Data Mining for the Security of Cyber Physical Systems Using Deep-Learning Methods

Bhagawan Nath[1,2], Timo Hamaleinen[1] and Soundararajan Ezekiel[3]
[1]University of Jyavaskyla, Finland
[2]Ton Duc Thang University, HCMC, Vietnam
[3]Indiana University of Pennsylvannia, PA, USA
bnath@student.jyu.fi
timo.t.hamalainen@jyu.fi
sezekiel@iup.edu

Abstract: Cyber Physical Systems (CPSs) have become widely popular in recent years, and their applicability have been growing exponentially. A CPS is an advanced system that incorporates a computation unit along with a hardware unit, allowing for computing processes to interact with the physical world. However, this increased usage has also led to the security concerns in them, as they allow potential attack vendors to exploit the possibilities of committing misconduct for their own benefit. It is of paramount importance that these systems have comprehensive security mechanisms to mitigate these security threats. A typical attack vector for a CPS is malicious data supplied by compromised sensors that are part of the CPSs. To combat this attack vector, many systems are secured through fault tolerance, including methods such as checkpointing to recover the system. Looking at the diverse nature of attacks and their ever growing complexities, traditional security approaches may not counter them efficiently, which creates a vacuum to be filled with sophisticated state-of-the-art techniques. In this paper, Deep Learning methods such as autoencoders, and Support Vector Machines are proposed to secure CPSs against these attacks. The networks in these applied methods are trained with a normal data profile devoid of any malicious data. Data collected from the system's sensors at specified intervals is used to form a data series and input to the neural networks. The networks compare and analyze new data to the normal profile to detect anomalies, if there is any. In the presence of anomalous data, the networks generate corrective action(s) for these sensors and the physical states they are recording. Through detection of anomalies, effective security of CPSs may be improved in addition to providing protection for the sensors. Moreover, the proposed method of securing CPSs opens up the possibility of further research by showcasing the applicability of neural networks in securing CPSs.

Keywords: Cyber Physical System, autoencoder, Support Vector Machine, Fault tolerance, Sensor Data, Cyber Attack

1. Introduction

Cyber Physical Systems (CPSs) have provided a way for communication and computing processes to interact through sensors and actuators with the physical world. As technology has progressed over the years and all systems have become connected to computer networks instead of being isolated and stand alone, thereby their functionality and interoperability needs to broaden the requirements for CPSs to interact with other systems in a more real-time fashion. As more services and applications have been added there has been an increase in the exposure area and hence exploitable vulnerabilities. The interactions a CPS has with the physical world are what makes this system more vulnerable to attacks that go beyond conventional cyber-attacks (Cardenas et al, 2008), (Meshram et al, 2017). It has been shown that the operation of a car can be disrupted and even disabled using simple methods (Checkoway et al, 2011). Another study shows that a Denial of Service attack can compromise the Controller Area Network bus and its functions on a system (Cho et al, 2016). A CPS can also be compromised through attacks on the sensors. By compromising the physical environment that the sensor reads, input can be generated that opens the system to attack. Other sensors that can be targeted by this are cameras, Light Detection and Ranging (LiDAR), and Global Positioning Systems (GPS) and can be on systems ranging from autonomous vehicles to ships.

The threat of these attacks has prompted many into finding ways to improve the attack resilience of CPSs, focusing specifically on the sensors, actuators, and communication components. Efficient solutions are typically achieved by developing a method that can approximate the normal states of the system accurately, allowing for continued control even with compromised components. This is advantageous, as both the error correction and the normal operations run from the same controller. This method has led into the creation of CPS checkpointing and recovery.

One way a Cyber Physical System can retain its integrity is through fault tolerance. In this process, the distinction between the physical and cyber physical worlds are utilized to provide seamless instruction, even upon system failure. This is achieved through roll-back – roll-forward schemes. Two operations take place in this scheme, the roll-back recovery focuses on the cyber state while the roll-forward recovery focuses on the physical state. The cyber states are composed of computational information, such as data values, and the physical states regard the physical information available, such as a light sensor detecting light. Roll-back refers to the CPS accessing the previous checkpoint, where it can then circumvent the error. Roll-forward recovery refers to keeping the system rolling at current time. This method essentisally handles failed estimated states caused by either attacks or failures.

This study is a based on a study by Njilla et al, 2019, titled "Internet of Things Anomaly Detection using Machine Learning", which investigates the application of DL methods (autoencoders) to detect anomalies. This study implements multiple DL models, autoencoder and one-class-SVM, specifically on CPS datasets and these models are tested to find the efficacy towards the stated problem.

The remainder of this paper is organized as follows: Section II describes the technical background of the techniques relevant in this study. Section III summarizes the methodology of securing CPSs with DL techniques to do such, that are used in this study. Section IV shows the results of our study and displays qualitative data to support these results. Section V discusses the impact of our study including the viability and effectiveness while exploring the future direction of our work.

2. Technical Background

2.1 Cyber Physical Systems

A Cyber Physical System is the improvement upon an embedded system, where a mechanism is either controlled or monitored by the system. These mechanisms can include autonomous vehicles, smart grids, robotic systems, and more. Rather than having just a single system controlling the machine, systems can be combined to create a system of systems. As demonstrated in Figure-1, these systems can share information, hardware, networks, and physical space to function more efficiently. In such a system, the details of its deployment must be clear during its creation, rather than on the go. This can be likened to cars at an intersection. In normal instances, the vehicles would communicate to each other in coming way to indicate who is going where. With a CPS, a system would emerge while the cars interact with each other to provide functionality but would not exist before or afterwards. This makes it so that the two vehicles would communicate to the intersection, rather than to each other, eliminating the need for hardware compatibility (Mosterman et al, 2016).

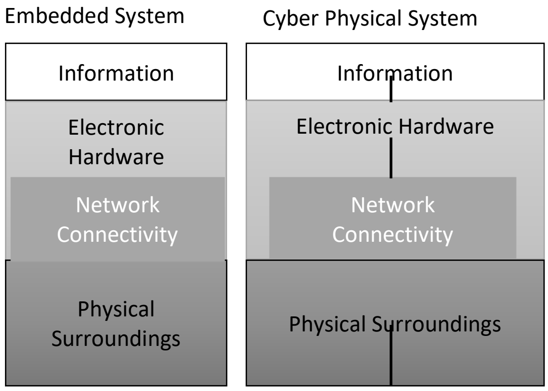

Figure-1: Embedded System vs a Cyber Physical System

There are numerous ways to model a CPS, with one of them being the 5-Layer model, shown in Figure-2, clearly demonstrates the architecture. At the base is connection, as acquiring data from the machines and components is the first step in a CPS. The correct sensors and equipment must be selected to give the system the most accurate data. Above that is the conversion layer, as the information gained through the connections must be translated into meaningful information. This allows for the machines in the CPS to have a form of self-awareness. At the centre is the cyber level, which acts as an information hub. Here, all the data is collected from the connected machines and analysed, allowing for the CPS to compare the performance of one machine to the rest in the system. The cognition level uses the previously obtained information and comparisons to make decisions on task prioritization. At the top is configuration, where feedback is given from the cyber side to physical side. Here the decisions made in the cognition level are passed to the system and monitored (Lee et al, 2015).

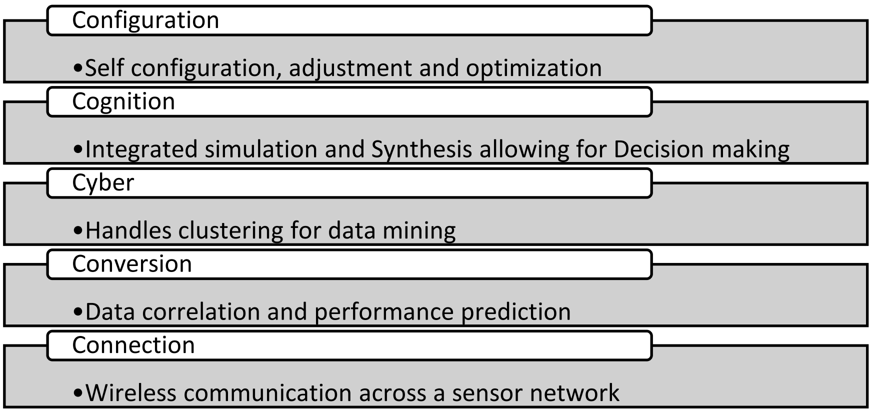
Configuration
•Self configuration, adjustment and optimization
Cognition
•Integrated simulation and Synthesis allowing for Decision making
Cyber
•Handles clustering for data mining
Conversion
•Data correlation and performance prediction
Connection
•Wireless communication across a sensor network

Figure-2: Five Layer Cyber Physical System Architecture

2.2 Generative Adversarial Networks

A Generative Adversarial Network (GAN) is an unsupervised deep learning technique that operates with two parts, the generative and the discriminative networks. These two models compete with each other in a form of a game setting. The GAN model would be trained on both - real data and fake data (generated by the generator). The discriminator's job is to determine fake data from real ones. The generator is a learning model, so starting from very basic at its initial stage, it is likely to produce low or even completely noisy data that does not reflect the real distribution or the properties of the real data.

The primary goal of the generator is generating artificial data that can pass the discriminator successfully. The model starts taking some noise, usually trying to convince the discriminator that the data produced by the generator is a legitimate one. The generator must learn how to trick the discriminator and win a positive classification (producing data classified as real). The generator's defeat is computed whenever any of those generated data is detected successfully as "fake". The discriminator, in turn, has to learn how to identify those fake data progressively. The discriminator is a loser whenever the model fails to recognize a fake data. The key concept is the simultaneous training of the generator and the discriminator. Hence, enhancing the overall performance of the system.

2.3 Deep convolutional neural networks

Deep convolutional neural networks (DCNNs) utilize deep, feed-forward architectures to learn the most significant features of their input. The trained model can then be used to classify new input into labels based on the learned features of the training data. In this regard, DCNNs are specific to the data on which they were trained, and new data must be able to be classified into existing labels (Staar et al, 2019). The feed-forward architecture consists of a varying number of layers stacked onto one another, using the previous layer's output as input for the next connected layer (LeCun et al, 2004), shown in Figure-3. The model is first initialized with all filters, weights, and parameters set to random values. Training data is then input into the network, which goes through a forward propagation step starting in the convolutional layer, which is comprised of filters that calculate an activation map containing the output of each convolution over the entire input (LeCun et al, 2004), (Lee et al, 2009). By activating when specific details are detected, the filters are used to learn the features of the input. The pooling layers of the network map the locations of the features of the data in relation to other features. The fully connected layers of the network then receive the activation mappings of the previous layers that contain the learned features, which are used for high-level reasoning and classification. At the output layer, the summation of the error is then calculated across all the classes and a backpropagation step is then applied using gradient descent to update the filter values and weights. The parameters of the model are subsequently optimized to improve classification accuracy.

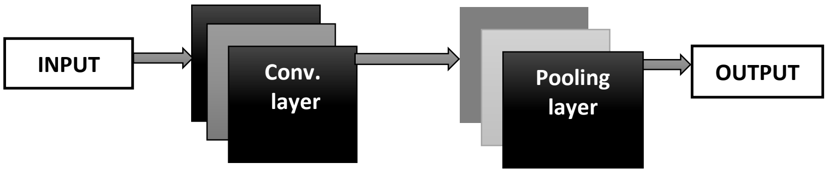

Figure-3: DCNN topology

2.4 Anomaly Detection

An anomaly is an out-of-way or abnormal behaviour shown by a system and anomaly detection is a data mining process that attempts to pinpoint data points, events or behaviours, that deviate from the normal behaviour of a dataset. Anomalies can be quite diverse and complex depending on various point of views, and transforming such an exceptional behaviour or an anomaly and its detection into an automated process can be even more challenging and time consuming task. The work starts with how we look at and define an anomaly, and it continues with the multitude of possible anomalies and different anomaly sources. An anomaly may be caused by unintentional or non-malicious causes such as communication errors, defective devices, noise in signals, significant environmental changes etc., which are usually unavoidable, often unpredictable and part of the system. But the other type of anomaly may be caused by intentional or malicious actions, such as a virus or another form of a malicious program, an intruder or a theorist. Overall, an anomaly detection method is usually picked based on a number of factors – different forms of anomalies, different anomaly sources, different types of systems (system behaviours) and different application domains (Sebestyen et al, 2017). Usually, an anomaly detection is used for data cleaning, fraud detection, intrusion detection, ecosystem instabilities, event detection in sensor networks etc.

It can be shown that the principal components are eigenvectors of the data covariance matrix. Therefore, Eigen decomposition of the data covariance matrix or singular value decomposition of the data matrix are often used to compute the principal components. PCA is the simplest of the true eigenvector-based multivariate analyses and it is closely related to factor analysis, which is a statistical method used to describe variability among observed, correlated variables in terms of a potentially lower number of unobserved variables called factors (Njilla et al, 2019).

2.5 Autoencoders

Autoencoders are a type of learning model, that are used to create a lower dimensional representation of their input, through encoding. This allows for the most significant features of the input data to be learned by the model, with extraneous noise being filtered out, which can reduce high dimensional datasets without a significant loss of information. Autoencoders are comprised of two fully connected feedforward networks, an encoder and decoder, with a hidden layer between them, shown in Figure-4. Input data is first received by the encoder network, that extracts the most significant features and correlations from the data while reducing it into a compressed representation, in a latent space. The input then goes through a forward propagation step, through the layers of the encoder, with a variable number of nodes per layer. The hidden layer between the encoder and the decoder, obfuscates the data as to ensure that, the decoder does not duplicate the input. In general, the smaller the size of the hidden layer, the more compressed the encoded representation is. The decoder, which is an inverse architecture of the encoder receives the output of the hidden layer, and attempts to reconstruct the original data from the compressed representation. A loss function is then applied to measure the error, between the original and reconstructed data, and used to optimize the model. Autoencoders are specific to the data on which they are trained and can be used for tasks such as dimensionality reduction, denoising, and anomaly detection. In the case of multivariate data, for example, an autoencoder model can be trained to learn the normal conditions of a dataset. When new, previously unseen anomalous data is introduced to the network, it will encode, and attempt to reconstruct the original input. As the anomalous data contains data that differs from the normal learned conditions, it would have increased reconstruction loss.

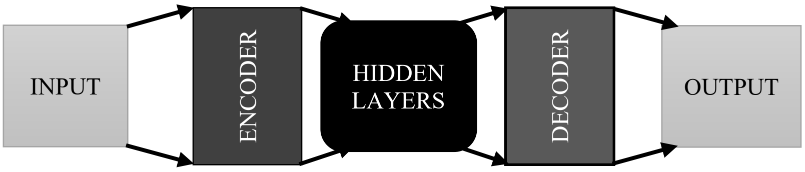

Figure-4: Autoencoder procedure

2.6 Principal Component Analysis

Principal Component Analysis (PCA) is a dimensionality-reduction method that is often used to reduce the dimensionality of large data sets, by transforming a large set of variables into a smaller one that still contains most of the information in the large set. The goal of PCA is to extract the most significant information from the data, compress the dataset by keeping only this most significant information, and analyze the structure of the data observations and the variables (Markopoulos et al, 2014). To achieve these goals, PCA calculates new variables called principal components which are obtained as linear combinations of the original variables. The first principal component is required to have the largest possible variance or inertia and therefore this component will "explain" or "extract" the largest part of the inertia of the data. The second component is calculated under the constraint of being orthogonal to the first component and to have the largest possible inertia. The other components are computed likewise. The values of these new variables for the observations are called factor scores. The factor scores can be interpreted geometrically as the projections of the observations onto the principal components.

2.7 Data preprocessing

Data preprocessing is a very crucial step in any data mining process, as it determines the quality and relevance of meaningful insights from the data. It is due to the fact that the real-life data usually are incomplete, inaccurate, inconsistent and often lacks specific attribute values/trends (missing values). Thus, a careful screening of the data is must for such problems in order to produce clean, well formatted and organized ready-to-use raw data for the machine learning processes (Oliveri et al, 2019). The figure-5 demonstrates the application of data preprocessing needed prior to using them in a ML process.

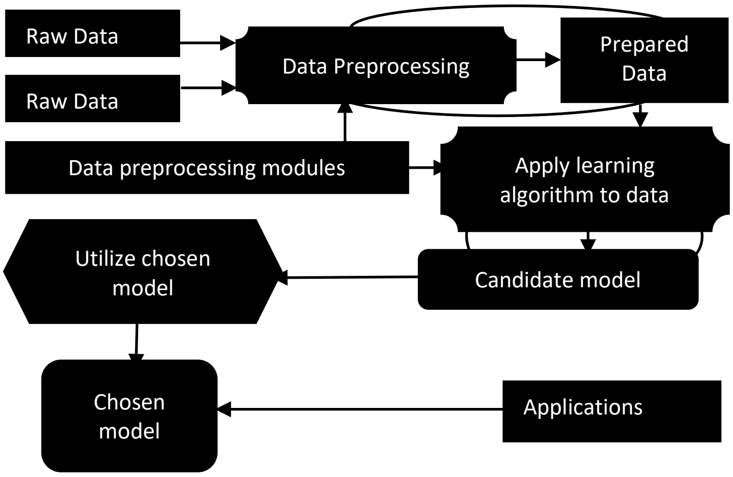

Figure-5: Data preprocessing

Data preprocessing starts with acquiring the data, cleaning the data, selection of instances, normalization, transformation, encoding the categorical data, splitting of datasets, feature scaling etc. Data preprocessing needs a careful insight as it may affect the way the outcomes of the final data processing can be interpreted (Oliveri et al, 2019), (Chatzigiannakis et al, 2006), (Simone et al, 2021).

2.8 Data mining

Data mining refers to the various techniques employed to detect correlations, anomalies, or patterns within the data so that insights about the data can be made. The mining does not refer to extraction of the data itself, but rather to the knowledge about the data that can be obtained. For this reason, an alternative term used in place of data mining is Knowledge Discovery in Databases (KDD). The KDD process consists of seven steps, shown in Figure-6. First is data cleaning, where missing values and noisy data are removed. This can be done through data discrepancy detection and data transformation tools. Next is data integration, where heterogeneous data from multiple sources are combined into a common source. This can be done through data migration or synchronization tools or through an extract-load-transformation process. Data selection comes next, deciding which data is relevant to the analysis and retrieving it from the data collection. Neural networks, decision trees, naïve bayes, clustering, and regression can be used to perform this selection. The fourth process is data transformation, converting the data into an appropriate form for the mining process using data mapping and code generation.

Next step is data mining, which is a process to identify interesting patterns and knowledge from a large amount of data. Pattern evaluation follows next, and it involves identifying interesting patterns representing some form of knowledge. Data summarization and visualization methods are used to convert the data to a form that users can understand. Finally comes Knowledge representation step, this is where data visualization and knowledge representation tools can be used to represent the mined data for a better and clearer understanding.

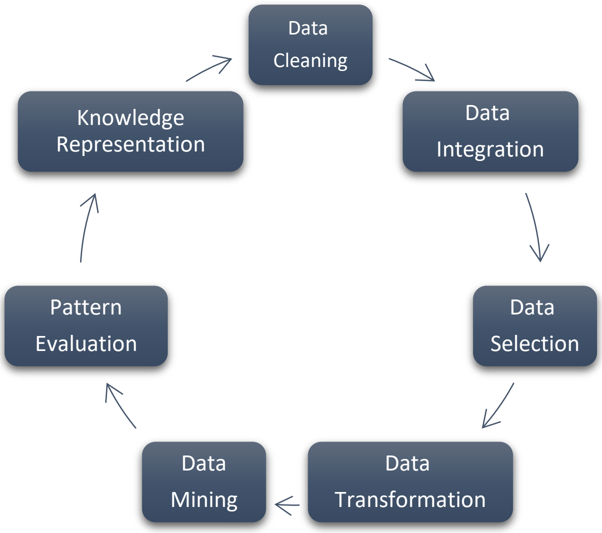

Figure-6: Data mining (KDD)

3. Methodology

The study is primarily based on two CPS datasets, the first one being a Water Distribution Testbed (WDT) dataset, which have been acquired from a water distribution hardware-in-the-loop testbed which emulates water passage between nine tanks via solenoid-valves, pumps, pressure and flow sensors, and the dataset is broadly divided into physical and network data. The other dataset, on the other hand, is a Supervisory Control And Data Acquisition (SCADA) Gas dataset.

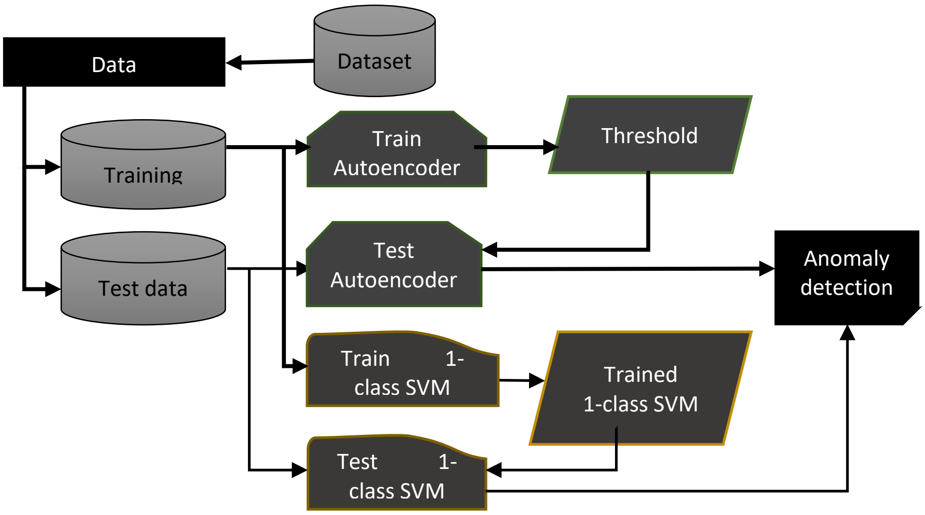

Figure-7: Basic methodology

The basic methodology is depicted in the figure-7, which starts with collecting CPS datasets from different sources, and then preprocessing them for a better study. The datasets are firstly preprocessed using one-hot encoding to represent certain categorical variables as binary vectors to ease the performance of the study, and the datasets are then normalized using Min-Max scaler. These datasets are individually divided into two parts – train and test data, out of which train data is picked as only normal data (without any anomalies) and test data containing both normal as well as anomalous data. The train data is further split into train and validation, certain amount of the train data is reserved to validate the models prior to doing the testing. Autoencoder and one-class SVM models are used in this study, they are trained using non-anomalous normal training data, validated with another set of data (validation data) and both the models are fine-tuned, wherever possible, for a better performance. For instance, in case of the autoencoder model, the parameters contributing to the reconstruction loss are determined and adjusted to minimize the loss. The one-class SVM is adjusted with its hyperplane parameters like kernel, gamma and nu, to achieve a satisfactory result. The models thus obtained are used to predict the anomalies from the test data and their performances are checked and compared individually.

4. Results

In case of autoencoder, different specifications of the important hyper-parameters such as depth of encoder-decoder, no. of nodes per layer in both, bottleneck etc. are tried and chosen to best suit the model. The model is later trained for 100 epochs with a batch size of 64, helping the encoder part of the model to compress and encode the parameters to a lower dimensional space. The latent space thus obtained is used by the decoder of the model to reconstruct the input data. The threshold is computed from the distribution of the training loss, which plays the key role in determining whether an observation is normal or anomalous. In case of one-class SVM, the upper bound on the fraction of outliers is determined carefully and set, the stopping tolerance that affects the number of iterations used while optimizing the model, and depends on the stopping criterion value, is also well thought and set, to enhance the performance of the model at an optimum level.

Looking at the purpose of the study being determining whether an observation is normal or anomalous, which is a classification problem, the performances of both the models are compared using the confusion matrix. It is a mechanism that not only provides accuracy, but also helps in estimating correct classification and misclassification. From the confusion matrix, important metrics such as accuracy, sensitivity, specificity, and precision are calculated using the following equations:

$$Accuracy = \frac{\mathrm{TP} \ + \mathrm{TN}}{\mathrm{TP} + \mathrm{FP} + \mathrm{FN} + \mathrm{TN}}$$

$$Sensitivity = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

$$Specificity = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}}$$

$$Precision = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}$$

$$F1 - score = \ 2 \times \frac{\mathrm{Sensitivity} \ \times \mathrm{Precision}}{\mathrm{Sensitivity} + \mathrm{Precision}}$$

Where, TP = True Positive, TN = True Negative, FP = False Positive and FN = False Negative.

Sensitivity or recall and precision are critical class-level metrics, maximizing one can pull the value of the other one down and vice-versa, as they are inversely proportional to each other. In most of the problems, we could either prioritize maximizing either sensitivity or precision, depending on the nature of the problem. Thus, it demands a trade-off between sensitivity or recall and precision, and this is where the F1-score may be handy, which is the harmonic mean of these two metrics. A model is considered perfect if its F1-score is 1, and 0 indicates the model as a total-failure.

Table-A: Performance metrics of the one-class SVM model

One class SVM				
Accuracy	**Sensitivity**	**Specificity**	**Precision**	**F1-score**
94.09	94.05	95.56	99.87	96.87
95.19	95.22	92.43	99.93	97.52
97.03	96.95	97.62	99.69	98.30
96.60	96.63	91.87	99.94	98.26
96.60	96.64	94.36	99.88	98.24
95.90	**95.90**	**94.37**	**99.86**	**97.84**

Table-B: Performance metrics of the autoencoder model

Autoencoder				
Accuracy	**Sensitivity**	**Specificity**	**Precision**	**F1-score**
99.58	99.87	85.88	99.70	99.78
98.77	99.72	85.93	98.97	99.34
99.82	99.87	99.76	99.76	99.82
99.84	99.93	99.75	99.74	99.84
99.82	99.94	99.70	99.69	99.82
99.56	**99.87**	**94.20**	**99.57**	**99.72**

The performance metrics of both autoencoder and one-class SVM models are computed, and the vital metrics are represented in Table-A and Table-B in tabular format. In both the models, the performance metrics are computed multiple times to determine the consistency and also to rule out any foul play which may occur.

5. Conclusion and Future work

The results thus generated by both the models, with the datasets under consideration, indicate that the models perform well if they are individually fine-tuned, however it requires a considerable amount of time to find out the parameters and their efficient settings. While the two models are compared to each other, it is found that the autoencoder model performance is relatively better compared to that of the one-class SVM. However, the study doesn't generalize it as the scenario may be very different while using diverse datasets and the models may require a different tuning of the relevant parameters.

In future, this method may be applied into a constantly running CPS, allowing for the performance of corrective action(s) to be taken once an anomaly is found, which is beyond the scope of the current study.

References

Cardenas, A. A., Amin S., and Sastry, S. (2008) Secure control: Towards survivable cyber-physical systems. In International Conference on Distributed Computing Systems Workshops (ICDCSW). IEEE.

Chatzigiannakis, V. et al, (2006) Hierarchical Anomaly Detection in Distributed Large-Scale Sensor Networks, 11th IEEE Symposium on Computers and Communications (ISCC'06), pp 761-767.

Checkoway, S. et al. (2011) Comprehensive Experimental Analyses of Automotive Attack Surfaces. In USENIX Security Symposium. San Francisco.

Cho, K.-T. and Shin, G. K. (2016) Error Handling of In-vehicle Networks Makes Them Vulnerable. In ACM Conference on Computer and Communications Security (CCS).

LeCun, Y, Huang, FJ and Bottou, L. (2004) 'Learning methods for generic object recognition with invariance to pose and lighting', Proceedings of the *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. II97-II104.

Lee, H.et al. (2009) Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. Proceedings of the 26th Annual International Conference on Machine Learning. ACM, pp 609–616.

Lee, J., Bagheri, B., and Kao, H.-A. (2015) A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems, Manufacturing Letters, Volume 3, ISSN 2213-8463, pp 18-23

Markopoulos, Panos P., Karystinos, George N. and Pados, Dimitris A. (2014). Optimal Algorithms for L1-subspace Signal Processing. IEEE, Vol 62, pp 5046-5058.

Meshram, A., Haas, C. (2017) Anomaly Detection in Industrial Networks using Machine Learning: A Roadmap. In: Beyerer J., Niggemann O., Kühnert C. (eds) Machine Learning for Cyber Physical Systems. Technologien für die intelligente Automation (Technologies for Intelligent Automation). Springer Vieweg, Berlin, Heidelberg.

Mosterman, P.J., and Zander, J. (2016) Industry 4.0 as a Cyber-Physical System study. Softw Syst Model 15, pp 17–29.

Njilla, L. et al, (2019) Internet of Things Anomaly Detection using Machine Learning, 2019 IEEE Applied Imagery Pattern Recognition Workshop (AIPR), pp 1-6.

Oliveri, P. et al, (2019). The impact of signal pre-processing on the final interpretation of analytical outcomes – A tutorial. Analytica Chimica Acta. 1058: pp 9–17.

Sebestyen, G. and Hangan, A. (2017) Anomaly detection techniques in cyber-physical systems, Acta Universitatis Sapientiae, Informatica, vol.9, no.2, pp 101-118.

Simone, G. et al (2021). A hardware-in-the-loop water distribution testbed (WDT) dataset for cyber-physical security testing. IEEE Dataport

Staar B., Lütjen M., Freitag M. (2019) Anomaly detection with convolutional neural networks for industrial surface inspection, Procedia CIRP, vol. 79, pp 484-489