2024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT) | 979-8-3503-7024-9/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICCCNT61001.2024.10724326

Performance Analysis of Online Machine Learning Frameworks for Anomaly Detection in IoT Data Streams

Santosh Kumar Ray
Department of Information Technology
Delhi Technological University
Delhi-110042, India
raysantosh806@gmail.com

Seba Susan
Department of Information Technology
Delhi Technological University
Delhi-110042, India
seba_406@yahoo.in

***Abstract*— With the rapid progress of technology, the Internet of Things (IoT) is vital in connecting real-time data sources. These sources generate a substantial volume of streaming data through various applications. When dealing with streaming data, there can be a significant amount of abnormal data, referred to as anomalies, which are completely unknown and can negatively affect the system's performance. In this context, unsupervised anomaly detection methods are expected to perform better than supervised methods. In this paper, we evaluate the performance scores of nine unsupervised anomaly detection models belonging to two recently introduced online machine learning frameworks: River and Python Streaming Anomaly Detection (PySAD). The methodologies investigated in the River framework include Half Space Trees (HST), Quantile Filter One-Class Support Vector Machine (QF_OCSVM), Quantile Filter Half Space Trees (QF_HST), and Threshold Filter Half Space Trees (TF_HST). The Histogram-Based Outlier Scores (HBOS), Kitsune's core Algorithms (KitNet), Robust Random Cut Forest (RRCF), Isolation Forest Anomaly Streaming Data (IForestASD), and K Nearest Neighbor Conformal Anomaly Detection (KNNCAD) belong to the PySAD framework. The models are compared in terms of Accuracy, Sensitivity, Specificity, and ROCAUC on the IoTID20 streaming dataset. The paper concludes with a comparative study through performance plots on both frameworks. The River models HST, QF_HST, and TF_HST attained the highest Sensitivity scores of 1.00, whereas only the PySAD model KitNet passed this Sensitivity level. In terms of ROCAUC, the PySAD model IForestASD performs best (0.7937) followed by the River model QF_HST (0.7810), implying that QF_HST performs consistently best among all models.**

Keywords*— *Anomaly detection, Online machine learning, IoT data streams, Unsupervised machine learning.

I. Introduction

In the past decade, significant changes have been observed in data stream mining. Numerous real-time data sources generate a large volume of diverse and dynamic data. Because of the varying nature of this data, it becomes challenging to use it collectively or in batch form. Alternatively, the data instances can be sequentially processed as streaming data [1, 2], where they are processed continuously, one by one. However, there is a high probability of encountering unexpected behavior or patterns within the data, which are referred to as anomalies, outliers, or novelties [3, 4]. These anomalies can be categorized into different types [5], such as single instance anomaly, group anomaly, local anomaly, and collective anomaly. Currently, numerous machine-learning techniques are employed to identify abnormalities in large real-time datasets. Conventional machine learning methods rely on various assumptions. One of these assumptions [6] is that the entire dataset could fit into the memory, which is no longer true in the current context. Online supervised methods [7] utilize labeled training data to construct a model potential of detecting deviants in real-time data streams. They may, however, confront difficulties when adapting to evolving patterns within the streaming data.

A unique approach to anomaly detection in data streams is utilizing the Hierarchical Temporal Memory (HTM) [8], an online sequence memory algorithm. A fundamental algorithm is specifically designed to handle real-world data streams that include labeled abnormalities. The Numenta Anomaly Benchmark (NAB) dataset is used as a reference to evaluate its performance. The Preprocessed Isolation Forest (PiForest) [9] is an anomaly detection concept that is appropriate for resource-limited environments and performs well on streaming data. It effectively handles streaming data by employing a sliding window mechanism, which organizes the data into sequential blocks for systematic processing. However, detecting anomalies within streaming data is challenging, as it necessitates real-time data processing and simultaneous learning and prediction. An influential anomaly detection model [10] derived from an online sequence memory method has demonstrated successful outcomes in the real-time detection of anomalies in financial metrics through a live application.

The emergence of the Internet of Things (IoT) has revolutionized numerous domains, such as healthcare, transportation, manufacturing, and smart cities, among others. It involves numerous devices that generate enormous volumes of data, necessitating considerable computational resources. Anomaly detection and security pose significant challenges in the IoT domain. A comprehensive survey [11] highlights the primary research issues and challenges of utilizing deep learning for anomaly detection for resource-constrained devices in real-life IoT hurdles.

The application of online machine learning algorithms on streaming data presents several challenges such as the need for real-time processing, limited availability of labeled data and ground truth, handling high volume and dimensionality of the data, as well as dealing with imbalanced class distribution. To address the existing challenges, extensive research opportunities are available. In this context, two advanced streaming frameworks, River and PySAD, are used to implement the nine online machine learning models used in this study. The models were evaluated for their effectiveness in identifying the anomalies in the streaming data on the basis of the performance metrics: - Accuracy, Sensitivity, Specificity, and ROCAUC, on the widely used IoT-based IoTID20 streaming dataset.

This research paper is structured as follows: - section II describes the related literature work, section III defines the system's process flow, section IV explains the experimental setup and discusses the results with a comparative study and Section V concludes the research.

II. RELATED WORK

Anomaly detection has become crucial in various applications, including computer security and sensor networks. Over the past few years, there has been increased curiosity in online machine learning anomaly detection concepts for streaming data. Researchers have put forth diverse approaches to address the challenges involved in spotting anomalies in real-time data streams. An efficient isolation tree-based concept, HS-Trees [14] is a rapid one-class anomaly detection approach designed for dynamic data streams. It operates efficiently using only normal data for training and demonstrates effectiveness in scenarios where anomalous data is infrequent. Notably, HS-Trees constructs trees without relying on any data. Ding and Fei proposed a novel anomaly detection framework considering the sliding window approach and accounting for concept drift [15]. Within this framework, a streaming data anomaly detection algorithm called iForestASD is composed, which is adapted from the iForest algorithm. A variant of the Isolation Forest algorithm called PCB-iForest was developed in [16], which serves as an extended version capable of accommodating any ensemble-based online outlier detection method for streaming data. This analysis emphasizes the need for dynamic and productive resolution, which is effectively addressed by PCB-iForest. A groundbreaking approach [26] leverages K-Nearest Neighbors (KNN) for outlier detection in IoT-based streaming data. The process entails three essential steps. Firstly, it employs the innovative grid indexing concept to handle streaming data, utilizing self-adjustable cells that effectively filter out abnormal objects. Secondly, it incorporates the concept of a minimum heap to determine an object's minimum and maximum distances. Lastly, it effectively distinguishes between normal and abnormal instances by detecting the presence of outliers. An online anomaly detection procedure was proposed in [20] that gradually evaluates data to identify deviants in time series data streams. It consists of two states. The primary phase utilizes a fully online density assessment approach, which is minimax optimal concerning log loss and gains satisfactory performance. In the second stage, a threshold calibration method is introduced, which is minimax optimal compared to the most suitable threshold, retrospectively, using surrogate logistic loss. Streaming data mining faces significant challenges, particularly the issue of class imbalance, which can negatively affect the predictive ability of online methods. Numerous existing methods for online learning lack an adequate operation to address big-size streaming data with imbalanced class distributions. Consequently, this leads to diminished model performance and limited effectiveness in addressing class imbalance. The cost-sensitive regularized dual averaging (CSRDA) model [21] significantly expands upon the well-known regularized dual averaging approach. It achieves this by introducing a novel convex optimization function, thereby enhancing the method's capabilities. Liu et al. [22] have introduced a novel anomaly detection framework for infrequent data streams. Their approach incorporates two main components: a monitoring module that dynamically determines measurement windows for scrutiny of data and a tracking module that employs a segregation partition strategy to assess the irregularity level of every latest observation. Boateng et al. [27] presented a new method for detecting anomalies without needing labeled data. Their approach is based on a weighted voting ensemble concept and employs a stacked-based ensemble model with isolation forest as the meta-learner. The framework utilizes a programmable logic controller and an industrial control system to monitor and manage resources in various attack scenarios by identifying anomalies. Anomaly detection becomes challenging when dealing with streaming data from Internet of Things (IoT) sources. The imbalance in data due to fewer anomalies in IoT sources is addressed by employing a dynamic ensemble approach [28]. This approach leverages IoT sources and utilizes normal data synthesis with borderline-SMOTE to mitigate the imbalanced problem.

III. WORKFLOW OF THE STUDY

The process flow for anomaly detection from data streams is adapted to evaluate the classification performance of two recently introduced streaming frameworks, River and PySAD. The online machine learning models from both frameworks are trained in an unsupervised manner on the streaming data. The IoT-based streaming dataset, IoTID20, is used for the evaluation. The performance metrics are Accuracy, Sensitivity, Specificity, and ROCAUC. The details of the River and PySAD models are given below, and the process flow diagram is shown in Fig. 1.

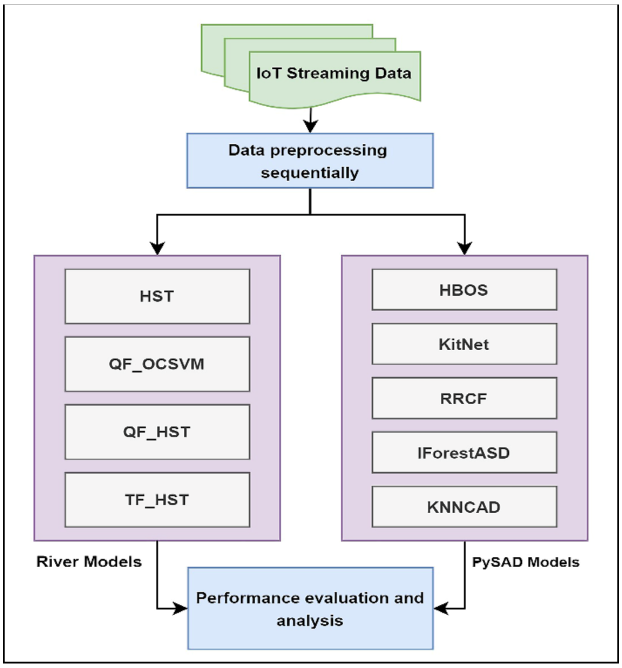

Fig. 1. Process flow diagram of the system.

A. River Framework

1. Half Space Trees (HST): Tan et al. [14] executed a fast one-pass streaming-based model named HST. The HS-Trees method exclusively utilizes normal data during its training phase because anomalous data is infrequent and might not be accessible for the training process. Updating the model is both straightforward and rapid since it necessitates no alterations to the tree structure when handling streaming data.

2. Quantile Filter One-Class Support Vector Machine (QF_OCSVM): During the initial training phase, OCSVM focuses on normal data, creating a boundary encompassing these non-anomalous data points in a high-dimensional space. This involves setting a decision boundary to separate normal instances from the rest of the space. The use of One-Class SVM involves integrating a Quantile Filter to enhance and fine-tune the anomaly detection process.

3. Quantile Filter Half Space Trees (QF_HST): The pipeline starts by normalizing features using a min-max scaler and follows up with anomaly detection using Half Space Trees within a Quantile Filter, emphasizing data points within a specific quantile. This configured pipeline is ready for both training on existing data and making predictions on new data instances.

4. Threshold Filter Half Space Trees (TF_HST): The anomaly scores generated by the Half Space Trees are then passed through a Threshold Filter. This filter is configured with a specific threshold value, suggesting that instances with anomaly scores above this threshold will be considered anomalies. This configured pipeline is ready for both training on existing data and making predictions on new data instances.

B. PySAD Framework

1. Histogram-Based Outlier Scores (HBOS): Goldstein et al. [12] designed an unsupervised outlier detection algorithm called HBOS, which uses histograms to represent the densities of univariate features. During the training phase, both normal and abnormal observations are employed, and the model identifies outliers as part of the fitting process. HBOS utilizes a referential windowing approach; the window size determines the quantity of data points employed for model training, and the sliding size governs how frequently the model is updated.

2. Kitsun's core Algorithm (KitNet): Mirsky et al. [29] designed a network intrusion detection-based methodology known as KitNet. The KitNet model utilizes unsupervised autoencoders, where only normal samples are involved in the training phase. Abnormal samples are excluded from training and are later employed for anomaly detection.

3. Robust Random Cut Forest (RRCF): Guha et al. [30] concentrated on a dynamic streaming-based anomaly detection concept. In this concept, the model learns a favorable score threshold using a training set and evaluates its effectiveness on a separate test set. The training set comprises all points before time t, while the test set comprises all points after time t.

4. Isolation Forest Anomaly Streaming Data (IForestASD) [15]: The IForestASD algorithm is unsupervised and does not require subclass attributes in the anomaly detection process. Anomalies labels are solely employed for assessing the last anomaly detection performance. When updating the detection model, there is an option to discard old instances while simultaneously incorporating new ones selectively.

5. K Nearest Neighbors Conformal Anomaly Detection (KNNCAD): Burnaev et al. [31] deployed a standardized distance and density-based approach, during the training phase. This approach calculates the probability that for each new observation, a consideration with a more utmost value of a Non-Conformity Measure (NCM) exists in the training set compared to the NCM value of the new observation.

IV. EXPERIMENTAL SETUP AND RESULTS

A. Experimental Setup

In this paper, we utilize the online machine learning concept and employ well-known open-source libraries for streaming data, PySAD [17, 18] and River (ver. 0.20.0) [1], along with the scikit-learn Python library [19], for anomaly detection in IoT data streams. The entire experimental setup was implemented on Jupyter Notebook version 6.5.4, running on a PC with the configurations D25U2P3, Intel Core i5-8265U CPU @ 1.60GHz 1.80 GHz, and 8.00 GB of RAM. To assess the performances of the River and PySAD framework's anomaly detection models, a comparison was made against four River-based anomaly detection models: HST, QF_OCSVM, QF_HST, and TF_HST and five PySAD-based anomaly detection models: HBOS, KitNet, RRCF, IForestASD, and KNNCAD. The evaluation was based on four reliable metrics, namely Accuracy, Sensitivity, Specificity, and ROCAUC. The performance evaluation was conducted using the widely used IoTID20 streaming dataset, and the results demonstrated that the River framework has more consistent models than the PySAD framework, which outperformed in terms of satisfactory scores. Typically, the hyperparameter configuration defaults to all nine streaming anomaly detection models.

B. Details of IoTID20 Streaming Dataset

The IoTID20 dataset [25] is widely recognized in IoT research for detecting anomalies. It mimics a real IoT setting, encompassing diverse network traffic data from IoT devices. The dataset encompasses 20 device categories such as cameras, smart bulbs, and thermostats, generating both normal and anomalous network behavior. Notably, it contains crucial features such as packet size, inter-arrival time, protocol, and destination port. As a result, the IoTID20 dataset proves instrumental in assessing anomaly detection algorithms and crafting resilient solutions to safeguard IoT networks.

C. Performance Metrics

Let t_p depict true positives, t_n depict true negatives, f_p depict false positives, and f_n depict false negatives.

1. Receiver Operating Characteristics Area Under Curve (ROCAUC): AUC suggests measuring the whole two-dimensional space below the total ROC curve. In other words, AUC quantifies the extent of the area covered by the ROC curve, encompassing all its data points.

2. Sensitivity: Sensitivity evaluates the proportion of original positive instances correctly detected by a method. Mathematical descriptions are shown as

$$Sensitivity = \frac{t_p}{[t_p + f_n]} \qquad (1)$$

where t_p depicts the count of true positives, f_n signifies the count of false negatives.

3. Specificity: Specificity evaluates the proportion of original negative instances correctly detected by a method. Mathematical descriptions are shown as

$$Specificity = \frac{t_n}{[t_n + f_p]} \qquad (2)$$

where t_n depicts the count of true negatives, f_p is the count of false positives.

4. Accuracy: Accuracy assesses the entire correctness of the model's predictions by computing the ratio of correct predictions to the whole count of predictions made. Mathematical descriptions are shown as

$$Accuracy = \frac{[t_p + t_n]}{[t_p + t_n + f_p + f_n]} \qquad (3)$$

D. Discussion on Results

The nine online machine learning models from the River, and PySAD streaming frameworks are evaluated for their performance on anomaly detection from IoT data streams using reliable metrics like Accuracy, Sensitivity, Specificity, and ROCAUC. The performance scores are summarized in Table I.

TABLE I. PERFORMANCE METRICS OF RIVER AND PYSAD FRAMEWORK'S MODELS

Frameworks	**Methods**	**IoTID20 Streaming Dataset**			
		Performance Metrics			
		Accuracy	**Sensitivity**	**Specificity**	**ROCAUC**
River Models	HST	0.8608	1.0000	0.9194	0.4426
	QF_OCSVM	0.6386	0.8485	0.6242	0.6837
	QF_HST	0.9257	1.0000	0.9890	0.7810
	TF_HST	0.8306	1.0000	0.8874	0.4432
PySAD Models	HBOS	0.8393	0.8839	0.8847	0.5040
	KitNet	0.9360	1.0000	1.0000	0.5000
	RRCF	0.8254	0.8791	0.8606	0.4954
	IForestASD	0.5019	0.8346	0.8562	0.7937
	KNNCAD	0.8430	0.8950	0.8937	0.5000

Sensitivity quantifies the correct identification of true anomalies; among River models, HST, QF_HST, and TF_HST show higher sensitivity scores of 1.00, while QF_OCSVM shows an average score of 0.8485. Similarly, among the PySAD models, KitNet shows a higher sensitivity score of 1.00, and the remaining models, HBOS, RRCF, IForestASD, and KNNCAD achieve average scores of 0.8839, 0.8791, 0.8346, 0.8950, respectively. Thus, the River models are more consistent than the PySAD models. Specificity measures the correct identification of normal instances. The specificity of River models: QF_HST and HST show higher scores of 0.9890 and 0.9194, while the remaining two models show average scores. Among PySAD models, the KitNet model shows a higher score of 1.00, and the remaining models show satisfactory scores. ROCAUC gauges the model's ability to distinguish between anomalies and normal instances, with a higher ROCAUC indicating superior performance. QF_HST shows a higher ROCAUC score of 0.7810 for the River framework, while the other models demonstrate average performance. Similarly, most of the PySAD models show average ROCAUC scores, except for IForestASD, which shows a higher score of 0.7930. Accuracy assesses the overall correctness of the anomaly detection model. Both framework models show good accuracy, but IForestASD shows an average accuracy score. These performance metrics are valuable in evaluating the performance of online anomaly detection methods and facilitating comparisons between different approaches. The selection of metrics is contingent upon the specific requirements of the application and the significance of different error types in the context of anomaly detection. Figs. 2 and 3 visually depict the overall performance comparison of the models in the two frameworks, River and PySAD.

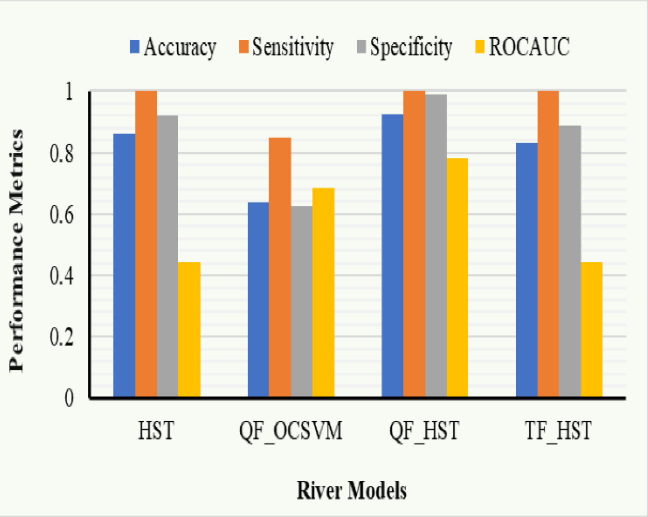

Fig. 2. Performance comparison of River framework's anomaly detection models on the IoTID20 streaming dataset.

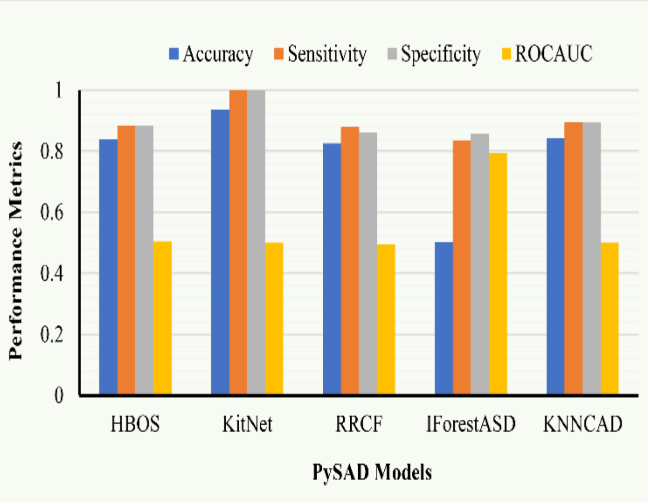

Fig. 3. Performance comparison of PySAD framework's anomaly detection models on the IoTID20 streaming dataset.

V. CONCLUSION

In this research, we have evaluated and analyzed nine online machine learning models belonging to two popular streaming frameworks, River and PySAD, for anomaly detection in IoT data streams. Using these two frameworks, we select nine popular unsupervised anomaly detection models (HST, QF_OCSVM, QF_HST, TF_HST, HBOS, KitNet, RRCF, IForestASD, and KNNCAD) to compute the key performance metrics (Accuracy, Sensitivity, Specificity, and ROCAUC) that signify the presence or absence of an anomaly in the benchmark IoTID20 streaming dataset. It is observed that the River framework models generally outperform the PySAD models in terms of Accuracy, Sensitivity, Specificity, and ROCAUC. The River models HST, QF_HST, and TF_HST attained the highest Sensitivity scores of 1.00, whereas only the PySAD model KitNet passed this Sensitivity level, In terms of ROCAUC, the PySAD model IForestASD performs best (0.7937) followed by the River model QF_HST (0.7810), implying that QF_HST performs best among all methods. Although detecting anomalies in streaming data remains challenging, our findings suggest promising future applications for these concepts for various real-time data. In future endeavors, we intend to broaden this research to tackle imbalanced data in the setting of data streams and will focus on the initial likelihood using varying window concepts.

REFERENCES

[1] Montiel, Jacob, Max Halford, Saulo Martiello Mastelini, Geoffrey Bolmier, Raphael Sourty, Robin Vaysse, Adil Zouitine et al. "River: machine learning for streaming data in python." The Journal of Machine Learning Research 22, no. 1 (2021): 4945-4952.

[2] Ray, Santosh Kumar, and Seba Susan. "Performance Evaluation using Online Machine Learning Packages for Streaming Data." In 2022 International Conference on Computer Communication and Informatics (ICCCI), pp. 1-6. IEEE, 2022.

[3] Ben-Gal, Irad. "Outlier detection." Data mining and knowledge discovery handbook (2005): 131-146.

[4] Pimentel, Marco AF, David A. Clifton, Lei Clifton, and Lionel Tarassenko. "A review of novelty detection." Signal processing 99 (2014): 215-249.

[5] Samariya, Durgesh, and Amit Thakkar. "A comprehensive survey of anomaly detection algorithms." Annals of Data Science 10, no. 3 (2023): 829-850.

[6] A L'heureux, Alexandra, Katarina Grolinger, Hany F. Elyamany, and Miriam AM Capretz. "Machine learning with big data: Challenges and approaches." Ieee Access 5 (2017): 7776-7797.

[7] Ma, Jiangang, Le Sun, Hua Wang, Yanchun Zhang, and Uwe Aickelin. "Supervised anomaly detection in uncertain pseudoperiodic data streams." ACM Transactions on Internet Technology (TOIT) 16, no. 1 (2016): 1-20.

[8] Ahmad, Subutai, Alexander Lavin, Scott Purdy, and Zuha Agha. "Unsupervised real-time anomaly detection for streaming data." Neurocomputing 262 (2017): 134-147.

[9] Jain, Prarthi, Seemandhar Jain, Osmar R. Zaïane, and Abhishek Srivastava. "Anomaly detection in resource constrained environments with streaming data." IEEE Transactions on Emerging Topics in Computational Intelligence 6, no. 3 (2021): 649-659.

[10] Ahmad, S., & Purdy, S. (2016). Real-time anomaly detection for streaming analytics. arXiv preprint arXiv:1607.02480.

[11] Sharma, Bhawana, Lokesh Sharma, and Chhagan Lal. "Anomaly detection techniques using deep learning in IoT: a survey." In 2019 International conference on computational intelligence and knowledge economy (ICCIKE), pp. 146-149. IEEE, 2019.

[12] Goldstein, Markus, and Andreas Dengel. "Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm." KI-2012: poster and demo track 1 (2012): 59-63.

[13] Angiulli, Fabrizio, and Fabio Fassetti. "Detecting distance-based outliers in streams of data." In Proceedings of the sixteenth ACM conference on Conference on information and knowledge management, pp. 811-820. 2007.

[14] Tan, Swee Chuan, Kai Ming Ting, and Tony Fei Liu. "Fast anomaly detection for streaming data." In Twenty-second international joint conference on artificial intelligence. 2011. ISBN:978-1-4503-0000-0/18/06.

[15] Ding, Zhiguo, and Minrui Fei. "An anomaly detection approach based on isolation forest algorithm for streaming data using sliding window." IFAC Proceedings Volumes 46, no. 20 (2013): 12-17.

[16] Heigl, Michael, Kumar Ashutosh Anand, Andreas Urmann, Dalibor Fiala Martin Schramm, and Robert Hable. "On the improvement of the isolation forest algorithm for outlier detection with streaming data." Electronics 10, no. 13 (2021): 1534.

[17] Yilmaz, Selim F., and Suleyman S. Kozat. "Pysad: A streaming anomaly detection framework in python." arXiv preprint arXiv:2009.02572 (2020).

[18] Zhao, Yue, Zain Nasrullah, and Zheng Li. "Pyod: A python toolbox for scalable outlier detection." arXiv preprint arXiv:1901.01588 (2019).

[19] Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel et al. "Scikit-learn: Machine learning in Python." the Journal of machine Learning research 12 (2011): 2825- 2830.

[20] Gokcesu, Kaan, and Suleyman S. Kozat. "Online anomaly detection with minimax optimal density estimation in nonstationary environments." IEEE Transactions on Signal Processing 66, no. 5 (2017): 1213-1227.

[21] Chen, Zhong, Victor Sheng, Andrea Edwards, and Kun Zhang. "An effective cost-sensitive sparse online learning framework for imbalanced streaming data classification and its application to online anomaly detection." Knowledge and Information Systems 65, no. 1 (2023): 59-87.

[22] Liu, Fengrui, Yang Wang, Zhenyu Li, Hongtao Guan, and Gaogang Xie. "AD2S: Adaptive anomaly detection on sporadic data streams." Computer Communications (2023).

[23] Kriegel, Hans-Peter, Peer Kröger, Erich Schubert, and Arthur Zimek. "LoOP: local outlier probabilities." In Proceedings of the 18th ACM conference on Information and knowledge management, pp. 1649-1652. 2009.

[24] Manzoor, Emaad, Hemank Lamba, and Leman Akoglu. "xstream: Outlier detection in feature-evolving data streams." In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1963-1972. 2018.

[25] https://sites.google.com/view/iot-network-intrusion-dataset/home

[26] Zhu, Rui, Xiaoling Ji, Danyang Yu, Zhiyuan Tan, Liang Zhao, Jiajia Li, and Xiufeng Xia. "KNN-based approximate outlier detection algorithm over IoT streaming data." IEEE Access 8 (2020): 42749-42759.

[27] Boateng, Emmanuel Aboah. "Unsupervised Ensemble Methods for Anomaly Detection in PLC-based Process Control." *arXiv preprint arXiv:2302.02097* (2023).

[28] Jiang, Jun, Fagui Liu, Yongheng Liu, Quan Tang, Bin Wang, Guoxiang Zhong, and Weizheng Wang. "A dynamic ensemble algorithm for anomaly detection in IoT imbalanced data streams." *Computer Communications* 194 (2022): 250-257.

[29] Mirsky, Yisroel, Tomer Doitshman, Yuval Elovici, and Asaf Shabtai. "Kitsune: an ensemble of autoencoders for online network intrusion detection." *arXiv preprint arXiv:1802.09089* (2018).

[30] Guha, Sudipto, Nina Mishra, Gourav Roy, and Okke Schrijvers. "Robust random cut forest based anomaly detection on streams." In *International conference on machine learning*, pp. 2712-2721. PMLR, 2016.

[31] Burnaev, Evgeny, and Vladislav Ishimtsev. "Conformalized density- and distance-based anomaly detection in time-series data." *arXiv preprint arXiv:1608.04585* (2016).