A Deep Learning Model for Anomalous Wireless Link Detection

Conference Paper · September 2021

DOI: 10.1109/WiMob52687.2021.9606264

4 authors:

Blaz Bertalanic
Jožef Stefan Institute
13 PUBLICATIONS **7** CITATIONS
SEE PROFILE

Halil Yetgin
Bitlis Eren University
41 PUBLICATIONS **869** CITATIONS
SEE PROFILE

Gregor Cerar
Jožef Stefan Institute
22 PUBLICATIONS **77** CITATIONS
SEE PROFILE

Carolina Fortuna
Jožef Stefan Institute
102 PUBLICATIONS **747** CITATIONS
SEE PROFILE

A Deep Learning Model for Anomalous Wireless Link Detection

Blaz Bertalanic*^, Halil Yetgin*‡, Gregor Cerar*†, Carolina Fortuna*
*Department of Communication Systems, Jožef Stefan Institute, SI-1000 Ljubljana, Slovenia.
^Faculty of Electrical Engineering, University of Ljubljana, 1000 Ljubljana, Slovenia
†Jožef Stefan International Postgraduate School, Jamova 39, SI-1000 Ljubljana, Slovenia.
‡Department of Electrical and Electronics Engineering, Bitlis Eren University, 13000 Bitlis, Turkey.
{blaz.bertalanic | halil.yetgin | gregor.cerar | carolina.fortuna}@ijs.si

***Abstract*—Machine learning (ML) techniques play a significant role in detecting anomalous wireless links. However, to date, to the extent of our knowledge, there is no robust classifier that would work in a realistic scenario where various anomalies could appear concurrently in the time-series gleaned from the network monitoring tools. In this paper, we propose a new deep learning based classifier and show that is able to outperform the state of the art for existing link layer anomalies. Our evaluation results demonstrate that the state-of-the ML models perform with an average accuracy of about 63%, whereas the average accuracy of the proposed DL model is around 90%, indicating a significant improvement of 27% anomaly detection performance.**

***Index Terms*—anomaly detection, deep learning, machine learning, ensemble classifier, wireless networks, wireless links**

I. Introduction

Modern wireless networks, both cellular and non-cellular, besides exploring operations in higher frequencies from mm-wave up to THz, are undergoing also a number of other transformations, such as heavily relying on network and service virtualization, e.g., virtual network functions and orchestration, robust connectivity through massive multiple-input multiple-output (MIMO), and beamforming techniques. The networks are becoming more complex and more agile, and thus they can be configured much faster and more efficiently through software control compared to the conventional and manual methods when dedicated network devices came pre-configured. Additionally, Internet of Things (IoT) [1] devices are also often deployed on various legacy infrastructures that are monitored and the resulting data is used to optimize various processes.

Increasingly large and complex wireless infrastructures necessitate to be monitored, maintained and serviced similar to any other infrastructure, e.g., legacy IT infrastructure, industrial robots, machinery, electrical grids, logistics and health. Minimizing maintenance costs while ensuring network reliability is challenging when the number of devices is relatively small, i.e. in their tens, and becomes prohibitive in actual implementation when in their thousands or tens of thousands. To aid in effective management of such networks, automatic network monitoring [2], malfunction detection [3] and specific anomaly shape recognition [4] solutions, have been proposed. Many use machine learning as one of the most prominent approaches to tackle automatic knowledge extraction from large amounts of data that cannot be managed manually.

Assume a large IoT network of thousands of nodes and corresponding wireless links that (inter)connect them to a network. An automatic network monitoring system would collect, show and perhaps trigger an alert when the link between two nodes would behave as in Figure 1. A human operator would then study a number of dashboards with metrics corresponding to that link and, based on experience, understand what might be the underlying issue that causes the anomaly in order to trigger fixes. However, in large IoT networks, with thousands to millions of nodes, providing an automatic solution that would detect the shape of a given anomaly and identify the possible causes, would improve the reaction time of the human, cut the time to remedy and cut the overall operational costs of maintaining it. For the example shape presented in Figure 1, the underlying cause of the anomaly could be caused by buffer congestion or a software bug that determine the watchdog to periodically reboot the node, a radio remaining in excessive active state and requiring recalibration or an obstacle blocking the communication for some time as discussed more in detail in [4].

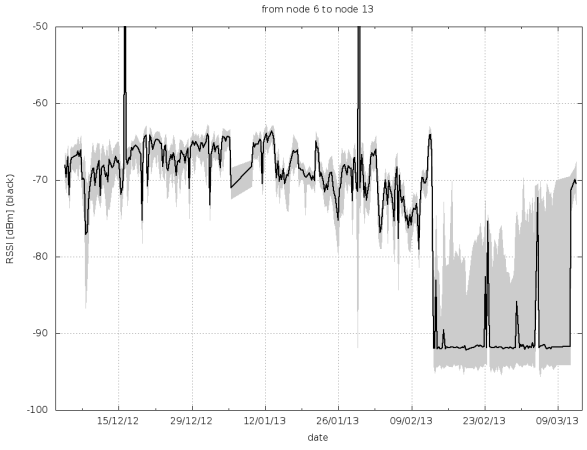


Figure 1. Example anomalous link observed over 3 months in a real-world wireless IoT network.

In this paper we propose a new, robust and accurate

supervised anomaly detector based on new deep learning architecture. We develop the detector to recognize the four types of anomalies introduced in [4], namely sudden link degradation (SuddenD), sudden link degradation with recovery (SuddenR), instantaneous link degradation (InstaD) and slow link degradation (SlowD). We show that the proposed model outperforms the state of the art link level anomaly detection models by up to 27% when all anomaly classes are taken into account. To the extent of our knowledge, this is the first attempt on anomaly detection in wireless networks considering a new DL approach employed over a set of predefined real-world wireless link anomalies.

This paper is organized as follows. We discuss the related work in Section II. Section III provides the formal problem statement, while Section IV discusses the model development for all the classifiers to be analyzed in the paper. Section V compares the performance of the resulting classifiers, while Section VI concludes the paper.

II. Related work

Wireless network malfunctioning can also be referred to as network and device anomalies, and to date, these anomalies have been defined in several ways, in particular from the perspective of wireless networking-related aspects. Machine learning is often used for assisting the development of anomaly detection systems.

A. Machine learning for wireless network anomaly detection

A large portion of the current research on anomaly detection in wireless networks focuses on intrusion, fraud and fault detection, event detection in wireless sensor networks (WSN), system health monitoring, and natural disaster [5].

Considering only classical machine learning (ML) algorithms, Salem *et al.* [6] evaluated the performance of five ML algorithms for anomaly detection in medical wireless sensor networks for health monitoring. Using SVM, decision trees, logistic regression, Naive Bayes, and decision table, they were able to distinguish between irregular variations in a patient's health parameters and faulty sensor data.

In our recent work [4], we considered the classical ML algorithms for link layer anomaly detection. We used three unsupervised (local outlier factor, isolation forests and one class SVMs) and three supervised ML algorithms (logistic regression, random forests and SVMs) for detecting 4 different anomalies by single class classification. Also deep learning techniques, more specifically autoencoders, were used but their scope was limited to feature generation only. More importantly, their training and evaluation process considered only one type of anomaly at a time, while in a realistic production environment all anomalies are likely to appear in the data.

Wazid [7] proposed an unsupervised way of detecting intrusions in wireless networks. They were able to distinguish between anomalous and non-anomalous links on traffic data using k-means algorithm achieving high recognition performance. Another anomaly detection with unsupervised ML was proposed by [8], where they detected misdirection and blackhole attacks in wireless environment on traffic data. For unsupervised learning they used k-medoid algorithm and successfully detect the anomalies with high accuracy.

As described in [9], there is a large list of possible ML models for anomaly detection and selecting a suitable ML algorithm for a particular application can be a challenging task, e.g., anomaly detection of wireless links [10].

B. Deep learning for wireless network anomaly detection

Anomaly detection with deep learning (DL) algorithms is often performed with unsupervised learning algorithms in the form of autoencoders (AE), while a few adopts supervised learning algorithms.

Recently, the authors of [11] evaluated the performance of four AE models for intrusion, anomaly detection and classification on captured IEEE 802.11 network data, and achieved high overall accuracy in classifying the attacks through this solution. Another study on cyber attack anomaly detection in wireless networks was conducted by [12], where they gathered data from transport layer traces. With the use AE network they increased performance results in detecting cyber attacks compared to the supervised approaches.

Similarly, the authors of [13] evaluated the performance of their proposed semi-supervised AE approach for anomaly detection. They used captured IEEE 802.11 network data for distinguishing between 4 different cyber attacks and achieved high overall performance accuracy. Additionally, another work on intrusion anomaly detection was realized by [14], where they encoded captured IEEE 802.11 network data into images and used supervised anomaly detection approach consisting of convolutional neural networks to distinguish between different cyber attack types with high accuracy.

Finally, an anomaly detection model was developed by [15], where they were trying to detect anomalies in wireless sensor data in the form of spikes and burst. By using an AE network, they were able to achieve high true positive rate of detection anomalies in data.

There is a plethora of work proposed for anomaly detection focusing on intrusion detection, albeit to the extent of our knowledge, this is the first attempt on anomaly detection in wireless networks using a new DL approach employed over a set of signal-level anomalies.

III. Problem Statement

Suppose an anomaly detection scenario in wireless networks as depicted in Figure 2. On the left side of the figure, wireless smart devices, IoT sensors and LPWAN residing on the premises of smart infrastructures and homes are depicted. To ensure high quality and possibly uninterrupted data communication for business processes, the operator of the smart infrastructure has in place an automated infrastructure that monitors the wireless link and notifies the technicians, represented on the right side of the figure, of malfunctions that are to be remedied. We assume that in this particular scenario, the anomalies need to be classified based on their

possible causes, as discussed in [4], before a decision on how to proceed with the remedy is taken. In some cases, the anomalies might be mitigated remotely through software upgrades, whereas in others an on-site visit will be required. We formulate the anomaly detection system portrayed in

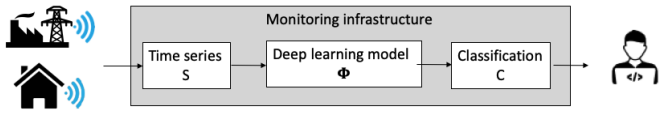
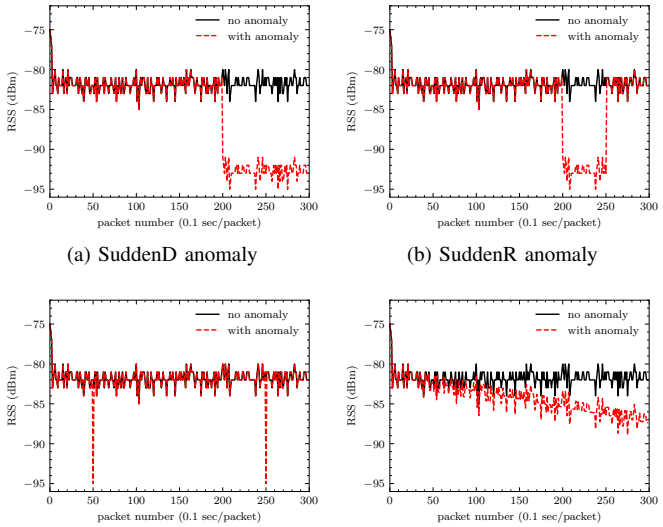


Figure 2. A high-level overview of the proposed anomaly detection system using DL model.

Figure 2 as a classification problem in which an input time series S is passed to the function of Φ that maps the input to a set of target classes C as provided in Eq. (1).

$$C = \Phi(S) \tag{1}$$

The cardinality of the set C, also denoted as $|C|$, provides the number of classes to be recognized. In this work, we start from the four types of anomalies introduced in [4] and consider a five-class classification problem with $|C| = 5$ where the set of target classes is defined to be $C = \{SuddenD, SuddenR, InstaD, SlowD, normal\}$. By doing so, we are able to detect the specific types of anomalies defined in [4] that are also portrayed in Fig. 3. Those anomalies appearing in time series of RSSI values are recorded as raw time-ordered values, thus forming the time series S. Fig. 3 presents the time-value representation of an ordinary link with solid black lines and its anomaly injected representation with dashed red lines.



(a) SuddenD anomaly (b) SuddenR anomaly

(c) InstaD anomaly (d) SlowD anomaly

Figure 3. Time-series representation of the normal links and of the four types of anomalies considered in [4]. These time series attempt to approximate long term link behaviours observed in a real-world deployment.

The mapping function Φ, representing the classifier, can be realized using various ML techniques. In this work, we propose a supervised DL model consisting of convolutional neural networks [16]. We compare the performance of the proposed model against ensembles of recent state of the art models for link layer anomaly detection [4] developed using supervised ML techniques; namely, i) Logistic Regression (LR) from Regression Analysis [17], ii) Random Forest (RForest) from tree ensemble class [18], and iii) Support Vector Machines (SVM) from kernel-method class [18].

DL models [16] are trained using ML algorithms based on artificial neural networks (ANN) and imitates the way how human brain works. The most common type of DL algorithms are convolutional neural networks (CNN). CNNs are able to successfully capture the temporal and spatial dependencies within an input data through the application of various filters. CNN uses these filters to apply significance to various aspects of the input data which enables them to differentiate between those aspects. In the remainder of the paper, we will refer to the resulting model as *deep learning classifier (DLC)*.

Logistic Regression [17] is a modified linear regression that is able to work on classification problems. In linear regression, the goal is to fit a line to data samples and minimize loss. Similarly, logistic regression aims for fitting sigmoid function with the goal to minimize loss at predicting any two classes. In the remainder of the paper, we will refer to the resulting model as *logistic classifier (LRC)*.

Random Forest [19] is an ensemble method that uses a number of decision tree classifiers followed by a voting mechanism to perform multi-class classification. The trees are learnt by randomly splitting a relatively large feature space into smaller subspaces. Each tree provides a class in which a specific data point falls into, where the class corresponds to the "vote" of that tree. The final outcome of the classifier then uses a mechanism, such as majority voting to provide the final result. In the remainder of the paper, we will refer to the resulting model as *random forest classifier (RFC)*.

Support Vector Machine [20] is a learning algorithm that belongs to the family of kernel methods. Generally speaking, SVMs attempt to learn a hyperplane that best splits a set of data into two classes. The shape of the hyperplane depends on the type of selected kernel function for the algorithm. For instance, when non-linear kernels are chosen, i.e., RBF kernel [21], then the learnt hyperplane is also non-linear, and thus the obtained model is better suited to approximate or discriminate non-linear random variables. In the remainder of the paper, we will refer to the resulting model as *support vector machine classifier (SVMC)*.

IV. Model Development

The proposed model is depicted in Figure 4 and consists of a seven layer deep network elaborated as follows. The time series S consisting of instances of length 300 is used to train the convolutional layers of the model. The first two convolutional layers consist of 64 filters of kernel size 3. The third layer has 32 filters with the same kernel size as the previous ones. The final two convolutional layers both use 16 filters of kernel size 7. The output data is then flattened and

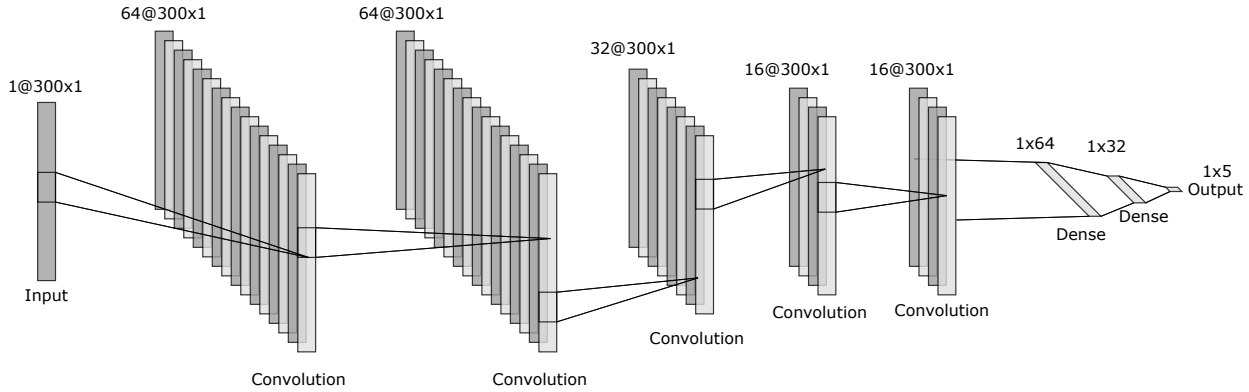


Figure 4. Proposed deep learning network model for five-class anomaly detection.

fed into a dense layer with 64 nodes followed by another dense layer with 32 nodes. Finally, the data is feed into the output layer of size 5 with sigmoid activation function, which enables the model to detect anomalies that could appear concurrently in the time-series. All other layers of the network for activation use the ReLU function and do convolution with stride 1.

We evaluate the proposed DL model against three ensemble classifiers: LRC, RFC, SVMC as described in Section III and depicted in Figure 5. Since logistic regression and SVM natively support only binary classification, we design the corresponding classifiers LRC and SVMC as a sequence of one versus all models. For instance, for the case of LRC, Classifier 1 from Figure 5 decides whether the anomaly is SuddenD. If the answer is positive, then it yields a prediction. Alternatively, it moves on to Classifier 2 which decides whether the anomaly is of type SuddenR. If the answer is positive, then it yields a prediction. It can alternatively move on to Classifier 3 which decides whether the anomaly is of type InstaD, and finally Classifier 4 is mapped to SlowD anomalies, whereas Classifier 5 is mapped to normal link behaviour. By doing so, we train five different binary classifiers, one for each anomaly type and one for non-anomalous (normal) link classification. Consequently, each classifier provides its own decision boundary, where together they represent a multi class classifier leveraged for final decision.

The RFC models are constructed using a multitude of decision trees at training time that together forms an ensemble. Figure 5 can be also used for a high level representation of the RFC model, where Classifier 1 represents the first decision tree estimator, Classifier 2 represents the second, and so on. In Figure 5, we present RFC with five decision trees, albeit in practice, the number of estimators varies and is expected to be much higher. Each of the decision trees on its own performs as a multi-class classifier and yields its prediction. The final decision is realized by performing the majority vote on the given predictions from all decision trees, which means that the final predicted class is the one selected by the most decision trees.

A. Dataset

To train all the models used in this paper, namely DLC, LRC, RFC and SVMC, we utilized the Rutgers WiFi dataset [22] as our real-world measurement dataset containing records from 29 nodes, each of which contains 300 measurements. Each link is measured with five different noise levels and we assume that each measurement is recorded as a different link. We use this real-testbed dataset and synthetically inject four anomalies that were defined by [4]. We only considered links without packet loss, which reduced our dataset from 4 060 to 2 123 ($\approx 52\%$). Similar to [4], we injected one anomaly type at a time over 33% of those links. Anomalies were injected according to guidelines in Table I, while other links are kept untouched.

Table I
SYNTHETIC ANOMALY INJECTION METHOD SIMILAR TO [4].

Type	Links	Affected	Appearance	Persistence
SuddenD	2 123	33% (700)	once, [200$^{\text{th}}$, 280$^{\text{th}}$]	for ∞
SuddenR			once, [25$^{\text{th}}$, 275$^{\text{th}}$]	for [5, 20]
InstaD			on $\approx$1% of a link	for 1 datapoint
SlowD			once, [1$^{\text{st}}$, 20$^{\text{th}}$]	for [150, 180]†

† $\text{RSSI}(x, \text{start}) \leftarrow \text{RSSI}(x) + \min(0, -\text{rand}(0.5, 1.5) \cdot (x - \text{start}))$

B. Model tuning

For each of the classifiers, Table II lists the parameters utilized in our experiments. For the DLC, the network architecture was designed with an iteration process of testing different combinations of number of layers, kernel sizes and strides. The DL model produced by this method, and presented in Table II, yielded the best combination of results and simplicity out of all tested combinations.

For the LRC, we leverage the LR implementation in SciKit-learn[1] that enables setting 12 different parameters. For most parameters, we selected standard values that have been proven to work on large number of cases by the ML community. For certain parameters that should be optimized such as the

[1]https://scikit-learn.org/stable/

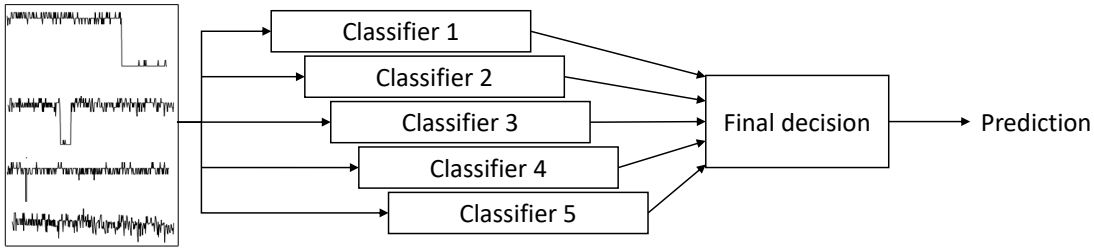


Figure 5. Design of the ensemble classifier for five-class anomaly detection using classical models: LRC, FRC, SVMC.

Table II
ML ALGORITHMS AND THEIR ASSOCIATED PARAMETERS.

Technique	Implementation	Parameters and their range
Deep Learning Classifier (DLC)	CNN Tensorflow	Input -> 2xCNN(num_filters=64, kernel=3) -> CNN(num_filters=32, kernel=3) -> -> CNN(num_filters=16, kernel=7) (all CNN with activation = ReLu and stride = 1) -> -> Dense(64) -> Dense(32) -> Output(5, activation = softmax)
Logistic Regression Classifier (LRC)	LogisticRegression from sklearn	penalty='l2', dual=False, tol=1e-4, C= (1e-3, 1e-2, 1e-1, 1., 10., 100.) fit_intercept=True, intercept_scaling=1, class_weight=None, solver='lbfgs', l1_ratio=None
Random Forest Classifier (RFC)	BaggingClassifier from sklearn	base_estimator=None, n_estimators=[10, 20, 30, 40, 50, 70, 100], max_samples=1.0, max_features=1.0, oob_score=False, intercept_scaling=1,
Support Vector Machine Classifier (SVMC)	SVC from sklearn	C=(1e-3, 1e-2, 1e-1, 1.0, 10., 100.), kernel=('linear', 'rbf'), gamma=('auto', 'scale'), tol=1e-3, decision_function_shape='ovr', break_ties=False,

regularization strength C in this case, we search for the best configuration by assessing a list of possible values that is presented in Table II and ultimately select the best performing regularization factor. The implementations for the other two classic ML algorithms also include over ten possible input parameters. For RFC, we vary the number of base estimators. For SVMC, we use the RBF kernel and vary the regularization factor C and kernel coefficient $gamma$. All ML algorithms are developed by searching for the optimal hyper-parameters through Grid search.

As some of the models are sensitive to scaling, we also scaled our dataset using min-max scaler to limit the values between 0 and 1, and used 10-times Shuffled split approach in order to ensure credible results.

V. RESULTS

The performance comparison of the proposed deep learning classifier, namely DLC, against the three selected classical ML algorithms is presented in Table III using the precision, recall and F1 metrics that are widely used for evaluating the performance of classifiers in the ML literature. In a nutshell, it can be seen from Table III that DLC performs equally with the other models for the SuddenD anomaly and outperforms by 4% and up to 86% for the other anomalies.

Analysing the SuddenD anomaly detection performance, we can see that neither model had any difficulties correctly detecting it. This is due to the fact that this anomaly is the most apparent out of all, since it always occurs at the end of the time series trace with a steep drop in the RSSI value, as depicted in Figure 3a.

Compared to the SuddenD anomaly, SuddenR is harder to detect since the occurrence of this anomaly can be anywhere within the trace as per Figure 3b and Table I. This can be observed in F1 score of classic ML algorithms which range from 0.57 for LRC to 0.87 for RF model. There is also a slight performance drop in DLC model, compared to SuddenD anomaly detection, albeit it still performs significantly better than other classical ML algorithms. DLC classifier with an F1 score of 0.97 is better by 0.10 from the F1 score of RFC, which is the best performing classical ML algorithm, and by up to 0.40 for the LRC, which is the worst performing classical ML algorithm, for the detection of SuddenR anomaly.

Detecting InstaD anomaly shows similar behaviours when compared to the SuddenR anomaly. However, owing of the instantaneous changes in samples, it is naturally difficult to accurately detect this type of anomaly. This can be readily observed in Table III, where all classical ML algorithms exhibit extremely bad performance in detecting this type of anomaly. The best out of three is the LRC model, which performs an F1 score of 0.13, followed by SVMC performing an F1 score of 0.11, while RFC exhibits the worst detection performance with an F1 score of 0.02. DLC algorithm shows superior performance than other three counterparts with F1 score of 0.88. Apparently, InstaD anomaly detection, owing to its rather arbitrary nature, reveals the superiority of DLC model over the classical ML models.

SlowD anomaly can be easily miss-classified by the models

Table III
PERFORMANCE RESULTS OF THE CLASSIFIERS.

Class	DLC			LRC			RFC			SVMC		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
SuddenD	1.00	1.00	**1.00**	1.00	1.00	**1.00**	1.00	1.00	**1.00**	1.00	1.00	**1.00**
SuddenR	0.98	0.96	**0.97**	0.49	0.67	0.57	0.99	0.77	0.87	0.76	0.68	0.72
InstaD	0.97	0.81	**0.88**	0.10	0.20	0.13	0.08	0.01	0.02	0.37	0.07	0.11
SlowD	0.62	0.85	**0.72**	0.54	0.81	0.65	1.00	0.21	0.34	0.37	0.68	0.47
No anomaly	0.97	0.95	**0.96**	0.93	0.77	0.84	0.85	0.99	0.92	0.88	0.89	0.89

since the slope of anomaly is usually not as distinguished as it is with other anomalies. This can be observed from the results in Table III, where performance results in the form of F1 score for all classic ML algorithms span between 0.34 for RFC and 0.65 for LRC classifier. Yet again, DLC model outperforms the classical ML algorithms with an F1 score of 0.72 which is higher by up to 0.38, albeit compared to the DLC classification of other anomaly types it performs with the lowest F1 score for SlowD anomaly.

In the last row of Table III, performance results of classifying links without anomaly are presented. Similar to the previous anomaly types, DLC model for normal link detection (F1 score of 0.96) outperforms the classical ML models by 0.04 for the RFC (F1 score of 0.92) and by up to 0.12 for the LRC (F1 score of 0.84).

VI. CONCLUSIONS

In this paper, we proposed a robust DL-based anomalous link classifier, namely DLC, for the classification of wireless link anomalies. We showed that the DLC model significantly improves the state of the art on link layer anomaly detection that currently considers four wireless link anomalies and a number of classical ML algorithms with various approaches to feature extraction. Unlike the state of the art, the proposed model is a single block that includes automated feature extraction using five convolution layers and automated model training using dense layers.

For SuddenD anomaly, DLC performs equally with the other traditional ML models, albeit it can outperform the traditional ML models by up to 86% detection performance. The considered ML models perform with an average accuracy of about 63%, while the average accuracy of the proposed DL model is around 90%, which leads to an improvement of 27% anomaly detection performance.

REFERENCES

[1] J. Davies and C. Fortuna, *The Internet of Things: From Data to Insight*. John Wiley & Sons, 2020.

[2] J. D. C. Silva, J. J. P. Rodrigues, K. Saleem, S. A. Kozlov, and R. A. Rabêlo, "M4DN. IoT-A Networks and Devices Management Platform for Internet of Things," *IEEE Access*, vol. 7, pp. 53 305–53 313, April 2019.

[3] A. Sheth, C. Doerr, D. Grunwald, R. Han, and D. Sicker, "Mojo: A distributed physical layer anomaly detection system for 802.11 wlans," in *Proceedings of the 4th international conference on Mobile systems, applications and services*. ACM, 2006, pp. 191–204.

[4] G. Cerar, H. Yetgin, B. Bertalanic, and C. Fortuna, "Learning to detect anomalous wireless links in iot networks," *IEEE Access*, vol. 8, pp. 212 130–212 155, 2020.

[5] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, vol. 41, no. 3, July 2009.

[6] O. Salem, A. Guerassimov, A. Mehaoua, A. Marcus, and B. Furht, "Anomaly detection in medical wireless sensor networks using svm and linear regression models," *International Journal of E-Health and Medical Communications (IJEHMC)*, vol. 5, no. 1, pp. 20–45, 2014.

[7] M. Wazid and A. K. Das, "An efficient hybrid anomaly detection scheme using k-means clustering for wireless sensor networks," *Wireless Personal Communications*, vol. 90, no. 4, pp. 1971–2000, 2016.

[8] B. Ahmad, W. Jian, Z. A. Ali, S. Tanvir, and M. S. A. Khan, "Hybrid anomaly detection by using clustering for wireless sensor network," *Wireless Personal Communications*, vol. 106, no. 4, pp. 1841–1853, 2019.

[9] A. Cook, G. Mısırlı, and Z. Fan, "Anomaly detection for IoT time-series data: A survey," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6481–6494, July 2020.

[10] M. Kulin, C. Fortuna, E. De Poorter, D. Deschrijver, and I. Moerman, "Data-driven design of intelligent wireless networks: An overview and tutorial," *Sensors*, vol. 16, no. 6, p. 790, 2016.

[11] V. L. Thing, "Ieee 802.11 network anomaly detection and attack classification: A deep learning approach," in *2017 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, 2017, pp. 1–6.

[12] Z. Chen, C. K. Yeo, B. S. Lee, and C. T. Lau, "Autoencoder-based network anomaly detection," in *Wireless Telecommunications Symposium (WTS)*, Phoenix, AZ, USA, April 2018.

[13] J. Ran, Y. Ji, and B. Tang, "A semi-supervised learning approach to ieee 802.11 network anomaly detection," in *2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)*. IEEE, 2019, pp. 1–5.

[14] T. Kim, S. C. Suh, H. Kim, J. Kim, and J. Kim, "An encoding technique for cnn-based network anomaly detection," in *2018 IEEE International Conference on Big Data (Big Data)*, 2018, pp. 2960–2965.

[15] T. Luo and S. G. Nagarajan, "Distributed anomaly detection using autoencoder neural networks in wsn for iot," in *2018 IEEE International Conference on Communications (ICC)*, 2018, pp. 1–6.

[16] Y. Bengio, "Learning deep architectures for ai," *Foundations*, vol. 2, pp. 1–55, 01 2009.

[17] R. Malouf, "A comparison of algorithms for maximum entropy parameter estimation," in *Proceedings of the 6th Conference on Natural Language Learning - Volume 20*, ser. COLING-02. USA: Association for Computational Linguistics, 2002, p. 1–7. [Online]. Available: https://doi.org/10.3115/1118853.1118871

[18] C. C. Aggarwal, "Outlier analysis," in *Data mining*. Springer, 2015, pp. 237–263.

[19] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[20] V. Vapnik, *The nature of statistical learning theory*. Springer science & business media, 2013.

[21] L. L. Z. Xiao-long, "Optimization of SVM with RBF Kernel [J]," *Computer Engineering and Applications*, vol. 29, 2006.

[22] S. K. Kaul, I. Seskar, and M. Gruteser, "CRAWDAD dataset rutgers/noise (v. 2007-04-20)," Downloaded from https://crawdad.org/rutgers/noise/20070420/RSSI, Apr. 2007, traceset: RSSI.