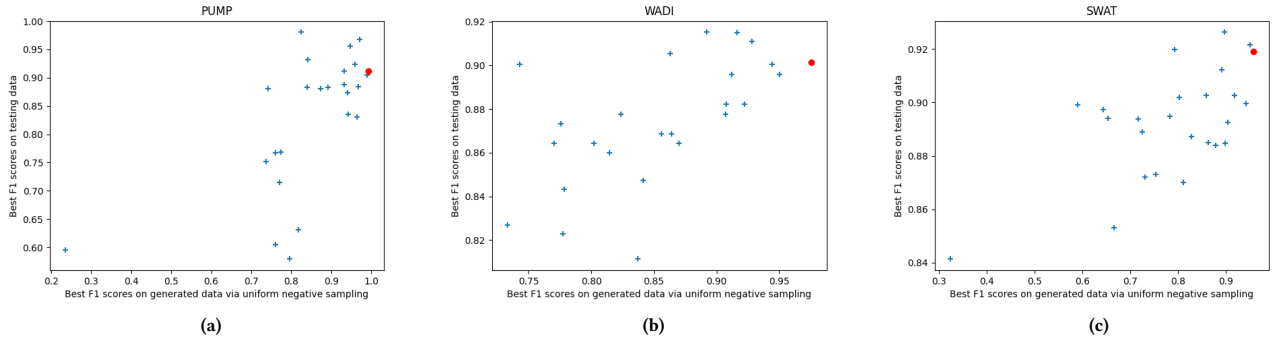
Figure 4: The best F1 scores on the generated datasets via uniform negative sampling during hyperparameter tuning, and the best F1 scores if the corresponding models are used for anomaly detection on the testing data for the three benchmark datasets. For each benchmark dataset, the red circle is selected as the metric for comparing with baselines in Table 2.

qualitative one) as we find it consistently achieves a good performance. For all datasets, we split the train data to a training set and a validation set with the ratio 3:1. To demonstrate the importance of Bayesian filtering, we also include NSIBF-RECON and NSIBF-PRED as baselines.

5.3 Uniform Negative Sampling for Guiding Hyperparameter Tuning

To effectively train the neural network for system identification in NSIBF, there are quite a few hyperparameters such as the dimension of the hidden state vector, the number of hidden layers and the hidden dimensions for the three subnets. We can simply use the loss function in Equation (3) as the criterion to tune hyperparameters, however it is not optimal for anomaly detection. On the other hand, in practice it is generally impossible to have a dataset with sufficient labelled anomalies to test our model's anomaly detection performance. Therefore, we propose to use the uniform negative sampling algorithm as introduced in [52] to generate anomalous data from the validation set D_{val} that only contains normal data. Different from [52] where the uniform negative sampling algorithm is used to generate anomalous data for training classification models

for anomaly detection, we utilize the generated anomalous data to guide the process of hyperparameter tuning of our model.

Specifically, we generate datasets by injecting anomalous data into D_{val}. Let M be the dimension of $\mathbf{x}^t$, N be the number of time points in D_{val}, for each dimension $d \leq M$, we define $lim_d = [min(U_d) - \delta, max(U_d) + \delta]$ as a range bounded by the extrema of the normal samples U_d extended by a conservative positive length δ that extends lim_d beyond the normal space. With a given sample ratio $r_s \in [0, 1]$, we randomly select $r_s \times N$ time points from D_{val}. For each selected time point t, we replace $\mathbf{x}_t$ by a vector whose value is uniformly sampled from lim_d for each $d \leq D$. Concretely, in our experiments, we generate 20 datasets with uniform negative sampling in which $r_s = [0, 0.05, 0.1, ..., 0.95]$ and $\delta = 0.05$ (after normalization). We use the randomised grid search algorithm to tune the hyperparameters of NSIBF, the model with the best overall anomaly detection performance (normal data are treated as outliers in the evaluation since most sequences will contain anomalies) on the 20 generated datasets is selected for anomaly detection on the testing data. The values and ranges for the hyperparameters of NSIBF are given in the appendix.

5.4 Evaluation Metrics and Results

As discussed in [55], for time series anomaly detection tasks, people generally do not care about the point-wise metrics. Instead, point-adjusted metrics are favored in which a contiguous anomalous segment is considered as successfully detected as long as any point inside the segment is classified as anomaly. Moreover, in practice it is more important to have an excellent F1 score at a certain threshold than to just have generally good performance on most thresholds. As a result, following the evaluation methods in [53], for the performance of a specific method on a dataset, we enumerate all possible anomaly thresholds to search for the best point-adjusted F1 score as the main evaluation metric.

The best F1 scores and their corresponding precision and recall for each method on all the benchmark datasets are reported in Table 2. Furthermore, we also report the best F1 scores on the generated datasets via uniform negative sampling during hyperparameter tuning, and the best F1 scores if the corresponding models are used for anomaly detection on the testing data for the three benchmark datasets in Figure 4. As can be seen from Figure 4, using uniform negative sampling for guiding hyperparameter tuning of NSIBF can generally provide a good model for anomaly detection even without real anomalies. By selecting the model with the best performance on the generated datasets via uniform negative sampling, we simulate the scenario in practice where the model (hyperparameter setting) with the best performance on testing data is generally unknown in advance. Importantly, from Table 2 we notice that NSIBF still achieves 2.9% improvement at the F1 score on the PUMP dataset, 3.7% on the WADI dataset and 7.6% on the SWAT dataset compared with the second best methods. Furthermore, NSIBF also significantly outperforms NSIBF-RECON and NSIBF-PRED on all the three datasets. These results clearly demonstrate the superiority of using a neural identified state-space model and Bayesian filtering to track the uncertainty of hidden states as the basis for anomaly detection in CPS signals with unknown amount of process and sensor measurement noises over time.

6 RELATED WORK

Deep learning-based time series anomaly detection in CPS has become an active research area in the recent years [42]. Till now, most of the methods focus on residual error-based detection. For example, recurrent neural networks (RNNs) are widely applied to capture the temporal dependency of time-series data in CPS [16, 24, 34, 54]. Autoencoders are also commonly used to capture the correlation of different sensors [4, 8]. Prediction errors and reconstruction errors are used as anomaly scores. We have shown that NSIBF which is a density-based detection method tailored for CPS, can achieve better anomaly detection performance than the commonly used residual error-based methods. There are also many other semi-data-driven methods for anomaly detection in CPS. Examples are those utilize invariant rules [12, 17], sensor and process noise fingerprints [3] in specific systems. NSIBF has better generality and lower implementing difficulty compared with those methods as it is purely data-driven and does not require any case-specific domain knowledge.

To avoid the need for explicit model specification and improve the ability to capture complex time series dynamics, combining

state-space models with deep neural networks has been proposed before. For example, the close relationship between RNNs and state space models that both use an internal hidden state to drive forecasts and observations is discussed in [14]. Applications of recurrent variational autoencoder architectures have been investigated for constructing non-linear state space models [28, 32, 33] and learning disentangled representations [18]. In [39, 48, 50], RNN-based deep state space models are proposed for probabilistic time series forecasting. Different from previous approaches which implicitly condense both the model specification and Bayesian filtering steps into the learned neural network, we explicitly decouple the two steps in NSIBF. In this way, the state transition (process) noises and sensor measurement noises that naturally exist in CPS are captured in our framework and this strategy can provide more accurate and consistent temporal information in noisy time series, making NSIBF tailored for anomaly detection in CPS data.

7 CONCLUSION

In this paper, we introduce a novel method, which we call the Neural System Identification and Bayesian Filtering (NSIBF), to learn system dynamics and conduct time series anomaly detection in CPS data. By leveraging the expressive power of neural networks and the ability of Bayesian filters to track uncertainties, our method can accurately detect anomalies in noisy sensor data from complex CPS. Based on the qualitative experiment on the simulated CPS example and the quantitative experiments on three real-world CPS datasets, the benefits of our method can be clearly seen from the improvements on the anomaly detection metrics compared with various start-of-the-art anomaly detection methods. Importantly, as our method is highly generalizable and easy to implement, we see great potential for application of NSIBF to a wide range of use-cases in the IoT applications where CPS become more ubiquitous and commoditized.

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*. 265–283.

[2] Chuadhry Mujeeb Ahmed, Venkata Reddy Palleti, and Aditya P Mathur. 2017. WADI: a water distribution testbed for research in the design of secure cyber physical systems. In *Proceedings of the 3rd International Workshop on Cyber-Physical Systems for Smart Water Networks*. 25–28.

[3] Chuadhry Mujeeb Ahmed, Jianying Zhou, and Aditya P Mathur. 2018. Noise matters: Using sensor and process noise fingerprint to detect stealthy cyber attacks and authenticate sensors in cps. In *Proceedings of the 34th Annual Computer Security Applications Conference*. 566–581.

[4] Abdulrahman Al-Abassi, Jacob Sakhnini, and Hadis Karimipour. 2020. Unsupervised Stacked Autoencoders for Anomaly Detection on Smart Cyber-physical Grids. In *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 3123–3129.

[5] Jinwon An and Sungzoon Cho. 2015. Variational autoencoder based anomaly detection using reconstruction probability. *Special Lecture on IE* 2, 1 (2015), 1–18.

[6] M Sanjeev Arulampalam, Simon Maskell, Neil Gordon, and Tim Clapp. 2002. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on signal processing* 50, 2 (2002), 174–188.

[7] Julien Audibert, Pietro Michiardi, Frédéric Guyard, Sébastien Marti, and Maria A Zuluaga. 2020. USAD: UnSupervised Anomaly Detection on Multivariate Time Series. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 3395–3404.

[8] Mikel Canizo, Isaac Triguero, Angel Conde, and Enrique Onieva. 2019. Multi-head CNN–RNN for multi-time series anomaly detection: An industrial case study. *Neurocomputing* 363 (2019), 246–260.

[9] Raghavendra Chalapathy and Sanjay Chawla. 2019. Deep learning for anomaly detection: A survey. *arXiv preprint arXiv:1901.03407* (2019).

[10] Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2009. Anomaly detection: A survey. *ACM computing surveys (CSUR)* 41, 3 (2009), 1–58.

[11] Jinghui Chen, Saket Sathe, Charu Aggarwal, and Deepak Turaga. 2017. Outlier detection with autoencoder ensembles. In *Proceedings of the 2017 SIAM international conference on data mining*. SIAM, 90–98.

[12] Yuqi Chen, Christopher M Poskitt, and Jun Sun. 2018. Learning from mutants: Using code mutation to learn and monitor invariants of a cyber-physical system. In *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 648–660.

[13] Charles K Chui, Guanrong Chen, et al. 2017. *Kalman filtering*. Springer.

[14] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C Courville, and Yoshua Bengio. 2015. A recurrent latent variable model for sequential data. *Advances in neural information processing systems* 28 (2015), 2980–2988.

[15] Derui Ding, Qing-Long Han, Xiaohua Ge, and Jun Wang. 2020. Secure State Estimation and Control of Cyber-Physical Systems: A Survey. *IEEE Transactions on Systems, Man, and Cybernetics: Systems* (2020).

[16] Cheng Feng, Tingting Li, and Deeph Chana. 2017. Multi-level anomaly detection in industrial control systems via package signatures and LSTM networks. In *2017 47th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 261–272.

[17] Cheng Feng, Venkata Reddy Palleti, Aditya Mathur, and Deeph Chana. 2019. A Systematic Framework to Generate Invariants for Anomaly Detection in Industrial Control Systems. In *26th Annual Network and Distributed System Security Symposium, NDSS 2019, San Diego, California, USA, February 24-27, 2019*. The Internet Society.

[18] Marco Fraccaro, Simon Kamronn, Ulrich Paquet, and Ole Winther. 2017. A disentangled recognition and nonlinear dynamics model for unsupervised learning. In *Advances in Neural Information Processing Systems*. 3601–3610.

[19] Jairo Giraldo, David Urbina, Alvaro Cardenas, Junia Valente, Mustafa Faisal, Justin Ruths, Nils Ole Tippenhauer, Henrik Sandberg, and Richard Candell. 2018. A survey of physics-based attack detection in cyber-physical systems. *ACM Computing Surveys (CSUR)* 51, 4 (2018), 1–36.

[20] Jonathan Goh, Sridhar Adepu, Khurum Nazir Junejo, and Aditya Mathur. 2016. A dataset to support research in the design of secure water treatment systems. In *International Conference on Critical Information Infrastructures Security*. Springer, 88–99.

[21] Jonathan Goh, Sridhar Adepu, Marcus Tan, and Zi Shan Lee. 2017. Anomaly detection in cyber physical systems using recurrent neural networks. In *2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE)*. IEEE, 140–145.

[22] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[23] Abdulmalik Humayed, Jingqiang Lin, Fengjun Li, and Bo Luo. 2017. Cyberphysical systems security—A survey. *IEEE Internet of Things Journal* 4, 6 (2017), 1802–1831.

[24] Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Soderstrom. 2018. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 387–395.

[25] Simon J Julier. 2002. The scaled unscented transformation. In *Proceedings of the 2002 American Control Conference (IEEE Cat. No. CH37301)*, Vol. 6. IEEE, 4555–4559.

[26] Simon J Julier and Jeffrey K Uhlmann. 2004. Unscented filtering and nonlinear estimation. *Proc. IEEE* 92, 3 (2004), 401–422.

[27] Rudolph Emil Kalman. 1960. A new approach to linear filtering and prediction problems. (1960).

[28] Maximilian Karl, Maximilian Soelch, Justin Bayer, and Patrick Van der Smagt. 2016. Deep variational bayes filters: Unsupervised learning of state space models from raw data. *arXiv preprint arXiv:1605.06432* (2016).

[29] Tung Kieu, Bin Yang, Chenjuan Guo, and Christian S Jensen. 2019. Outlier Detection for Time Series with Recurrent Autoencoder Ensembles.. In *IJCAI*. 2725–2732.

[30] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[31] Genshiro Kitagawa. 1996. Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *Journal of computational and graphical statistics* 5, 1 (1996), 1–25.

[32] Rahul Krishnan, Uri Shalit, and David Sontag. 2017. Structured inference networks for nonlinear state space models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 31.

[33] Rahul G Krishnan, Uri Shalit, and David Sontag. 2015. Deep kalman filters. *arXiv preprint arXiv:1511.05121* (2015).

[34] Sungmoon Kwon, Hyunguk Yoo, and Taeshik Shon. 2019. RNN-based anomaly detection in DNP3 transport layer. In *2019 IEEE International Conference on Communications, Control, and Computing Technologies for Smart Grids (SmartGridComm)*. IEEE, 1–7.

[35] Roger R Labbe. 2018. FilterPy Documentation. (2018).

[36] Roger R Labbe. 2019. *Kalman and Bayesian Filters in Python*.

[37] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature* 521, 7553 (2015), 436–444.

[38] Edward Ashford Lee and Sanjit A Seshia. 2017. *Introduction to embedded systems: A cyber-physical systems approach*. Mit Press.

[39] Bryan Lim, Stefan Zohren, and Stephen Roberts. 2019. Recurrent Neural Filters: Learning Independent Bayesian Filtering Steps for Time Series Prediction. *arXiv preprint arXiv:1901.08096* (2019).

[40] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2008. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*. IEEE, 413–422.

[41] Lennart Ljung. 1999. System identification. *Wiley encyclopedia of electrical and electronics engineering* (1999), 1–19.

[42] Yuan Luo, Ya Xiao, Long Cheng, Guojun Peng, and Danfeng Daphne Yao. 2020. Deep Learning-Based Anomaly Detection in Cyber-Physical Systems: Progress and Opportunities. *arXiv preprint arXiv:2003.13213* (2020).

[43] Pankaj Malhotra, Anusha Ramakrishnan, Gaurangi Anand, Lovekesh Vig, Puneet Agarwal, and Gautam Shroff. 2016. LSTM-based encoder-decoder for multi-sensor anomaly detection. *arXiv preprint arXiv:1607.00148* (2016).

[44] Larry M Manevitz and Malik Yousef. 2001. One-class SVMs for document classification. *Journal of machine Learning research* 2, Dec (2001), 139–154.

[45] Goeffrey J McLachlan. 1999. Mahalanobis distance. *Resonance* 4, 6 (1999), 20–26.

[46] Andrew Ng et al. 2011. Sparse autoencoder. *CS294A Lecture notes* 72, 2011 (2011), 1–19.

[47] Daehyung Park, Yuuna Hoshi, and Charles C Kemp. 2018. A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder. *IEEE Robotics and Automation Letters* 3, 3 (2018), 1544–1551.

[48] Syama Sundar Rangapuram, Matthias W Seeger, Jan Gasthaus, Lorenzo Stella, Yuyang Wang, and Tim Januschowski. 2018. Deep state space models for time series forecasting. *Advances in neural information processing systems* 31 (2018), 7785–7794.

[49] Maria Isabel Ribeiro. 2004. Kalman and extended kalman filters: Concept, derivation and properties. *Institute for Systems and Robotics* 43 (2004), 46.

[50] David Salinas, Valentin Flunkert, Jan Gasthaus, and Tim Januschowski. 2020. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting* 36, 3 (2020), 1181–1191.

[51] Stanley F Schmidt. 1966. Application of state-space methods to navigation problems. In *Advances in control systems*. Vol. 3. Elsevier, 293–340.

[52] John Sipple. 2020. Interpretable, multidimensional, multimodal anomaly detection with negative sampling for detection of device failure. In *International Conference on Machine Learning*. PMLR, 9016–9025.

[53] Ya Su, Youjian Zhao, Chenhao Niu, Rong Liu, Wei Sun, and Dan Pei. 2019. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2828–2837.

[54] Shahroz Tariq, Sangyup Lee, Youjin Shin, Myeong Shin Lee, Okchul Jung, Daewon Chung, and Simon S Woo. 2019. Detecting anomalies in space using multivariate convolutional LSTM with mixtures of probabilistic PCA. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2123–2133.

[55] Haowen Xu, Wenxiao Chen, Nengwen Zhao, Zeyan Li, Jiahao Bu, Zhihan Li, Ying Liu, Youjian Zhao, Dan Pei, Yang Feng, et al. 2018. Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In *Proceedings of the 2018 World Wide Web Conference*. 187–196.

[56] Chuxu Zhang, Dongjin Song, Yuncong Chen, Xinyang Feng, Cristian Lumezanu, Wei Cheng, Jingchao Ni, Bo Zong, Haifeng Chen, and Nitesh V Chawla. 2019. A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 1409–1416.

[57] Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeki Cho, and Haifeng Chen. 2018. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International Conference on Learning Representations*.

A BASELINE IMPLEMENTATIONS

Isolation Forest comes from the scikit-learn implementation[2]. DAGMM comes from a Tensorflow implementation in the Github repository[3]. OmniAnomaly comes from a Tensorflow implementation in the Github repository[4]. USAD comes from a PyTorch implementation in the Github repository[5]. Sparse-AE, EncDec-AD and LSTM-Pred are implemented by us using Tensorflow. All experiments are run on a Linux machine with 64 GiB memory, 8 4.2GHz Intel Cores and a GTX 1080 GPU.

B HYPERPARAMETER CONFIGURATIONS OF NSIBF

Hyperparameter	Dataset	Value or Range
w_1	All	0.45
w_2	All	0.45
w_3	All	0.1
l	PUMP	15
l	WADI	36
l	SWAT	36
Num. dimensions for $\mathbf{z}_t$	PUMP	$[1, 120]$
Num. dimensions for $\mathbf{z}_t$	WADI	$[1, 200]$
Num. dimensions for $\mathbf{z}_t$	SWAT	$[1, 75]$
Num. hidden layers for g net	All	$[1,3]$
Num. hidden layers for h net	All	$[1,3]$
Num. dense layers for f net	All	$[1,3]$
Num. LSTM layers for f net	All	$[1,3]$
Num. dimensions in hidden layers	All	$[32,256]$
Training epochs	All	100

---

[2]https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.IsolationForest.html
[3]https://github.com/tnakae/DAGMM
[4]https://github.com/NetManAIOps/OmniAnomaly
[5]https://github.com/robustml-eurecom/usad