Intrusion Detection for Industrial Control Systems by Machine Learning using Privileged Information

Moojan Pordelkhaki
School of Computing and Digital Technology
Birmingham City University, UK

Shereen Fouad
School of Informatics
and Digital Engineering
Aston University, UK

Mark Josephs
School of Computing and Digital Technology
Birmingham City University, UK

***Abstract*—The continuous operation of an industrial process, such as water treatment or power generation, is governed by an Industrial Control System (ICS). Cyber-attacks on industrial networks are of growing concern because of the disruption they can cause, leading to loss of revenue, and the possibility of harm to workers, plant and surroundings. Operators therefore need a Network Intrusion Detection System (NIDS) to analyse industrial network traffic in real time for adversarial behaviour. Machine Learning (ML) is applicable to the problem of network intrusion detection. This paper investigates the possibility of training an ML-based NIDS for an ICS (specifically, the well-known Secure Water Treatment testbed) by combining network traffic data and physical process data. In the supplied dataset, data had already been labelled "according to normal and abnormal behaviours"; the labelling of data collected around the start and end of each attack was scrutinized and, where found to be problematic, labelled data were excluded in order to improve the effectiveness of supervised learning. The ML technique of "Learning using Privileged Information" was evaluated and found to be superior to six baseline ML algorithms trained on network traffic data alone.**

***Index Terms*—Network Intrusion Detection System, Industrial Control System, Machine Learning, Learning using Privileged Information**

I. Introduction

INDUSTRIAL Control Systems (ICS) provide autonomous control of physical processes, such as those found in manufacturing, chemical processing, power generation and water treatment. These cyber-physical systems are susceptible to attack on their network infrastructure. Cyber-attacks are of growing concern because of the disruption they can cause, leading to loss of revenue, and the possibility of harm to workers, plant and surroundings; theft of operational information is another possible impact.[1]

The application of Machine Learning (ML) to intrusion detection and, more generally, anomaly detection for ICS is an active area of research. ML can be categorised as supervised, semi-supervised and unsupervised; examples of each kind can be found in [1]–[3]. Supervised learning involves data that have been labelled, a manual, expensive operation that requires domain knowledge. It is known for high predictive ability, whereas semi-supervised and unsupervised learning (which use unlabelled data) suffer from low predictive performance. Indeed, a comparative analysis conducted in [4] found that supervised ML algorithms outperform semi-supervised/unsupervised algorithms in an ICS context.

Research into supervised ML-based Network Intrusion Detection Systems (NIDSs) for ICS has mainly focused on utilising industrial network traffic while ignoring physical process data. For example, [5] uses the One-Class Support Vector Machine (SVM) algorithm, [6] uses a knowledge-based analysis technique to detect and classify attacks, and [7] uses an autoencoder feature learning technique for network packets to identify attacks. However, a survey of supervised ML-based intrusion detection systems argued that "aggregation and correlation of variables from multiple sources" (an "holistic perspective") was important [8]. For example, an adversary can implant a malicious message in a payload of a network packet without violating any protocol or communication pattern; a NIDS would not be able to detect it. The importance of incorporating domain-knowledge and process context in detecting cyber-attacks in ICS through modelling the physical processes was indicated in [9]. However, generating an accurate model requires a thorough understanding of the physical processes and algorithms.

Other studies have focused on supervised ML-based anomaly detection using process data only. For example, Hink *et al.* proposed an attack discrimination technique by monitoring normal, natural and attack events based on measurement from power systems [10]. Junejo and Goh demonstrated the effectiveness of supervised techniques on behaviour-based anomaly detection while monitoring the process data and reported a low false-positive rate as well as high precision and recall [11]. Another method relying on the laws of physics to detected an insider attack based on observing the dynamics of the system [1].

It should be noted, however, that anomalies can arise because of device or system failure for reasons other than cyber-attack. Anomaly detection based on physical process data alone cannot distinguish between a faulty sensor, for example, and a cyber-attack. Also, NIDS can identify the reconnaissance phase of a cyber-attack, which takes place without affecting physical processes.

Training and testing an ML-based NIDS for an ICS using both network traffic data and physical process data may lead to higher accuracy than network data alone. However, the

[1]See https://collaborate.mitre.org/attackics/index.php/Impact for a detailed classification of cyber-attack impact and real-life examples.

integration of the two sources of data during testing (i.e., at run-time) is likely to be impractical because of differences in the method and rate of data collection from those sources. In particular, network traffic is monitored at high frequency (e.g. in milliseconds), whereas the process data are reported periodically at a fixed rate depending on process parameter configuration (e.g. in seconds/minutes). In addition, during run-time both process and network data are stored in different locations.

In this paper, we present an ML-based NIDS that integrates network and process data during the training phase; the model solely uses network data during testing. The proposed method utilises the Learning Using Privileged Information (LUPI) framework [12], [13]. In LUPI, a supervised ML model is constructed by integrating additional informative features, known as privileged information, during the training phase. We hypothesise that this integration will improve the accuracy of intrusion detection compared with models utilizing only network data.

The LUPI framework has proved to be effective in various domains, including computer vision [14]–[16], astronomy [17]–[19] and medical diagnosis [20], [21]. The application of LUPI in the cybersecurity domain has also been investigated in the context of detecting malicious botnet activities in IT networks [22]. LUPI was also investigated in [23] to train anomaly detection systems using forensic data as privileged information in a variety of security applications, such as face authentication, fast-flux Bot Detection and Malware Traffic Detection. However, the application of LUPI to NIDS for ICS has not been investigated before.

The effectiveness of the proposed methodology is evaluated in the context of the Secure Water Treatment (SWaT) testbed, an ICS for a scaled-down water treatment plant [24]. There is a large body of work investigating ML-based anomaly detection for the SWaT testbed but, to the best of our knowledge, this paper is the first to apply the LUPI framework.

The rest of the paper is organised as follows: Sections II and III provide background information about LUPI and the SWaT testbed, respectively. Section IV describes our proposed framework for NIDS using process data as privileged information. Experimental results are evaluated in Section V and conclusions are drawn in Section VI.

II. Learning Using Privileged Information (LUPI) versus classical Machine Learning (ML) algorithms

Supervised classical ML algorithms aim to learn the distribution pattern of labelled training data presented in n number of training pairs (x_i, y_i), where $i = 1, ..., n$, $x_i \in X$, and $y_i \in \{+1, -1\}$. During training, a mapping function $f : X \rightarrow +1, -1$ is formulated that can map an input instance (x_i) to a predicted output (y_i) with the lowest error possible. In classical supervised learning problems, the same data features are used for both training and testing (run-time).

In some pattern recognition problems, there may be additional helpful information about the training samples that will not be available during the testing phase. Such data would often be discarded by classical ML algorithms since models have been trained based on training input features only. Recently, there has been a trend in designing ML models that incorporate this additional information (referred to as "privileged information"), alongside the main the training samples. The framework of Learning Using Privileged Information (LUPI) was originally proposed by Vapnik and Vashist [12], [13] in the context of the Support Vector Machine (SVM) classifier, where a triplet of training data is provided $(x_i, x_i*, y_i), i = 1, ..., n$, $x_i \in X$, $x_i^* \in X^*$, and $y_i \in \{+1, -1\}$. Similar to classical ML, the goal is to find a function $f : X \rightarrow +1, -1$ that can predict labels with the lowest error possible. The idea is that the privileged information might improve the learning process and help the ML model converge to a better decision boundary in the input space.

A. SVM+ Learning Algorithm for LUPI

SVM [25], is a popular supervised learning algorithm for solving non-linear classification problems. The aim is construct a non-linear hyperplane with maximum margin that separates two classes (in the case of binary classification). The SVM allows the decision margin to make some violations known as slack variables (ξ_i). The task here is to find a decision function $f(x) = sgn[\langle w, x\rangle + b]$, where $w \in X, b \in \mathbb{R}$ and they are obtained by solving the following optimization problem

$$\min_{w,b,\xi_i} \ \frac{1}{2}||w||_2^2 + \gamma \sum_{i=1}^{n} \xi_i \tag{1}$$

under the constraints,

$$\forall \quad 1 \leq i \leq n, \quad [y_i \langle w, x_i\rangle + b] \geq 1 - \xi_i, \quad \xi_i \geq 0, \tag{2}$$

where $\gamma \geq 0$ is a hyper-parameter that controls the tradeoff between margin maximization and margin violation. If the slacks ξ_i are all equal to zero then we call the set of given examples separable, otherwise they are non-separable.

In SVM+ [12], [13], [19], which is based on LUPI, the additional information ${x_i}^* \in X^*$ will be available during training but not at the test stage. Unlike SVM which uses a correcting slack variable ξ_i, the SVM+ uses a slack function $\xi_i = [\langle w^*, x_i^*\rangle + b^*]$, where $w^* \in X^*$, $b^* \in \mathbb{R}$ and they are obtained by solving the following optimization problem:

$$\min_{w,w^*} \ \frac{1}{2}||w||_2^2 + \frac{\rho}{2}||w^*||_2^2 + \gamma \sum_{i=1}^{n} [\langle w^*, x_i^*\rangle + b^*] \tag{3}$$

under the constraints,

$$\forall \quad 1 \leq i \leq n, \quad [y_i \langle w, x_i\rangle + b] \geq 1 - [y_i \langle w^*, x_i^*\rangle + b^*], \quad [\langle w^*, x_i^*\rangle + b^*] \geq 0 \tag{4}$$

In SVM+, correcting functions control the slack variables based on the privileged information. The objective function of SVM+ contains two hyper-parameters $\gamma, \rho > 0$. The ρ is a non-negative parameter that reflects the imposition of smoothness in the slack model.

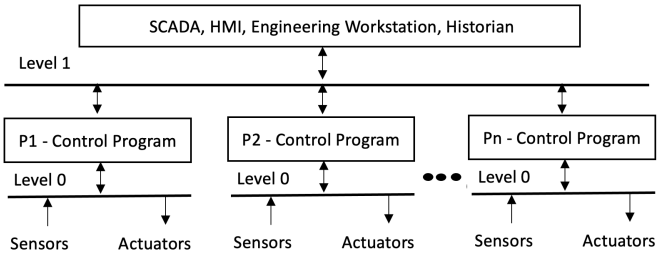

Fig. 1. Overview of SWaT Network Architecture, adapted from [24].

III. Secure Water Treatment Testbed

The Secure Water Treatment (SWaT) testbed was constructed at the Singapore University of Technology and Design [24]. It is an operational scaled down water treatment plant with six process stages which can produce 5gal/min of double filtered water. Figure 1 presents the Network Architecture, a distributed control system where each stage of the process is under the control of programmable logic.

There are various datasets available on request for the SWaT testbed. For this study, we use SWaT A1 & A2-Dec 2015 dataset, which contains recordings from network traffic as well as 51 field instruments over 11 days of continuous operation. For the first seven days, data were collected under normal operation, whereas the remaining four days included 36 attacks. In this dataset, attackers hijacked data packets through the Level 1 communication link of Figure1, manipulated the sensor data in the packets and sent them to the PLCs [24].

While all the physical properties of the field instruments were recorded periodically as process data in the Historian server, network data was captured from Level 1 at much higher frequency. The latter is claimed to only include data valuable for intrusion detection. All process and network data has a timestamp, allowing the provider of the dataset to flag data occurring within an attack.

IV. Network intrusion detection for the SWaT testbed using process data as privileged information

Here, we introduce a NIDS for the SWaT testbed using the LUPI framework for supervised ML. Unlike current ML-based NIDS, our ML method incorporates the process data as privileged information during training, whilst using only network data for testing.

The proposed framework is presented in Figure 2. In brief, the process and network features are extracted from their original sources preserving the labelling, where $+1$ supposedly indicates a record collected during an attack, and -1 indicates a record collected under normal operation. Any misalignment in data labels (discussed below) will be addressed. The process and network data will then be integrated in order to create the training data. The LUPI framework will then be applied via the Support Vector Machine plus (SVM+) algorithm, an extension of the SVM which integrates privileged information during training to help construct an optimal hyperplane. Lastly, the trained SVM+ model will be applied (during run-time) on the testing set, which consists of network data only.

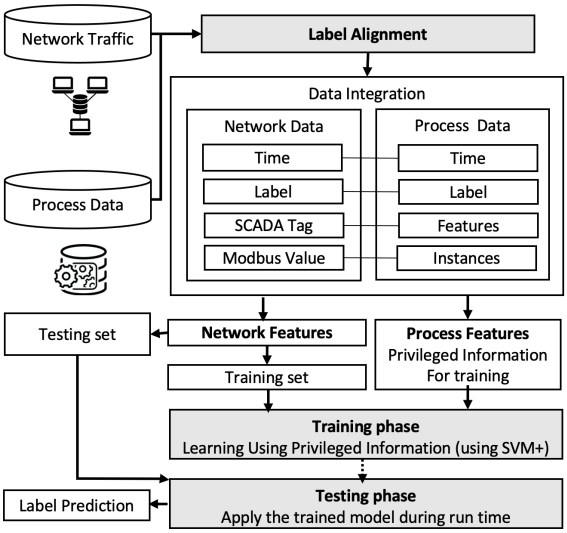

Fig. 2. Network intrusion detection framework for the SWaT testbed using process data as privileged information.

A. Network and process data selection for ML training

1) ***Network traffic (original training set)****:* Network dataset, provided by iTRUST [24], contains 19 selected features captured from communication between SCADA and PLC, they considered to be valuable for intrusion detection, namely, *Date, Time, Origin, Type, Interface Name, Interface Direction, Source IP, Destination IP, Protocol, Proxy Source IP, Application Name, Modbus Function Code, Modbus Function Description, Modbus Transaction ID, SCADA Tag, Modbus Value, Service/Destination Port, Source Port.* Note that the hex-encoded binary payload of the Modbus protocol includes multiple register readings. When analysing this dataset we merged the request and response messages that were part of the same transaction into a single record. We extracted the values representing the readings of the available SCADA Tag in the dataset, and used only the first value of the Modbus field. Note that network traffic is monitored at high frequency (e.g. in milliseconds). In this study, the network dataset includes 493,001 records and was randomly divided into 70% and 30% for training and testing, respectively, see Figure 2.

2) ***Process data (privileged information)****:* There are two versions of process dataset for the physical recordings of the SWaT normal state that are available at [24]. The first version, records activities started when the plant was draining the water storage tank for 30 minutes. Since in general this is part of the maintenance process out of normal operation, the second version was generated by removing the first 30 minutes of data. The latter version is used here which represents the normal operations and includes 495,000 records from 51 field instruments. In SWaT testbed, the process data are reported periodically to an historian server at a fixed rate depending on process parameters configuration and characteristics (e.g.

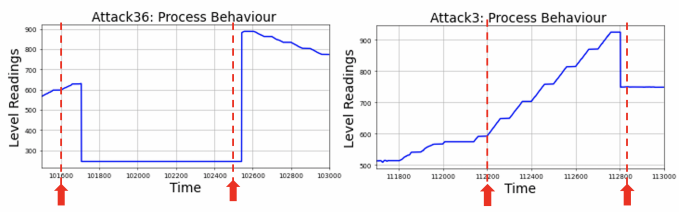

Fig. 3. Illustration of the misalignment between indicated end points of attack (in red) and process behavior change (in blue) for Attack 3 and 36 within SWaT testbed

in seconds in SwaT testbed). In this study, the process data will be used as privileged Information for the model training only, but not for testing, see Figure 2.

B. Label Alignment and Data Integration

In this section we identify two shortcomings in data labels of the SWaT dataset. The first shortcoming was found in the process data where records were labelled as attacks but this was not evident in the physical process behaviour. This problem has been briefly mentioned in [4] but not thoroughly investigated nor addressed. The second shortcoming was found in the network traffic where each record appears to inherit its label from process data by a coarse-grained mapping of timestamps.

Two attacks (numbered 3 and 36 in the supplied dataset documentation) have been selected here to illustrate the first shortcoming, Figure 3. These attacks target a sensor in the first processing stage of the water treatment plant. Attack 36 sets the level reading (LIT101) to a value lower than the process lower warning limit. As a consequence, the outlet pump turns off and the tank inlet valve opens to increase the water level, which causes the tank to overflow. As shown in Figure 3, the start (end) of Attack 36 has been indicated earlier than the drop (jump) in the level reading of the sensor. Attack 3 increases the level reading by 1mm per second. As a consequence, a pump is required to operate unnecessarily, which will cause it to fail earlier than expected — this is a good example of a stealthy attack [26]. Figure 3 shows that the gradual increase in the LIT101 reading starts earlier than the indicated start of the attack and the reading returns to the correct level before the indicated end of the attack. Similar to the process data, the inconsistency between labels and behavioural change can also been observed in the network traffic for the above attacks.

As mentioned above, network data has been recorded at a higher sampling rate in comparison to process data. There is a many-to-one relationship between network and process data under a timewise mapping. Consequently, a single instance of process data with a given timestamp S will be integrated with multiple network data records with timestamps within one second of S and they will inherit its label.

In the context of ML-based intrusion detection systems, it is important to ensure that the labelling of attacks is well aligned with behavioural change to facilitate valid, accurate data analysis. Rather than alter labels in the dataset supplied by iTRUST, we have simply filtered out any labelled data around the start and end of attacks where we considered the labelling to be problematic. This improved the effectiveness of supervised learning, as we shall see.

V. Experiments and Analysis

The effectiveness of the proposed methodology, integrating and incorporating process data as privileged information in learning, alongside network data, was evaluated in the context NIDS for the SWaT testbed. Firstly, the proposed label alignment approach has been evaluated on the network data. Secondly, we validate our hypothesis that using process data as privileged information during learning, via the LUPI framework, improves the effectiveness of supervised learning for network intrusion detection.

A. Data pre-processing and experimental setup

In the data pre-processing step, numerical features of process and network data were normalized to zero mean and unit variance to ensure that all features contribute equally to the classification task. The unequal distribution of classes within the data (no. -1 *vs.* no. +1) was addressed using the under sampling method, often used as a pre-processing step in ML practice for dataset that has a skewed class distribution, in our case the number of samples of normal activity (-1). In network data, categorical data were transformed using one-hot encoding framework [27]. The network data was randomly divided into 70% and 30% for training and testing, respectively. The process data records corresponding to the network training instances were used by SVM+ (LUPI) for training purposes.

In this study, we applied a feature selection method [28] to assess and compare the predictive ability of the network and process features using the Decision Tree (DT) ML algorithm. This experiment was conducted on the integrated training set, which combines both network and process data after resolving the data labels misalignment. We ranked the predictive features according to their weight importance with respect to the classification task. The analysis revealed that the process features act as stronger predictors for intrusion detection in SWaT testbed when compared to the network features, therefore, used here as privileged information in the training phase.

Our classification results were evaluated using four performance measures:

- Classification accuracy, measures all of the correctly identified cases,

$$\frac{TP+TN}{TP+FP+TN+FN,} \tag{5}$$

  where TP, FP, FN and TN denotes true positives, false positives, false negatives and true negatives, respectively
- Precision, the ratio of correctly predicted positive records to the total predicted positive records,

$$\frac{TP}{TP+FP} \tag{6}$$

- Recall, the ratio of correctly predicted positive records to the all data records in a class,

$$\frac{TP}{TP + FN} \tag{7}$$

- The F1-score, it conveys the balance between the precision and the recall.

$$\frac{2 \times (Recall \times Precision)}{(Recall + Precision)} \tag{8}$$

In all experiments, we applied supervised ML models to identify one type of attack (Attack 36) using network data of the SWaT testbed. We particularly evaluated the performance of six classical and popular ML models including, K-Nearest Neighbour (K-NN), Logistic Regression (LR), Decision Tree (DT), Multilayer Perceptron (MLP), one-dimension Convolutional Neural Network (CNN), and Support Vector Machine (SVM) [29]. (Hyper-)parameters of all classification algorithms were tuned via 5-fold cross-validation on the training set. For fair comparisons, the experiments were run five times for each classifier and we reported the average results over five runs. All experiments were run using Python (scikit-learn libraries) and Jupyter hosted on Google's Colab platform.

B. Evaluate the effectiveness of the label alignment approach on the NIDS

In this experiment, we applied six classical ML models (mentioned above) to identify *Attack 36* in network data of the SWaT testbed. The NIDS performance results *before* resolving the label misalignment problem is reported in Table I. Table II presents the NIDS performances *after* excluding the records with inconsistent data labels, which ensures that labels (attack or not) are consistent with the behaviour change.

As shown in Tables I and II, the detection performance obtained after applying the label alignment method outperforms results obtained using the original data labels, which is an evident that this strategy reduces the number of false negatives. On average, across all the assessed ML models applying the proposed label alignment method on network data improves the NIDS performance by 3.04%, 2.44%, and 5.02%, in terms of Accuracy, Recall and F1-score, respectively.

C. Evaluate the Effectiveness of LUPI on the NIDS

In this experiment, we integrated the process and network dataset of SWaT testbed using the proposed method in Figure 2. Then, we applied the SVM+ (LUPI), discussed in section II-A, to identify *Attack 36* using the integrated training set. The NIDS performance obtained by SVM+ algorithm is reported in Table III. Note that the SVM+ algorithm will be tested using the network data only.

Results obtained here can be compared against the six classical ML models (in Table II), trained and tested on network traffic only, without considering the impact of the process data. Note that, on average, the accuracy, precision, Recall and F1-scores results obtained by SVM+ algorithm outperform the results obtained by all classical ML algorithms by 12.49%, 22.57%, 16.71%, and 19.45%, respectively. Note that, the accuracy obtained by SVM+ outperforms all baseline ML algorithms except for the DT classifier. However, due to the nature on the imbalanced class dataset, the results reported by F1-score maybe more reliable than the accuracy measure in our scenario.

VI. Conclusion

In this paper, we presented an alternative ML-based NIDS that uses the LUPI framework. Unlike classical ML algorithms for anomaly detection which rely upon only one source of data for learning, our NIDS incorporates process data as privileged information during the training phase. This allows for a more accurate and resilient ML-based NIDS than is possible using baseline methods, while requiring similar computational resources at run-time because the testing phase only involves one source of data, namely, the industrial network. To the best of our knowledge, this is first attempt to combine industrial network traffic and physical process data in supervised learning for a NIDS.

We used the SWaT testbed to assess the effectiveness of our proposed methodology. Before integrating process and network data, we ensured that data labels (attack or not) were aligned with behavioural change. The effectiveness of the label alignment method was validated experimentally by comparing the NIDS accuracy obtained before and after the label alignment. Our second experiment demonstrated that SVM+, trained on both network as well as process features using the LUPI framework, outperforms six baseline ML models trained on network features only. Our work suggests that LUPI framework has good potential for application to ML-based NIDSs for ICS. However, there is still considerable room of improvement in its performance; hence, we are planning to attempt other approaches involving LUPI such as knowledge transfer and distillation [23].

References

[1] A. Agrawal, C. M. Ahmed, and E.-C. Chang, "Poster: Physics-based attack detection for an insider threat model in a cyber-physical system," in *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*, 2018, pp. 821–823.

[2] K. Yau, K.-P. Chow, S.-M. Yiu, and C.-F. Chan, "Detecting anomalous behavior of PLC using semi-supervised machine learning," in *2017 IEEE Conference on Communications and Network Security (CNS)*. IEEE, 2017, pp. 580–585.

[3] W. Gao, T. Morris, B. Reaves, and D. Richey, "On SCADA control system command and response injection and intrusion detection," in *2010 eCrime Researchers Summit*. IEEE, 2010, pp. 1–9.

[4] G. Bernieri, M. Conti, and F. Turrin, "Evaluation of machine learning algorithms for anomaly detection in industrial networks," in *2019 IEEE International Symposium on Measurements & Networking (M&N)*. IEEE, 2019, pp. 1–6.

[5] L. A. Maglaras and J. Jiang, "Intrusion detection in SCADA systems using machine learning techniques," in *2014 Science and Information Conference*. IEEE, 2014, pp. 626–631.

[6] A. Patel, H. Alhussian, J. M. Pedersen, B. Bounabat, J. C. Júnior, and S. Katsikas, "A nifty collaborative intrusion detection and prevention architecture for smart grid ecosystems," *Computers & Security*, vol. 64, pp. 92–109, 2017.

[7] P. Schneider and K. Böttinger, "High-performance unsupervised anomaly detection for cyber-physical system networks," in *Proceedings of the 2018 Workshop on Cyber-Physical Systems Security and PrivaCy*, 2018, pp. 1–12.

Classifier	Accuracy	Precision	Recall	F1-score
KNN	65.057%(±2.580%)	66.214% (±1.723%)	66.717% (±0.858%)	63.972%(±0.663%)
LR	59.770% (±2.666%)	64.594% (±1.817%)	61.602% (±1.435%)	60.089% (±2.305%)
DT	84.215% (±1.686%)	61.570% (±2.259%)	63.238% (±1.990%)	60.888% (±2.226%)
MLP	59.847% (±3.726%)	60.754% (±9.690%)	59.378% (±4.821%)	53.219% (±3.269%)
CNN	60.327%(±4.249%)	66.084% (±6.244%)	61.382% (±3.637%)	58.033% (±5.693%)
SVM	60.690%(±1.202%)	66.696% (±3.681%)	60.688% (±2.284%)	60.062% (±3.019%)

TABLE I

NIDS PERFORMANCES USING NETWORK TRAFFIC *before* APPLYING THE LABEL ALIGNMENT METHOD. WE REPORT THE MEAN VALUES OF ACCURACY, PRECISION, RECALL AND F1-SCORE, ALONG WITH STANDARD DEVIATIONS (±) ACROSS 5 TRAINING/TEST RE-SAMPLING.

Classifier	Accuracy	Precision	Recall	F1-score
KNN	65.779% (±2.164%)	65.151% (±2.242%)	65.666% (±2.826%)	64.492% (±2.308%)
LR	60.913% (±1.282%)	61.790% (±2.020%)	62.610% (±2.153%)	61.107% (±1.312%)
DT	84.639% (±1.963%)	62.269% (±2.429%)	61.814% (±1.847%)	64.193% (±1.501%)
MLP	65.095% (±2.324%)	62.548% (±7.562%)	66.192% (±1.054%)	62.060% (±2.969%)
CNN	61.714% (±2.470%)	63.828% (±2.811%)	61.726% (±1.849%)	60.269% (±2.473%)
SVM	62.890% (±2.186%)	62.698% (±2.986%)	63.598% (±2.299%)	61.088% (±2.721%)

TABLE II

NIDS PERFORMANCES USING NETWORK TRAFFIC *after* APPLYING THE LABEL ALIGNMENT METHOD. WE REPORT THE MEAN VALUES OF ACCURACY, PRECISION, RECALL AND F1-SCORE, ALONG WITH STANDARD DEVIATIONS (±) ACROSS 5 TRAINING/TEST RE-SAMPLING.

Classifier	Accuracy	Precision	Recall	F1-score
SVM+ (LUPI)	74.2534% (±1.022%)	77.251% (±0.849%)	74.1692% (±1.173%)	73.4782% (±1.375%)

TABLE III

NIDS PERFORMANCE OF SVM+ ALGORITHM (WITH LUPI FRAMEWORK). THE TABLE REPORTS THE MEAN VALUES OF ACCURACY, PRECISION, RECALL AND F1-SCORE, ALONG WITH STANDARD DEVIATIONS (±) ACROSS 5 TRAINING/TEST RE-SAMPLING.

[8] J. Suaboot, A. Fahad, Z. Tari, J. Grundy, A. N. Mahmood, A. Almalawi, A. Y. Zomaya, and K. Drira, "A taxonomy of supervised learning for IDSs in SCADA environments," *ACM Computing Surveys (CSUR)*, vol. 53, no. 2, pp. 1–37, 2020.

[9] D. Hadžiosmanović, R. Sommer, E. Zambon, and P. H. Hartel, "Through the eye of the PLC: semantic security monitoring for industrial processes," in *Proceedings of the 30th Annual Computer Security Applications Conference*, 2014, pp. 126–135.

[10] R. C. B. Hink, J. M. Beaver, M. A. Buckner, T. Morris, U. Adhikari, and S. Pan, "Machine learning for power system disturbance and cyber-attack discrimination," in *2014 7th International symposium on resilient control systems (ISRCS)*. IEEE, 2014, pp. 1–8.

[11] K. N. Junejo and J. Goh, "Behaviour-based attack detection and classification in cyber physical systems using machine learning," in *Proceedings of the 2nd ACM International Workshop on Cyber-Physical System Security*, 2016, pp. 34–43.

[12] V. Vapnik and A. Vashist, "A new learning paradigm: Learning using privileged information," *Neural networks*, vol. 22, no. 5-6, pp. 544–557, 2009.

[13] V. Vapnik, *Estimation of dependences based on empirical data*. Springer Science & Business Media, 2006.

[14] A. Momeni, K. Tatwawadi, and U. Stanford, "Understanding LUPI (learning using privileged information)," *Ionosphere*, vol. 201, no. 7, p. 6.

[15] X. Yang, M. Wang, and D. Tao, "Person re-identification with metric learning using privileged information," *IEEE Transactions on Image Processing*, vol. 27, no. 2, pp. 791–805, 2017.

[16] X. Xu, W. Li, and D. Xu, "Distance metric learning using privileged information for face verification and person re-identification," *IEEE transactions on neural networks and learning systems*, vol. 26, no. 12, pp. 3150–3162, 2015.

[17] S. Fouad, P. Tino, S. Raychaudhury, and P. Schneider, "Incorporating privileged information through metric learning," *IEEE transactions on neural networks and learning systems*, vol. 24, no. 7, pp. 1086–1098, 2013.

[18] S. Fouad and P. Tiňo, "Ordinal-based metric learning for learning using privileged information," in *The 2013 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2013, pp. 1–8.

[19] S. Fouad, "Metric learning for incorporating privileged information in prototype-based models," Ph.D. dissertation, University of Birmingham, 2013.

[20] T. A. Shaikh, R. Ali, and M. S. Beg, "Transfer learning privileged information fuels CAD diagnosis of breast cancer," *Machine Vision and Applications*, vol. 31, no. 1, p. 9, 2020.

[21] L. Shen, Q. Wang, and J. Shi, "Single-modal neuroimaging computer aided diagnosis for schizophrenia based on ensemble learning using privileged information," *Sheng wu yi xue Gong Cheng xue za zhi= Journal of Biomedical Engineering= Shengwu Yixue Gongchengxue Zazhi*, vol. 37, no. 3, pp. 405–411, 2020.

[22] A. Sapello, C. Serban, R. Chadha, and R. Izmailov, "Application of learning using privileged information (LUPI): botnet detection," in *2017 26th International Conference on Computer Communication and Networks (ICCCN)*. IEEE, 2017, pp. 1–8.

[23] Z. B. Celik, P. McDaniel, R. Izmailov, N. Papernot, R. Sheatsley, R. Alvarez, and A. Swami, "Detection under privileged information," in *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*, 2018, pp. 199–206.

[24] J. Goh, S. Adepu, K. N. Junejo, and A. Mathur, "A dataset to support research in the design of secure water treatment systems," in *International Conference on Critical Information Infrastructures Security*. Springer, 2016, pp. 88–99.

[25] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[26] A. A. Cárdenas, S. Amin, Z.-S. Lin, Y.-L. Huang, C.-Y. Huang, and S. Sastry, "Attacks against process control systems: risk assessment, detection, and response," in *Proceedings of the 6th ACM symposium on information, computer and communications security*, 2011, pp. 355–366.

[27] C. Guo and F. Berkhahn, "Entity embeddings of categorical variables," *arXiv preprint arXiv:1604.06737*, 2016.

[28] T. M. Phuong, Z. Lin, and R. B. Altman, "Choosing snps using feature selection," in *2005 IEEE Computational Systems Bioinformatics Conference (CSB'05)*. IEEE, 2005, pp. 301–309.

[29] A. Géron, *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*. O'Reilly Media, 2019.