International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 07 Issue: 05 | May - 2023 Impact Factor: 8.176 ISSN: 2582-3930

NETWORKS ATTACK PREDICTION USING RANDOM FOREST ALGORITHM

[1]*S. Sharmila*, [2]**CH Puurna Hari,** [3] **V. Srinivas ,** [4] **BR Lokesh**
[1]Assistant Professor, Dhanalakshmi College of Engineering, Chennai
[2,3,4] Students, Dhanalakshmi College of Engineering, Chennai

---

ABSTRACT:

The purpose of this article is to compare conventional machine learning with modern deep learning approaches for the task of detecting anomalies in self-organizing networks. While deep learning has gained a significant move, especially in application scenarios where large volumes of data can be collected and processed conventional methods can nevertheless offer strong statistical alternatives, especially when the right methods are used learning representations. For example, support vector machines have previously shown superior potential in many binary classification applications and can be further used with various representations such as single-class learning and data augmentation In four different application scenarios, we are presenting evidence for the first time that conventional machine learning can achieve superior performance compared to prior deep learning approaches by an average of 15%, as demonstrated on a publicly available dataset that has been previously published. Our results further indicate that the computational speed has improved by almost two orders of magnitude and an order of magnitude reduction in trainable parameters, conventional machine learning provides robust alternative for self-organizing 5G networks, especially when execution and detection times are critical.

INTRODUCTION

As a next-generation telecommunication technology, 5G brings a new perspective and innovative solutions for the increased demand of people and autonomous devices [1]. Five areas are the main focus of this technology, which includes dense device structures, high-frequency carriers like millimetre wave (VMware), massive MIMO enabling multi-connectivity, smart devices, and massive machine-type communications. To meet these demands in such a dynamic digital world, next-generation cellular networks must be adaptable with predictive capabilities due to the ever-changing landscape of nested services that interact with each other [2]. Therefore, artificial intelligence (AI) has gained increased interest as a potential 5G technology to handle the dynamic environment in analyzing and contributing to network realization. Examples of intelligent applications for massive machine-type communication (mMTC) in 5G or massive MIMO in wireless sensor networks (WSNs) are provided by the Associate Editor responsible for overseeing the review of this manuscript. The primary objectives of such applications include improving IoT connectivity, extending battery life, and increasing spectral efficiency. These goals are achieved through the use of a channel-based decision fusion methodology, as described in references [3] and [4] approval for publication was by Bilal Alates. For example, Ericsson lost at least 100 million dollars because 61744 This work is licensed under a Creative Commons Attribution 4.0 License. AirHop's eSON technology is an example of autonomous mobile networks or self-organizing networks (SONs) with features such as self-scheduling, self-configuration, self-optimization, and self-healing that have proven to be effective in reducing network failures and improving performance without the need for human intervention[5]. This was demonstrated in previous mobile networks like 4G, and further information can be found in volume 10 of 2022, which is available at https://creativecommons.org/licenses/by/4.0/. However, current solutions on the market generally lack smart features, especially for managing cell outage

(COM) to recover autonomously [6]. If there is no traffic due to a specific network problem, this indicates an anomaly where one or more cells may be out, and it is important to detect the outage and restore service as soon as possible. grid outage [7] that could have been prevented with AI- powered SONs. To tackle the challenges posed by anomaly detection in mobile networks, the Touchless Automation Research Group was established by the European Telecommunications Standards Institute (ETSI). This group is dedicated to advancing machine learning and artificial intelligence techniques, particularly in the area of deep learning, to improve the detection of anomalies in mobile networks. This paper has four main contributions to the field, as summarized below. A thorough examination of a conventional machine learning technique for detecting anomalies in 5G self-organizing networks (5G-SON) is provided in this article. We also compare this method to a widely-used deep learning alternative, utilizing various learning representations such as one-class and binary learning. Our article asserts that we achieve state-of-the-art performance on a publicly available dataset [8] that examines various usage scenarios for detecting anomalies in 5G self-organizing networks (5G-SON). Our method outperforms the best-performing autoencoder-based deep setup, achieving an average improvement of 15%. Our study demonstrates, for the first time, that the performance of binary-mode anomaly detection can be further enhanced by using data augmentation methods, even when employing conventional algorithmic techniques such as support vector machines on a sufficiently large dataset. Moreover, we achieve a significant improvement in computational speed, nearly two orders of magnitude, and a reduction in trainable parameters by an order of magnitude using conventional machine learning. This approach provides a robust alternative for 5G self-organizing networks, especially when the execution and detection times are crucial. The remainder of this article is organized as follows: Section II provides a brief overview of prior research on anomaly detection in modern mobile and self-organizing networks. Section III describes the methods employed in this study. Section IV explains the experimental setup and includes details about hyperparameters, dataset characteristics, implementation, evaluation metrics, and information necessary for repeatability. The results and discussion are presented in Section V, and the conclusions are summarized in Section VI. Additionally, Table 1 lists the abbreviations used in this article for easy reference.

II. RELATED RESEARCH

Anomaly detection in communications has been an active area of research in the last decade. For example, abnormal activity in the wireless spectrum was investigated in [9]. Specifically, the authors used Power Spectral Density (PSD) information to identify and determine anomalies in the form of either unwanted signals present in the licensed band or the absence of a desired signal. The information obtained from the PSD was processed using a combination of adversarial auto-encoders, convolutional neural networks and recurrent neural networks with long-term short-term memory.

Abbreviations used in this study. In another example, [6] uses measurements and handover statistics (inHO) from neighbouring cells in a mobile communication network to detect abnormalities and outages. Monitoring in this manner provides a potential cell outage condition where the in HO information becomes zero. A novel online system for detecting anomalies in key performance indicators (KPIs) of mobile networks was proposed in reference [10]. The proposed system consists of a training and detection/tracking block. The system learns the most damaging anomalies in the training block while retrieving each last KPI and tracks its status until the end of the second block. Anomaly detection was thus set to prefer high-possible anomalies in the long run. Additionally, the system works against providing the minimum amount of anomalies by keeping the positive rate low in favor of network operators trying to resolve only true anomalies. In addition, the system can be extended to next-generation networks through automatic adaptation functions to the new network behavior profile.

Anomaly detection in mobile and self-organizing networks has been an area of active research in recent years. For instance, in [9], abnormal wireless spectrum activity was investigated using Power Spectral Density (PSD) information

to identify unwanted signals or the absence of desired signals. Meanwhile, in [10], an online mobile network anomaly detection system was proposed to identify anomalies in key performance indicators (KPIs).

Mobility-related anomalies were detected using unsupervised learning through Mobility Robustness Optimization (MRO) in [11], while in [12], reinforcement learning was used to reduce call drop rates. In [13], the authors utilized an autoencoder-based network anomaly detection method to detect nonlinearity changes in features, achieving better performance than conventional wireless communication methods for cyberattack detection. AI applications supporting 5G technologies can be implemented in both the physical and network layers, as discussed in [14], where a comprehensive overview of deep learning (DL) applications is presented.

In recent years, there has been an increasing interest in applying artificial intelligence (AI) to support 5G technologies, not only in the network layer but also in the physical layer. As an example, the authors in [14] provide a comprehensive overview of deep learning (DL) applications for the physical layer. One of the proposed applications is a modified version of the standard autoencoder, known as the "channel autoencoder." Unlike a typical autoencoder, the aim of the channel autoencoder is to find the most robust representations of the input signals to account for channel degradation by adding redundancies instead of removing them. The authors extend this concept to a multiple transmitter/receiver pairs adversarial network to increase capacity. Additionally, they introduce radio transformer networks (RTNs) that incorporate channel domain information and simplify symbol detection at the receiver's end by employing a neural network estimator to obtain the best detection parameters.

The augmented DL models on complex I/Q samples for modulation classification demonstrated that the DL models outperformed the classification methods based on expert features. Another study discussed in [15] introduces designing mobile traffic classifiers based on the DL utilization. A systematic framework of new DL-adopted Traffic Classification (TC) structures is introduced and analyzed. Rather than mobility, the study includes a wide allocation range to encrypt the TC.

Deep neural network methods have been widely used in studies that rely on multi-class applications, learning features such as key performance indicators, reference signal received power and quality (RSRP and RSRQ), handover statistics, and the number of connection drops and failures. However, there is ongoing discussion in the research community about the potential of other learning representations, such as one-class learning and deep unsupervised methods, to become strong alternatives for anomaly detection in SONs . Similarly, an ever-present debate investigates the comparative effectiveness of empirical deep learning models and conventional approaches such as feature engineering for a range of temporal applications [16]. A categorical analysis of literature discussed in this section can be found in Table 2, which displays the important characteristics and features such as the dataset type, topology of the network used, which learning methodology is applied for which application, etc.

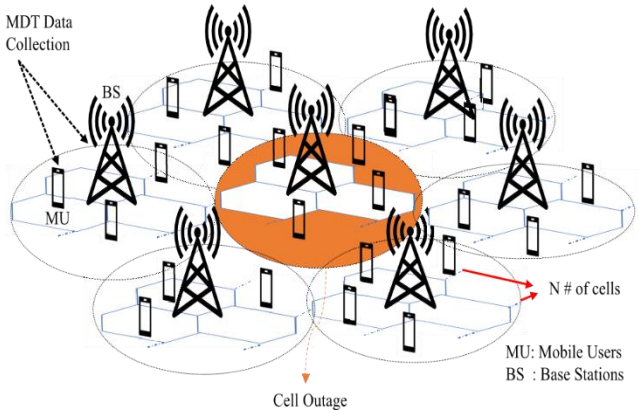
III. METHODS

A. ANOMALY DETECTION PROBLEM DEFINITIONS

In this study, we focus on anomaly detection scenarios in which 105 mobile users are evenly distributed among 7 base stations (BS). Each BS consists of 3 regularly spaced cells, resulting in a total of 21 cells, as shown in Figure 1.The objective is to detect anomalies where a BS fails to communicate with a user, resulting in below-par key performance indicator (KPI) margins. Data is collected using a Minimization of Drive Test (MDT) report, with KPIs such as Reference Signal Received Power (RSRP) and Reference Signal Received Quality (RSRQ) collected from each of the 7 BS at a 5kHz sampling rate for a total of 50 seconds. Each MDT report is assigned a class label from the set {0, 1, 2, 3}, where labels 0, 1, and 2 indicate normal network status with varying levels of transmitter power, while label 3

indicates an anomaly where the BTS has failed to communicate with the user. The study includes three different scenarios with different numbers of users and cells to explore the AD application for a wider range of networks and users.

Figure 1:



To perform the AD task, we use the support vector machine (SVM), a popular machine learning algorithm that is often used in binary classification and unsupervised feature learning [17]. Regenerate response An SVM model represents and separates various classes by building a set of hyperplanes in a multi-dimensional space. Separation of the classes from each other is performed iteratively by the SVM algorithm through finding the optimal hyperplane (the decision region between a group of objects from different classes). In this context, the optimal hyperplane maximizes the distance gap (margin) between the two lines closest to the data points from different classes based on the support vectors (data sample points where the classification error is minimum)

B. SUPPORT VECTOR MACHINE

The support vector machine (SVM) [17] is a popular machine learning algorithm that plays important roles in binary classification and unsupervised feature learning. An SVM model represents and separates various classes by building a set of hyperplanes in a multi-dimensional space. Separation of the classes from each other is performed iteratively by the SVM algorithm through finding the optimal hyperplane (the decision region between a group of objects from different classes). In this context, the optimal hyperplane maximizes the distance gap (margin) between the two lines closest to the data points from different classes based on the support vectors (data sample points where the classification error is minimum).

TABLE 2.

Paper Name and Reference	Features	Dataset	Topology	Learning	Application	Data Preprocessing
Asghar et. al. [8]	KPIs (RSRP, RSRQ)	User-Cell Based Four Use Case Datasets	AE (Autoencoder), k-NN (k-Nearest Neighbor)	Unsupervised	Cell Outage Detection	Yes
Rajendran et. al. [9]	PSD (Power Spectral Density)	One Synthetic Spectrum Dataset, two real wireless datasets (SDR, PSD Sensor Data)	VAE (Variational Autoencoder), AAE (Adverserial Autoencoder), LSTM (Long-Short Term Memory)	Unsupervised/ Semi-Supervised	SAIFE (Spectrum Anomaly Detector with Interpretable Features)	Yes
Bandera et. al. [6]	InHO-KPIs (Incoming Handover Statistics Key Performance Indicators)	Live LTE Network Simulator	LTE Cellular Layout	Unsupervised	COD (Cell Outage Detection)	No
Burgueño et. al. [10]	KPIs (Key Performence Indicators)	Real LTE Advanced Mobile Network	DBSCAN (Density Based Spatial Clustering)	Supervised	Online Anomaly Detection	Yes
Moysen et. al. [11]	PM (Performance Management), CM (Configuration Management)/IM (Inventory Management)	Commercial LTE Networks Datasets	PCA (Principal Component Analysis), HDBSCAN (Hierarchical Density Based Spatial Clustering of Applications with Noise)	Unsupervised	Identifying Cells Experiencing Performance Degradation	Yes
Chen et. al. [13]	TCP (Transmission Control Protocol), UDP (User Datagram Protocol), ICMP (Internet Control Message Protocol) Traffics	NSL-KDD	PCA (Principal Component Analysis), AE (Autoencoder), CAE (Convolutional Autoencoder), k-NN (k-Nearest Neighbor), SVM (Support Vector Machine), TANN (Triangle Area Based Nearest Neighbors)	Unsupervised/ Supervised	Network Anomaly Detection	Yes
Kanjilal et. al. [16]	ADL (Activities of Daily Living)	UniMIB-SHAR	SVM (Support Vector Machine), ANN (Artificial Neural Network), LSTM (Long-Short-Term Memory), CNN (Convolutional Neural Network)	Unsupervised	Human Activity Recognition	Yes

O"Shea et. al. [14]	Real Signal Messages, I/Q Samples	RML2016.10b	AE (Autoencoder), RTN (Radio Transformer Networks), CNN (Convolutional Neural Network)	Unsupervised	Deep Learning Applications on Physical Layer-Survey	Yes
Aceto et. al [15]	HTTP Traffic, User-Website Fingerprint, Flow-based	Multi-Class/Binary Dataset, FB/FBM (Face-book/Facebook Messenger)	SAE (Sparse Autoencoder, CNN (Convolutional Neural Network), LSTM (Long-Short Term Memory)	Unsupervised/ Supervised	Real Human Users' Activity	Yes

Summary of related research.

We implement SVM in two different ways

1) ONE CLASS ANOMALY DETECTION WITH SVM

SVM with one-class classification is a well-known method for detecting anomalies in an unsupervised setting [18]–[21]. Typically, SVM is utilized for binary classification tasks. However, in the context of one-class anomaly detection, the algorithm is trained solely on observations from the majority class in order to learn what constitutes "normal" samples. When new data is presented to the SVM algorithm, the decision probability for "anomalous" samples is lower in comparison to observations from the majority class under ideal conditions.

2) BINARY ANOMALY DETECTION USING SVM

In the case of binary anomaly detection using SVM, a balanced dataset is necessary to achieve the most accurate results. When the dataset is imbalanced, the SVM classifier tends to favor the majority class, resulting in poor performance for the minority class and reduced generalization. Several techniques have been proposed to overcome data imbalance, with one of the most popular being the synthetic minority resampling technique (SMOTE) [22]. SMOTE aims to balance the class distribution by randomly generating "synthetic" patterns based on features rather than raw data, thereby increasing the minority class. The resampling process for the minority class (C) starts by selecting each sample (X) and interpolating synthetic instances along the lines connecting minority instances and their k-nearest neighbours x. The value of k is randomly chosen according to the hyperparametric resampling rate N based on the number of minority samples. First, a distance-based method such as "Euclidean Distance" is used to calculate the distance between a feature vector and its neighbours . Second, the distance is multiplied by a random number between (0,1] and added to the previous feature sample. Consequently, new and synthetic features are generated along

the lines between the two original samples, mathematically expressed as X0 = X + rand(0,1)∗|X − Xk (1), where X0 represents the new set of synthetic samples and Xk is the set of randomly selected k-nearest neighbour samples.

C. SMOTE ALGORITHM

Autoencoder, a specific topology of an artificial neural network, is another method for anomaly detection. Autoencoder has the same input and output layer, where training is performed by simultaneously presenting the same input data to both layers. The general structure of an autoencoder includes a visible input layer x, a series of hidden layers h, and an output layer constructed from it with a series of nonlinear activation functions f applied in different layers.

DATASET STRUCTURE

Time	UserID	LocationX	LocationY	RSRP1	RSRQ1	...	RSRP10	RSRQ10	LABEL
0.4	1	2003.53	648.252	97.8906	-5.9521	...	-116.126	-24.1876	0
0.4	2	1434.95	283.589	92.1017	3.84048	...	-113.364	-25.1028	0
0.4	3	1982.94	712.61	-100.5	7.79236	...	-117.207	-24.4995	0
0.4	4	1721.1	872.081	80.8191	4.40083	...	-115.727	-39.3085	0
:	:	:	:	:	:		:	:	:
0.6	1	2006.21	646.261	7.8847	5.95526	...	-115.727	-39.3085	1
:	:	:	:	:	:		:	:	:
22.4	65	1162.72	369.994	94.3135	4.38688	...	-112.918	-22.9913	0

FIGURE 2. Dataset structure for the first use case.

In the first use case, the dataset follows a specific structure. During the training process, the auto-encoder takes input data x that belongs to the real-valued space Ry. The auto-encoder then maps the input data to hidden layers with smaller dimensions compared to the input data. This results in a compressed representation of the original data, which is equivalent to the size of the code or latent layer represented by H in the real-valued space Rh. The encoder step is responsible for creating this compressed information, which is then used by the decoder to map it back to the output layer through a process called reconstruction. Mathematically, these two steps can be formulated as follows:

$$H \equiv fWH\,(x) = f\,(WHx + bH) \quad (2)$$

$$z \equiv gWz(x) = g(WzH + bZ\,) \quad (3)$$

In the first use case, the dataset follows a specific structure. During the training process, the auto-encoder takes input data x that belongs to the real-valued space Ry. The auto-encoder then maps the input data to hidden layers with smaller dimensions compared to the input data. This results in a compressed representation of the original data, which is equivalent to the size of the code or latent layer represented by H in the real-valued space Rh. The encoder step is responsible for creating this compressed information, which is then used by the decoder to map it back to the output layer through a process called reconstruction. Mathematically, these two steps can be formulated as follows:

$$H \equiv fWH\,(x) = f\,(WHx + bH) \quad (2)\ z \equiv gWz(x) = g(WzH + bZ\,)$$

Given N samples of input data, the following loss function is used to determine the parameters "WH,WZ,bH, and bZ" using a backpropagation algorithm commonly used in feedforward neural networks:

As previously mentioned, the main objective of autoencoding is to extract meaningful information at the coding layer by minimizing the reconstruction error. To achieve this, a backpropagation algorithm commonly used in feedforward neural networks is employed to determine the parameters "WH, WZ, bH, and bZ" using the following loss function:

L(WH,WZ,bH,bZ) = 1/N ∑i=1N ||x_i - g(f(x_i;WH,bH);WZ,bZ)||^2

where N is the number of input samples, x_i is the i-th input sample, f(.) and g(.) are the encoding and decoding activation functions, such as a sigmoid function or a rectified linear unit, and ||.||^2 denotes the L2 norm. The parameters are updated iteratively using the gradient of the loss function with respect to the weights and biases.

N

X 2

LAE||xk − zk|| (4)

N

k=1

In the context of this application, the autoencoder is used for anomaly detection by training the network with normal observations and, after training, compares the reconstruction error of normal and anomalous samples with a threshold for detection. Normal samples are expected to provide smaller reconstruction errors compared to anomalous samples, which can be easily characterized using receiver operating characteristics (ROC In the research paper [8], an autoencoder model was used with a specific structure. The input vector size was 20, which corresponds to 10 RSRP and 10 RSRQ measurements. This was followed by four hidden layers consisting of 12, 6, 6, and 12 neurons, respectively. The overall topology of the model was 20-12-6-6-12-20.

IV. EXPERIMENTAL SETUPS

A. DATASETS

For our study, we utilize a dataset that was generated by a SON simulator, as described in a previous publication [8]. This dataset has been made publicly available. We present ROC curves for the four datasets, which enable us to compare modern and conventional machine learning approaches with the results reported in the original paper.

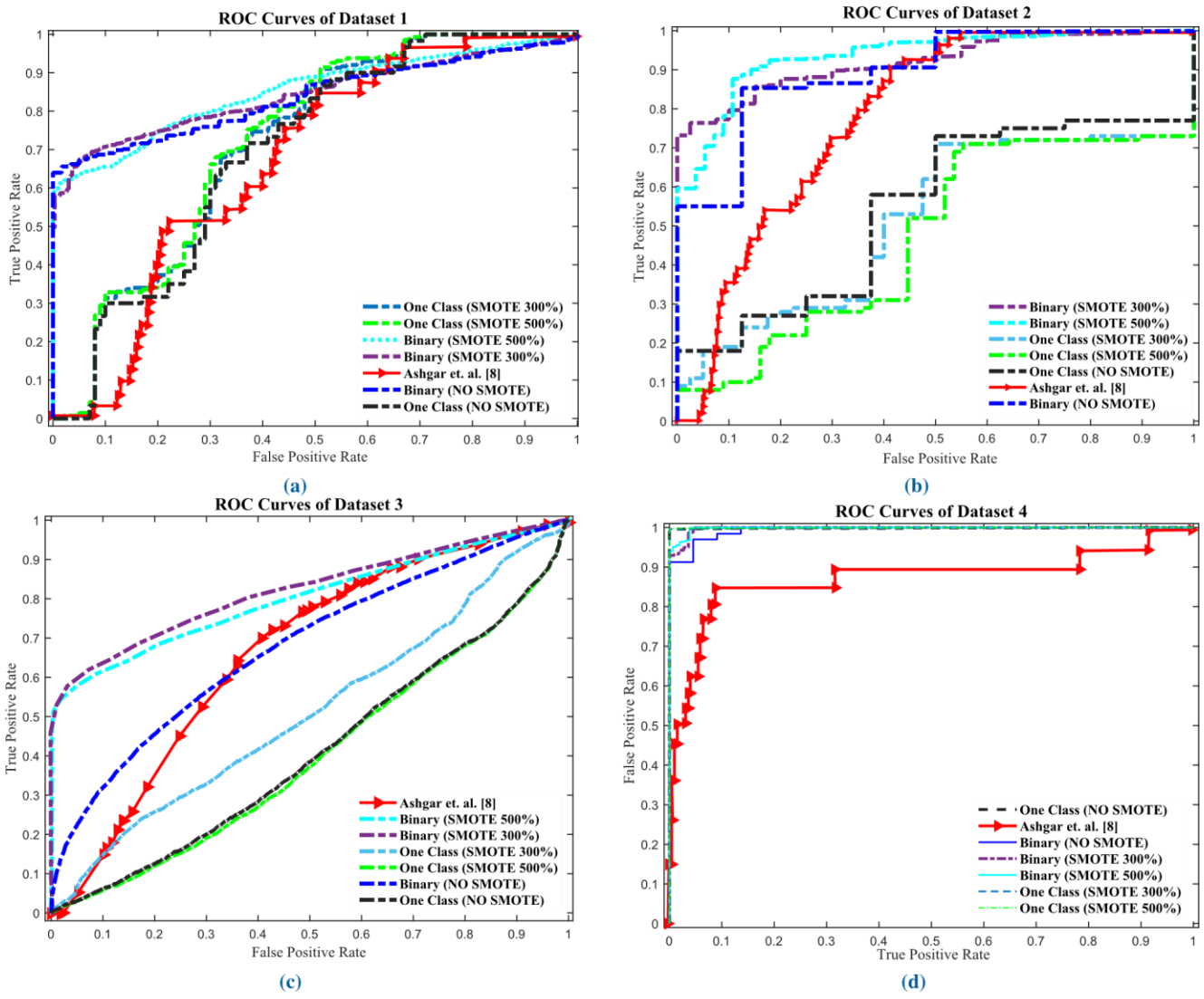


FIGURE 3.
The ROC curves for the four datasets for the comparative analysis of modern and conventional machine learning with the results reported in the source paper.

For further research [23]. The dataset contains four different application scenarios where data is collected periodically (with a sampling frequency of 5 kHz) from the minimization of the drive test report [24], which contains mobile user information regarding user activities recorded in closed areas around base stations. (cells) in certain measurement periods. There are four different datasets with different numbers of users and use cases. Giant. 2 shows the basic structure of the data set for the first use case (data set 1), which consists of the time of measurement, the unique ID assigned to each user, the coordinates of the users' location in two dimensions at the time of measurement, the received reference signal power (RSRP), the quality of the received reference signal (RSRQ) and a label indicating whether the associated item (ie the collection of measurements associated with this user) is anomalous (1) or not (0). The LTE network KPIs RSRP and RSRQ are crucial measures of signal strength and quality. They are used to determine whether the collected information is anomalous. The first dataset used in this work is taken from a SON simulator introduced

in [8], and it is publicly available. It consists of 11,674 observations with only 60 anomalous measurement samples, which accounts for a ratio of 1 in 200. The dataset includes 25 features, such as feature vector, user ID, location, and class labels. The second dataset has a similar number of features and measurements but a much lower anomaly rate, with 8382 observations and only eight anomalous samples, accounting for a ratio of approximately 1 in 1000.

Datasets 3 and 4 have 114 features each and a longer recording time (80 s), resulting in a significantly larger observation base of 42,000. These datasets have different anomaly rates, with dataset 3 having a much larger number of anomalous measurements (9635) compared to dataset 4, where only items below the -120 dB RSRP measurement threshold were flagged as anomalous, resulting in only 22 anomalous measurements.

B. HYPERPARAMETERS

In this study, the resampling rate N and the number of nearest neighbors k are the most significant hyperparameters. These hyperparameters are particularly important for the SMOTE algorithm when applying resampling for binary classification. Two different sets of hyperparameters were tested for SMOTE, namely N=300 and k=4, and N=500 and k=5.

C. IMPLEMENTATION

For this study, we used MATLAB 2020b to implement one-class and binary SVM models on all datasets. To address the issue of unbalanced datasets, we employed the SMOTE algorithm with a binary SVM model. Preprocessing was performed on the datasets, where time, user ID, and location features were excluded from the training process for fairness. About 10% of normal and abnormal samples were reserved for testing purposes. Normalization was also applied to both training and test samples within the range of (0,1).

In the case of the one-class SVM model, only normal samples were used for training with Gaussian RBF kernels. After training, we obtained SVM probability outputs for both normal and anomalous test samples to generate ROC curves and calculate area under the curve (AUC) scores as a performance metric.

For the binary SVM model, we followed the same process as the one-class SVM model, but with the additional step of using SMOTE to resample anomaly samples for generating balanced datasets before training and testing the algorithm.

D. EVALUATION METRICS

To evaluate the performance of the models in this study, we utilized ROC curves and AUC scores as metrics. The ROC curve illustrates the model's ability to identify the positive class accurately, with TPR on the y-axis and FPR on the x-axis. TPR represents the proportion of correctly classified positive outputs, while FPR represents the proportion of incorrectly classified positive outputs, as shown below:

TP FP

TPR = FPR = (5)

TP + FP FP + TN

AUC, on the other hand, provides a summary number as an indication of how strong the model is in distinguishing between classes with a mathematical expression as below:

$$\frac{TP + TN}{TP + FP + FN + TN} \quad \text{AUC} = \quad (6)$$

V. RESULTS

Figure 3 displays the ROC curves for all four datasets, where the red curve represents the performance reported in [8] using a deep autoencoder. To compare the results, we applied various SVM models on the same datasets. Additionally, Figure 4 shows the area under the curve (AUC) scores across the different datasets. For comparison, different SVM implementations

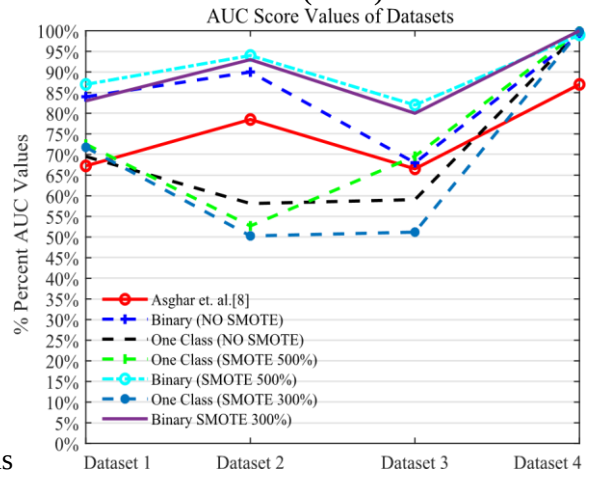


FIGURE 4. Area under curve (AUC) scores across different datasets.

TABLE 3:
The AUC scores for both one-class and binary SVM classifiers, with and without SMOTE augmentation, were graphed in various colours to depict their performance across all datasets.

			Dataset 1	Dataset 2	Dataset 3	Dataset 4
SVM	Binary (No Smote)		84%	90%	68%	99%
	One Class (No Smote)		69.52	58.125%	59.08%	99.84%
Smote Algorithm	N=500% K=5	One Class	72.52%	52.7%	69.6%	99.91%
		Binary	**87%**	**94%**	**82**	99%
	N 300 % K=4	One Class	71.76 %	50.28 %	51.20 %	).86 %
		Binary	83%	93%	80%	**100**%
Ashgar et al. AUC [8]			67.25%	78.49%	66.54%	87%

The outcomes were generally consistent, with the exception of dataset 4, where all SVM combinations outperformed the deep autoencoder at all levels of true positive rate and false positive rate. In datasets 1-3, the deep autoencoder yielded better results than the SVM implementations without augmentation. However, when SMOTE was applied, both one-class and binary SVM models demonstrated markedly improved performance compared to the original research paper, sometimes to a significant degree.

In Figure 4, an overview of AUC scores across different datasets and algorithms is provided, with binary and one-class SVMs (depicted by the top three curves) showing superior performance over the deep autoencoder approach (indicated by the red curve). Table 3 furnishes the AUC values with a 5% significance level. The most effective combination of

SVM modality and SMOTE outperformed the deep autoencoder by 19.75%, 15.5%, 15.5%, and 13% for datasets 1, 2, 3, and 4, respectively, resulting in an average improvement of over 15% across all scenarios. It is noteworthy that traditional machine learning utilizing SVMs still outperforms the previous state-of-the-art methods reported in the literature, even without artificial augmentation of the dataset. However, the disparity in performance was less noticeable.

To assess whether SMOTE could offer a similar performance boost to the autoencoder setup, we focused on the first two datasets with significant class imbalance, as reported in the Asghar et. al. [8] study. Our findings indicate that the average detection accuracy using SMOTE with hyperparameters N=300, k=4 and N=500, k=5 was 64.03% and 57%, respectively, for dataset 1 and 68.87% and 51.64%, respectively, for dataset 2. These results reveal insignificant improvements over the baseline performance or, in some cases, even worse results compared to not utilizing SMOTE on the dataset. There could be various explanations for this outcome, but the most plausible explanation is that the operational structure of the latent space in an autoencoder closely resembles the way in which SMOTE generates augmented samples. In other words, the benefit of using SMOTE is diminished in the latent coding layer of the autoencoder itself.

The lack of significant or even negative impact of SMOTE on the performance of one-class SVM topology is an important issue to consider. SMOTE generates samples by leveraging the nearest neighbour similarities of intraclass samples and differences of interclass samples. This approach works best when training a binary classifier where one class may be underrepresented compared to the other class, as demonstrated by the performance boost observed in binary SVM training. However, in one-class learning representation, only the majority class is utilized in training. Therefore, SMOTE can only have an indirect effect on the number and quality of samples generated for the normal class and does not directly contribute to performance improvement. The decline in performance, in some instances, can be attributed to the quality of the anomaly class samples being generated, which may not compensate for the additional information that cannot be utilized in the training process.

A. COMPUTATIONAL COMPLEXITY ANALYSIS

Computational complexity analysis is an important step in identifying the strengths and weaknesses of conventional algorithm such as one-class and binary SVMs compared to more modern approaches such as autoencoders used in anomaly detection. In this paper, we focused on both the raw computation times specific to the test phase for each of the four dataset scenarios, as well as the number of trainable parameters for both algorithms. We also compared how SMOTE affected complexity. [3] All measurements are performed using the latest version of MATLAB at the time of writing (2021b) using standard timing scripts. On the first dataset, for the one-class SVM Algorithm, the testing time took 90ms without SMOTE and 140ms with SMOTE, while the autoencoder took 9000ms to run, a nearly 100-fold increase. The additional complexity of SMOTE is more than outweighed by its significant contribution to accuracy. We observed similar computation times for the rest of the usage scenarios (i.e., for the second dataset, single class SVM test times were 110–130ms (SMOTE) compared to 7120 ms for AE), where there were orders of magnitude improvements in test times. We performed the computational speed analysis using a one-class SVM due to the fact that the implementation was done using a native script, while for the binary SVM a GUI toolbox was used with better visualization capabilities that affect the speed. However, there should be no difference in test times since the SVM topologies are identical, the only difference being the way the data is represented by the algorithms. By analysing the number of trainable parameters, we can gauge the computational complexity of an algorithm. In our study, we observed that the average number of support vectors for SVM models was 23, whereas the autoencoder required the training of 660 weight and 76 bias parameters, which sums up to 736 parameters. Based on these measurements of computational time and complexity, we conclude that SVM-based approaches, even with the use of SMOTE, exhibit lower

complexity, have competitive AUC scores, and are more suitable for time-sensitive scenarios like anomaly detection and outage recovery compared to the autoencoder.

VI. CONCLUSION

The aim of our study was to compare the performance of conventional machine learning with deep learning for anomaly detection in SON. Although deep learning has been widely used in this field, traditional methods can still provide effective statistical alternatives for accurate learning representations. For our research, we examined SVMs with both one-class and binary learning scenarios using a publicly available dataset. Our findings indicate that while deep learning is highly competitive, standard SVMs with RBF kernels can be trained to outperform the deep autoencoder approach. Furthermore, we discovered that both single-class and binary classifications can significantly benefit from synthetic dataset augmentation using SMOTE, with an average improvement of up to 15% in detection accuracy across four different application scenarios. To expand the scope of this study, future research could explore the impacts of incorporating dataset augmentation into different types of machine learning algorithms, including those that use statistical methods such as variational autoencoders. Moreover, the outcomes of this research have broader implications and can be employed in other domains and applications beyond anomaly or outage detection .In particular, increased attention has been paid to modulation detection in next-generation mobile wireless networks, where fast, robust and lightweight machine learning models could enable time-critical applications in signal classification and modulation detection. Speed improvements can be realized both at the algorithm level and in the data pre-processing stages using techniques such as principal component analysis to identify the most important features for classification and detection. Finally, statistical learning algorithms such as Gaussian Process Regression, which have gained immense popularity as alternatives to deep learning, can be applied to various scenarios, especially when data is not present in large enough volumes to properly train DL models with many parameters.

VII. REFERENCE

[1] F. Boccardi, R.W. Heath, A. Lozano, T.L. Marzetta, and P. Popovski, “Five Disruptive Technology Directions for 5G,” IEEE Commun. Mag., Vol. 52, no. 2, pp. 74–80, February 2014.
[2] J. G. Andrews, S. Buzzi, W. Choi, S. V. Hanly, A. Lozano, A. C. Soong, and J. C. Zhang, ‘‘What will 5G be?’’ IEEE J. Sel. Areas Commun., Vol. 32, No. 6, pp. 1065–1082, June 2014.
[3] A.-S. Bana, E. de Carvalho, B. Soret, T. Abrão, J. C. Marinello, E. G. Larsson, and P. Popovski, ‘‘Massive MIMO for Internet of Things (IoT) connectivity,’’ Phys. Commun., Vol. 37, December 2019, Article No. 100859.
[4] D. Ciuonzo, P. S. Rossi, and S. Dey, “Massive MIMO channel-aware decision fusion,” IEEE Trans. Signal Process., Vol. 63, no. 3, pp. 604–619, February 2015.
[5] Airhop. Accessed: 18 June 2021. [Online]. Available: http://www. airhopcomm.com
[6] I. de-la-Bandera, R. Barco, P. Muñoz, and I. Serrano, “Cell outage detection based on handover statistics,” IEEE Commun. Lett., Vol. 19, No. 7, pp. 1189–1192, July 2015.
[7] Erikson. Network outage. Accessed: 18 June 2021. [Online]. Available: http://telecoms.com/494091/
[8] M. Z. Asghar, M. Abbas, K. Zeeshan, P. Kotilainen and T. Hämäläinen, ‘‘Assessment of deep learning Method for self-organizing 5G networks’’ Appl. Sci., vol. 9, No. 15, p. 2975, July 2019.
[9] S. Rajendran, W. Meert, V. Lenders, and S. Pollin, “Unsupervised wireless spectrum anomaly detection with interpretable features,” IEEE Trans. Cognit. Commun. Netw., Vol. 5, No. 3, pp. 637–647, September 2019.
[10] J. Burgueño, I. de-la-Bandera, J. Mendoza, D. Palacios, C. Morillas, and R. Barco, ‘‘An Online Anomaly Detection System for Mobile Networks,’’ Sensors, vol. 20, No. 24, p. 7232, December 2020.

[11] J. Moysen, F. Ahmed, M. Garcia-Lozano and J. Niemela, ‘‘Unsupervised learning for detection of mobility related anomalies in commercial LTE networks’’, Proc. Eur. Conf. Network Commun. (EuCNC), June 2020, pp. 111–115.
[12] W. Qin, Y. Teng, M. Song, Y. Zhang, and X. Wang, ‘‘AQ-learning approach for mobility robustness optimization in lte-son,’’ in Proc. 15th IEEE Int. Conf. Commun. Technol., Nov. 2013, pp. 818–822.
[13] Z. Chen, C. K. Yeo, B. S. Lee, and C. T. Lau, ‘‘Autoencoder-based network anomaly detection,’’ in Proc. Wireless Telecommun. Symp. (WTS), April 2018, pp. 1–5.