An Evaluation of Time-Series Anomaly Detection in Computer Networks

Hong Nguyen[1], Arash Hajisafi[2], Alireza Abdoli[2], Seon Ho Kim[2], Cyrus Shahabi[2]
[1]*Department of Electrical and Computer Engineering*
[2]*Department of Computer Science*
Viterbi School of Engineering, University of Southern California
Los Angeles, USA
hongn, hajisafi, aabdoli, seonkim, shahabi@usc.edu

***Abstract*—One critical issue in any network systems is failure detection. Failures not only impact the network it originates from but also propagate through other communicating networks due to its butterfly effect, making it even more challenging to find the origin. Therefore, the need to detect failures or anomalies in computer networks is fundamental. Due to the nature of computer networks, data is received in a time-series format where each time-point has temporal dependencies on each other. As a result, time-series analysis stands out as a potential approach to deal with the task of anomaly detection. In this paper, we conduct studies on multivariate time series anomaly detection, varying from traditional machine learning to deep learning models in computer networks. We show that the choice of models is not as important as the choice of preprocessing techniques. Interestingly, non-linear normalization can boost the performance of deep detectors by around 20% in terms of F1 score and balance the preference of deep detectors for specific types of anomalies. We also study the bias of anomaly types to deep detectors, time-performance trade-offs, shortage of data, and weak labeling on both synthetic and real-world datasets to fill out the missing insights in the literature.**

***Index Terms*—Time-series Data Analysis, Anomaly Detection, Computer Networks, Deep Learning, Unsupervised Learning**

I. Introduction

The requirements of upcoming network applications, along with advances in virtualization and software-defined networks (SDNs) have driven networks to grow in size, depth, and complexity and to become highly dynamic. Thus, new challenges arise in terms of network management, including traffic analysis, congestion control, resource and fault management, QoS / QoE management, and security issues. One critical challenge with such a complex computer network system is to detect the cause of failures when the system does not operate as expected. A failure in network communications can quickly lead to disruption of services, degraded quality of user experience, financial losses, and may even endanger human lives when it affects a safety-critical infrastructure [1], [2]. Hence, the need to improve troubleshooting procedures and quickly recover from potential incidents only keeps increasing as we rely more on network infrastructures such as the Internet in everyday life.

The origins of network failure can be categorized into three groups, including non-human activities (e.g., defective switch/router, environmental effects), human activities (e.g., rush hours, cyclic action of administrators), and malicious behaviors (e.g., DDoS, Flood attack, Worm attack) [2]. As for the location of occurrence, failures happen in the physical layer (hardware), data link layer, network layer, and application layer. Among many causes of failures, this paper focuses malicious attacks as the main cause of network failures and investigates time-series anomaly detection techniques to detect these attacks.

Why time series analysis? Computer network traffic data are, in nature, continuous over time with a huge volume. Apart from batch processing, time-series samples are ordered by time and contain temporal dependencies. Real-world problems show different temporal properties, such as trend, seasonality, and periodic patterns. For instance, the DNS server in Los Angeles got an increasing traffic load from 6 pm PST to 10 pm PST, which is normal since it is the time duration that people access the Internet the most. Due to the advances in machine learning and deep learning, time series analysis have gained significant attention from researchers in handling complex dynamic networks [3], [4].

Challenges and Contributions Real world time-series analysis usually suffers from weak labeling and shortage of labeled data, resulting in the performance degradation of anomaly detection techniques. Particularly in computer networks, massive packet flows are being generated every second and make it challenging for deep learning models to perform efficiently in time-series analysis. Furthermore, we have little ground truth to decide on anomaly labels at the packet level. Instead, network operators might observe system level behavior to gain a textual description of an anomaly for a specific period of time. Thus recent deep learning methods for multivariate anomaly detection are learning the normal patterns from a anomaly-free training set rather than learning anomaly patterns from a well labeled dataset [5]–[11]. However, guaranteed anomaly-free data is not always available in reality either so it leaves a question regarding how sensitive deep anomaly detectors to the training set that contains missed labeled anomalies. Another critical challenge is the weak generalization of different types of anomalies [12], [13] due to the shortage of labeled anomalies in training data.

To address these challenges, we derived several experiments on different time-series anomaly detectors for computer net-

works. We investigate factors that may affect the performance of anomaly detectors in real-world scenarios, such as weak labeling, shortage of labeled data, bias of anomalies, and time-performance trade-offs. To summarize, the main contributions of our work are:

- We conduct an evaluation of multivariate time-series anomaly detectors from traditional unsupervised ML to DL models on both synthetic and real-world computer network datasets. Surprisingly, classical OCSVM and Isolation Forest gain a reasonable performance even though they do not require an explicit training step.
- We propose a Time Aggregation and Pre-processing (TAP) procedure to convert computer network data from packet-indexed to time-indexed. We show that the selection of pre-processing methods might be more important than the choice of the underlying models.
- We investigate the bias/favoritism of deep anomaly detectors on different types of anomalies. Interestingly, non-linear normalization can balance this bias and boost the performance of deep detectors by around 20% in terms of F1 score. In addition, we conduct several studies on anomalous training dataset, time-performance trade-offs, and weak labeling in real-world datasets.

The remainder of this paper is organized as follows. In Section II, we conduct a literature survey on anomaly detection methods for time-series data. Section III describes the formal problem definition followed by our methodology. Experiment results are discussed in Section IV. We conclude in Section V with a brief overview of our contributions.

II. Related work

Over the last decade, there has been increasing attention to anomaly detection on multivariate time series. In this section, we present the previous work on this topic.

Traditional Unsupervised Machine Learning Models Traditional methods for time-series mostly rely on statistical metrics [5] such as distance, density, and entropy to detect anomalies in time series. For instance, One-Class SVM [14] estimates the density distribution of the dataset and classifies data points as normal or abnormal based on the difference in variance from the true distribution. K-Nearest Neighbor (KNN) [15] and Local Outlier Factor (LOF) [16] compute point-wise anomaly scores of sample data points according to the group of neighbors for each point. Random Forest [17] constructs a multitude of decision trees at training time and classifies the points selected by most trees as anomalies. Isolation Forest (IForest) [18], which is built upon the idea of random forests, assumes that anomalous data points can be separated in fewer steps than the normal data points, which are closer to each other. However, these methods are designed for batch data (where the order of samples do not matter), not for time-series data; Therefore, they ignore the inherent temporal dependencies of time-series. To capture temporal dependencies, time series decomposition approaches, *e.g.*, Bayesian Structural Time Series [19], Autoregressive Integrated Moving Average [20] (ARIMA) and its variants have been proposed to model time series as a combination of trends, seasonal and periodic components, and noise. After this decomposition, ARIMA reconstructs the time series to identify anomalies based on the error between the original and reconstructed series. The drawback of these methods is the sensitivity to noise, as it may affect the generalization capability of temporal prediction and increase the likelihood of false positive detections.

Deep Learning Models Along with the traditional methods, deep learning models have been proposed to detect anomalies in multivariate time series [6]–[11] in an unsupervised fashion. Existing DL models for anomaly detection can be categorized into reconstruction-based, prediction-based, and linguistic-based models. The primary task of these models is to learn normal time-series patterns from normal training samples and predict anomalous samples in the test set. Reconstruction-based Models like MSCRED [6] and THOC [7], encode normal patterns of multivariate time series into latent embeddings and use reconstruction error to detect abnormal patterns. Prediction-based Models like DAGMM [21], OmniAnomaly [22], InterFusion [8], and GANF [23], use probabilistic methods to find a fitted distribution or random process that time-series samples are generated from. The scoring will be in the form of an anomaly probability. More recently, Linguistic-based Models [24], [25] make an adaptation of Transformer to time-series anomaly detection.

III. Time-Series Anomaly Detection

A. Problem Statement

We consider a network system that monitors n network attributes, $x \in \mathbb{R}^n$. Each attribute is collected in a streaming fashion from a univariate time series. The period between each time-points is Δt seconds. Assume that we have anomaly-free time-series X_{train} containing τ continuous time points.

$$X_{\text{train}} = \{x_0, x_1, ..., x_\tau\} \in \mathbb{R}^{\tau \times n} \tag{1}$$

For any test sequence $X_{\text{test}} = \{x_0, x_1, ..., x_\Upsilon\}$ having the same level of time aggregation Δt with X_{train}, we want to detect anomaly labels at system-level $y_{\text{pred}} = \{y_0, y_1, ..., y_\Upsilon\} \in \mathbb{R}^\Upsilon$ assuming that anomalies have inter-variable and temporal dependency property [8].

B. Anomaly Detection Algorithms

In this work, we first implement several underlying models from machine learning to deep learning families to detect attacks in computer networks. Then, we study factors in real-world attacks that may degrade anomaly detectors' performance.

We generalized deep anomaly detectors into four blocks [26], as shown in Fig. 1, including pre-processing, underlying model, scoring, and thresholding, respectively. The network time-series first go through the pre-processing block for data aggregation for every Δt period of time, followed by normalization. In the training phase, models take time sequences as input and output the reconstruction version or predicted version of the input or predict the next several time-points.

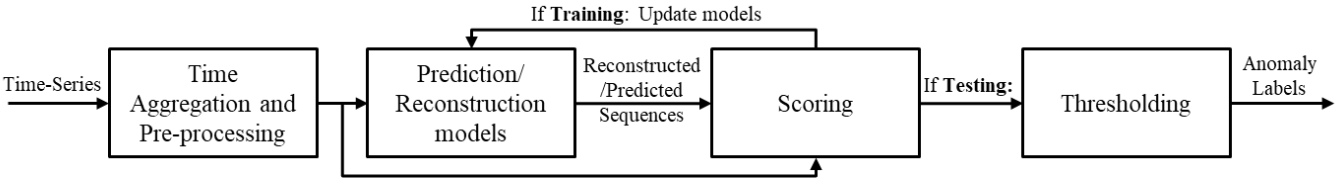

Fig. 1: Main blocks for deep time-series anomaly detection

In the third module, the scoring function compares the reconstructed/predicted sequence with the original sequence to output multi-channel errors. It also combines multi-channel errors to a single value error for each time-point. The last module is the thresholding function, which puts a threshold to discriminant anomaly scores into binary labels.

1) Varying models of detectors: On the same training set X_{train} and test set X_{test}, we alternatively test each of the detectors while varying pre-processing methods and underlying models. The choices of pre-processing, models, scoring, and thresholding function are described as follows.

Time Aggregation and Pre-processing (TAP): Network messages received at receivers come at an unpredictable rate. Plus, the attack behaviors usually are not recognizable from packet-level view-point but from traffic-level. For instance, a DDOS attack generates a massive number of packets to the target server. Each individual packet is normal during the attack, but the traffic behavior is unusual. Therefore, the task of aggregating messages to a higher level is indispensable. We aggregate network data for every Δt seconds with respect to several attributes (total number of packets, average message length, average inter-arrival time, etc.) and choose the best features that are highly correlated with traffic load by using a correlation function. The hyper-parameter Δt is selected during validation. For the normalization task, we investigate linear (Standard) and non-linear (Power Transformer, Quantile Transformer) normalizers. Standard normalization is widely used in machine learning problems by removing the mean and scaling data to unit variance $X_{\text{norm}} = \frac{X-\mu}{\sigma}$ where μ is the mean of the training samples and σ is the standard deviation of training samples. On the other hand, the Power Transformer scales each feature to be more Gaussian-like in order to stabilize variance and minimize skewness. One other method of non-linear normalization, Quantile Transformer, allows features to be scaled to uniform or Gaussian distribution that reduces the impact of marginal.

Choice of models: In this work, we evaluate traditional machine learning methods and deep learning methods for multivariate time-series anomaly detection with a case study of attacks in computer networks. For better generalization and get up-to-date methods, we select 1-2 models from each family of algorithms categorized by its mechanism. For traditional machine learning models, we evaluate tree-based (Isolation Forest), cluster-based (KNN), discriminant-based models (OC-SVM), density-based (LOF), and pattern recognization models (Auto-Regression, Matrix Profile). For deep learning, we take into account (a) reconstruction-based family - MSCRED, VAE, HVAE, (b) families of generative networks - DAGMM, USAD, and (c) linguistic-based models that gain much attention in natural language processing - TranAD.

Scoring and thresholding function: Because deep learning models are integrated with different built-in scoring functions, the task of synchronizing the same scoring function for every model is not an easy task. We will leave this task for future investigation. In this work, we utilize the default scoring setting of models. For thresholding, we use a static best-F1-score threshold that learns from the training set to discriminate between normal and abnormal time points.

2) Factors affecting anomaly detection on real-world data: The real-world anomaly detection task faced weak labeling data. The term 'weak labeling' refers to the case when network experts miss-label anomalies in the dataset or the abnormal period is not strictly correct. Meanwhile, deep learning models strictly require anomalous-free training data to learn the latent embedding of normality. To simulate the real-world scenario, we use training data that contain anomalies and test on different detectors. We discussed the robustness of each deep detector to anomalous training data in section IV-B1.

One other problem in real-world datasets is the bias or favoritism of anomaly detectors to different anomaly types. For instance, we investigate the detection abilities of detectors on different types of attacks in section IV-B2. We figure out that using non-linear normalization will make detectors less biased to different attack types. More discussion will be conducted in section IV-B2. In addition, we record the run-time of each detector to study the time-performance trade-off.

IV. Experimental Results

A. Datasets

We used a real-world network traffic dataset B-Root-ISI and a synthetic dataset CIC-IDS. Attack events were considered anomalies. The detailed statistics of both datasets are described in Table I.

TABLE I: Statistics of CIC-IDS and B-Root-ISI datasets.

Statistics	**CIC-IDS**	**B-Root-ISI**
Time aggregation	1 mins	1 mins
# Attributes (features)	6	12
# Anomalies portion	11%	7%
# Training set	1 day	3 days
# Test set	4 days	3 days

TABLE II: Anomaly detection results on two datasets. The two best F1 and AUC scores are highlighted in bold.

Datasets	B-Root-ISI		CIC-IDS	
Models	F1	AUC	F1	AUC
NonAD	0.907	0.500	0.733	0.500
IForest	0.875	0.731	0.861	0.699
AR	0.870	0.480	0.783	0.571
OCSVM	0.874	0.736	0.879	0.807
KNN	0.878	0.475	0.818	0.620
LOF	0.874	0.479	0.733	0.499
MatrixProfile	0.867	0.473	0.780	0.585
EncDec-AD	0.906	0.746	0.774	0.529
DeepLog	0.867	0.473	0.759	0.534
MSCRED	0.908	0.750	0.892	0.793
InterFusion	0.910	0.704	**0.908**	0.824
OmniAnomaly	**0.919**	0.708	**0.909**	**0.829**
TranAD	**0.919**	**0.769**	0.905	0.815
USAD	0.918	**0.768**	0.841	**0.959**
DAGMM	0.918	**0.768**	0.899	0.819

- B-Root-ISI dataset: The Domain Name Server (DNS) at B-Root is operated by the Networking and Cybersecurity Division at the Information Sciences Institute (ISI) - University of Southern California. The dataset is a collection of host-only DNS requests collected at B root. Message packets are collected in PCAPs format, then they are aggregated every 1 minute into a CSV file. Attack types include flood attacks, DDOS, IP fragmented queries.
- CIC-IDS dataset: CIC-IDS [27] is a synthetic dataset that consist of labeled network flows collected by Canadian Institution of Cybersecurity. The dataset contains benign and the most up-to-date common attacks (DDOS, SQL Injection, Infiltration, Botnet, Patator,...). The CIC-IDS dataset included one day of normal traffic and 4-days of traffic containing attack flows.

Evaluation Metrics: To evaluate performance of anomaly detection models, we use classic F1-score and Area under the ROC Curve (AUC).

B. Results

We used the same test set for both unsupervised ML models and DL models. While traditional unsupervised ML models do not need training phase [5], the training step is required for deep detectors. Anomaly-free training data is fed to deep learning models with default settings. We do not take into account the availability of a validation set to imitate the shortage of data in real-world problems. We introduce NonAD model, which predicts every time-points as normal, as a ground base for comparison. The NonAD model missed all anomaly points, hence, have the most false-negative predictions.

1) Effect of Models: For the synthetic dataset, CIC-IDS, as shown in Table II, both ML and DL detectors do better than the NonAD model. Compared with the NonAD detectors, traditional models gain up to 15% of F1 score with OCSVM. On the other hand, DL models achieve up to 18% with predictive-based detectors, InterFusion, and OmniAnomaly. In Table II, we observe that DL methods got 5-10% higher F1 score and 10-15% higher recall than traditional ML methods. Surprisingly, OCSVM and IForest demonstrate outstanding F1-score among ML models while being time-efficient. The time-performance trade-off will be discussed later in section IV-B3.

TABLE III: Comparasion of DL anomaly detectors results with and without anomalies in the training set for CIC-IDS dataset. The best F1 and AUC scores of each algorithm in two cases are highlighted in bold.

Datasets	Case 1		Case 2	
Models	F1	AUC	F1	AUC
MSCRED	**0.889**	0.770	0.880	**0.788**
InterFusion	0.890	0.771	**0.911**	**0.790**
TranAD	**0.886**	**0.752**	0.781	0.525
OmniAnomaly	**0.890**	**0.760**	0.763	0.502
USAD	**0.886**	**0.747**	0.793	0.595
DAGMM	**0.880**	**0.758**	0.786	0.536

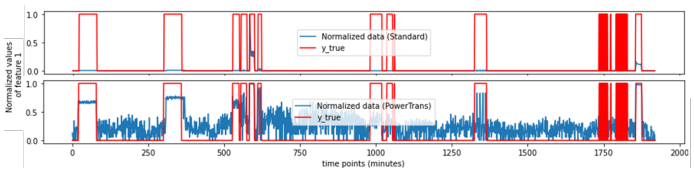

Fig. 2: Bias of anomalies

On the real-world dataset, B-Root-ISI, deep learning models show outstanding performance compared to ML models. Plus, It is noticeable that the F1-score of ML models is lower than that of NonAD methods. The reason is that DLs are trained on normal traffic while ML models do not have a training phase. Hence, DL models have more information for correctly thresholding the anomaly score. As a result, DL reduces a large amount of false positive samples while ML got a lot of False Positive due to the un-optimized threshold. Another reason is that the real-world dataset, B-Root-ISI, contains more complex patterns, such as trends and seasonalities, than the synthetic one, CIC-IDS. Among ML models, IForest and OCSVM show high AUC compare to nonAD. It is because these models reasonably capture anomalies but got a lot of false-positive predictions from un-optimized thresholding. Among DL models, TranAD and OmniAnomaly got the highest F1 score, while MSCRED, Interfusion, USAD, DAGMM get very close performance. There also have a simple difference between the two datasets in terms of detectors' performance. On the synthetic side, detectors can easily outperform NonAD models. Meanwhile, for the real-world dataset, only DL models may outperform NonAD model. The reason is that synthetic traffic is normaly

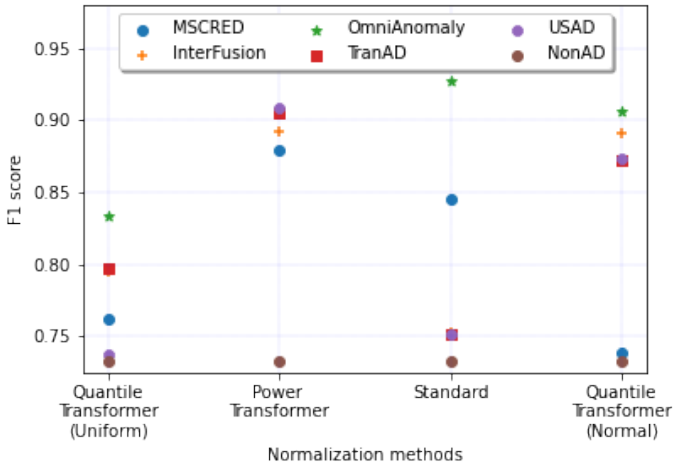

Fig. 3: F1 score of different normalization methods of CIC-IDS

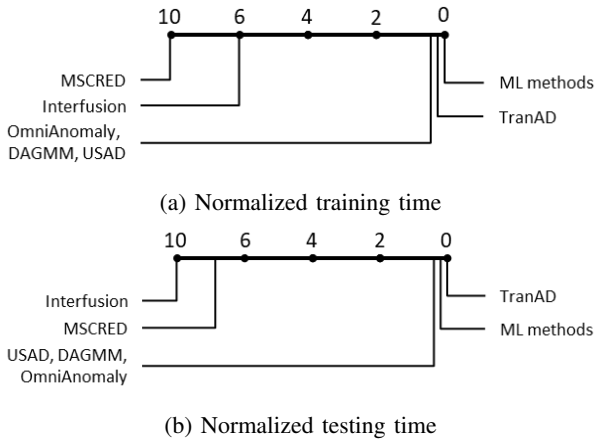

Fig. 4: Normalized run-time of (a) training phase and (b) testing phase.

generated while real-world data contain seasonality, trends, and periodic patterns that can only learn by complex DL models.

2) Bias of anomalies: First and second plots of Fig. 2 show a single dimension of CIC-IDS normalized by the Standard and Power Transformer, respectively. From the top plot, there are DDOS attacks around time-points 600 that are significantly more recognizable than other types of attacks. As a result, models tend to treat this peak of DDOS attacks as more important and reduce the score of other types of attacks to a minimum. Therefore, the thresholding decision is biased on the more important anomalies. To tackle this bias problem, we figured out that by applying non-linear normalization such as Power Transformer, it is likely to transform point-wise to pattern-wise anomalies (as shown in Fig. 2). Hence, the traits of different types of attacks in the case of non-linear normalization can be fairly recognizable.

Fig. 3 shows the F1 score of different models nornalized by four methods. As can be seen, Gaussian-based methods like Power and Quantile Transformer (with normal kernel) outperform Standard and Quantile Transformer (with uniform kernel). In addition, non-linear normalization boost DL's performance by 10-23% in comparison to the nonAD approach, while the Standard normalization only gives by 5% more than the ground. Interestingly, Interfusion show best F1 score even with Standard normalization. The reason is that Interfusion is intergrated with a built-in pre-processing module named "prefiltering strategy" which reduces the risk of overfitting to anomalous patterns while preserving the flexibility of the intermetric and temporal embeddings. In conclusion, the vary of up-to-date deep learning models in section IV-B1 have merely effects on performance while the change of pre-processing methods can boost the F1 score up to 23%.

3) Time-efficiency trade-off: Fig. 4a and 4b show normalized training and testing time across multiple models, respectively. Since no explicit training step is required, traditional unsupervised machine learning models remain the first place in Fig. 4a. MSCRED and Interfuion took the highest time for training and testing, around 2000 seconds, which is 20 times slower than linguistic models like TranAD. The reason is that TranAD deploys parallel data processing from the idea of language translation. Overall, light-weight models such as TranAD, OmniAnomaly, DAGMM, and traditional unsupervised ML work well in streaming environments while deeper models like MSCRED, and Interfusion take longer run-time, which should be better off with offline settings.

4) Effect of anomalous training set: Because of imperfect training set and weak labeling, training set often contain missed labeled anomalies which makes deep learning models to learn the wrong normal patterns. Table III shows two experiment settings where we train deep learning models on anomalous-free traffic and anomaly-contained traffic, respectively. The length of both training sets is one day. We use the same pipeline and the same test set for both cases for a trustworthy comparison. It can be seen that five out of six models are sensitive to anomalies-contained training data as the F1 score degraded by around 10%. Interfusion, on the other hand, shows the robustness to noise due to a built-in pre-filtering strategy that gets rid of possible outliers. Once again, it is not the model that makes a big difference in performance, but the additional module in pre-processing makes the detector robust with noise.

5) Anomaly detection on real-world data: Fig. 5 shows real DNS traffic under a real TCP SYN flood attack with spoofed sources. A TCP SYN flood DDoS attack occurs when the attacker floods the system with SYN requests in order to overwhelm the target and make it unable to respond to new real connection requests. The first plot show one of 12 attributes (Total number of query message) of the DNS server located at the Amsterdam site. The plot shows a sudden drop in the number of query messages due to an overflow of the DNS server to a new connection. It is temporally dependent since the drop significantly deviates from the normal pattern. Moreover, such a drop is also inter-variable depended since it

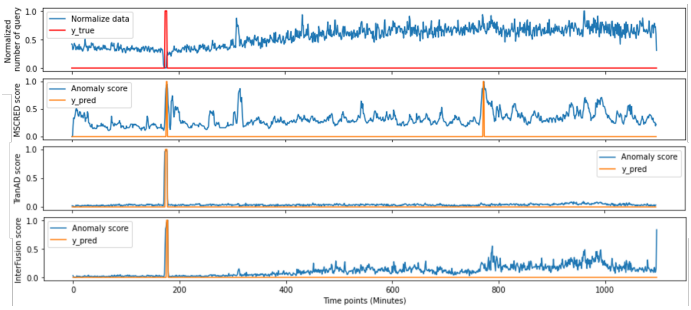

Fig. 5: Anomaly detection results on DNS traffic under a real TCP SYN flood attack with spoofed sources.

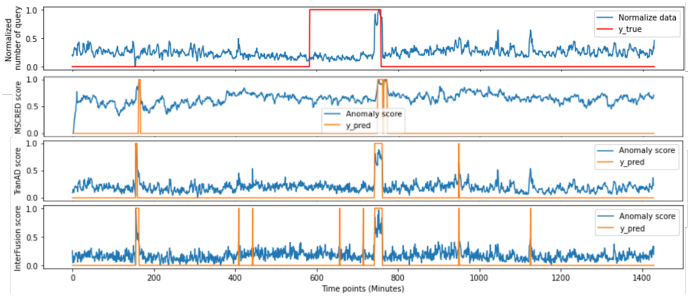

Fig. 6: Anomaly detection results on DNS traffic under large IP fragmented queries.

happened on most of the 12 features.

Fig. 6 shows attack that contain IP fragmented queries (large packet size) at Los Angeles' DNS server. The period of attack is claimed to be 09:54 UTC to 12:49 UTC by network experts. As can be seen on the first plot of figure 6, the real peak is only 20 minutes starting from 12:30 UTC. This is caused by weak labeling in the real-world dataset.

V. Conclusion

In this work, we have conducted a comprehensive evaluation of machine learning and deep learning models for time-series anomaly detection on both synthetic and real-world network data. Traditional unsupervised ML models such as OCSVM and IF showed comparable performance to up-to-date deep learning models while being time-efficient at the same time. While DL models outperform OCSVM by around 5% in the best pipeline setting, the selection of non-normalization boosts F1 score by 23%. The highest F1 score belongs to the predictive-based family, Interfusion and OmniAnomaly. Additionally, we discussed some lacking insights from literature, such as the bias of anomaly types to deep detectors, time-performance trade-off, shortage of data, and weak labeling.

References

[1] K. Roshan and A. Zafar, "Deep learning approaches for anomaly and intrusion detection in computer network: A review," in *Cyber Security and Digital Forensics*. Singapore: Springer, 2022, pp. 551–563.

[2] M. łgorzata Steinder and A. S. Sethi, "A survey of fault localization techniques in computer networks," *Science of Computer Programming*, vol. 53, no. 2, pp. 165–194, 2004, topics in System Administration.

[3] L. N. A. S. S. N. E.-S. F. C. O. M. A. Boutaba Raouf, Salahuddin Mohammad and C. Ji, "A comprehensive survey on machine learning for networking: evolution, applications and research opportunities," *Journal of Internet Services and Applications*, vol. 9, 2018.

[4] G. Bontempi, S. Ben Taieb, and Y.-A. Le Borgne, *Machine Learning Strategies for Time Series Forecasting*. Springer Berlin Heidelberg, 2013.

[5] S. Schmidl, P. Wenig, and T. Papenbrock, "Anomaly detection in time series: A comprehensive evaluation," *Proc. VLDB Endow.*, vol. 15, no. 9, p. 1779–1797, may 2022.

[6] C. Zhang, D. Song, Y. Chen, X. Feng, C. Lumezanu, W. Cheng, J. Ni, B. Zong, H. Chen, and N. Chawla, "A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data," in *AAAI*, 2019.

[7] L. Shen, Z. Li, and J. Kwok, in *Advances in Neural Information Processing Systems*.

[8] Z. Li, Y. Zhao, J. Han, Y. Su, R. Jiao, X. Wen, and D. Pei, "Multivariate time series anomaly detection and interpretation using hierarchical inter-metric and temporal embedding," in *KDD*, 2021, pp. 3220–3230.

[9] J. Audibert, P. Michiardi, F. Guyard, S. Marti, and M. A. Zuluaga, "Usad: Unsupervised anomaly detection on multivariate time series," *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020.

[10] Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun, and D. Pei, "Robust anomaly detection for multivariate time series through stochastic recurrent neural network," 2019.

[11] D. Li, D. Chen, L. Shi, B. Jin, J. Goh, and S.-K. Ng, "Mad-gan: Multivariate anomaly detection for time series data with generative adversarial networks," in *ICANN*, 2019.

[12] Z. Ye, Y. Chen, and H. Zheng, in *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*.

[13] A. Tong, G. Wolf, and S. Krishnaswamy, "Fixing bias in reconstruction-based anomaly detection with lipschitz discriminators," *Journal of Signal Processing Systems*, vol. 94, 02 2022.

[14] G. Huang, H. Chen, Z. Zhou, F. Yin, and K. Guo, "Two-class support vector data description," *Pattern Recognition*, vol. 44, no. 2, pp. 320–329, 2011.

[15] V. Hautamäki, I. Kärkkäinen, and P. Fränti, "Outlier detection using k-nearest neighbour graph," in *ICPR*, 2004.

[16] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "Lof: Identifying density-based local outliers." ACM, 2000.

[17] S. D. D. Anton, S. Sinha, and H. Dieter Schotten, "Anomaly-based intrusion detection in industrial data with svm and random forests," in *2019 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, 2019, pp. 1–6.

[18] K.-H. Lai, D. Zha, J. Xu, Y. Zhao, G. Wang, and X. Hu, "Revisiting time series outlier detection: Definitions and benchmarks," in *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021.

[19] M. F. J. Steel, *Bayesian time series analysis*. London: Palgrave Macmillan UK, 2010, pp. 35–45.

[20] A. H. Yaacob, I. K. Tan, S. F. Chien, and H. K. Tan, "Arima based network anomaly detection," in *2010 Second International Conference on Communication Software and Networks*, 2010, pp. 205–209.

[21] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. ki Cho, and H. Chen, "Deep autoencoding gaussian mixture model for unsupervised anomaly detection," in *ICLR*, 2018.

[22] Y. Su, Y. Zhao, C. Niu, R. Liu, W. Sun, and D. Pei, "Robust anomaly detection for multivariate time series through stochastic recurrent neural network." ACM, 2019.

[23] E. Dai and J. Chen, "Graph-augmented normalizing flows for anomaly detection of multiple time series," in *ICLR*, 2022.

[24] S. Tuli, G. Casale, and N. R. Jennings, "Tranad: Deep transformer networks for anomaly detection in multivariate time series data," *Proc. VLDB Endow.*, vol. 15, no. 6, p. 1201–1214, feb 2022.

[25] J. Xu, H. Wu, J. Wang, and M. Long, "Anomaly transformer: Time series anomaly detection with association discrepancy," in *ICLR*, 2022.

[26] A. Garg, W. Zhang, J. Samaran, R. Savitha, and C.-S. Foo, "An evaluation of anomaly detection and diagnosis in multivariate time series," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 6, pp. 2508–2517, 2022.

[27] I. Sharafaldin, A. Habibi Lashkari, and A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," 01 2018, pp. 108–116.