UNIVERSITY OF OULU

FACULTY OF INFORMATION TECHNOLOGY AND ELECTRICAL ENGINEERING

**Moustafa Khairi**

# MACHINE LEARNING BASED ANOMALY DETECTION IN RELEASE TESTING OF 5G MOBILE NETWORKS

Master's Thesis
Degree Programme in Computer Science and Engineering
April 2022

**Khairi M. (2022) Machine Learning Based Anomaly Detection in Release Testing of 5G Mobile Networks.** University of Oulu, Degree Programme in Computer Science and Engineering, 65 p.

## ABSTRACT

**The need of high-quality phone and internet connections, high-speed streaming ability and reliable traffic with no interruptions has increased because of the advancements the wireless communication world witnessed since the start of 5G (fifth generation) networks. The amount of data generated, not just every day but also, every second made most of the traditional approaches or statistical methods used previously for data manipulation and modeling inefficient and unscalable. Machine learning (ML) and especially, the deep learning (DL)-based models achieve the state-of-art results because of their ability to recognize complex patterns that even human experts are not able to recognize. Machine learning-based anomaly detection is one of the current hot topics in both research and industry because of its practical applications in almost all domains. Anomaly detection is mainly used for two purposes. The first purpose is to understand why this anomalous behavior happens and as a result, try to prevent it from happening by solving the root cause of the problem. The other purpose is to, as well, understand why this anomalous behavior happens and try to be ready for dealing with this behavior as it would be predictable behavior in that case, such as the increased traffic through the weekends or some specific hours of the day.**

**In this work, we apply anomaly detection on a univariate time series target, the block error rate (BLER). We experiment with different statistical approaches, classic supervised machine learning models, unsupervised machine learning models, and deep learning models and benchmark the final results. The main goal is to select the best model that achieves the balance of the best performance and less resources and apply it in a multivariate time series context where we are able to test the relationship between the different time series features and their influence on each other. Through the final phase, the model selected will be used, integrated, and deployed as part of an automatic system that detects and flags anomalies in real-time. The simple proposed deep learning model outperforms the other models in terms of the accuracy related metrics. We also emphasize the acceptable performance of the statistical approach that enters the competition of the best model due to its low training time and required computational resources.**

**Keywords: anomaly detection, time series, times eries anomaly detection, time series forecasting, machine learning, deep learning, supervised learning, unsupervised learning, artificial intelligence, 5G, BLER, wireless communications.**

# TABLE OF CONTENTS

# FOREWORD

The output of this work is to be integrated as part of an automatic testing system at Nokia Solutions and Networks Oy in Oulu. I am grateful for having the opportunity to do my thesis in such an interesting topic in the telecommunications domain and I want to thank my Nokia supervisor, Jukka Lämsä, for his continuous support and help during the thesis work. I also want to thank my line manager, Tero Nieminen and my collegaues at Nokia for being a great part of this experience.

Finally, I want to thank my University supervisors, Pekka Siirtola and Jaakko Suutala for their guidance, help and time all the time.

Oulu, April 6th, 2022

Moustafa Khairi

# LIST OF ABBREVIATIONS AND SYMBOLS

| | |
|---|---|
| 3GPP | Third Generation Partnership Project |
| 5G | Fifth-Generation |
| ACK | Acknowledgement |
| AI | Artificial Intelligence |
| AIC | Akaike's Information Criteria |
| ANN | Artificial Neural Networks |
| AR | Autoregressive |
| ARMA | Autoregressive Moving Average |
| ARIMA | Autoregressive Indexed Moving Average |
| BIC | Bayesian Information Criteria |
| BLER | Block Error Rate |
| CV | Computer Vision |
| CNN | Convolution Neural Networks |
| CQI | Channel Quality Indicator |
| DBSC | Density-Based Spatial Clustering |
| DES | Double Exponential Smoothing |
| DL | Deep Learning |
| ES | Exponential Smoothing |
| FIR | Finite Impulse Response |
| FN | False Negatives |
| FP | False Positives |
| GRU | Gated Recurrent Unit |
| IoT | Internet of Things |
| K-NN | K-nearst Neighbor |
| KPI | Key Performance Indicator |
| LOF | Local Outlier Factor |
| LSTM | Long Short Term Memory |
| MA | Moving Average |
| MCS | Modulation Coding Scheme |
| ML | Machine Learning |
| MLP | Multi-layer Perceptron |
| NACK | Negative Acknowledgement |
| NLP | Natural Language Processing |
| NN | Neural Networks |
| OC-SVM | One-Class Support Vector Machine |
| OLS | Ordinary Least Squares |
| PCA | Principal Component Analysis |
| PCI | Predicitoon Confidence Interval |
| QQ | Quantile-Quantile |
| UE | User Equipment |
| RAN | Radio Access Network |
| ReLU | Rectified Linear Unit |
| RL | Reinforcement Learning |
| RMSE | Root Mean Squared Error |

| | |
|---|---|
| RNN | Recurrent Neural Network |
| RSSI | Received Signal Strength Indication |
| SEO | Search Engine Optimization |
| SFN | System Frame Number |
| SINR | Signal to Interference Noise Ratio |
| SNR | Signal to Noise Ratio |
| STSC | Subsequence time-series Clustering |
| TES | Triple Exponential Smoothing |
| VR | Virtual Reality |

| | |
|---|---|
| $X_t$ | target value for time step t |
| $e_i$ | anomaly score |
| $y_i$ | true label |
| $f_i$ | predicted label |
| $e_i$ | anomaly score |
| $e_i$ | anomaly score |

| | |
|---|---|
| $\beta$ | shape factor |
| $\epsilon_\mathrm{k}$ | residual coefficient |
| $\Phi_\mathrm{k}$ | state transition matrix at instant k |
| $\theta$ | parameter vector |
| $\mu$ | mean |
| $\sigma^2$ | variance |

# 1. INTRODUCTION

The introduction of the 5G technology for broadband cellular networks has succeeded in establishing many achievements for different applications in different domains, such as the internet of things (IoT) [1], drones, virtual reality (VR) [2], and autonomous robotics. On the other hand, all these advancements and applications introduce complex challenges in terms of the latency, reliability, connectivity and more. Having a strong and reliable receiver system is essential for maximizing the data throughput and having high-quality connections. Being able to stream high-quality videos, downloading gigabytes of data in a couple of minutes, browsing tons of social media channels and having real-time updates with no interruptions are not a luxury anymore but a need for the mobile users. The quality of both the transmitter and receiver in both the cases of uplink and downlink cases affects the mobile devices' users' experience.

In many critical applications in different domains, it is crucial to meet the strict requirements of latency and reliability. For example, the modern critical surgeries in the healthcare domain, the real-time transmission of the digital sensors' data to the medical devices is not a negotiable constrain. Any small error might cause the death of the patients. In self-driving cars, the car must have real-time connection with all the cloud servers so that the radar, lidar and cameras' readings are communicated, pre-processed and fed to the algorithms for making decisions in real-time. Any small margin or error can cause fatal accidents and the losses of humans' life. A huge part of the entertaining industries nowadays is based on the Augmented Reality (AR) and VR. Streaming high-quality videos in real-time is one of the main factors the AR industry is based on. With the rapid increase of the previously mentioned applications and many more, maintaining the capacity and latency requirements becomes more and more challenging for both the technical side and the business side as well.

At the same time, artificial intelligence (AI), especially, the machine learning and deep learning models have been achieving breakthroughs in industry and research in all the domains. In E-commerce, machine learning models have been used in building personalized recommendation engines that we use on a daily basis for many different purposes starting from watching YouTube videos till the recommendation of strong passwords for our personal emails [3]. In healthcare, deep learning has been contributing to detecting fatal diseases such as cancer, enough time beforehand so that chances of saving patients life increase [4]. In digital marketing, machine learning has been used for identifying and clustering users' personas, predicting the conversion and retention rates and optimizing the advertising campaigns [5]. In finance, machine learning has been used for detecting fraud transactions and forecasting the stock markets [6]. There are also many sub-tracks under the umbrella of AI such as computer vision (CV) or machine vision, natural language processing (NLP), time series analysis and anomaly detection and more.

Anomaly detection for time series data is one of the very important topics that are essential and has a lot of applications in many domains and not just the Telecommunications domain. In network activities-related data sets, anomaly detection can be used for detecting intrusion attacks [7]. In the finance domain, anomaly detection can be used for detecting the unusual behaviors for users' purchases for instance [8]. Anomaly detection can be used for tracking the organic traffic and keywords ranking in search engine optimization (SEO) [9] in the digital marketing domain. At the same time, anomaly detection can be used for detecting the unusual behavior in heart rate signals in addition to some cancer and diabetic diseases rare cases [10] in the medical domain. Real estate domain has witnessed different applications of anomaly detection as well for the purpose of tracking and analyzing prices [11]. Even recommendation systems and recommendation engines have many application based on anomaly detection as it can be used for detecting the rare customer personas that might bias the performance of the machine or deep learning models responsible for making the recommendations [12]. In almost any domain, anomaly detection can be used and applied whether through applying basic and simple statistical approaches, classic machine learning models or deep neural networks.

## 1.1. Thesis Scope

As discussed earlier, there are many factors and constrains that can be used as evaluation metrics for the receiver side. Our main focus in this work is the BLER constrain as it is a crucial feedback measurement that significantly affects whether wireless communications processes are accepted, declined, or need enhancements. There are different studies that tackle the problem of time series anomaly detection in the telecommunications domain in general but, to the best of our knowledge, there is no focus on detecting anomalies on the time series based BLER target.

This work is organized as follows: Chapter 2 discusses the time series analysis and anomaly detection concepts, an overview of machine learning addition to an introduction to the fifth-generation network receiver side with an emphasize on the BLER target as an evaluation metric of the data reception process. Chapter 3 discusses the related work for time series anomaly detection using statistical approaches, machine learning-based approaches and deep learning based-approaches in addition to anomaly detection in the telecommunications domain. Chapter 4 discusses the data set description and the most important features it has. It also discusses the data analysis including the data cleaning and pre-processing and the exploratory data analysis and visualization. Chapter 5 discusses the methods used in this work and their advantages and disadvantages. Chapter 6 discusses the results of the four models selected for the forecasting and anomaly detection purposes. Chapter 7 discusses the results we got in addition to the limitations of the work and potential improvements. Chapter 8 concludes the work done and the potential future steps.

## 1.2. Contribution

The motivation behind this work is to build a reliable baseline system for automatic anomaly detection for the wireless communication time series features. One of the common challenges of any anomaly detection problem is the lack of labeled data. Labeling data points or sequences as anomalous or not is a time consuming process that requires a lot of resources and domain expertise. Being able to build and deploy a system that can detect anomalies in not only one specific time series feature, but also other wireless communications related features, in real-time, can contribute significantly to the company business and as a result, the end users.

The generalization of such a system to different telecommunications features is a challenging yet rewarding mission.

One of the main goals of this work is to overcome the problem of unlabeled anomalies by proposing a flagging method based on the assumption of the percentage of anomalies in the data set. Having labeled data will impact the system performance positively but it should not be a barrier for having good model performance using supervised machine learning or deep learning models. The data set used for this work has been recorded in an automated test environment and because of its generic use, it is not labeled for the anomaly detection purpose. Our approach tackles this change by representing the results in a severity based approach given the assumption of the percentage of anomalies in the data set.

This work is considered phase one of three phases. The objective of the first phase, the thesis work, is to select the best model for detecting anomalies in the univariate time series BLER feature based on the model accuracy related-metrics and computational resources. Through the second phase, the best model selected from phase one will be used in a multi-variate environment in order not to be just able to detect the anomalies but also, understand why this behavior happen in the first place. One of the advantages of applying anomaly detection in a time series multi-variate environment is that we will be able to verify the model performance and its ability to generalise better in different context in addition to testing potential correlation and causation among the time series features. In the final phase, the best model will be optimized and deployed as part of an automated system for the time series anomaly detection purpose in real-time.

# 2. BACKGROUND

In this chapter we give an overview about machine learning and its different sub-areas. We also introduce the main topic of this work, time series anomaly detection and dive deeper into the foundations of time series analysis and anomaly detection. In addition, we give a basic background about the BLER feature and the data reception process.

## 2.1. Machine Learning

Machine learning is one of the main sub-fields of AI that has been revolutionizing many industries due to its ability to outperform human experts' performance in solving complex tasks in an automated manner. There are many sub-tracks under the umbrella of machine learning such as supervised machine learning, unsupervised machine learning, reinforcement learning (RL), semi-supervised learning[13], self-supervised learning [14] and more. Supervised learning is the most popular sub-track due to its applicability and scalability to a wide range of applications and industries. The main advantage of supervised learning is that we have labeled data set. In other words, we are able to supervise the model's learning process as we have both the input and output known. The model needs to learn the process or the mapping from this input to that output so that when we have a future input, we can just pass it to the mapping process the model learned to predict the future output.

In supervised learning, we have mainly two problems: regression-based problems and classification-based problems. In the regression-based problems, the model forecasts a continuous value such as the prices of stock markets. In the classification-based problems, the model forecasts a discrete value, a category or a class such as forecasting whether the value of the stock market will increase or not. The discrete values in classification problems can take any number as far as there are no infinite possibilities of the values of that number and we know these possibilities beforehand. For instance, in computer vision, we can have one million classes that the model needs to distinguish and it is considered a classification problem. In the case of anomaly detection problems, if we have a labeled data set, it is considered a classification problem as the models needs to predict whether a point or a sequence is an anomaly or not.

In unsupervised learning, we have only the input but we do not know the output beforehand which makes unsupervised machine learning problems more challenging. In RL, we do not have either the input or the output defined. The learning process in RL depends on the interaction of an agent with the surrounding environment and the feedback signal it gets based on that interaction. Other areas such as self-supervised learning, one-shot learning [15] and few-shot learning [16] are based on the fact that we have both labeled and unlabeled data where the model tries to generalize its learning experience using few labeled data and enhance this learning experience using the unlabeled data. Both supervised and unsupervised machine learning models can be used for applying anomaly detection. The models selected for this work are from both the categories.

One of the most common challenges in anomaly detection problems is that most of the time, we do not have the anomalies themselves labeled as anomalies or not. If they are labeled, we can transform the problem into a supervised classification problem and then, apply a suitable classification model that yields good performance but as discussed, this is not the common case. In the case of unlabeled anomalies data set, sometimes we need the participation of domain experts to label the data and provide some guidelines and directions for the whole labeling process. This is not the most efficient way to solve the problem of unlabeled anomalies as it is really resource-consuming in terms of time and cost. Getting domain experts to do this labeling themselves can be quite expensive as well.

Another solution for the unlabeled anomalies problem is to have some meta data or additional information about the data set such as the potential percentage of anomalies in the data or the contamination value. By knowing such information, we can apply confidence interval-related statistical approaches and flag the different anomalies based on the model performance. This is the approach that we will follow in our thesis given that we do not have the anomalies labeled in our data set.

## 2.2. Time Series Analysis

Time series is a sequence of information that attaches a time period to each value and this value can be represented in almost everything such as prices, humidity, or the number of people. As far as the value recorded is unambiguous, any medium could be measured with time series. There are not any limitations regarding the total span of a time series. It can be a minute, a day, a month, or even a century. The most important thing is to have a starting and an ending point. The interval between one value and value next to that value is called a time period or time step. Frequency is referred to as how often values of the data set are recorded. To be able to analyze and extract insights from time series data, all time steps must be equal and clearly defined, which would result in a constant frequency. This frequency is a measurement of the time and can range from a few milliseconds to decades. Generally, patterns observed in time series are expected to persist in the future. That is why we try to predict the future by analyzing the past recorded values.

Time dependency is also an important feature of time series data which means that the values for every period are affected by outside factors and by the values of past periods. Time series data suffer from seasonality as some values depend on the time of day or season of the year for instance and since it is a repeated pattern, we can anticipate these changes and account for them when making our predictions. Seasonality as a trade is not often observed in regular data. When there is no chronological order, we do not expect repeated cycles. We express time series variables with capital letters such as $X$ and $Y$. We express the entire period covered by a time series sequence with $T$ while we use $t$ to describe a single period of an interval. Since $t$ represents the order of the period we are interested in, we use $t-1$ to express the previous period with one time step difference. Similarly, we can express the next period as $t+1$.

The intervals between the time steps or time periods need to be identical and if this is not the case, it is usually because of the missing values. The values between consecutive periods usually affect each other. We can also adjust the frequency of the data set based on the values we are interested in. For example, if we have daily data and we want to conduct a monthly analysis, we need to compute some average values for the recorder's daily values. In this way, by aggregating the data, we will try to determine the best values to describe a month. Sometimes, we have to increase the frequency and this leads to an increase in the number of time steps within an interval and consequently, the time period. Then, comes the need to impute values for the missing ones. As mentioned before, unlike regular data, time series require chronological order. From the machine learning perspective, this is inconvenient as we cannot shuffle the data into train and test sets. What we do instead is to pick a cut-off point and the data before the cut-off point is the training set and the data after this cut-off point is the testing or validation set.

Another odd of time series data is that its graphs do not follow any popular statistical distribution such as the normal distribution for instance. This is because time series data never satisfies Gauss-Markov assumptions, unlike regular linear regression data. Instead, time series data assumes that the past time patterns in the variable will continue unchanged in the future

The quantile-quantile (Q-Q) plot is one of the tools used in analytics to determine whether a data set is distributed a certain way. It usually showcases how the data fits a normal distribution. The Q-Q plot works by taking all the values a variable can take and arrange them in ascending order. The $Y$ axis represents the variable values and the $X$ axis represents the theoretical quantiles of the data set, in other words, how many standard deviations away from the mean these values are. The diagonal line represents what the data should follow if they are normally distributed. If the data is not normally distributed, we cannot use popular statistical techniques for making a forecast. However, this is what is usually expected from time series data.

Another important concept in time series is white noise [17]. White noise is a special type of time series where the data does not follow a pattern and since no pattern can be found, white noise is not predictable. In order for a series to be considered as a white noise series, it needs to satisfy these three conditions:

- Constant mean
- Constant variance
- No auto-correlation.

Auto-correlation means how correlated a series is with a past version of itself

$$p = corr(x_t, x_{t-1}), \tag{1}$$

where $p$ is the auto-correlation value. Hence, no auto-correlation means that there is no clear relationship between the past and present values of a time series. Thus, we can also say that white noise is a sequence of random data, where every value has a time period associated with it.

Another related concept is random walk. Random Walk is a special type of time series where values tend to persist over time and the differences between periods are simply white noise

$$X_t = X_{t-1} + \epsilon_t, \tag{2}$$

where $\epsilon_t$ is the residual coefficient. We assume that these residuals are white noise, so they are arbitrary and cannot be predicted. This suggests that the best estimator for the current variable value is the previous variable value and the best estimator for the next variable value is the current variable value and so on.

In order to understand the different tests that should be conducted before applying any statistical approaches to time series data sets, we need to understand the concept of time series stationarity [18]. Stationarity implies that taking conductive samples of data with the same size should have identical covariances regardless of the starting point. This characteristic of the data is also known as weak-form stationarity or covariance stationarity. In other words, we classify a time series data as stationary data if it satisfies their key assumptions:

- Constant mean
- Constant variance.

Consistent covariance between periods and identical distances from one another are key assumptions for time series stationarity. To satisfy this concept, we want to have the same covariance between the first and fourth period as we do between the third and sixth:

$$cov(x_1, x_4) = cov(x_3, x_6)\cdot \tag{3}$$

We can also say that a sample of a weak-form stationary process is white noise as the mean, in that case, is equal to 0 and the variance is constant. Additionally, auto-correlation between lags is always zero. Since covariance is simply correlation multiplied by the standard deviations, it will also equal zero and hence, white noise stratifies all the assumptions of a covariance stationary process. Generally, when we refer to stationarity, we refer to it as strict stationarity where samples of identical size will have distributions:

$$(x_t, x_{t+k}) \sim dist(\mu, \sigma^2), \tag{4}$$

$$(x_{t+\tau}, x_{t+\tau+k}) \sim dist(\mu, \sigma^2), \tag{5}$$

where $k$ and $\tau$ are the sampling factors to be chosen to achieve the matching of the samples size.

Statisticians David Dickey and Wayne Fuller developed a statistical test to check whether a data set is stationary and it is called Dickey-Fuller test [19]

- $H_0 : \phi_1 < 1$
- $H_1 : \phi_1 = 1$

The null of the Dickey-Fuller test ($H_0$) assumes non-stationarity. It assumes that one lag of the autocorrelation coeffient ($\phi_1$) is lower than one and when we compute the stastistic, we compare it to the values in fuller table. If it is lower than the critical value, then, we reject the null and hence, the data comes from a stationary process.

Seasonality suggests that certain trends will appear on a cyclical basis. One way to test whether a data set has a seasonality is to decompose the signal or split it into three components [20]:

- Trend: represents the pattern consistent through the data
- Seasonal: represents all the cyclical effects due to seasonality
- Residual: represents the error of prediction of the actual data and the model we fit.

The simplest decomposition is called a naïve decomposition. With naïve decomposition, we expect a linear relationship between the three parts observed in the time series. Naïve decomposition has two features which are additive and multiplicative. Additive assumes that for any period, the observed value is the sum of the trend, seasonal and residual for that period. Similarly, the multiplicative decomposition assumes that the observed series is a product of the trend, seasonal and residual components.

In order to be able to specify the number of lags that we will use for measuring any of the time series components such as the seasonality, we need to calculate the correlation between lags whether it is auto-correlation or partial auto-correlation. Correlation measures the similarity in the change of values of two series. Auto-correlation represents the correlation between a sequence and itself. In other words, it measures the level of resemblances between a sequence from several lags ago and the actual data. The lags mean a delayed version of the original one.

The number of significant lags are one of the main parameters to be fed to the models and tuning most of the forecasting models is based on tuning the lags' value. Sometimes, the number of lags even influences the choice of the selected model. For example, if we have only three to five significant lags, we can use any simple model including the statistical approaches. If we have a high number of significant lags, we need to use more complex models in order to be able to detect the patterns in such long sequences. In addition to the number of lags, There are also other factors that influence the model selection and its degree of complexity such as:

1. Significant coefficients
2. Parsimonious
3. Residuals

If the values of the coefficients are not that significant or very close to zero, that means that they are not contributing to our model results that much. In other words, they do not have predictive power. Parsimonious means as simple as possible as we generally prefer to use a simpler model than the complex one as the latter generally provide significantly better predictions. For us to be able to decide whether our predictions are good ones, we use a statistic test called the log-likelihood ratio test but it only can be applied to models with different levels of degree freedom. So, we need to compare the information criteria for each one. The lower the coefficients, the less data the model requires to make accurate predictions. The two most common features for this measurement are the Akaike's Information Criteria (AIC) and Bayesian

Information Criteria (BIC). Because we tend to prefer simpler models, we prefer the models with lower AIC and BIC values. Regarding the residuals, if our model fits well, there should be no trend we fail to account for and the residual of the model should resemble white noise. So, we can conclude that there are no other patterns we can account for when we overtrain our model. By overtrain we mean that the model is learning the data too well that it might not be able to generalize to a new data set.

## 2.3. Anomaly Detection

Before stating some popular definitions of anomalies, we need to highlight the fact that there is a distinction between anomaly detection and outlier detection. Despite that there is no clear measurement in terms of the definition between outliers and anomalies, there are some factors that can be considered to distinguish between them. Most of the time, the two terms, outliers and anomalies are used interchangeably.

One definition of outliers is as stated by Aggarwal [21]: “Outliers are also referred to as abnormalities, discordants, deviants, or anomalies in the data mining and statistics literature.”. Some also tend to define outliers as a broader and more generic concept of anomalies which consists of some noise in addition to the anomalies. Finally, Hawkins [22] defines outliers as follows: “An outlier is an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism.”. While they also define anomalies as irregular behaviors that share some patterns [23]. On the other hand, one the most popular definitions of anomalies is "Anomalies are patterns in data that do not conform to a well-defined notion of normal behavior.” – Chandola et al. [24]. Another popular definition of anomalies is "An anomaly is an observation or a sequence of observations which deviates remarkably from the general distribution of data. The set of the anomalies form a very small part of the data set." [25]. The main focus of our thesis work is on anomaly detection for time series data.

There are many different types of anomalies but we can categorize them into three main categories:

**Point Anomalies**: this is the case when we have some point values that deviate significantly from the other points’ values. For instance, an extreme temperature degree within a specific weather season. So, $X_t$ is considered a point anomaly if its value deviates significantly from all the points in the interval $[X_{t-n}, X_{t+n}]$, $n \in R$ and n is sufficient large.

**Collective Anomalies**: this is the case where each point value that deviates from the other values does not consider to be an anomalous point but the sequence of the values together is labeled as a collective sequential anomaly. An example of collective anomalies can be a very big withdrawal or purchase from a bank account each day within the same week. A big withdrawal every once in a while is not considered an anomaly as this might happen for different reasons but the sequence of the big withdraws within the same week is considered an anomalous behavior.

**Contextual Anomalies**: in contextual anomalies, the values themselves do not show an anomalous behavior but it depends mainly on the context of these values. Back to the temperature degree example, a +20 degree in Finland in Summer, for instance, is not considered an anomaly while the exact same temperature in Winter in the same country is considered an anomaly. The main differentiator is the context of the values and not the values themselves.

There is also a measure of how the anomalous values deviate from the other values. This measure is usually used to explain the strength of the anomalous values, or in other words, the likelihood of being an anomaly. This measure is referred to as an anomaly score. How an anomaly score is calculated differs from one approach to the other, for instance, whether it is a statistical approach or machine learning-based approach. Detecting anomalies in the case of univariate time series unlabeled data sets is applied by using a forecasting model first to which we fit our training data set and then, a test data set is used to make predictions and evaluate the model performance. A window is used with a certain number of lags is used to mark the labeling of the target. The number of lags depends on different factors such as the model used and whether it is a statistical approach, machine learning or deep learning model, the data set itself and whether it is stationary or not, the nature of the target variable itself, and more.

The anomaly score is computed by measuring the distance between the model prediction and the actual value of the target

$$e_i = d(x_i, x^i), \tag{6}$$

where $d$ is the chosen distance function. The distance function is also dependent on the model used and in the case of the machine learning models, it differs depending on whether it is a supervised or unsupervised machine learning model. The deviation $e_i$ can be referred to as the error function which is proportional to the anomaly score. For instance, if the error value is larger than a specific threshold, the point is considered an anomalous point.

## 2.4. 5G Networks and BLER Metric

5G is the next promising generation of mobile broadband that offers download and upload speeds that the mobile users have not experienced before. From the user perspective, and in addition to the higher download and upload speeds, mobile users enjoy low battery consumption which also allows the service providers to develop more efficient phone devices. 5G traffic fees is considered lower than the previous networks due to the infrastructure low maintenance resources required in terms of the cost. Moreover, increased throughput and lower network outage probability is one of the most important features that distinguishes the 5G network.

However, all of these powerful features of the 5G network imposes some challenges for reliability, latency, capacity and network outage. Controlling the network outage is not an easy task and it is one of the main goals of the current research. In simple terms, maintaining low outage means maintaining successful transmission and reception processes. There are many ways to enhance the transmission process but our main focus in this work is the receiver side. There are also many factors to measure the quality and accuracy of the data transmitted to the receiver side and we are going to focus on only the monitoring and evaluation of one factor in this work which is the BLER [26]. BLER is considered one of the most effective metrics of evaluating the demodulation process and the receiver sensitivity which is one of the most important measurements of the receiver performance. As a rule of thumb, the higher the modulation coding scheme (MCS) [27], the higher the spectral efficiency and as a result, higher data throughput.

The MCS selection is mainly based on two factors; the radio signal quality and the BLER. Signal to noise ratio (SNR) [28] is used to measure the radio signal quality as it represents the difference between the received signal power and the noise power. BLER is defined by 3GPP [29] as a physical layer error estimation technique. BLER is calculated as the number of unsuccessfully transmitted data blocks over the total number of transmitted blocks within a certain number of frames as follows

$$BLER = \frac{number\ of\ erroneous\ blocks}{total\ number\ of\ transmitted\ blocks}. \tag{7}$$

BLER can also be defined as the number of negative acknowledgment (NACKs) over the total number of transmitted blocks which is equal to the positive acknowledgment (ACKs) summed up to NACKs. A NACK can be considered as a negative feedback signal when the data block is transmitted unsuccessfully while the ACK can be considered a positive feedback signal that acknowledges that the data blocks were transmitted successfully, which leads to the following equation:

$$BLER = \frac{number\ of\ NACKs}{number\ of\ ACKs + number\ of\ NACKs}. \tag{8}$$

BLER is also used to measure the level of interference. The higher the SNR, the lower BLER expected for a certain modulation scheme, and the opposite is true. When we have a low SNR, we expect a lot of interference and as a result, the data blocks transmitted in error will increase and hence the BLER. When we have high BLER, we expect more drops in calls, poor reliability, handovers, and latency.

# 3. RELATED WORK

In literature, different terms are used interchangeably that express the same meaning of anomaly detection. Anomaly detection, deviant discovery, intrusion detection, fault detection, misuse detection [30] and more are used within the same context to express unfamiliar or unexpected behavior. As discussed earlier, anomaly detection methods can be divided into three categories; statistical approaches, machine learning-based approaches and deep learning-based approaches. For the machine learning approaches, we can categorize them into two categories; supervised machine learning approaches and unsupervised machine learning approaches. In this chapter, we discuss the related work in the three main categories for the time series anomaly detection generally. Then, we review some novel approaches for detecting anomalies in cellular networks, specifically.

## 3.1. Statistical Approaches

In addition to the auto-regressive integrated moving average (ARIMA) model that will be discussed in the methods chapter, there are many other different approaches. Double and triple exponential smoothing (DES, TES) [31] are considered an extension of the simple exponential smoothing (SES) model as it uses an additional parameter, beta, as a smoothing factor to smooth the trend in the time series target. Triple exponential smoothing is also considered an extension of the double exponential smoothing as it adds a third parameter, gamma, to influence the seasonality of the time series data set.

Another statistical approach is the prediction confidence interval (PCI) [32]. PCI uses a non-linear weighted technique over the time series sequence to forecast the next data point. After the forecasting, it applies a threshold to classify whether the forecasted point is an anomaly or not. To calculate $X_t$, it uses a window of past observed points of the series:

$$X_t = \frac{\Sigma_{j=1}^{2k} w_{t-j} X_{t-j}}{\Sigma_{j=1}^{2k} X_{t-j}}, \tag{9}$$

where $w_{t-j}$ is the weight for $X_{t-j}$ and it is proportional to the inverse of the distance between $X_t$ and $X_{t-j}$ . Then, it uses upper and lower boundaries to classify the anomalies given this equation:

$$PCI = X_t \pm t_{\alpha,2k-1} \cdot \sqrt[s]{1 + \frac{1}{2k}}, \tag{10}$$

where $t_{\alpha,2k-1}$ is the p-th percentile of the target feature t-distribution with $2k - 1$ degrees of freedom, $s$ is the standard deviation value, and $k$ is the window size. Thus, $s$ is calculated based on the window $k$. Another hyperparameter to be tuned to choose the PCI boundaries range is $\alpha$. The value of $X_t$ is checked whether it fits within the boundaries range. If it is outside the range, then, the point is considered an anomaly. Overfitting and underfitting are common challenges for this method and tuning all the hyperparameters helps in overcoming these challenges.

## 3.2. Machine Learning Approaches

By the machine learning approaches we mean the classic machine learning approaches and not the deep learning based one. As discussed earlier, deep learning is a sub-area of machine learning. The anomaly detection deep learning approaches will be discussed in the following section.

K-means clustering is used widely in unsupervised problems for clustering purposes. In time series-related problems, K-means clustering can be also used, and sometimes it is referred to as subsequence time series clustering (STSC). To apply K-means clustering for anomaly detection, a sliding window methodology is used [33]. The algorithm starts running on the data set until it converges which results in $K$ number of centroids [34]. A centroid is considered the mean of vectors in one cluster. To detect anomalies an error function is computed by calculating a distance function, usually, the euclidean distance, but other distance functions can be used as well depending on the data set and whether we are working on univariate of multivariate time series sequences. After calculating the distance function, a threshold is specified and if the error or distance is higher than the specified threshold, the point is considered an anomaly.

One of the main challenges of the K-means clustering algorithm generally is specifying the hyperparameter $K$, which needs a lot of experimentation. Another challenge is the time complexity of running this algorithm which scales drastically with the length of the data set. Keogh and Lin [35] showed that using a subsequence part of the data set and feeding it to the K-means clustering algorithm is complexity insignificant as the results are almost the same as using a random walk algorithm for selecting the K-centroids. They did this experimentation to try to overcome the time complexity challenge of the K-means clustering algorithm. They also experimented with different distance functions such as the Manhattan distance but the results were not significantly different from the ones that were based on the Euclidean distance.

Another algorithm is the Density-Based Spatial Clustering algorithm (DBSC) [36]. DBSC is another clustering algorithm that is used for time series anomaly detection. In addition to the clustering methodology, the Density-Based Spatial Clustering algorithm also analyzes and categorizes the points into three categories:

- Core points
- Border points
- Anomalies.

There are mainly two hyperparameters used to tune the performance of this algorithm, unlike the K-means clustering algorithm which had only one hyperparameter, the number of clusters. The two hyperparameters used in DBSC are the distance to the neighbors of the selected point and the minimum number of points that each cluster has to have in order for the model to perform well. To classify a point into one of the three categories mentioned before, the distance to the neighbors for each point has to be calculated first. Based on the calculated distances and a threshold to be determined, the point can be classified as a core point or in other words, a safe point, a border point or in other words, a suspicious point and an anomalous point.

Local outlier factor (LOF) is another clustering-based anomaly detection algorithm for time series data. It is important to mention that the LOF algorithm was used initially to detect outliers and anomalies in spatial data sets. After that, it was extended by Oehmcke et al. [37] to include time series data sets as well. Unlike the DBSC algorithm that is based on the density approach, it is based on the K clustering approach similar to the K-nearest neighbor (K-NN) algorithm [38]. K-NN is one of the most popular clustering algorithms. It assigns a point to a certain cluster based on the distance between it and the other $k$ neighbors. A time series sequence is to be divided into a train set A and a test set B. After that, a sliding window with window length $w$ is applied to both the training and testing data sets as a transformation function. To determine whether the new transformed sequence is an anomalous sequence or not, an anomaly score is calculated and if it is higher than a pre-defined threshold, it is considered as an anomalous sequence.

The anomaly score is calculated so that it measures how much the different transformed data sets are similar or different to each other. This algorithm has its own challenges as well:

**The 1st challenge**: is tuning the hyperparameter $K$ similar to the $K$ clustering algorithm which requires some domain knowledge about the data set.

**The 2nd challenge**: is that the algorithm depends mainly on the direct neighbors, which on the other hand, makes it more sufficient for detecting local anomalies.

**The 3rd challenge:** is choosing the error or distance function as it affects how the anomaly score will be calculated. For instance, using the Euclidean distance might not be suitable for this algorithm in the case of working with a multidimensional space.

**The 4th challenge**: is the time complexity of running this algorithm which is $O(n^2)$

Semi-supervised learning models are also used for the anomaly detection purpose in addition to the unsupervised learning models. One-class support vector machines (OC-SVM) [39] is a semi-supervised machine learning algorithm that is used for classification problems and hence, it can be used for classifying the anomalous points in time series data sets. The main support vector machine classification algorithm was extended by adding the kernel trick which enabled the algorithm to detect the non-linear patterns. After that, the algorithm was extended again to be able to detect anomalies [40]. The data set is divided into training and testing sets where the training set includes only the normal data. After feeding the training data to the model, the test set is used as a classification set and is compared to whether it is similar to the normal, training, data set or not. A threshold is pre-determined to be able to detect anomalies after feeding the training data to the model. If the classification error is higher than the specified threshold, then, the point is classified as an anomaly.

## 3.3. Deep Learning Approaches

The third category is the deep artificial neural networks or the deep learning-based models. The most basic format of neural networks is the multi-layer perceptron (MLP) which is a fully-connected neural network that is based on feedforward propagation. Hyndman and Athanasopoulos [41] used neural networks as autoregressive models for forecasting the time series signals. Despite the fact that they used only one hidden layer, they were able to predict the time series signal efficiently.

Haselsteiner and Pfurtscheller [42] used two different variations of MLP for forecasting and classification of anomalies. For the classification model, they used finite impulse response filters (FIR) that are based on MLP. On the other hand, they used a sliding window technique for forecasting purpose. After using MLP for forecasting the time series signal based on the sliding window approach, an error function is calculated based on the model prediction and labels and after that, an anomaly score is used to determine whether a point is considered an anomaly or not.

One of the biggest advantages of using MLP for anomaly detection is that we are able to forecast more than one time step at a time which allows for applying anomaly detection for time series sequences accurately and not just a single point. On the other hand, there are some challenges that are represented mainly in the hyperparameters that need to be tuned in the case of neural networks. Hyperparameters can be:

- number of hidden layers
- number of neurons of perceptrons per a hidden layer
- learning rate
- the activation function
- the optimization function
- the regularization function, if needed.
- the window length in case of the forecasting model.

Another popular deep learning architecture that has been used widely for computer vision applications such as image classification, object detection, semantic segmentation, and more is the convolution neural networks (CNN). CNNs usage has been extended to detecting anomalies in time series data. In contrast to MLP, which are full-connected layers, CNNs are based on the convolutional layers that are partially connected in addition to the pooling layers such as the maximum pooling layers [43].

Munir et al. used a CNN architecture (DeepAnT) [44] that is based on deep learning for the time series anomaly detection purpose. While we feed 2D images to the CNN architecture for classification purposes, in the case of DeepAnT, we feed a 1D univariate time series signal which leads to using 1D kernels as well. In the original architecture of DeepAnT, two convolutional layers followed by max-pooling layers were used in addition to a fully-connected dense layer at the end. ReLU or rectified are used as the activation function [45].

Batch normalization [46] sometimes is being used as a regularization technique that can replace the effect of the dropout layers. It also results in a higher learning rate and hence, easier parameters initialization. The same prediction methodology as the MLP one is being used in the case of CNN architectures as well.

Using CNN has its own challenges as well, in terms of hyperparameter tuning. So, in addition to the parameters that need to be specified in the case of MLP architectures, there are a couple of additional parameters that need to be tuned in the case of CNN architectures. These parameters can be:

- the actual architecture of CNN
- the design choice of batch normalization
- the design choice of pooling layers
- the design choice of dropout/regularization layers
- the depth of the convolutional layers
- the number of kernels per a convolutional layer
- the kernels' size.

Residual Neural Network (Resnet) [47] is one of the most important extensions of CNNs that has been used widely for many applications and not just in computer vision. When using deep CNN architectures with quite many pooling layers, the problem of vanishing gradient appears which affects most of the values of the parameters to be close to zero. The skip connections idea was developed as a solution to this problem where simply a connection from a certain layer at the start of the architecture, depending on the design choice is fed to a later layer in the middle or last part of the network as a shortcut feedback layer.

Wang et al. [48] used ResNet for time series anomalies classification. They used mainly three convolutional layers with batch normalization. ReLU was used as the activation function. In addition, they used three residential blocks with dimensions of 64, 128, and 128. The main challenge of the Resnet architectures in addition to all the CNNs challenges is that the Rent architectures can overfit the data easily, especially, in the case of limited data sets but the overfitting problem can be resolved using different techniques.

Wavenet [49] is another variation of CNN architecture that has been used widely in the audio-related applications. It is basically a deep generative model that was used to create audio waveforms. In other words, it is a probabilistic model that predicts the distribution of the signals by approximating the joint probability.

$$p(x) = \prod_{t=1}^{T} p(x_t|x1, \cdots, x_{t-1}) \tag{11}$$

The difference between the Wavenet architecture and the regular CNN architecture is that the Wavenet one is using dilated convolution layers instead of the regular convolution filters used in the case of CNN architectures. The dilated convolution layer enables the convolution operation to be applied on data with a size higher than the kernel size using the skipping trick. Borovykh et al. [50] used the concept of WaveNet for time series forecasting by applying the dilated concept to a regular CNN architecture which enabled the use of a much bigger window size in comparison to the one used in the case of regular convolution layer.

Long Short Term Memory (LSTM) [51] has been used a lot recently for many applications and especially, the time series based ones because of its strong ability to detect patterns in long sequences. LSTM architectures are one of the variations of the Recurrent Neural Network architectures (RNNs). RNNs architectures generally have the advantage of feedback connections that enable them to get feedback from previous sequences. As the name implies, LSTMs are able to detect patterns in both short and long sequences. LSTMs architectures also have the ability to overcome the vanishing gradient problem because of their different gates functionality. They have mainly two gates that are responsible for retaining the important information, patterns, and forgetting the unnecessary information. The other two gates are the input and output gates which are responsible for processing the input signal and output the most significant part of the long-short memory. Chauhan and Vig [52] used an LSTM architecture to predict (ECG) signals. By using the probability distribution of the output and after calculating the error based on the prediction, they were able to classify whether an input signal is considered an anomaly or not. Malhotra et al. [53] used an LSTM architecture of two layers for the same forecasting purpose but using a multivariate Gaussian distribution context. They applied maximum likelihood estimation (MLE) after calculating the prediction error to detect the anomalies in the time series signal. Gated recurrent unit (GRU) [54] is another variation of RNNs. It is different from the LSTM architectures as it includes the input and forgets gates in only one gate. In addition, the output gate functions in a different manner as the full-state vector is the output of every timestamp. GRUs have the advantage of requiring less computational resources in comparison to the LSTM architectures. Wu et al. [55] used a stacked GRU model for time series anomaly detection using the same approach used with LSTM architectures.

Autoencoders [56] are another deep learning-based architectures that belong to the feedforward neural networks family. They consist of two main blocks the encoder and decoder parts. The encoder block encodes the input into a project space called the latent space where the dimensionality of the data is reduced to include only the most significant information of the data patterns. The decoder part decodes the embedded vector into its original dimension back. The main reason that autoencoders based architectures are useful in the case of anomaly detection is that the embedded vectors of normal data and anomalous data can be distinguished easily in the latent space. Sakurada and Yairi [57] used an autoencoder architecture for detecting anomalies on time series. They benchmarked the results with a simple linear principal component analysis (PCA) technique. They used what they called a denoising autoencoder in addition to a normal autoencoder. The denoising autoencoder works on removing the noise from the original time series signal and hence, they are able to detect the anomalies after projecting the normal time series signal only on the latent space.

### 3.4. Anomaly Detection in Telecommunications

The autoregressive (AR) model was used not only for time series forecasting but also detecting anomalies in network related activities. Shao et al. [58] used the AR model was used to detect the abrupt change in time series network data sets. After using the AR model for forecasting, a sequential hypothesis test was applied to the output of the model to filter the anomalies. Time correlation and location correlation were key features that contributed to the model learning experience and helped in detecting the anomalies in addition to their occurring time and location.

A hybrid method of three models were used to study the mobile users' behavior in cellular networks, predict the number of calls within a specific time in a certain region and detect any potential anomalies [59]. The hybrid method used included a supervised machine learning model, generalized autoRegressive conditional heteroskedasticity (GARCH) [60], an unsupervised machine learning, K-means and a deep learning method, an MLP architecture. The $k$ parameter in the K-means model was specified to two in order to cluster the data set into two clusters. The first cluster includes the normal points and the second cluster includes the anomalous points. The GARCH model was used to stabilize the mean the data points by removing potential noise. The output of the GARCH model is to be fed to the Neural Network architecture for the forecasting purpose and detecting the anomalies.

Another hybrid model consists of two models was used to filter out outliers and understand their behavior in cellular networks [61]. One class support vector machine (OC-SVM), an LSTM architecture and K-means models were used for automatic labeling of the data set, forecasting the target feature and finally, detecting anomalies. The OC-SVM model was used to filter out the outliers in the first phase of the anomalies detection. After that, the LSTM architecture was used for a deeper understanding of the root cause for such behaviors in addition to successfully classifying both the false positives (FP) and false negatives (FN) anomalies. They used an anomaly filtering technique after studying the key performance indicators (KPIs) features' profile behavior for labeling the data set and transforming the problem into a supervised one.

While most of the research papers were focusing on solving the anomaly detection problems using novel methods, a traditional graph model was used but with a novel approach that focuses on modeling and reading the data set in a graph format was proposed [62]. The main idea is to transform the cellular network data set into a graph representation where a graph-based anomaly detection model can be used easily. This approach suggests using the history of the mobile users' phone calls and messages to detect the anomalous users' patterns. Transforming the data such into the graph format will make using any graph-based model a straightforward and efficient approach for reporting anomalies.

Topic modeling has been mainly used for text analysis purposes such as detecting patterns in documents such as the words count frequencies. In the work by [63], and given its statistical probabilistic property, the topic modeling algorithm was used as a clustering algorithm to cluster the normal point together and filter out the anomalous points in a separate cluster. They applied the algorithm directly to the data gathered from live Radio Access Networks (RANs) and they were able to cluster the anomalies with a performance that is very close to human experts' one.

# 4. DATA ANALYSIS

In this chapter, we discuss the data used for this work and the cleaning and pre-processing methods used and the reasoning behind using them. We also visualize both the uplink and downlink BLER features and apply different statistical tests to check the number of significant lags, stationarity of the data sets and isolate the main time series signal into its three components; trend, seasonality and residuals. Moreover, we conclude our findings.

## 4.1. Data Description

The data set was recorded in a field testing environment using an in-house agnostic tool. Using this tool, the granularity of the data sets can be specified in addition to features generated for both the downlink and uplink cases. The data set recorded for the purpose of this work has a granularity of 80 ms for each time step. The data set has 4614 records for both the uplink and downlink cases. These 4614 are in the time span of 7 minutes starting from the time 10.46 to 10.52. The data set has 89 features. Some of the 89 features are common for both the uplink and downlink cases such as the timestamp and the rest are case dependant. The uplink and downlink data sets were extracted from the main data set file so that the analysis and modeling can be applied to each of them separately. Modeling both the data sets separately was necessary because of the difference between the data sets and comparing their results to each other does not make sense. The uplink case represents the case of data transmission from the user equipment (UE) to the base station while the downlink case represents the case of data transmission from the base station to the UE.

Since our work focuses on the anomaly detection for univariate time series targets. We are mainly concerned about the BLER target for both the uplink and downlink cases. It is important to highlight that the BLER feature values are not labeled as anomalous or normal point. Solving this problem will be explained later in this chapter. Other features were used for the analysis purpose during the data pre-processing, cleaning, and analysis phases. Some of the features that the data sets include:

- Number of transmitted blocks
- Number of received blocks
- Number of ACKs
- Number of NACKs
- MCS: modulated coding scheme
- CQI: Channel Quality Indicator.

The MCS and CQI features were used during the exploratory analysis phase to get a better understanding of the BLER feature by checking the potential correlation and causation relation with the BLER feature. The MCS and CQI showed some correlation but it was not significant enough. The other four features; number of transmitted data blocks, number of received data blocks, ACKs and NACKs were used to calculate the BLER within specified window. The four features were also used to check the correlation between them and the BLER feature within smaller periods such as one minute or a couple of seconds instead of the whole data set period, seven minutes.

## 4.2. Data Cleaning and Preprocessing

During the data preprocessing phase, we focused mainly on handling four major aspects:

- Missing values
- Categorical features
- Duplicates
- Outliers
- Normalization.

**Missing values:** given that we are working on uni-variate time series target, we are mainly concerned about the target feature itself and the timestamp feature. The feature engineering phase will become handy for more complex cases of multi-variate time series analysis and forecasting. For the missing values, only five features had the dominant percentage of missing values. These features were representing the IDs of some events, and they were all unique. So, they are not contributing to our work scope. After dropping the five features, we are left only with fourteen data entry with missing values. Given that the 14 records represent less than 0.003 of the total number of records, we decided to drop them as well other than working on the imputation of these missing values.

**Categorical features:** for the categorical features, all of their classes were already encoded properly except for the ones with missing values that have been already handled in the previous phase. Only two categorical features had only one value consistently in all the records. So, they will not be helpful whether during the analysis or modeling phases. We dropped these two features as well.

**Duplicates:** during the initial check of duplicates in the data, we found that more than half of the records have duplicated values. After deeper investigation, we found that this problem was caused because of the timestamp features as the values of the timestamp features started repeating themselves again from the beginning after the record 2302. We found out this was a bug during the measurement phase and all the other features values have no duplicates. We tackled the duplicates problem by changing the duplicated values of the timestamp feature given that the granularity of the data set is 80 ms per each time step. We looped over the the duplicated timestamp values starting after the record 2302 and edited them manually to match the accurate measurement.

**Outliers:** because of the scope of the problem we are tackling, time series anomaly detection, we expect that the data has already anomalies or outliers. So, it is important to keep them to be able to train the different machine learning and deep learning models to classify the anomalous points in addition to the normal ones as well.

**Normalization:** we normalized all the BLER values for both the uplink and downlink values by using a simple min-max normalization so that the final values range is from 0 to 1. Due to the existence of the anomalies in the data set, using the BLER values as they are might influence the model performance and normalizing even the anomalous points helps the model to generalize better by understanding the pattern of the anomalies.

## 4.3. Analysis and Visualization

After cleaning the data and making sure it is in a good shape, we started by dividing the data set to uplink and downlink data sets based on their features. There were some shared features for both the uplink and downlink cases which were mostly the timestamp related ones. The other features were case dependant. So, we divided the main data set so that the downlink data set has its own features in addition to the shared features and the same applies to the uplink case.

### *4.3.1. Uplink*

We started by a basic visualization of the BLER feature. As we can see in Figure 1, most of the values are zero which is the optimal case of the BLER values. We can also see that there is a pattern that repeats representing increasing and decreasing trends of the BLER values. Visualizing the whole BLER signal does not reveal much information in this case and that is why we decided to divide the signal into smaller ones and visualize each of them separately. All the data set records were recorded through almost seven minutes, so, we divided the data set into 7 chunks, the first chunk represents the first minute, the second chunk represents the second minute and so on.

Some of the new divided data sets shared the same pattern as the original data set such as the pattern represented in Figure 2. While others showed the normal expected behavior of the BLER feature, to have all the values equal to zero or close to it such as the one illustrated in Figure 3. Others showed a lot of fluctuations in the signal as illustrated in Figure 4.

Before moving to the statistical tests, We checked the correlation between the BLER feature and the other features in the uplink data set but there was not a strong clear correlation between the BLER and any of the other features. So, we decided to check the correlation within shorter duration, such as a specific minute of time or through each couple of seconds. Some of the features started showing some correlation but it was not significant enough. Checking the correlation between the BLER feature and the other features helps us in understanding the potential important features that might be used in the feature engineering phase.

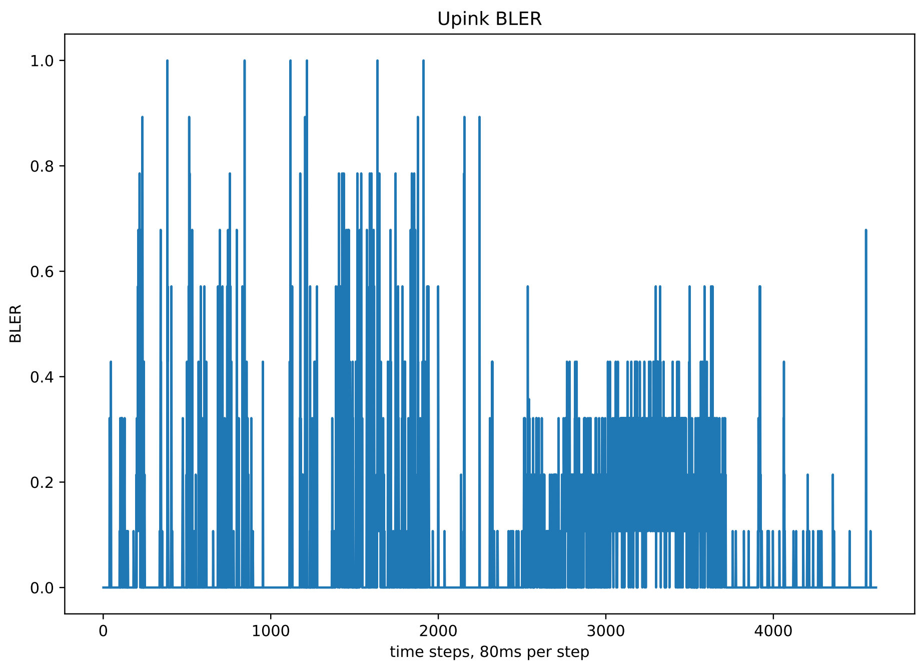


Figure 1. Uplink BLER feature.

## Statistical tests

Statistical tests represent a very important phase of any time series data set manipulation. Applying the statistical tests enables us to check the number of significant lags that we will feed to our models and might affect the choice of the model used in addition to verifying the stationarity of the data which is an essential property for any time series data set. We started by checking whether the distribution of the data follows one of the known distributions such as the normal distribution or the Bernoulli distribution as in that case we will be able to make many assumptions about the data set and apply direct statistical approaches. As we can see in Figure 5, if the distribution of data is following one of the known distributions, the curve should have aligned with the red line, which is not the case for our data. So, we cannot apply direct statistical rules to our data set. We also checked whether the data set is stationary by applying the Adfuller test [64] and the p value was very close to 0 which indicates that the data set is stationary. The zero value of the p value means that there is 0 percent chance of not rejecting the null, so, it is definitely stationary. Given the data is stationary, using only the autoregressive moving average (ARMA) model would be sufficient as in that case we do not need the integration factor represented in the ARIMA model that will be discussed in the following chapter, and we also do not need to transform the data set into a stationary one.

After making sure that the data is stationary, we need to decompose the time series signal into its three components; trend, seasonality and residuals so that we get a better understanding and a visual of the patterns, trends and outliers in the main signal. As discussed previously, We can work on extracting the signal components using the additive or multiplicative approaches. Given that the BLER values usually should be

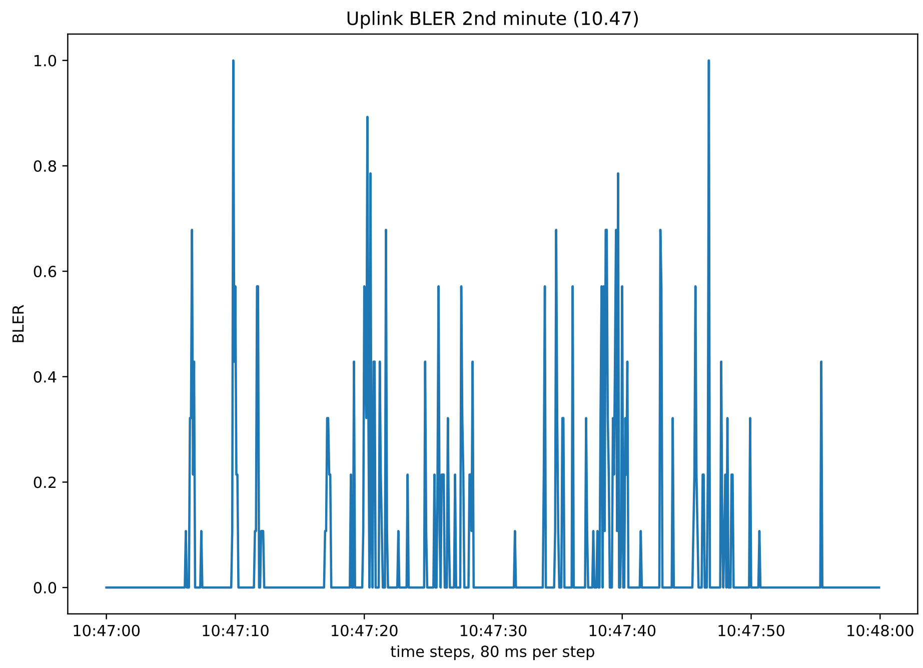


Figure 2. Uplink BLER 2nd minute (10.47).

zero or close to zero, we will use on the additive approach. As we can see in Figure 6, there is no one specific trend in the BLER signal as it has decreasing and increasing trends with different peaks. The seasonality pattern was captured efficiently from the original signal and it shows a repeating pattern. Many residuals or fluctuations were found around the zero value and this is expected because of the anomalous behavior of the data set.

Given that our data set is not labeled, we need to first work on improving the forecasting performance of our models whether they are statistical approaches or machine learning based models. In order for the models to forecast the future values of the BLER accurately, we need to specify how many lags or time steps in the past will affect the future forecasting positively and with what weights for each lags in case the lags have different weights. Checking the auto correlation and partial auto correlation is one of the ways to check the significance of the lags and visualise this significance.

For the partial auto correlation as illustrated in Figure 7, we can see the first ten lags are the most significant ones. The first lag is the main contributor to next time step value and the second lag is the second best contributor. Including all the first ten lags will slightly increase the model performance but it will require more computational resources, training time and inference time. The main objective is to select the best lags that will yield to a good performance while maintaining a good training and inference time. It is important to mention that the partial auto correlation is mainly used as a baseline for the autoregressive-based models.

While the partial auto-correlation graph showed that the first 10 lags are the most significant ones, the auto-correlation graph suggests that the more lags we include the better model performance we will get with the first lag as the most significant lag as

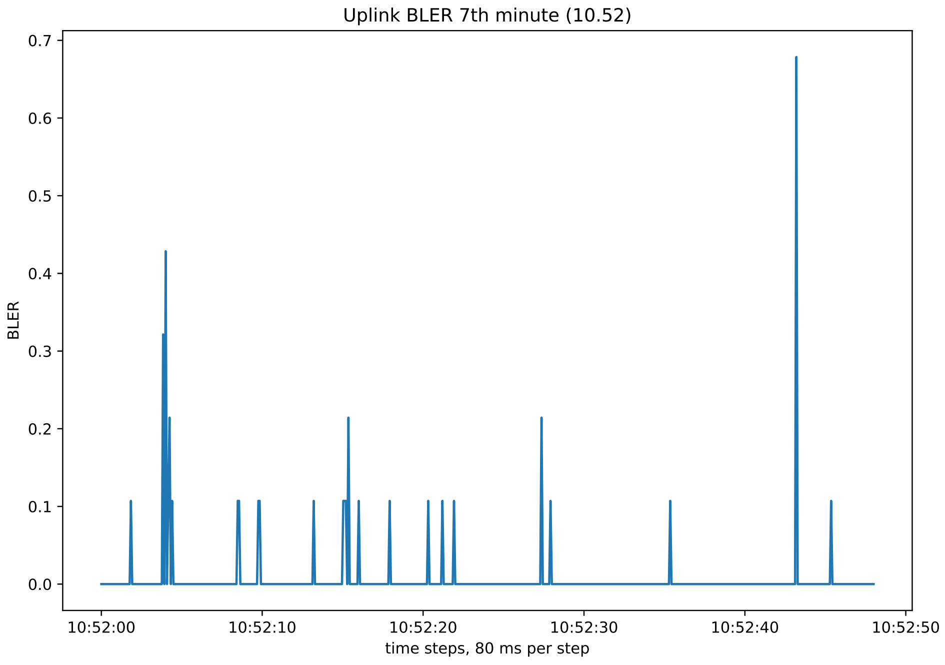


Figure 3. Uplink BLER 7th minute (10.52).

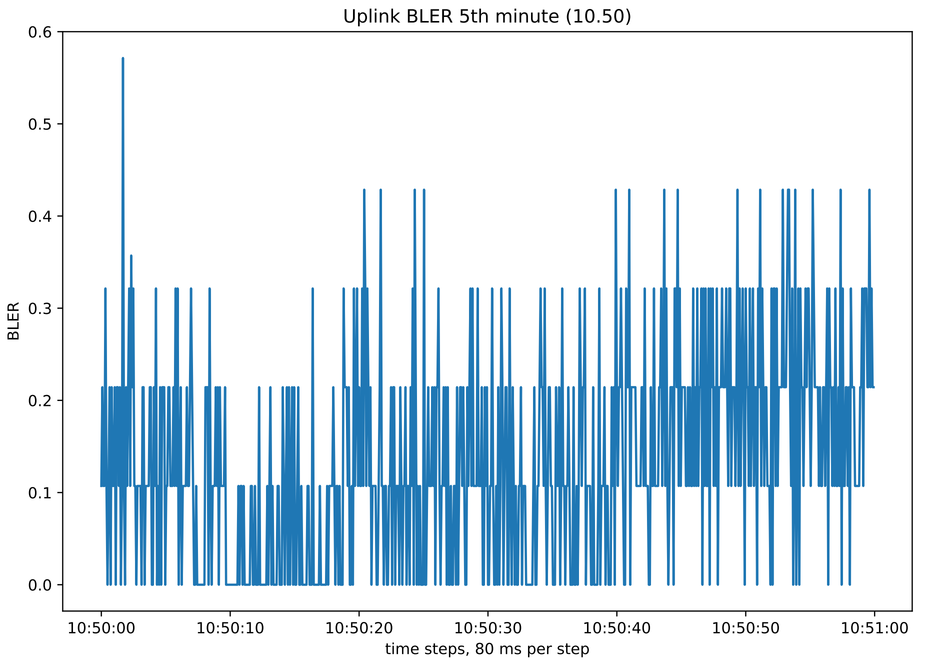


Figure 4. Uplink BLER 5th minute (10.50).

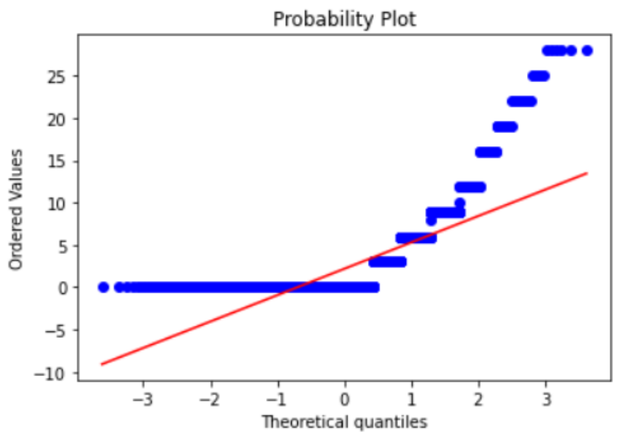


Figure 5. Uplink BLER Q-Q plot.

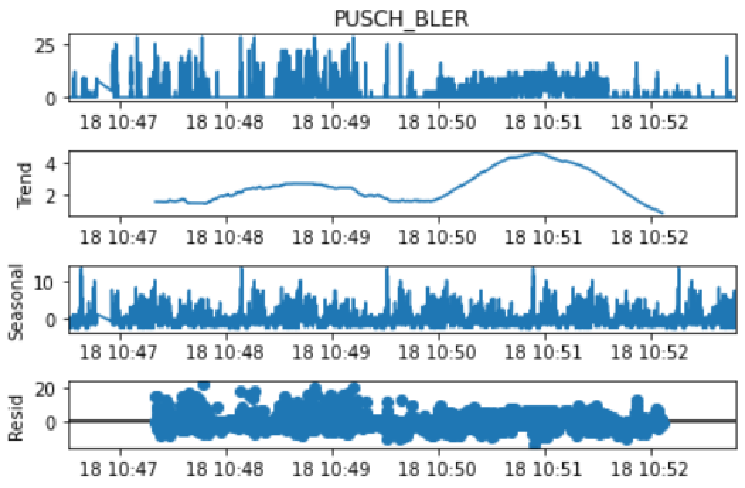


Figure 6. Uplink BLER trend, seasonality and residual components.

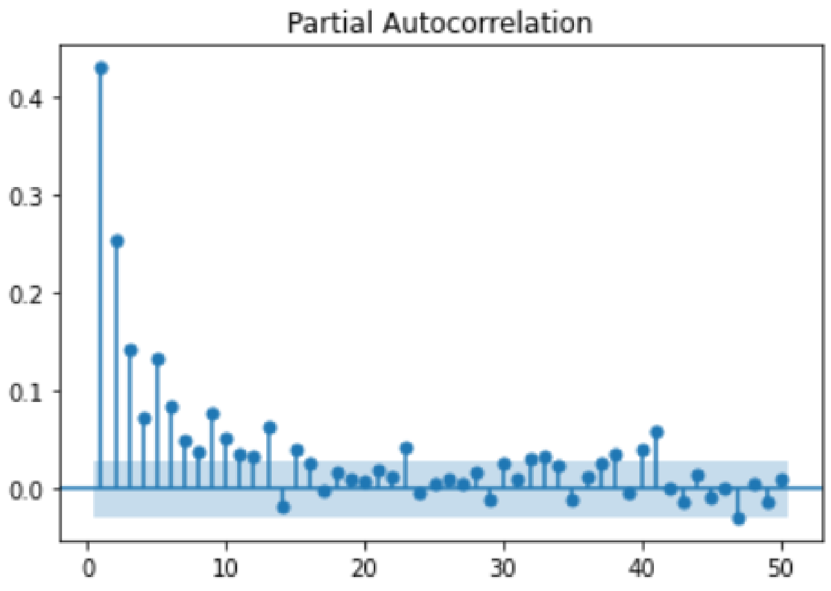


Figure 7. Uplink BLER partial auto correlation.

illustrated in Figure 8. We can also see that the significance of the lags decreases gradually and there is a consistently decreasing trend after the lag 60. It is important to highlight that the auto correlation is used as the main metric to determine the number of the significant lags for the moving average method in addition to the machine learning and deep learning models. Fine tuning the models will be required in all the cases.

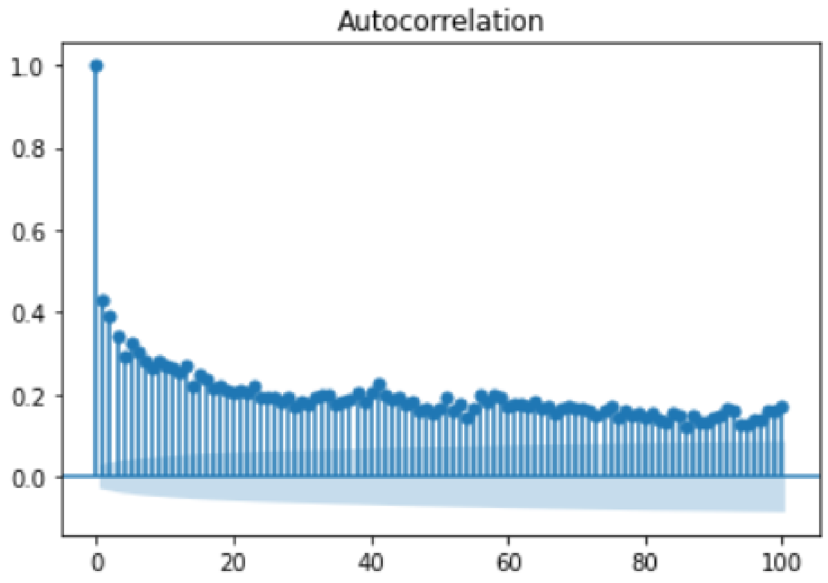


Figure 8. Uplink BLER auto correlation.

### *4.3.2. Downlink*

For the downlink data set, we followed the same steps used for the uplink data set. We started by a basic visualization of the BLER feature as illustrated in Figure 9. As we can see there is a clearer seasonality pattern in the downlink BLER time series feature in comparison to the uplink BLER time series feature. We can also see that there are generally less fluctuations or anomalies than in the case of the uplink. We used the same approach of visualizing each minute of the signal separately to be able to capture any trends, correlation or general patterns.

Some of the new divided data set has clear fluctuations or anomalies as illustrated in Figure 10. Others share a clear seasonality pattern with less fluctuations or anomalies as illustrated in Figure 11 We can also see that the range of the BLER values in Figure 10 is limited in comparison to the range of the BLER values in Figure 11.

After the exploratory visualization of the downlink data set, we checked the correlation between the BLER feature and the other features and there was not a strong clear correlation between the BLER and any of the other features as well. Then, we used the same approach used for the uplink data set and checked the correlation within shorter duration. Some of the features started showing some correlation but again it was not significant enough as it was around 0.3 to 0.4.

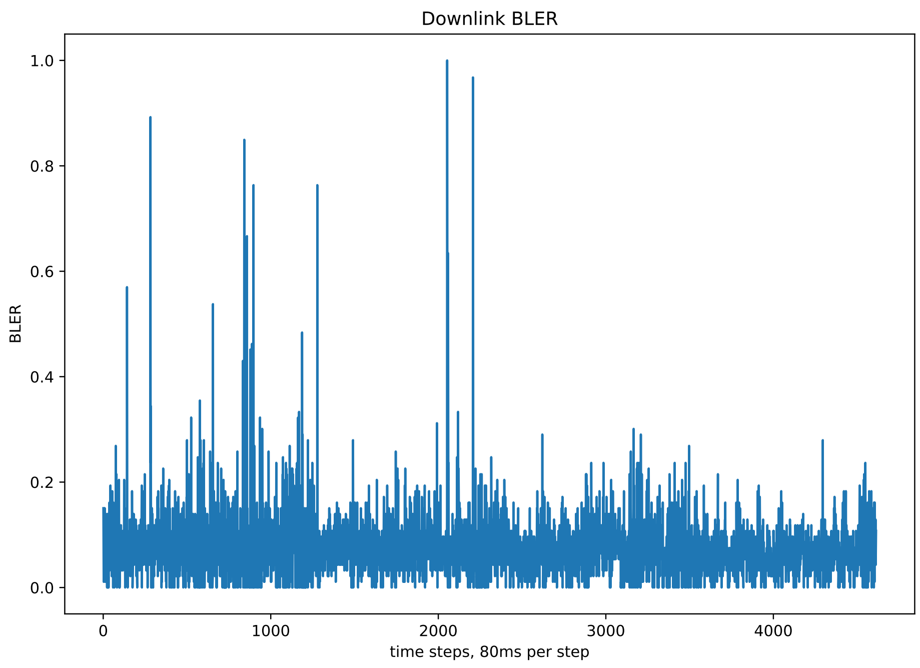


Figure 9. Downlink BLER feature.

## Statistical tests

We again started by checking whether the distribution of the data follows one of the known distributions. As we can see in Figure 12. if the distribution of data is following one of the known distributions, the curve should have aligned with the red line, which is not the case for our data. So, we cannot apply direct statistical rules to our data set. We also checked whether the data set is stationary by applying the Adfuller test and the p-value was very close to 0 which indicates that the data set is stationary.

For decomposing the main downlink signal into its components, we used the additive approach as illustrated in Figure 13. Unlike the uplink case, there is a clear decreasing trend in the BLER signal which can be interpreted in different ways. One way can be that there is some calibration of in the transmission and reception process at the beginning and after this calibration period, the number of ACKs starts increasing gradually and as a result, the BLER signal decreases. The seasonality pattern was captured efficiently from the original signal and it shows a repeating pattern. A very clear seasonality pattern is captured from the signal and it repeats periodically. Many residuals or fluctuations were found around the zero value as for the case of the uplink data set. Despite the stronger seasonality and repeated pattern, more residuals were found in the case of downlink.

For the partial auto correlation and as illustrated in Figure 14, we experimented with two approaches to check the significance of the lags. The first approach is to take the zero offset into consideration. This approach is important when the lags show weak partial auto correlation close to zero. In order to be able to distinguish the values representing the significance of the lags, we have to set the zero offset. Unlike the uplink case, we can see that the very first lags show lower significance than the later

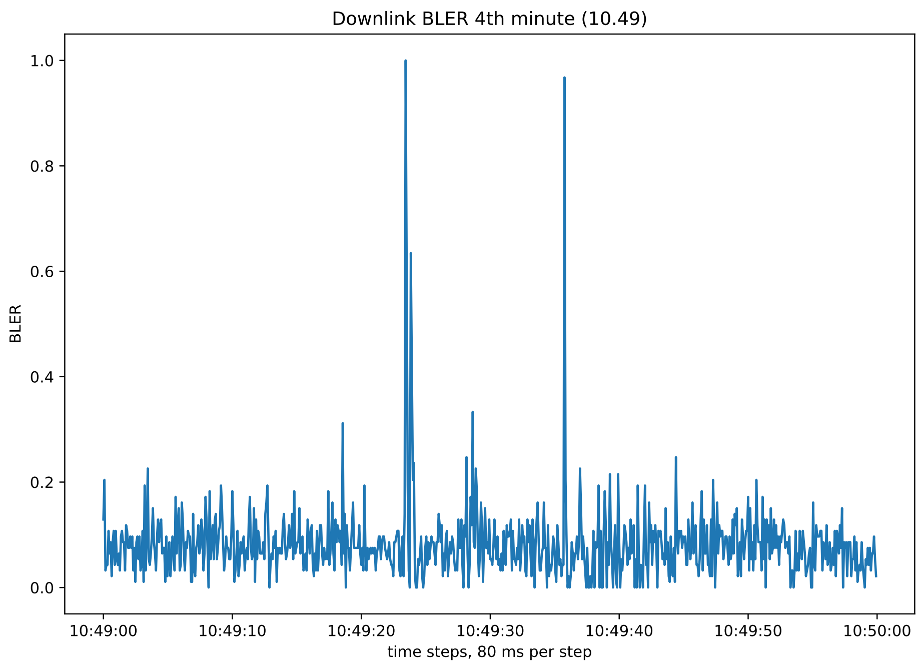


Figure 10. Downlink BLER 4th minute (10.49).

ones. For instance, the fourth lag shows more significance than the first three lags. The lags number nine and 16 shows the highest significance, in terms of affecting the model performance. This approach of testing the lags' significance might not be efficient eventually as all the partial auto-correlation values are very low, the highest ones are around 0.04.

The second approach for checking the lags' significance depends on calculating the ordinary least squares (OLS) [65] between the lags in addition to taking the zero offset into consideration as done in the same approach. OLS is one of the popular methods for estimating the regression parameters or weights in a linear regression model. This approach gives better indication of the significance of the lags as it assumes a basic forecasting model for making future predictions to be able to calculate the OLS which can be used an error metric and a feedback for evaluation the lags' significance. As we can see in Figure 15 the first lag is the most significance one. All the other lags were not significant as their partial auto-correlation value is very close to zero even with setting the zero offset. The output of this approach, the significance of the first lag, can be tested by using any of the statistical approaches for forecasting and evaluating the final output based on any regression metric such as root mean squared error (RMSE) for instance. Another way to evaluate the output of this approach is by calculating the auto-correlation as calculating the auto-correlation should yield similar results.

After experimenting with different numbers of lags to calculate the direct correlation between the lags in the downlink BLER case, we were able to get the same results we got using the OLS approach for calculating the partial auto-correlation between lags. As illustrated in Figure 16, the first lag is the most significant one while all the other

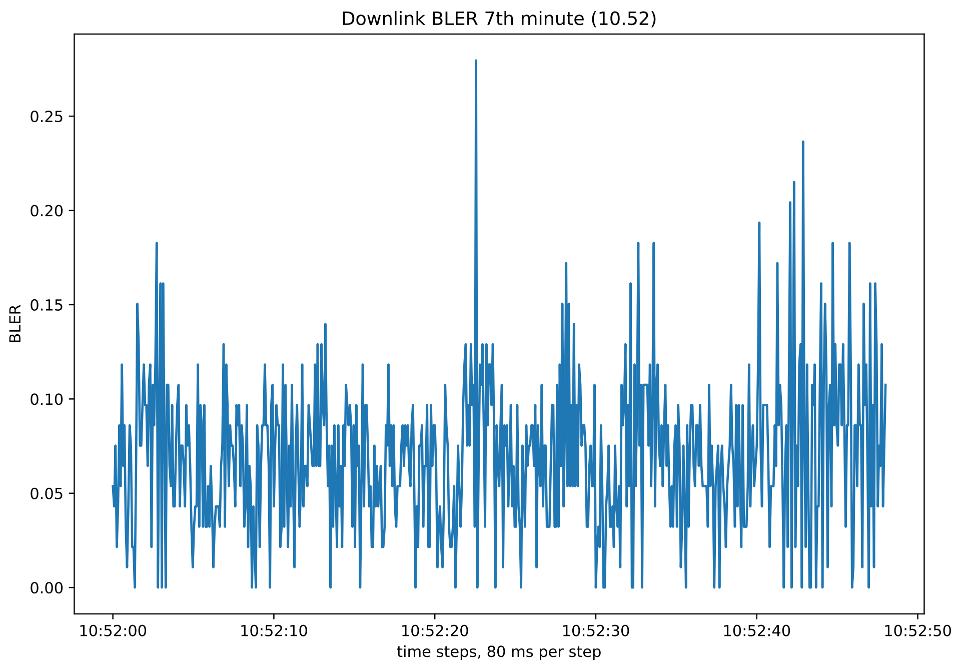


Figure 11. Downlink BLER 7th minute (10.52).

lags have correlation values equal or very close to zero even with setting the offset to zero.

It is important to mention that due to the anomalous nature of both the uplink and the downlink data sets, the data analysis, cleaning and preprocessing can differ because of the existence of the outliers or anomalies which should be handled first in normal case or otherwise we get false insights. Working on normal BLER data set whether for the uplink or downlink cases will give more accurate insights about the feature itself and might help determining the significance of the lags in a more accurate way for better model performance.

## 4.4. Data Analysis Conclusion

During the first phase of the data analysis we cleaned and pre-processed the data set and normalized the BLER features values. After that, we started by dividing the main data set into uplink and downlink ones so that we are able to get more accurate insights about the BLER feature in both cases. Then, we checked the distribution of the data sets and concluded that they do not follow any popular data distributions, which is the expected behavior of time series data sets. Finally, we made sure that the both the uplink and downlink features are stationary and ready for the modeling phase.

Through the second phase of the analysis, we mainly focused on checking the number of significant lags or in other words, previous time steps, that might influence the model performance. This is also important for the labeling technique we used for the forecasting purpose. For instance, if the number of lags selected is 60, that means

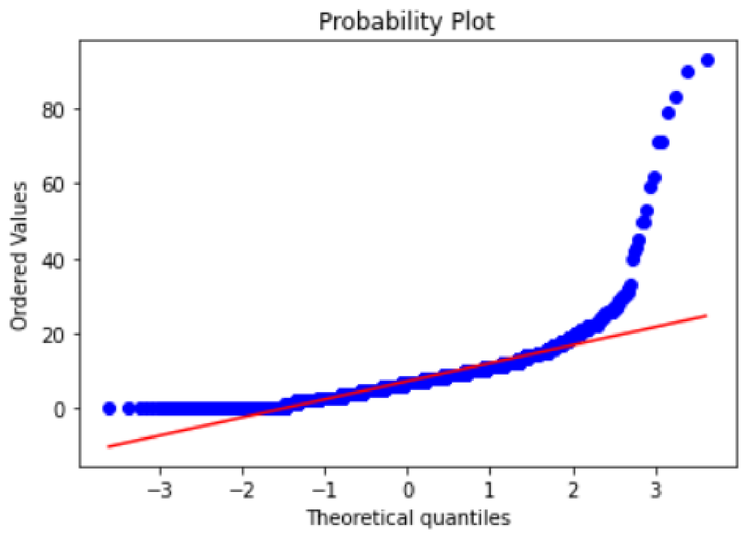


Figure 12. Downlink BLER Q-Q plot.

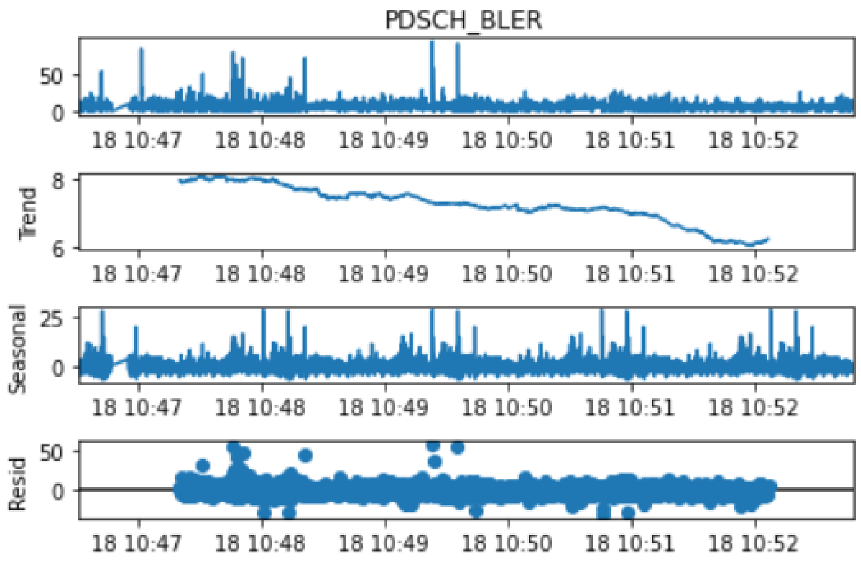


Figure 13. Downlink BLER trend, seasonality and residual components.

we use a sliding window of length 60 time steps and the label of the first 60 time steps is the 61 time step and for the next 60 time step, from two till 61, is the 62 time step and so on. Both the partial auto-correlation and main auto-correlation graphs suggest that the higher the number of lags, the better the model performance. The main objective is to find the focal point that represents the minimum number of lags that yields the best possible model performance. The exact number of lags is to be determined during fine tuning the four models.

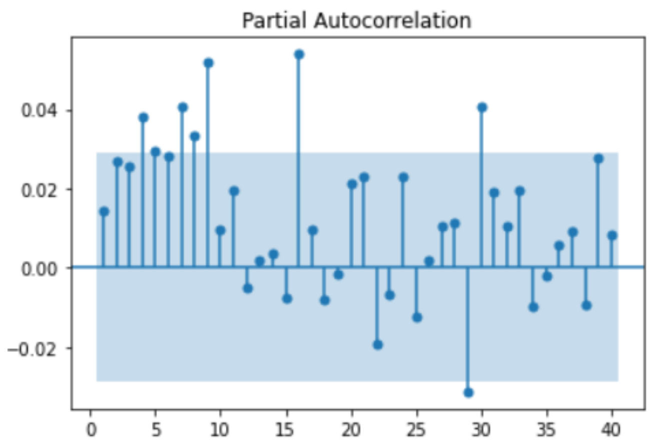


Figure 14. Downlink BLER partial auto correlation with zero offset.

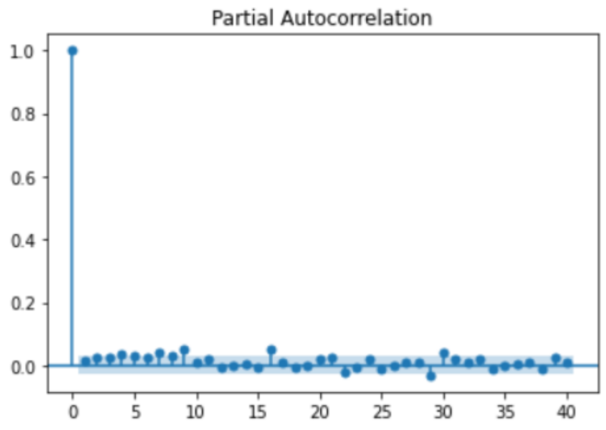


Figure 15. Downlink BLER partial auto correlation using OLS with zero offset.

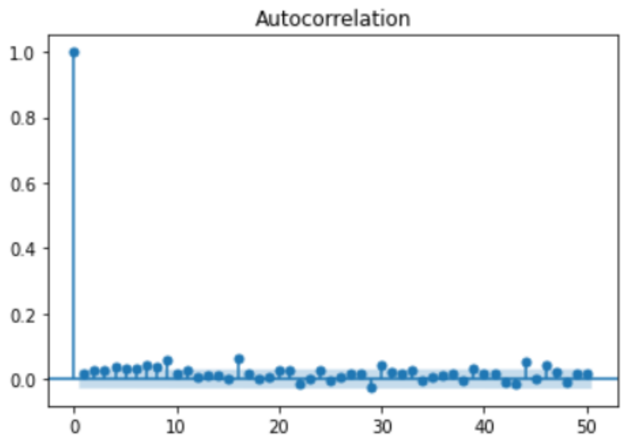


Figure 16. Downlink BLER auto correlation.

# 5. METHODS

In this chapter, we discuss the Four models used for the detecting the anomalies. We explain how these 4 models work and the advantages of using them. The Four models selected are from Four different categories to be able to compare the pros and cons of each category as well. The ARMA model belongs to the statistical approaches category. The isolation forest model belongs to the unsupervised machine learning category. The Extreme Gradient boosting regressor (XGBRegressor) belongs to the supervised machine learning category. Finally, the LSTM model belongs to the deep learning category.

## 5.1. ARMA

The autoregressive model [66] or AR model relies on past period values and uses them only to precise current period values. It is a linear model where current period values are a sum of past outcomes multiplied by a numeric factor

$$X_t = C + \phi + X_{t-1} + \epsilon \tag{12}$$

where $\phi$ is any numeric constant by which we multiply the lagged variable. This $\phi$ coefficient should be between -1 and 1.

The more lags we include in our model, the more complicated the model is. The more complicated the model is, the more coefficients we use. The more coefficients we use the more likely that some of them will not be significant. In general, the model takes into account more data to make a prediction. Auto-correlation function (ACF) captures the direct and indirect effects the previous value has on the current one. Since we want to select the best model, we want to only choose the lags that have significant or direct effects. By normalizing two time series, we compare how they perform to one another. In stationary data sets, ideally, the residuals should follow a random walk process

Generally, the autoregressive model does not handle unexpected behavior or events in the time series signal as it often relies on past data to make predictions. However, there exists a type of self-correcting model which takes past residuals into account. Such models adjust to unexpected behaviors or events very quickly because the predictions are corrected immediately following the error. The more errors we examine, the more adapted our model is to handle unseen errors. These models are known as moving average models or MA models because absorbing unexpected events allows the mean to move accordingly. They do exceptionally well in predicting random walk signals because they make adjustments from the error or previous period. This gives the model prediction a better chance to move in a similar direction to the values it is trying to predict. It also stops the model from greatly diverging and this is very useful in the case of stationary time series data:

$$r_t = c + \theta_1 + \epsilon_{t-1} + \epsilon, \tag{13}$$

where $\theta_1$ is a numeric coefficient for the value associated with the first lag. We can say that a simple moving average model or MA(1) is equivalent to an infinite auto-regressive model with certain restrictions. The main difference between the auto-regressive model and the moving average model is that the moving average model relies on the residual values while the auto-regressive model relies mainly on the variable values themselves. The absolute value of the coefficients in both the models' cases should be less than one as a restriction. This restriction is to prevent compounded effects from exploding in magnitude.

Also, both the models have some key differences. One such distinction comes in the form of determining the maximum number of lags we are willing to include in our model. While in the autoregressive model, we relied on the partial auto correlation function, with moving average, we relied on the auto correlation function. Determining which lagged values have a significant direct effect on the present time step value is not relevant in our case. The combination of both the models result into the ARMA model that share the strength points and overcome the negative side of both them. After choosing the best number of lags, or the degree of the model complexity, for both the AR and MA model separately, we apply fine tuning again for the final ARMA model. For example, if we have AR(4) and MA(3), the best ARMA model is not necessarily ARMA(4, 3) but it is expected to be of a lower degree or otherwise the new ARMA model will be of degree 7, which adds unneeded complexity for the model to perform well.

## 5.2. Isolation Forest

Isolation forest [67] is one of the unsupervised machine learning approaches for detecting anomalies in time series data sets. As the name implies, it isolates the anomalies in the data sets from the normal data points. The forest part of the algorithm name is due to the binary trees mechanism and the combination of these trees together forms the forest. Because of the nature of anomalies, it is expected that the isolated anomalies will be closer to the root of the trees. In isolation forest, the data is processed in a tree structure based on the time series signals. The sample points that go deeper into the tree structure are the normal points as it is hard for the algorithm to separate them based on the nature of their values.

As stated earlier, the isolation forest is basically an ensemble to decision trees. These are more detailed steps of how the algorithm actually works:

1. A random sample of the data is assigned to a binary tree.
2. Branching start randomly based on a randomly selected value or threshold within the range of the minimum and maximum values of the target.
3. If the current point value is less than the selected threshold, the point goes to the left of the partition of the tree structure and if it is higher, the point goes to the right partition of the tree structure.
4. This process continues recursively till we have all the points isolated or until we reach a pre-defined maximum depth of the tree.

5. The whole steps are repeated again for new binary trees.

After the whole steps are completed, an anomaly score is calculated for each point based on how deep this point traveled in each tree, and then, the final anomaly score of each point is an ensemble based on the contamination value. The contamination value represents the percentage of outliers in the data set and it is one of the parameters that need to be defined for the algorithm beforehand. An anomaly score of -1 is assigned to the anomalous points and an anomaly score of +1 is assigned to the normal points. Based on the steps explained previously, the algorithm does not take the time series sequence into consideration as it selects the points and the threshold values randomly and the timestamp feature is the main feature that distinguishes the time series features from any normal data features.

Ding and Fei [68] extended the main algorithm and uses a sliding window to be able to take the time series sequence feature into consideration. Now, the anomaly score is computed for each sequence and it is proportional to the average path length that the sequence travels within the binary trees.

$$S(x, w) = 2^{-\frac{E(h(x))}{c(w)}} \tag{14}$$

$$E(h(x)) = \frac{1}{L}\Sigma_{i=1}^{L} h_i(x) \tag{15}$$

where $h_i(x)$ denotes the length of the i-th iTree, $E(h(x))$ the average of $h(x)$ from a collection of iTrees and $c(p)$ is the average of $h(x)$ given $w$ and $L$, the number of iTrees.

The extended version of the algorithm has three main parameters that need to be defined or tuned:

- Window length
- Number of binary trees in the forest
- Contamination value.

Too short window length causes the loss of important information about the time series sequence which was the main reason for extending the algorithm functionality. On the other hand, too long window length might include less relevant information which can confuse and affect the model performance in a negative way. The higher the number of the binary trees, the closer the average value of the estimated value but on the other hand, this increases both the time and space complexity of the algorithm and as a result, the computational resources required for the training and inference phases. Contamination value or the percentage of anomalies in the data set: as discussed earlier, this value should be pre-defined as it affects the model performance significantly given that the algorithm is an unsupervised machine learning algorithm and hence, we do not have the data labeled. The contamination value can be determined in different ways and might require the help of domain experts.

## 5.3. XGBoost

XGBRegressor [69] is a supervised machine learning algorithm that is used for regression-based problems for forecasting or prediction purpose. XGBRegressor uses gradient boosted trees and we refer to them as the number of estimators when using the model. The boosting mechanism is done based on the combination of decision trees. The number of lags in XGBRegressor should be taken into consideration as they basically form the labels of the data sets. Depending on the data set, the number of lags can be only one or higher. In our case, we experimented with 5, 60, and 100 lags based on the auto-correlation and partial auto-correlation graphs to be able to determine the most significant number of lags that will contribute to better model performance.

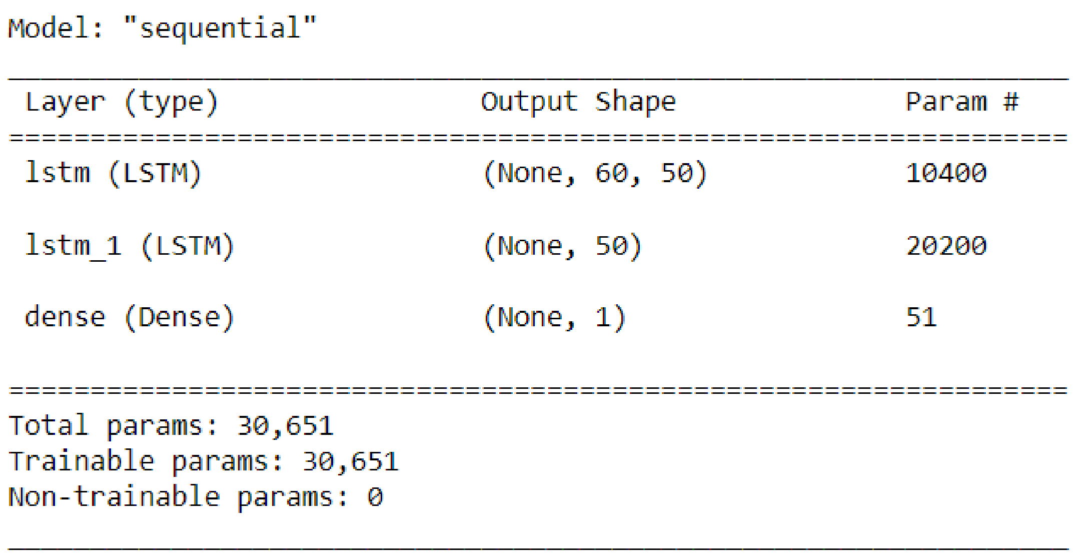
After using the model for the prediction purpose. We apply a threshold error based on the assumption of the percentage of anomalies in the data set and assign different flags for each assumption.

## 5.4. LSTM

As discussed earlier, LSTM has the advantage of learning the patterns on long time series sequences because of its different gates functionalities which enables it to discard the unnecessary information and learn the important patterns. After experimenting with different architectures, we found that deep architectures tend to overfit the data set pretty easily and requires more work for fine tuning the hyper-parameters, add regularization to overcome the over-fitting problem in addition to the increased time of training and inference and as a result the computational resources.

On the other hand, experimenting with shallow architectures that have three or less layers results yield almost the same or close to the performance we got from the deeper models. Using simple architectures helped in decreasing the training and inference time exponentially in addition to the time required for fine tuning the model hyper-parameters and experimentation. Generally, one of the disadvantages of using shallow networks is the problem of under-fitting but in our case this was not a problem as the model was able to detect the patterns and generalize for the test set as well.

For the loss function we used a basic mean squared error metric and for the optimizer we used adaptive moment estimation (Adam) [70] optimizer as it has the advantage of using adaptive learning rate and performs generally well on sparse and time series data set most of the time. We used a batch size of four and trained the model for only one epoch. Training the model for more epochs gets slightly better results but takes much longer time and computational resources. The exact architecture we used is illustrated in Figure 17. We used only one hidden layer in addition to the input layer and dense (fully-connected) layer. We did not need to add any dropout layers because of the simplicity of the model and as a result, no need for regularization.



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 60, 50)            10400

 lstm_1 (LSTM)               (None, 50)                20200

 dense (Dense)               (None, 1)                 51

=================================================================
Total params: 30,651
Trainable params: 30,651
Non-trainable params: 0
_________________________________________________________________
```

Figure 17. LSTM architecture.

# 6. RESULTS

For all the models selected we divided the data set into train and validation sets. The whole data set was almost 7 minutes length and after some experimentation, the first 5.5 minutes were assigned to the train set and the rest to the validation set without any shuffle as it is important to maintain the time series sequence. Only the BLER target with the timestamp features were used through the modeling phase without any other features. The reason behind that is that some of the models expect only the time series target such as the ARMA one. In order to compare and benchmark the results, we need to make sure that all the models have the same input. The inference and the evaluation were done on the validation data set. The following graphs for models' prediction and anomalies detection representing the output over the whole data sets, uplink and downlink ones. We used the trained weights for forecasting over only the validation set. One of the reasons of choosing a shallow LSTM architecture as shown in Figure 17 is to overcome the overfiting the data in addition to the constrain of the time and computation resources. We made sure that the model is not overfitting and by monitoring the validation loss for longer time and we can see that there is not overfitting as illustrated in Figure 18.

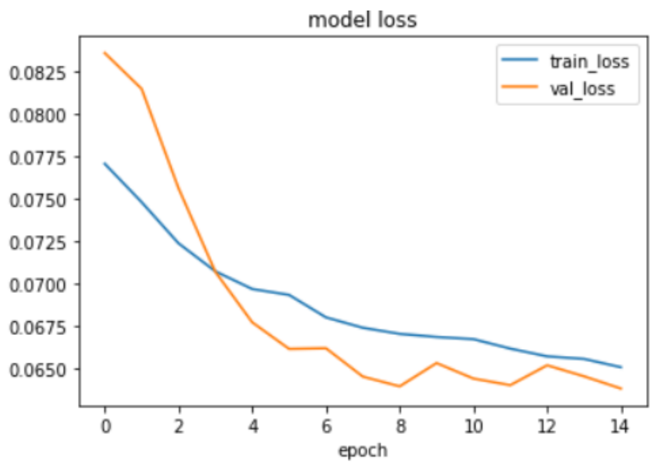


Figure 18. LSTM train and valid loss for 15 epochs.

## 6.1. Anomalies Flagging Methodology

The anomaly detection approach we followed to overcome the unlabeled anomalies is based a flagging technique and the assumption of the percentage of anomalies in the data set. After we forecast the output of the time series target, we calculate the error between the models' prediction and the actual values of the BLER target. Then, we sort the error values while saving their original indexes so that we are able to restore the time series sequence back using the saved indexes. Based on the assumption of the percentage of anomalies in the data set, we select the points with the highest error as anomalies and project them back into the original time series signal using their saved indexes. Finally, we use the flagging technique to determine the severity of those anomalies. If we have only 2% of anomalies in our data set, these points will be the ones labeled with the red points. If we have 5% of anomalies in the data set, these points will be the ones labeled with the red points in addition to the ones labeled with the yellow points.

## 6.2. Uplink

### *6.2.1. ARMA*

After the experimentation with different lags for both the AR and MA models, we found that the best model results are based on the model ARMA(4, 4). In Figure 19 we can see that the model was able to detect the general pattern in the data set and forecast the BLER values accurately but the main challenge is that the data set is anomalous and this results into the model learning the anomalous behavior of the BLER signal as well which could be solved by training the model on only a data set with normal BLER values as discussed earlier.

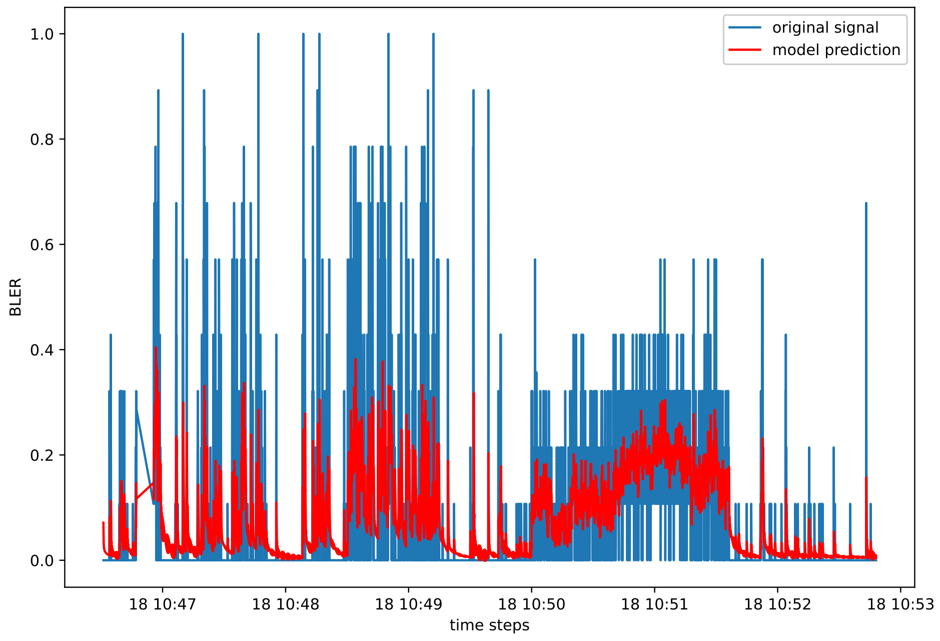


Figure 19. ARMA(4, 4) model prediction for the uplink BLER signal.

Given that we have the forecasted output, the next step is to calculate the error based on a statistical threshold and the assumption of the percentage of anomalies in the data set. As illustrated in Figure 20, we can see that the distribution of the error is centered around the zero value which matches our knowledge about the default or normal BLER values.

As illustrated in Figure 21, we can see that the model labeled some of the zero BLER values as anomalies by mistake and this is because of the nature of the data set.

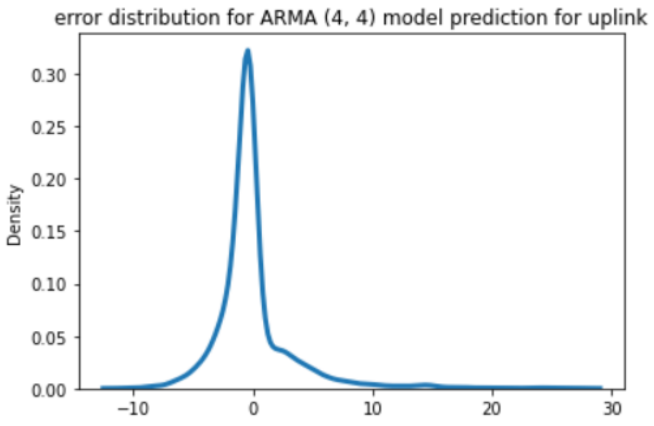


Figure 20. Error distribution for uplink BLER forecasting.

### *6.2.2. Isolation Forest*

As discussed previously, one of parameters that influences the isolation forest performance is the n estimator. After experimentation with different values of the n estimator hyperparamter, we found that using 20 estimator yields the best performance as increasing the number of n does not contribute to enhancing the model performance but just adding more complexity to the model. As illustrated in Figure 22, we can see that the dominant percentage of the BLER values are equal or close to the 0 value.

Given that the contamination value is the second main parameter that needs to be given to the model for accurate results, we used the same percentages we used previously for the other models to be able to compare and benchmark the results. As illustrated in Figure 23, given the 5% of anomalies, we can see that the model selected all the BLER points with values that match the contamination value as anomalies.

### *6.2.3. XGBoost*

We generally start by a low number of lags and start adding complexity to the model depending on the suggested number of lags we got during the data analysis phase. Following that approach, we found that using only five lags yields the best model performance as adding more lags do not contribute significantly to enhancing the model performance. As illustrated in Figure 24, we can see that the model was able to detect the general pattern in the data set while failing to recognizing the anomalies pattern. Most of the values forecasted were close to 0 but not actually 0 and this is due to the very small number of lags selected that gives higher weight to the very previous time steps regardless of whether they are anomalous or normal points. We experimented with very large number of of lags such as 100 and 200 and found that this yields worse model performance.

As illustrated in Figure 25, and given both the assumptions of 2% and 5% anomalies in the data set, we can see that model failed to detect some anomalies while assigning the same severity degree to the anomalies detected in both the cases of 2% and 5% anomalies.

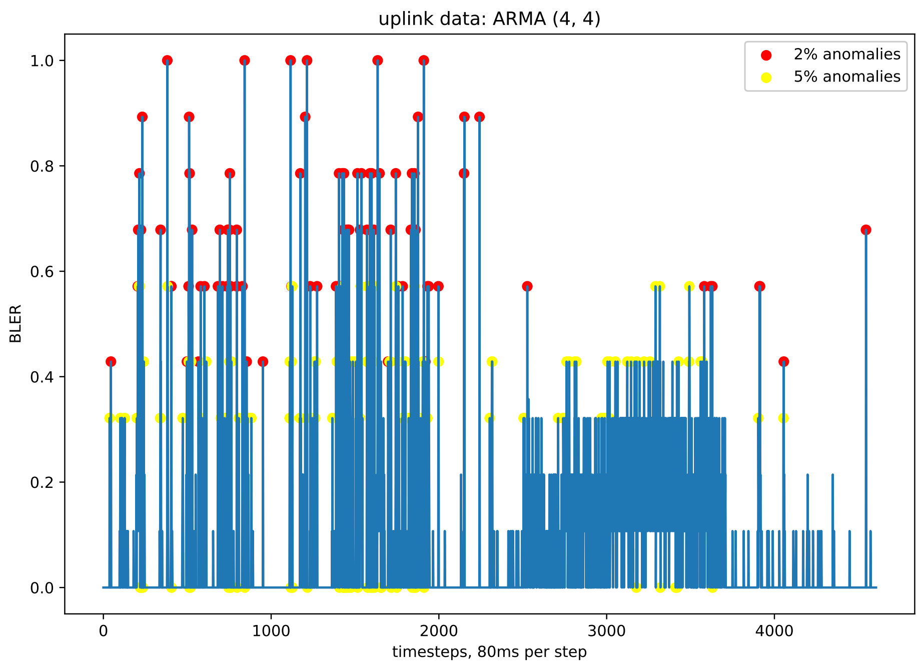


Figure 21. ARMA(4, 4) anomalies for uplink BLER.

### *6.2.4. LSTM*

Using the same approach used before for selecting the best number of significant lags, we found that using 60 lags yield the best LSTM model performance as illustrated in Figure 26. The model was able to forecast low BLER values while retaining the general pattern in the data set. Some fluctuations were forecasted as normal values.

As illustrated in Figure 27, we can see that the model was able to isolate the anomalies given the assumptions of percentage of anomalies in the data set. Identical BLER values were isolated in different cases based on the percentage of anomalies

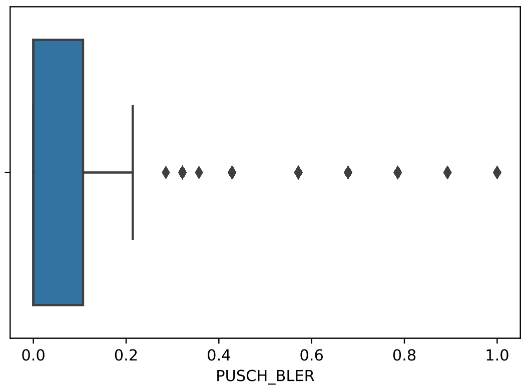


Figure 22. Uplink BLER box plot representation.

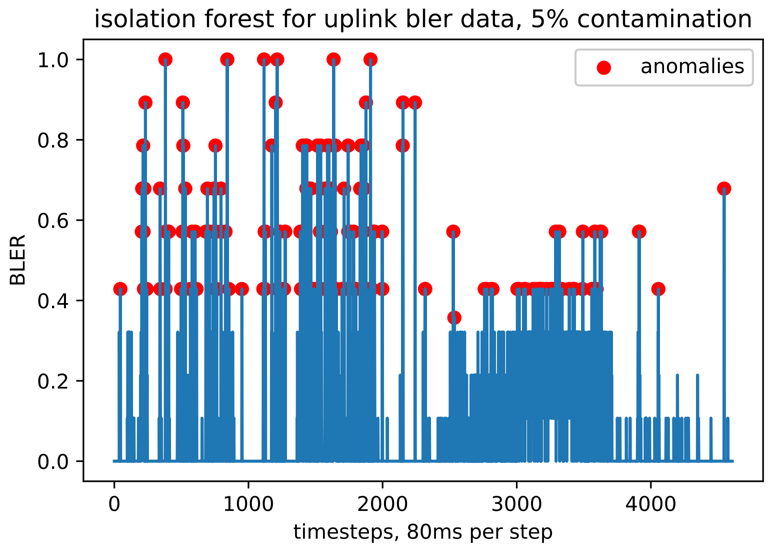


Figure 23. Isolation Forest for uplink BLER, contamination of 5.

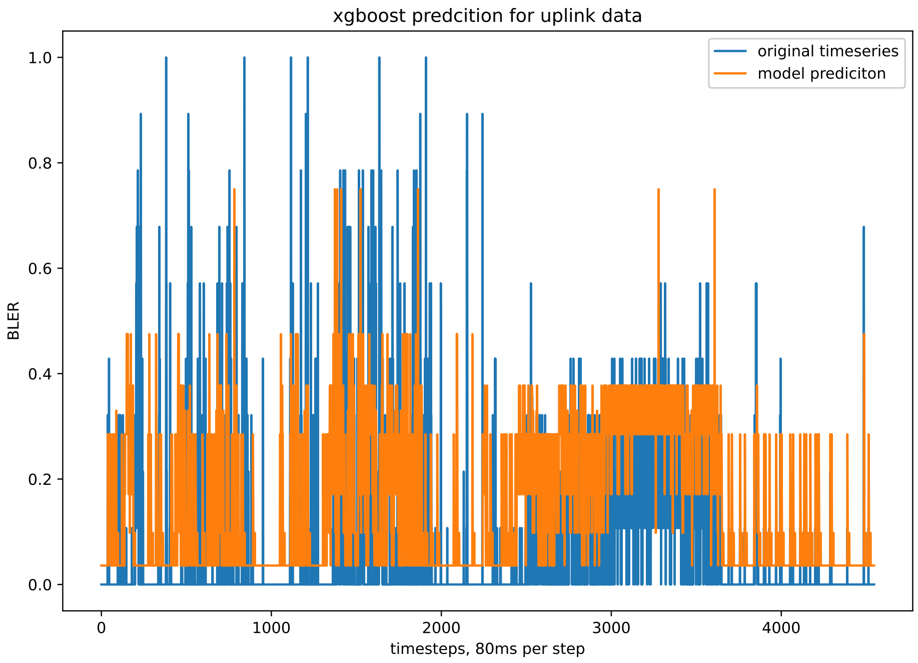


Figure 24. XGBRegressor uplink BLER forecast.

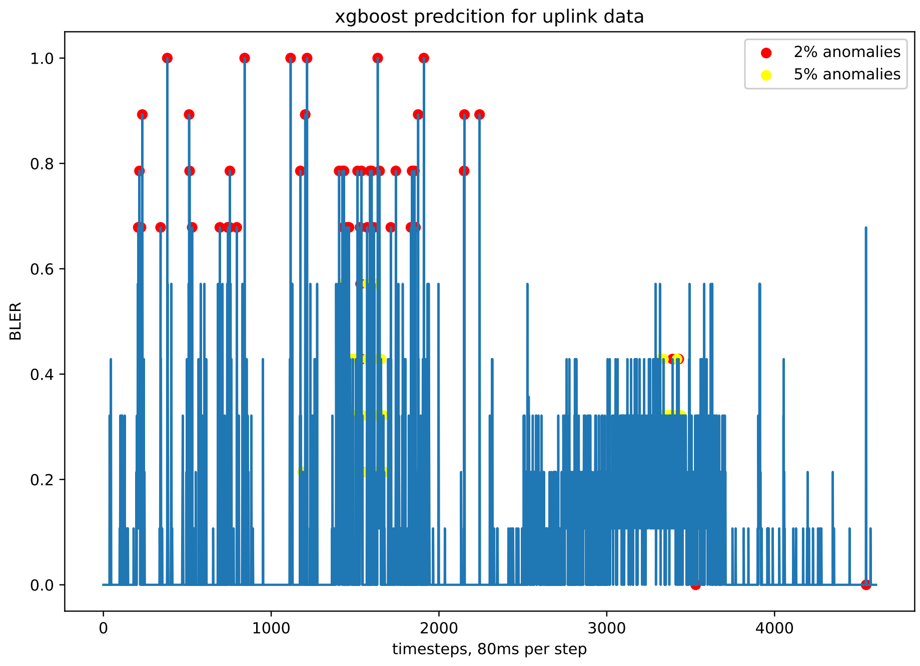


Figure 25. XGBRegressor uplink BLER anomalies detection.

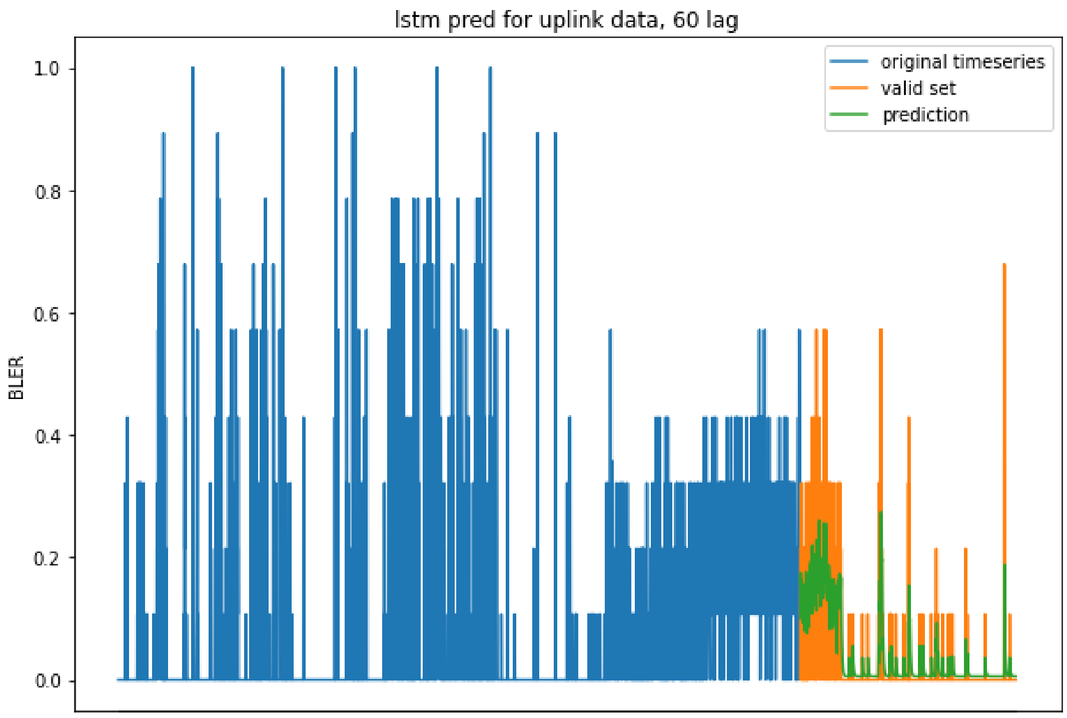


Figure 26. Uplink BLER LSTM prediction.

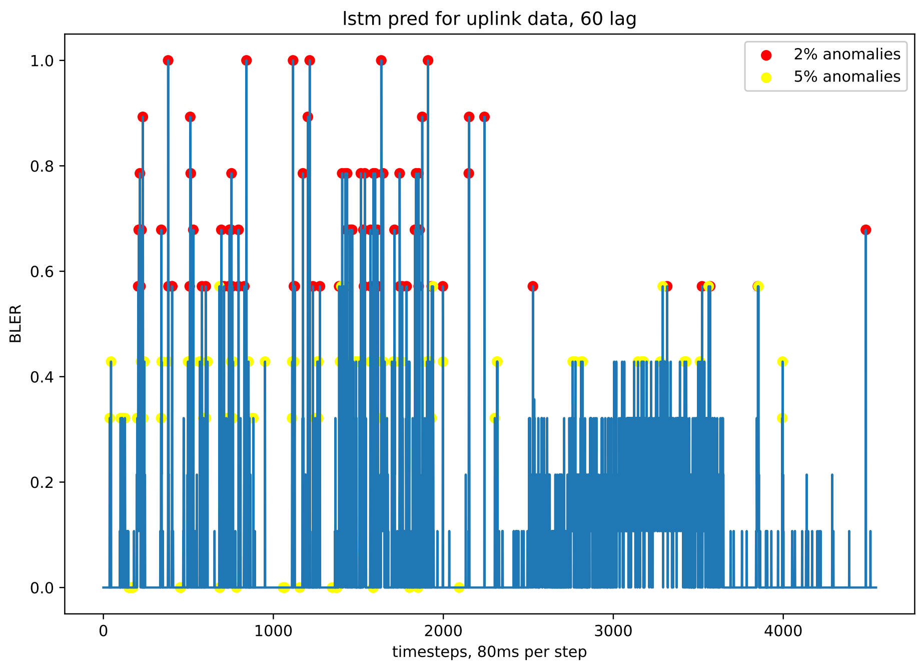


Figure 27. Uplink BLER LSTM anomalies detection.

which indicates that the model is learning how to distinguish the anomalies based on their pattern and not just the values.

### 6.3. Downlink

#### *6.3.1. ARMA*

Like the uplink case, we started by experimenting with different lags for both the AR and MA models and found that the best model results are based on the model ARMA(4, 3). In Figure 28 we can see that the model was able to detect the general pattern in the data set and forecast the BLER values accurately. Unlike the case of the uplink, the model forecasting is closer to the zero values of BLER as the downlink data set has more stable patterns with less fluctuations.

After forecasting, we used the same approach used previously to calculate the error. As illustrated in Figure 29, we can that the distribution of the error is centered around the zero value as well but the tail of the distribution is larger this time which indicates that the forecasting results are more accurate so that the anomalies are isolated at the tail of the distribution.

Through the final phase of the error values manipulation and as illustrated in Figure 30, we can see that most of the fluctuations were labeled as anomalies and the higher the values of the fluctuations, the faster they are isolated. Again the model labeled some of the zero BLER values as anomalies by mistake because of the reason discussed earlier.

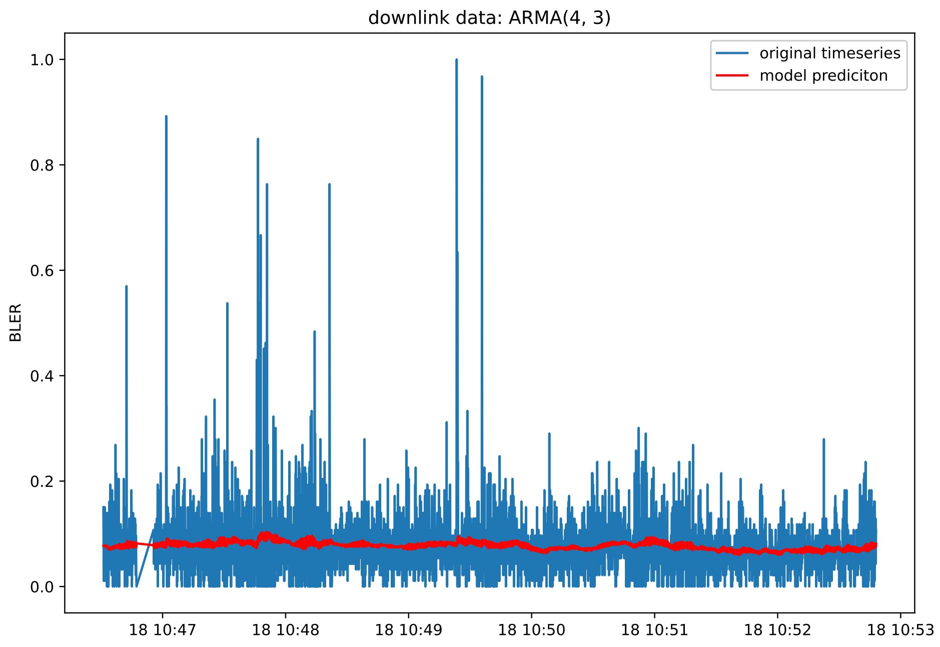


Figure 28. ARMA(4, 3) model prediction for the downlink BLER signal.

### *6.3.2. Isolation Forest*

Despite the difference of BLER values in the uplink and downlink data sets, we found that using n of 20 yields the best performance as in the case of uplink. As illustrated in Figure 31, we can see that the dominant percentage of the BLER values are forming one cluster together. After training the model and assigning the anomaly score based on the contamination value we can see the anomaly scores distributed over the original data set values as illustrated in Figure 32.

For the contamination value, we used the same percentages we used previously for the other models to be able to compare and benchmark the results. As illustrated in Figure 33, given the 5% of anomalies, we can see that the model selected most of the BLER points with values that match the contamination value as anomalies.

### *6.3.3. XGBoost*

Unlike the uplink case, we found that using only one lag yields the best model performance. As illustrated in Figure 34, we can see that the model was able to detect the general pattern but we can also see that the model predicted some fluctuations in the BLER values as normal values. The general range of the forecasted BLER values is low and stable which indicates that the model is learning the pattern.

As illustrated in Figure 35, we can see that the model failed to detect some fluctuations as anomalies. We can also see that the model detected a couple of 0 BLER

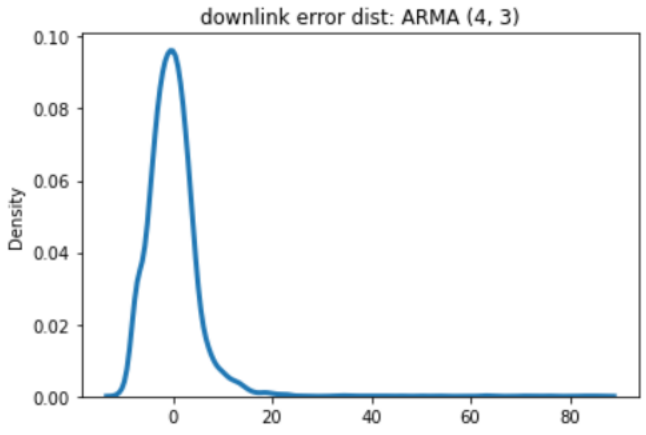


Figure 29. Error distribution for downlink BLER forecasting.

values, the normal value, as anomalies in both the assumption cases of 2% and 5% of anomalies in the data set.

#### *6.3.4. LSTM*

After experimenting with different lags, we found that using 100 lags yield the best LSTM model performance as illustrated in Figure 36. The model forecasted stable values of BLER that are close to the normal behavior of BLER values but the forecast still deviates from the default values of BLER, the zero values.

By getting a closer look at the prediction of the LSTM model over the validation set as illustrated in Figure 37, we can say that the LSTM mode is the closest to learning the expected normal behavior of the BLER feature despite of the anomalous data set. The model was able to learn the stable pattern with values close to zero and discarded all the fluctuations or anomalies in the BLER signal.

As illustrated in Figure 38, we can see that the model was able to isolate the anomalies given the assumptions of percentage of anomalies in the data set. Identical BLER values were isolated in different cases based on the percentage of anomalies which indicates that the model is learning how to distinguish the anomalies based on their pattern and not just the values.

## 6.4. Evaluation

As discussed earlier, only the validation set was used to evaluate the models' performance. In the case of the labeled data sets, supervised learning process, we tend to use classification metrics. On the other hand, we tend to use the regression based metrics in case of the time series forecasting or prediction problems and in case of unlabeled anomalies as well, the case of our work. One of the most popular used regression metric is the root mean square error because of its simplicity to calculate and its efficiency to indicate whether a regression model is performing well. RMSE is calculated as follows:

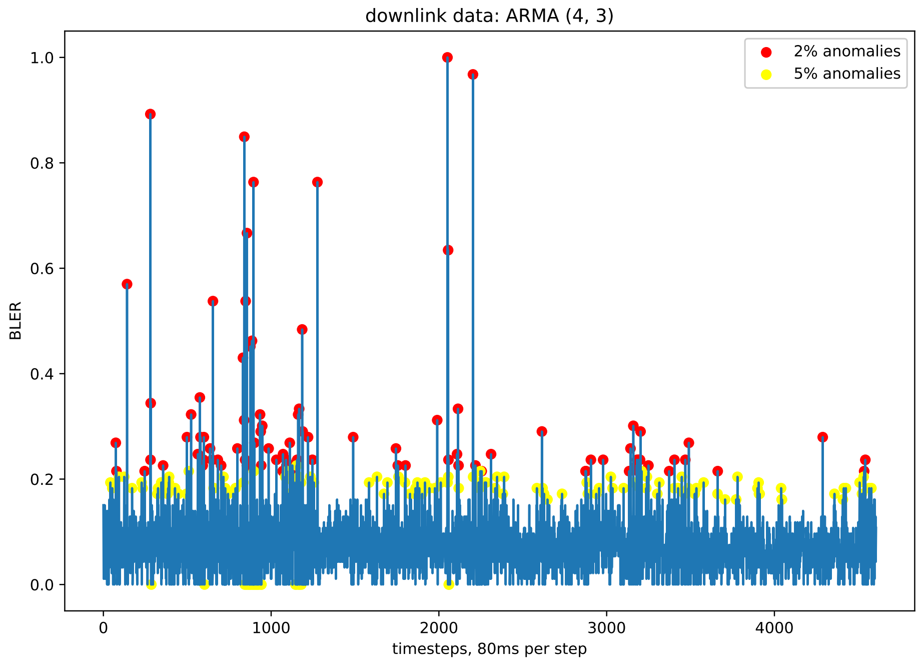


Figure 30. ARMA(4, 3) anomalies for downlink BLER.

$$RMSE = \sqrt{\frac{1}{n}\Sigma_{i=1}^{n}\Big(y_i - f_i)^2}, \tag{16}$$

where $n$ is the number of samples we have in the data set, $y_i$ is the true label value and $f_i$ is the predicted target value. We will be using RMSE as our main evaluation metric for benchmarking the supervised learning based-models' results.

We used RMSE for all the models except for the Isolation Forest model as it is an unsupervised machine learning model as shown in Table 1. The evaluation of the isolation forest model was based on the anomaly score as illustrated in Figure 32 and we can see that the anomalies were assigned higher anomaly score than the case of

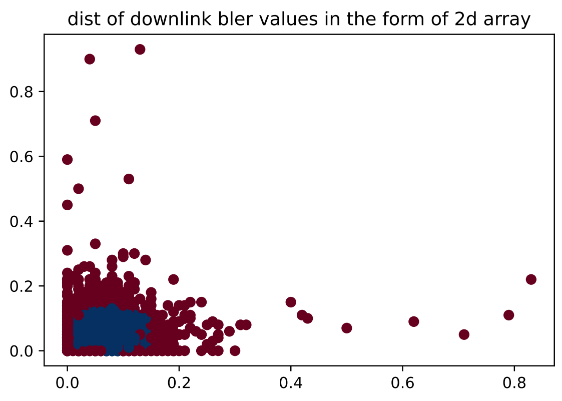


Figure 31. Downlink BLER 2D distribution in 2d array.

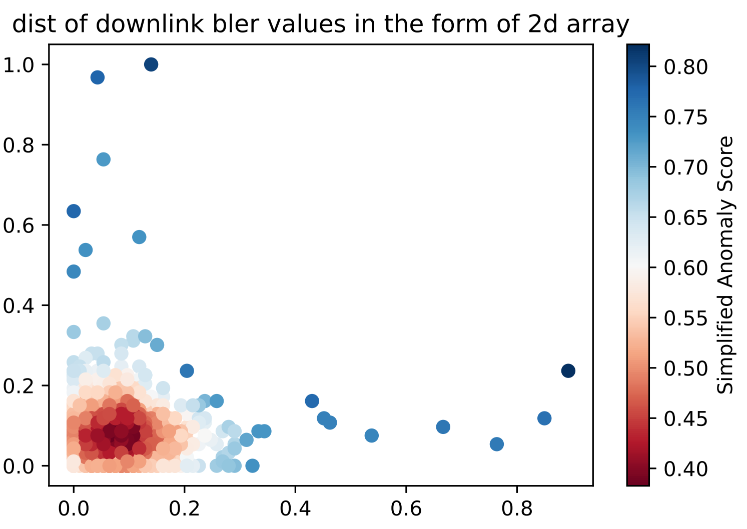


Figure 32. Downlink BLER anomaly score distribution for isolation forest.

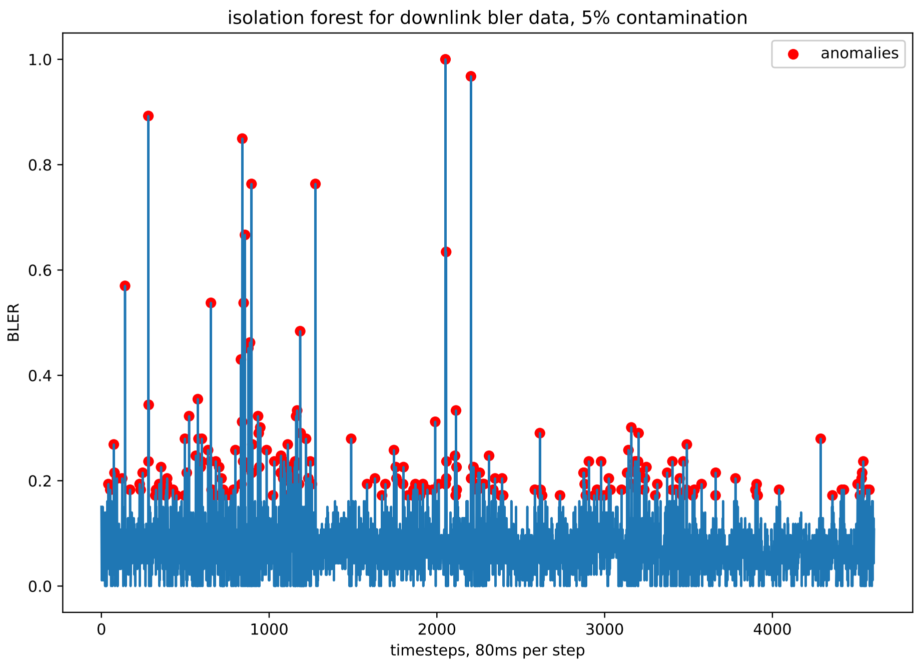


Figure 33. Isolation Forest for downlink BLER, contamination of 5.

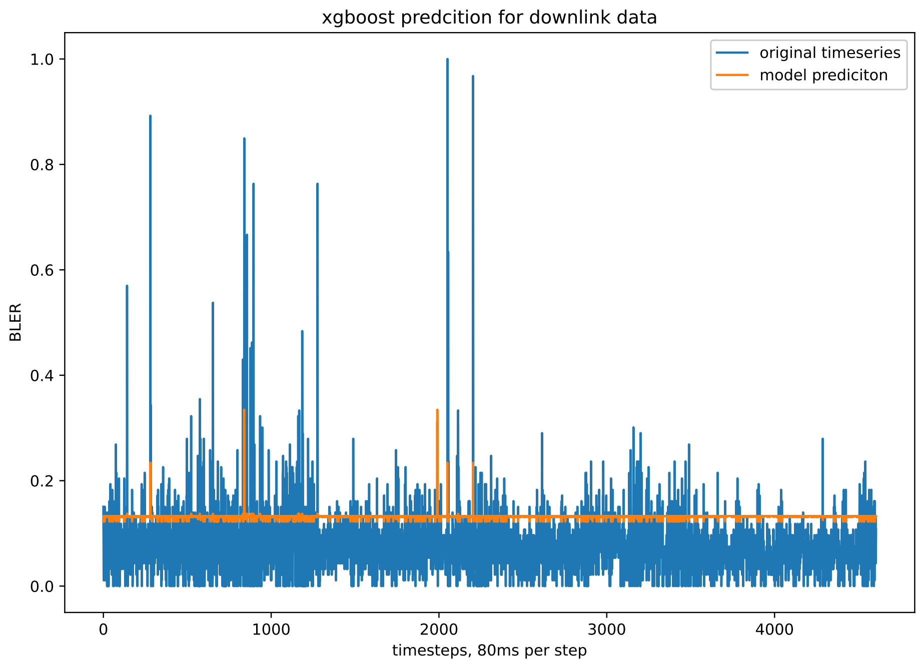


Figure 34. XGBRegressor downlink BLER forecast.

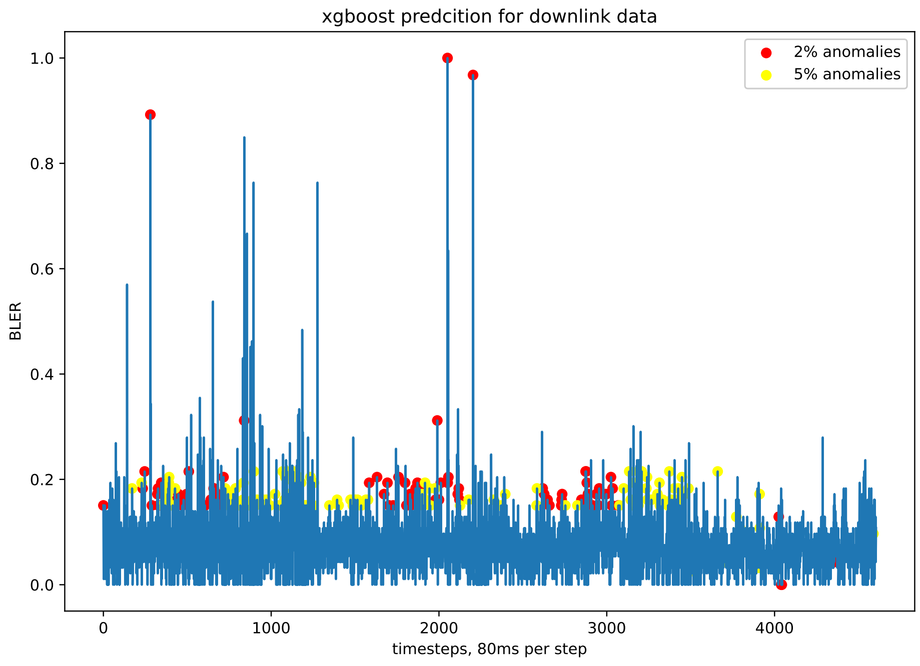


Figure 35. XGBRegressor downlink BLER anomalies detection.

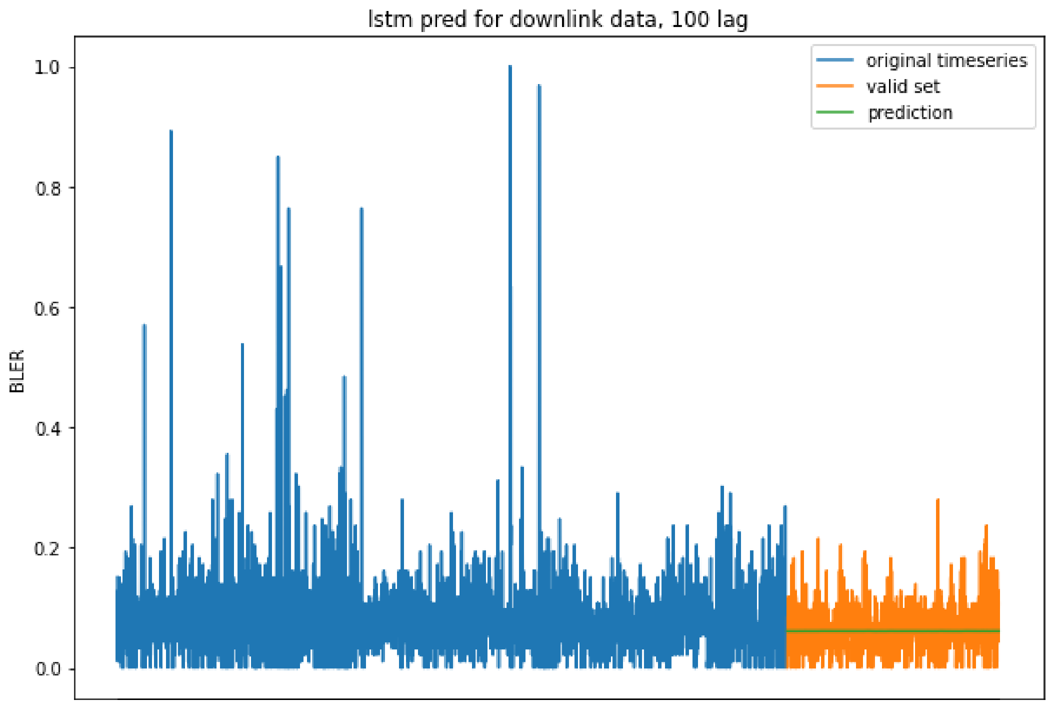


Figure 36. Downlink BLER LSTM prediction.

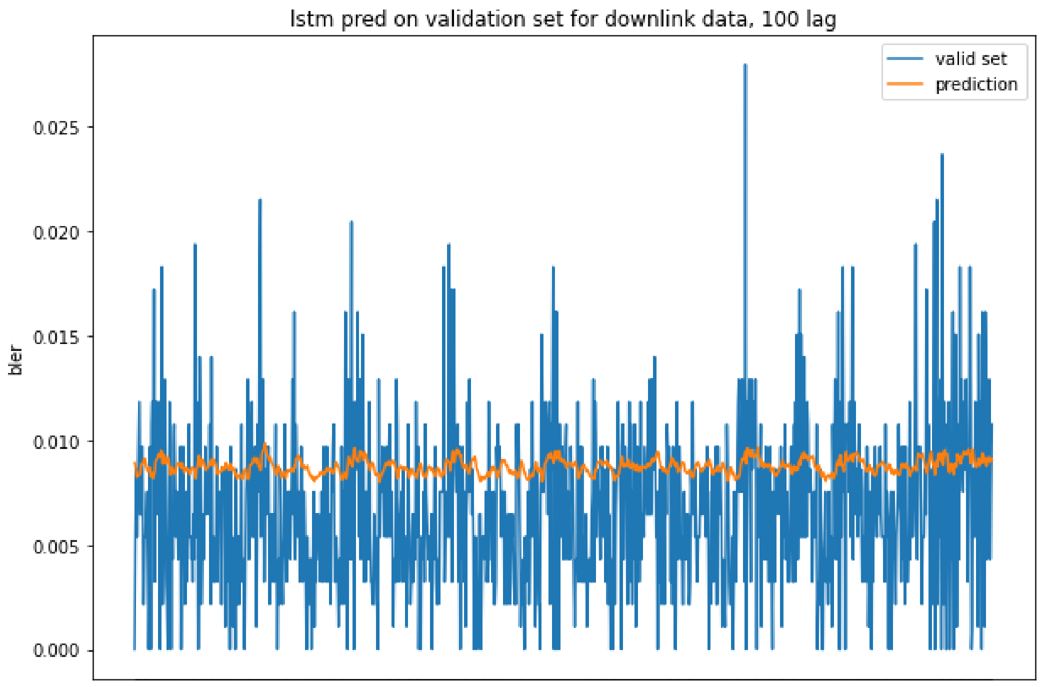


Figure 37. Downlink BLER LSTM prediction over valid set.

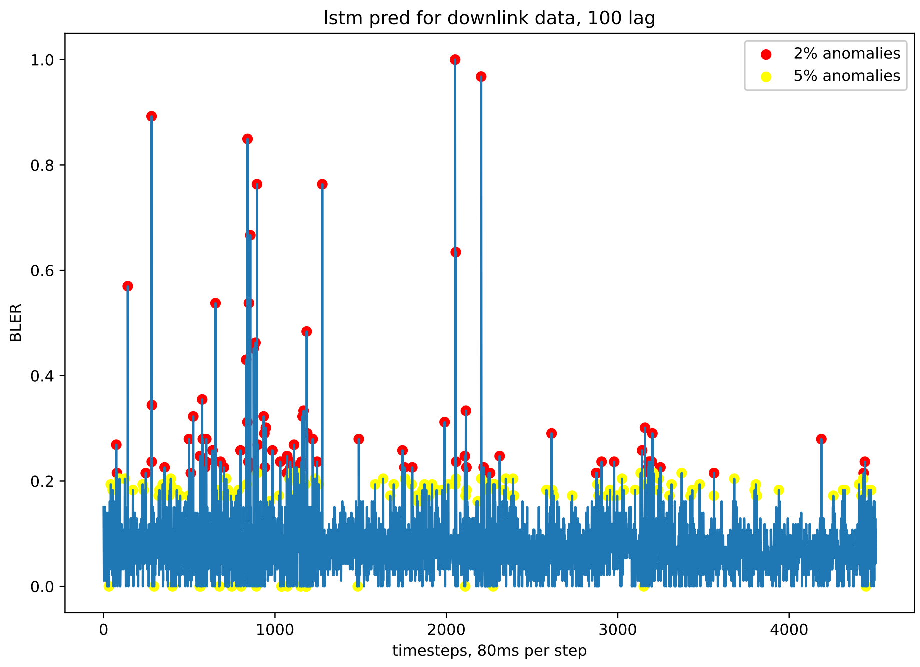


Figure 38. Downlink BLER LSTM anomalies detection.

the normal points. Anomalous points have an anomaly score of 0.65 or higher. The RMSE values represent the evaluation of the three models used for forecasting. The more robust the forecasting the more accurate we are able to detect the anomalies using the flagging technique explained previously.

Table 1. Models evaluation using RMSE on validation sets

| Models | Uplink | Downlink |
|---|---|---|
| ARMA | 2.023 | 3.349 |
| XGBRegressor | 2.577 | 3.852 |
| LSTM | 1.928 | 3.788 |

# 7. DISCUSSION

Exploratory data analysis is always important even for the case of uni-variate time series as it gives us a better understanding about the whole data set we are working with generally and the time series target we are interested in specifically. Understanding the context helps significantly through the data cleaning, preprocessing and modeling phases. For the time series data sets, it is always necessary to verify whether the data set is stationary and transform it into a stationary one if it is not already. It is also important to check whether the data distribution follows one of the popular distributions as in this case, applying only pure statistical rules can be sufficient.

The data set used for this work has two main limitations. The first one is that it is not labeled for the anomaly detection purpose. The approach we developed to overcome this limitation is sufficient but it also depends on the assumption of the percentage of anomalies in the data set which is also not known. The second limitation is that the data set has many anomalies which affects the models learning behavior. Training the model on a normal data set that includes only normal BLER values would have helped the model in learning the expected behavior of the BLER values and as a result, detecting any anomalous pattern more efficiently.

Choosing the number of lags that significantly influence the model's performance is one of the key factors for any time series forecasting problem. Using partial auto-correlation or main auto-correlation tests is one way to determine this number but it is important to highlight that these tests only suggest the range of the number of the significant lags while the exact number is to be determined during fine tuning the model used for the forecasting purpose.

The selection criteria of the four models selected was based on the literature review of these models and their final results' benchmark on different data sets for the time series anomaly detection problems. Each model represents a different category of algorithms. The ARMA one represented the statistical approaches. XGBRegressor represented the supervised machine learning approaches. Isolation Forest represented the unsupervised machine learning approaches. Finally, the LSTM model represented the deep learning approaches.

Based on the previous results we got, we can conclude that the Isolation Forest algorithm works as a clustering algorithm given its baseline version. That means that it will be able to isolate the outliers but not the anomalies as it does not distinguish whether these fluctuations are part of a pattern or just outliers. It does not also make use of the time series sequence which is a very important feature that influences the model performance in the case of the time series anomaly detection problems. On the other hand, being an unsupervised machine learning model, makes it easier to generalize using this model for unlabeled data sets, which is a common challenge most of the time. Using the enhanced version of Isolation Forest that takes the time series sequence into consideration might influence the model performance and makes it more suitable for the anomaly detection problems and not just the outliers detection.

There are advantages and disadvantages of the other three models. The ARMA model got the best performance on the downlink BLER data set and the second best performance on the Uplink BLER data set. It is also important to highlight the fact that despite it got the second best performance, its RMSE is very close to the best performance with less than 0.1 difference. On the other hand, it got the best performance on the BLER downlink data set with a difference of 0.44 RMSE to the second best performance. The training time is almost real-time but it depends on the length of the data set. The biggest odd of the ARMA or ARIMA based models generally is that it does not accept any other features except for the time series target we are concerned about. In some cases, we need to test the model performance in a multi-variate context to be able to check the influence and correlation of other features on our main time series target as working only on univariate time series target does not give enough insights in that case.

The XGBRegressor got the highest RMSE for the both the BLER uplink and downlink data sets. It is important to mention that even though it got the highest RMSE or worst results, its results are still good as they differ from the other models results with a small margin. Further fine tuning of the model in addition to more accurate labeling technique could have resulted in better RMSE. One of the advantages of the XGBRegrssor is that it can accept and train over additional features other than the main time series target which can be helpful in the multi-variate context. In addition, it has very decent training and inference time that we are able to get the final results in almost real-time.

The simple LSTM architecture we used got the lowest RMSE for the BLER uplink data set and the second best result for the BLER downlink data set. It is important to mention that there are many improvements that could be done to the model that will definitely enhance its performance and yield the best performance among the other models. Similar to the XGBRegressor, the model can accept and train over additional features other than the main time series target. We only used one hidden layer in our architecture and trained the model for only one epoch. Some improvements that can influence the model performance are:

- Using a deeper architecture
- Training the model for longer time
- Using a different batch size
- Changing the number of perceptrons in hidden layers.

The main goal of this work is not to prove the capability of a certain model or a category of models to detect anomalies in a time series target but to compare and benchmark the output of different models from different categories in terms of different performance metrics. The recommendation of the best model is based on the constraints of the system this model will be integrated in as there is no one best model that fits all the requirements. As mentioned previously, this is phase one of three phases. During the next phases, further filtration process is to be applied to the selected model or models from phase one.

# 8. CONCLUSION

Machine learning has been applied widely in different problems in the wireless communications domain. Time series anomaly detection-based machine learning topic has attracted a lot of research attention recently because of its increasing need in almost all the domains. The main factor that distinguishes machine learning models ability to detect anomalies from the statistical approaches is the automation factor. Detecting unseen anomalous patterns without the need of humans expert interference saves many resources and is considered a huge added value for both the research and industry aspects.

In parallel to the improvements and enhancements the 5G cellular network has witnessed, the importance of testing the 5G networks' environments became an essential need in terms of the networks' capacity, reliability and latency. Testing is done in both the transmitter and receiver sides. This work focuses mainly on one of the most important features and measurements of the quality of the reception process, the BLER.

In this work we applied four statistical, supervised machine learning, unsupervised machine learning and deep learning models for the anomaly detection purpose. Given that the data set does not have the anomalous points labeled, we developed a flagging technique for flagging the anomalies in the data set based on their severity and the assumption of the percentage of anomalies in the data set.

There will always be the trade-off between the performance and resources. If the performance is our main constrain, then, using complex deep learning models such as the LSTM model in our case will yield the best results. If training and inference time and the computational resources required is our main constrain, then, using simple statistical approach such as the ARIMA models and its variations is the best option as these models yield good results and have almost real-time performance.

The next phase of this work, after the selection of the model that matches the required constraints, is to apply the selected model in a multi-variate time series context where other features' behavior influence the main time series target. During the final phase and based on the output of the second phase, the best model is to be optimized and integrated as part of an automatic time series anomaly detection system that detect anomalies in a time series target with good performance in real-time.

# 9. REFERENCES

[1] Ejaz W., Anpalagan A., Imran M.A., Jo M., Naeem M., Qaisar S.B. & Wang W. (2016) Internet of things (iot) in 5g wireless communications. IEEE Access 4, pp. 10310–10314.

[2] Haus G., Ludovico L., Pagani E. & Scarabottolo N. (2019) 5g technology for augmented and virtual reality in education. pp. 512–516.

[3] Chavare S., Awati C. & Shirgave S. (2021) Smart recommender system using deep learning.

[4] Chatterjee J. (2020) Machine Learning with Health Care Perspective Machine Learning and Healthcare: Machine Learning and Healthcare.

[5] Ullal M., Hawaldar I., Soni R. & Nadeem M. (2021) The role of machine learning in digital marketing. SAGE Open 11, pp. 1–12.

[6] Trivedi D., Bhagchandani A., Ganatra R. & Mehta M. (2018) Machine learning in finance. pp. 1–4.

[7] Nascimento G. & Correia M. (2011) Anomaly-based intrusion detection in software as a service. Proceedings of the International Conference on Dependable Systems and Networks .

[8] Jain A., Arora M., Mehra A. & Munshi A. (2021) Anomaly detection algorithms in financial data. International Journal of Engineering and Advanced Technology 10, pp. 76–78.

[9] Elghanuni R., Ali M. & Swidan M. (2019) An overview of anomaly detection for online social network. pp. 172–177.

[10] Sabic E., Keeley D., Henderson B. & Nannemann S. (2021) Healthcare and anomaly detection: using machine learning to predict anomalies in heart rate data. AI SOCIETY 36.

[11] Choukuljaratsiri A., Lertwongkhanakool N., Thienprapasith P., Pratanwanich N. & Chuangsuwanich E. (2021) Anomaly Detection for Online Visiting Traffic as a Real-Estate Indicator: The Case of HomeBuyer. pp. 291–301.

[12] Krishnasamy S., Sen R., Oh S. & Shakkottai S. (2015) Detecting sponsored recommendations. ACM SIGMETRICS Performance Evaluation Review 43.

[13] Reddy Y., Pulabaigari V. & B E. (2018) Semi-supervised learning: a brief review. International Journal of Engineering Technology 7, p. 81.

[14] Liu Y. & Liu Q. (2021) Research on self-supervised comparative learning for computer vision. Journal of Electronic Research and Application 5, pp. 5–17.

[15] O' Mahony N., Campbell S., Carvalho A., Krpalkova L., Velasco-Hernandez G., Harapanahalli S., Riordan D. & Walsh J. (2019), One-shot learning for custom identification tasks.

[16] Lin Q., Liu Y., Wen W. & Tao Z. (2021), Ensemble making few-shot learning stronger.

[17] Mahan M., Chorn C. & Georgopoulos A. (2015) White noise test: detecting autocorrelation and nonstationarities in long time series after arima modeling.

[18] Greunen J., Heymans A., Van Heerden C. & Vuuren G. (2014) The prominence of stationarity in time series forecasting. Journal for Studies in Economics and Econometrics 38, pp. 1–16.

[19] Montañés bernal A. & Sanso A. (2001) The dickey-fuller test family and changes in the seasonal pattern. Annales d'Economie et de Statistique 61, pp. 73–90.

[20] Bellazzi R. & Magni P. (1999) Time series analysis (i) .

[21] Aggarwal C.C. (2016) Outlier Analysis. Springer Publishing Company, Incorporated, 2nd ed.

[22] Hawkins D. (1980) Identification of outliers. Monographs on applied probability and statistics, Chapman and Hall, London [u.a.]. URL: `http://gso.gbv.de/DB=2.1/CMD?ACT=SRCHA&SRT=YOP&IKT=1016&TRM=ppn+02435757X&sourceid=fbw_bibsonomy`.

[23] Baur C., Wiestler B., Albarqouni S. & Navab N. (2019) Deep autoencoding models for unsupervised anomaly segmentation in brain mr images. Lecture Notes in Computer Science , p. 161–169URL: `http://dx.doi.org/10.1007/978-3-030-11723-8_16`.

[24] Chandola V., Banerjee A. & Kumar V. (2009) Anomaly detection: A survey. ACM Comput. Surv. 41, pp. 15:1–15:58.

[25] Braei M. & Wagner S. (2020), Anomaly detection in univariate time-series: A survey on the state-of-the-art.

[26] Toyserkani A., Ström E. & Svensson A. (2010) An analytical approximation to the block error rate in nakagami-m non-selective block fading channels. Wireless Communications, IEEE Transactions on 9, pp. 1543 – 1546.

[27] Wang Y., Liu W. & Fang L. (2020) Adaptive modulation and coding technology in 5g system. In: 2020 International Wireless Communications and Mobile Computing (IWCMC), pp. 159–164.

[28] Vijay A. & Umadevi K. (2021) Developing optimal spectrum sharing protocol and optimal linear precoding for multi-carrier code-division multiple access using massive multiple input multiple output in 5g wireless networks. Wireless Personal Communications 119, pp. 1–26.

[29] Lin X., Rommer S., Euler S., Yavuz E. & Karlsson R. (2021), 5g from space: An overview of 3gpp non-terrestrial networks.

[30] Gupta M., Gao J., Aggarwal C.C. & Han J. (2014) Outlier detection for temporal data: A survey. IEEE Transactions on Knowledge and Data Engineering 26, pp. 2250–2267.

[31] Robert G. Brown R.F.M. (1961) The Fundamental Theorem of Exponential Smoothing. Operations Research.

[32] yf y., Zhu Y., Li S. & Wan D. (2014) Time series outlier detection based on sliding window prediction. Mathematical Problems in Engineering 2014.

[33] Dunning T. & Friedman E. (2014) Practical machine learning: A new look at anomaly detection.

[34] Bradley P.S. & Fayyad U.M. (1998) Refining initial points for K-Means clustering. In: Proc. 15th International Conf. on Machine Learning, Morgan Kaufmann, San Francisco, CA, pp. 91–99.

[35] Keogh E. & Lin J. (2005) Clustering of time-series subsequences is meaningless: Implications for previous and future research. Knowledge and Information Systems 8, pp. 154–177.

[36] Ester M., Kriegel H.P., Sander J. & Xu X. (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. In: KDD.

[37] Oehmcke S., Zielinski O. & Kramer O. (2015) Event detection in marine time series data.

[38] Breunig M., Kriegel H.P., Ng R. & Sander J. (2000) Lof: Identifying density-based local outliers. vol. 29, pp. 93–104.

[39] Amer M., Goldstein M. & Abdennadher S. (2013) Enhancing one-class support vector machines for unsupervised anomaly detection. pp. 8–15.

[40] Schölkopf B., Williamson R., Smola A., Shawe-Taylor J. & Platt J. (1999) Support vector method for novelty detection. vol. 12, pp. 582–588.

[41] Hyndman R. & Athanasopoulos G. (2018) Forecasting: Principles and Practice. OTexts, Australia, 2nd ed.

[42] Haselsteiner E. & Pfurtscheller G. (2001) Using time-dependent neural networks for eeg classification. IEEE transactions on rehabilitation engineering : a publication of the IEEE Engineering in Medicine and Biology Society 8, pp. 457–63.

[43] Wu H. & Gu X. (2015) Max-pooling dropout for regularization of convolutional neural networks. ArXiv abs/1512.01400.

[44] Munir M., Siddiqui S., Dengel A. & Ahmed S. (2018) Deepant: A deep learning approach for unsupervised anomaly detection in time series. IEEE Access PP, pp. 1–1.

[45] Nair V. & Hinton G. (2010) Rectified linear units improve restricted boltzmann machines vinod nair. vol. 27, pp. 807–814.

[46] Ioffe S. & Szegedy C. (2015) Batch normalization: Accelerating deep network training by reducing internal covariate shift .

[47] He K., Zhang X., Ren S. & Sun J. (2016) Deep residual learning for image recognition. pp. 770–778.

[48] Wang Z., Yan W. & Oates T. (2017) Time series classification from scratch with deep neural networks: A strong baseline. pp. 1578–1585.

[49] van den Oord A., Dieleman S., Zen H., Simonyan K., Vinyals O., Graves A., Kalchbrenner N., Senior A. & Kavukcuoglu K. (2016) Wavenet: A generative model for raw audio. In: Arxiv. URL: `https://arxiv.org/abs/1609.03499`.

[50] Borovykh A., Bohté S.M. & Oosterlee C.W. (2017) Conditional time series forecasting with convolutional neural networks. arXiv: Machine Learning .

[51] Hochreiter S. & Schmidhuber J. (1997) Long short-term memory. Neural computation 9, pp. 1735–80.

[52] Chauhan S. & Vig L. (2015) Anomaly detection in ecg time signals via deep long short-term memory networks. pp. 1–7.

[53] Malhotra P., Vig L., Shroff G. & Agarwal P. (2015) Long short term memory networks for anomaly detection in time series.

[54] Cho K., Merrienboer B., Gulcehre C., Bougares F., Schwenk H. & Bengio Y. (2014) Learning phrase representations using rnn encoder-decoder for statistical machine translation .

[55] Wu W., He L. & Lin W. (2019) Local Trend Inconsistency: A Prediction-driven Approach to Unsupervised Anomaly Detection in Multi-seasonal Time Series.

[56] Bank D., Koenigstein N. & Giryes R. (2020), Autoencoders.

[57] Bakrania M., Rae I., Walsh A., Verscharen D. & Smith A. (2020), Using dimensionality reduction and clustering techniques to classify space plasma regimes.

[58] Wu Q. & Shao Z. (2005) Network anomaly detection using time series analysis. In: Joint International Conference on Autonomic and Autonomous Systems and International Conference on Networking and Services - (icas-isns'05), pp. 42–42.

[59] Ghorbani N. & Bahrak B. (2022) Aggregated traffic anomaly detection using time series forecasting on call detail records. Security and Communication Networks 2022.

[60] Arum K. (2019) Volatility modelling using arch and garch models (a case study of the nigerian stock exchange). International Journal of Mathematics Trends and Technology 65, p. 58 to 63.

[61] Al Mamun S.M.A. & Valimaki J. (2018) Anomaly detection and classification in cellular networks using automatic labeling technique for applying supervised learning. Procedia Computer Science 140.

[62] Chaparro C. & Eberle W. (2015) Detecting anomalies in mobile telecommunication networks using a graph based approach. In: The Twenty-Eighth International Flairs Conference.

[63] Steinhauer H., Helldin T., Mathiason G. & Karlsson A. (2019) Topic modeling for anomaly detection in telecommunication networks. Journal of Ambient Intelligence and Humanized Computing .

[64] Mushtaq R. (2011) Augmented dickey fuller test. SSRN Electronic Journal .

[65] Dismuke C. & Lindrooth R. (2006) Ordinary least squares. Methods and Designs for Outcomes Research .

[66] Meek C., Chickering D. & Heckerman D. (2002) Autoregressive tree models for time-series analysis.

[67] Liu F.T., Ting K. & Zhou Z.H. (2009) Isolation forest. pp. 413 – 422.

[68] Ding Z. & Fei M. (2013) An anomaly detection approach based on isolation forest algorithm for streaming data using sliding window. In: ICONS.

[69] Chen T. & Guestrin C. (2016) Xgboost. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining URL: `http://dx.doi.org/10.1145/2939672.2939785`.

[70] Kingma D. & Ba J. (2014) Adam: A method for stochastic optimization. International Conference on Learning Representations .