Comparison of Machine Learning Techniques Applied to Traffic Prediction of Real Wireless Network

DARIA ALEKSEEVA1 (Student Member, IEEE), NIKOLAI STEPANOV2, ALBERT VEPREV2, ALEXANDRA SHARAPOVA2, ELENA SIMONA LOHAN1 (Senior Member, IEEE), ALEKSANDR OMETOV1 (Member, IEEE)

1Tampere University, Tampere, Finland
2National Research University Higher School of Economics, Moscow, Russia

Corresponding authors: Daria Alekseeva (email: daria.alekseeva@tuni.fi)

This paper is an extended version of the paper “Applying Machine Learning to LTE Traffic Prediction: Comparison of Bagging, Random Forest, and SVM,” presented at the 12th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT) and published by IEEE.

The code affiliated with this paper is available at: https://github.com/AlbertV11/ML_LTE_traffic_prediction

The work of D. Alekseeva was funded by the doctoral grants of the Information Technology and Communications Science Faculty at Tampere University. The work of E.S. Lohan was supported by the Academy of Finland under Project ULTRA.

ABSTRACT

Today, the traffic amount is growing inexorably due to the increase in the number of devices on the network. Researchers analyze traffic by identifying sophisticated dependencies, anomalies, and novel traffic patterns to improve the system performance. One of the fast development niches in this domain is related to Classic and Deep Machine Learning techniques that are supposed to improve the network operation in the most complex heterogeneous environment. In this work, we first outline existing applications of Machine Learning in the communications domain and further list the most significant challenges and potential solutions while implementing those. Finally, we compare different classical methods predicting the traffic on the LTE network Edge by utilizing such techniques as Linear Regression, Gradient Boosting, Random Forest, Bootstrap Aggregation (Bagging), Huber Regression, Bayesian Regression, and Support Vector Machines (SVM). We develop the corresponding Machine Learning environment based on a public cellular traffic dataset and present a comparison table of the quality metrics and execution time for each model. After the analysis, the SVM method proved to allow for a much faster training compared to other algorithms. Gradient Boosting showed the best quality of predictions as it has the most efficient data determination. Random forest shows the worst result since it depends on the number of features that may be limited. The probabilistic approach-based Bayesian regression method showed slightly worse results than Gradient Boosting, but its training time was shorter. The performance evaluation demonstrated good results for linear models with the Huber loss function, which optimizes the model parameters better. As a standalone contribution, we offer the source code of the analyzed algorithms in Open Access.

INDEX TERMS

Machine learning; Communication system traffic; Prediction algorithms; Optimization; Next generation networking.

I. INTRODUCTION

Every year, the number of users and devices is increasing tremendously, resulting in traffic growth and higher network congestion from various perspectives, e.g., causing a higher number of requests to be tackled with constrained network resources. In turn, that increased the device number, which is also affecting the complexity of the system’s architecture [1]. With this respect, researchers are already investigating the potential challenges for future wireless communications starting at 6G and beyond [2]–[5] that bring higher requirements in latency, reliability, security, privacy, etc.

Shifting the computing to the Edge is forecasted to
notably improve energy efficiency and reduce computing latency [6]. Therefore, the Information and Communication Technologies (ICT) researchers and integrators of the computing paradigms will face the challenge of complex system analysis. Implementation of the Machine Learning (ML) approaches is one of the solutions for managing and optimization such a massive and heterogeneous system [7]–[9].

One of the advantages of applying ML is the possibility to identify new dependencies in a large amount of data, which is already close to impossible to do manually, e.g., traditional network planning methods fail to manage Big Data flows [10] causing a need to shift the operations closer to the user. The use of ML in future networks is a promising direction for solving the problem of managing large data flows [11]. ML algorithms based on the input data could provide sub-optimal solution (depending on the task) potentially providing a completely new level of network and application management.

Overall, the fluctuations in, e.g., traffic may be caused by various factors, including crowds dynamically coming and leaving the area, i.e., waves of traffic load that may be non-fixed in time but somewhat fixed in space [12]. Moreover, new services, e.g., Ultra-Reliable Low-Latency Communication (URLLC) driven by 3rd Generation Partnership Project (3GPP), are now taking place where a high Quality of Service (QoS) should be guaranteed. Significantly, conventional solutions aiming at solving the traffic fluctuation problems, such as network offloading, densification, advanced Multiple-Input and Multiple-Output (MIMO) techniques and Cell on Wheels (CoW), require deployment/reuse of additional infrastructure, which adds more Capital expenditure (CAPEX)/Operating expense (OPEX) investments [13].

ML techniques may only require a logical overlay deployed on the Edge/in the Long-Term Evolution (LTE) network core to minimize the impact of, e.g., the hotspots.

Despite various challenges for network traffic prediction, ML models can bring benefits to different real-life scenarios for short periods as well as for long ones [14], [15]. From the ML perspective, short-time prediction, which ranges from milliseconds to minutes, can be applied for optimal resource planning, congestion control, and routing packets. Long-term forecasts can provide analysis for future capacity requirements, network security and reduce excess operation costs by bandwidth and energy optimization for predicted traffic. Richerzhagen et al. [16] considered the scenario of urban train station. In this scenario, long-term traffic prediction could be implemented for rescheduling network resources beside an urban railway station with numerous pedestrians forming crowds during busy times. These peaks are due to the periodical train arrival and departure results in an overload of the cellular communication network. Wang et al. [14] considered data from the major city of China and specified that during 24h traffic volume can change 100,000 times. So traffic has a daily trend, and it is useful to predict traffic for such a long period. From the short-time prediction view, it is useful to recognize correlations in real-time between overcrowded occasions from festivals to rallies and mobile traffic. That can become critical for network security and applications related to healthcare, security, logistics, or even mission-critical programs [12].

There is a large number of ML algorithms that could be applied to the network optimization tasks. The algorithm’s choice depends on the purpose, input data type, available resources, etc. It is not straightforward to compare those algorithms since a peculiar one can show different results under different circumstances. As the input data changes, the algorithms would require different parameters to work correctly [17], [18]. Thus, as a first step, we identify the main problem types and applications of ML in the communications domain, related metrics, and current problems to be solved to form a baseline for the selection and evaluation of the most applicable ones to the traffic prediction of a real-life dataset.

In our previous conference paper, we attempted to analyze three ML models as Bagging, Random Forest, and Support Vector Machines (SVM) [19]. Continuing our research, we have extended this work with four new models: Linear Regression, Huber Regression, Bayesian Regression, and Gradient Boosting. This way, we are offering an in-depth analysis of the ML techniques suitable for predicting the cellular network traffic. In order to test the ML techniques, we found and utilized a real-life measurement dataset titled “Predict traffic of LTE network” [20]. Jupyter notebook and Python 3.7 software were used to model the prediction algorithms. Finally, we offer a source code of the analyzed algorithms in Open Access1.

We attempted to answer the following research questions in this work:

RQ1: To which areas of future networks could ML be applied?
RQ2: What ML models are utilized by researchers and for which problem type?
RQ3: What are the current challenges and potential solutions of implementing ML for communications?
RQ4: Which ML technique would provide the best results for the cellular traffic prediction?

To answer the research questions, we identified the following tasks:

- To identify existing applications of ML in communications domain.
- To apply the selected ML models to the real-life collected dataset.
- To identify advantages and disadvantages of each ML model.
- To highlight the main challenges and overview potential solutions found in the literature.

1The code affiliated with this paper is available in Open Access at: https://github.com/AlbertV11/ML_LTE_traffic_prediction
A. LITERATURE REVIEW METHOD
To achieve better reproducibility of this research, we have included a brief research methodology subsection, which is based on PRISMA guidelines [21]. Commonly, the initial step is to specify eligible criteria, i.e., year, language, and publication status. Criteria to including papers were works in English where ML strategies are applied to the network, starting from 2017. We formed a search entry for Scopus and Web of Science – the two most conventional research databases in the computer science domain. Based on initial literature analysis, the following search expression was formed: (“LTE” OR “traffic” OR “prediction”) AND “machine learning”.

From the set of publications that are potentially relevant to the topic, we excluded grey literature, duplicates, and pre-prints. Then, we investigated the abstracts, titles, and keywords of the publications to identify papers and articles that report the application of ML in the communications domain. The following paper exclusion criteria were defined:

- Not related to the application of ML;
- Do not describe the used ML models;
- Review articles and survey papers;
- No technical content present;
- No full text available.

After careful reading of the papers, we have extended the list of identified papers by relevant ones found in the inner references.

B. STRUCTURE OF THE PAPER
The rest of the paper has the following structure. First, Section II provides a brief description of ML: types, models, metrics, and main techniques required for the data preprocessing. Next, Section III presents an overview of existing works and faced ML challenges. Further, Section IV describes the traffic prediction strategies utilized in this work and results obtained with each of the applied models, i.e., quality metrics and algorithm running time, as well as the impact of noise in the dataset on the results. Section V summarizes the challenges on the applicability ML for communication networks. The last section concludes the paper and provides future work.

II. BACKGROUND INFORMATION
At the beginning of the 21st century, researchers were no longer in doubt that developing the technology that would allow computers to learn based on their experience is a very promising direction [22]. We can define ML as a computer science field applying mathematical-based algorithms to analyze significant amounts of data in a semi-automatic way.

Historically, ML methods have their roots in the second half of the 20th century, but they did not find their application until the beginning of the 21st century due to the lack of computing power [23]. Today, ML could be divided into two parts – Classic and Deep Learning [24]. Classic ML includes linear models, for example, Linear Regression, Logistic Regression, Huber Regression, Bayesian Regression, SVM, and ensemble models, such as algorithms as Bootstrap Aggregating (abbreviated Bagging) and Gradient Boosting. Linear models use linear functions, i.e., the function takes a fixed number of numerical inputs and searches the most advantageous matrix of factors to minimize the loss function. The ensemble model, or nonlinear, consists of several algorithms for more accurate prediction than each algorithm itself. Deep Learning was a 2012th breakthrough generated after George E. Dahl’s team won the Merck Molecular Activity Challenge. Dahl’s group utilized various neural networks in order to find complex relationships in the data [25].

All tasks of ML application could be bound to one of the following ML types. There are Supervised Learning, Unsupervised Learning, Reinforcement Learning, and Hybrid models. Each task could be solved by one of the types of ML (see Fig. 1).

A brief description of the algorithms used in this work is provided below. Those may be applied to predict different variables such as traffic, time of the transaction, detection of some incorrect or fraudulent transactions, etc. The problem types they solve and examples of the real applications are provided in Table 1.

- Linear Regression is a model based on relationships between variables by fitting the training data to a linear model based on least-squares fitting. Regression is a simple and precise linear model with a high variance but low bias. It is widely applied in prediction cases.
- SVM is based on creating plane-bounded regions and by maximizing the distance from each dividing plane. For the regression and classification problem, the model uses reference objects (located on the class boundary) [26].
- Random Forest is based on the Decision Tree approach, i.e., by creating a tree-like model of decisions and looking at their possible consequences. Each of the Trees is trained on a random sample and a random function. The prediction is made based on the result from each Tree.

FIGURE 1: Types of ML and related characteristics
TABLE 1: Most popular ML problems types in telecommunications and related pivot metrics

| Pr. type  | Description                                                                 | Telecom examples                                                                 | ML models                                                                                     | Key metrics                                                                 | Brief definition                                                                 |
|-----------|------------------------------------------------------------------------------|----------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------|----------------------------------------------------------------------------------|
| Classification | To arrange the data in groups according to established criteria. It takes a definite response based on a set of features. | Decision Tree, k-NN, SVM, Random Forest, Logistic Regression | Accuracy (Acc)                                                                           | The number of correct predictions divided by the total number of predictions. |

- **Bootstrap Aggregation** (abbreviated Bagging) is very similar to the Random Forest model. For Bagging, it is essential to train several independent Decision Trees on sub-samples with return (bootstrap) but using the entire feature data space simultaneously.
- **Gradient Boosting** is an ensemble model of converting weak learners into strong ones.
- **Bayesian Regression** is a probabilistic way of representing a target. The idea of this approach is to assert that our target variable linearly depends on random variables — regression coefficients, the distribution laws of which we choose based on the logic of the problem.
- **Huber’s regression model** is a more advanced version of the linear model. Its main difference lies in the application of Huber’s piecewise loss function, which consists of two parts — a quadratic and a linear function.

Nonetheless, it is essential to compare the accuracy of each model, thus, **metrics** play a crucial role in ML algorithms, as those represent the quality or quantity indexes that give information about a particular process. All metrics could be divided into different categories based on the ML model according to a problem type. For example, the most common metrics for Classification problems are accuracy and precision. Accuracy is defined as the number of correct predictions divided by the total number of predictions. It shows the percentage of hits into the required class. Precision is a class-specific performance metric. It is defined as the number of true-positive results divided by the difference between true-positive and false-positive results.

According to the problem type, the most common metrics for various algorithms are highlighted in Table 1 (based on works from Section III of this paper), i.e., Absolute Error (AE) and Relative Error (RE), RMSE, MAE, Coefficient of Determination ($R^2$).

a: Absolute and Relative Error

AE is the difference between the real output $y_i$ and the predicted value $\hat{y}_i$, as $AE = |y_i - \hat{y}_i|$. RE means the ratio of AE to the real value, such that $RE = \left( \frac{|y_i - \hat{y}_i|}{y_i} \right)$.

b: Root Mean Square Error

RMSE is a well-known metric, which measures how well the model fits the predicted variables. It is defined as

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2},$$

where $y_i$ is a real output, $\hat{y}_i$ is a predicted output, and $N$ is the number of variables. RMSE represents the root mean square of the prediction error measured as the difference
between real and predicted output; the larger RMSE value corresponds to the larger error.

c: Mean Absolute Error
MAE is similar to the RMSE, but it takes the sum of the absolute values instead of taking the sum of absolute square values, such that

$$
MAE = \frac{1}{N} \sum_{i=1}^{N} |y_i - \hat{y}_i|,
$$

thus, showing the average over the data set of the absolute differences between the real observation and the predicted value and it can range from 0 to infinity. The lower the MAE value is, the better the prediction is.

d: Coefficient of Determination
The coefficient of determination ($R^2$) is a ratio of variances, and it is a measure of how much the variance the estimated value is, the proportion of the predicted output variance. $R^2$ is defined by

$$
R^2 = 1 - \left( \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2} \right),
$$

where $y_i$ is the real (or true) observed data, $\hat{y}_i$ is the predicted output for $y_i$, and $\bar{y}$ is the mean of the observed data.

The $R^2$ is the proportion of the predicted output variance, which varies between 0 % and 100 %, i.e., a higher $R^2$ shows a higher correlation between the predicted and the observed variables. For example, $R^2 = 100 \%$ means that the model explains all the variance in the data.

The remaining aspect to highlight with respect to the ML execution is related to data preprocessing, which is sometimes required before the actual ML application. In particular, the data must be presented in an accessible form suitable for the import, i.e., it can be numerical or nominal (categorized). Before applying the ML model, it is necessary to present them in the required form for processing (for example, binary numbers) and at the same scale. We further describe such methods as One-Hot Encoding (OHE) and Min-Max Scaling, which are used to encode and normalize data accordingly.

e: One-HotEncoding
As above-mentioned, the data can be classified into two classes: categorical (nominal) type or a numerical type. Some algorithms, such as an XGBoost or Catboost, can work with both categories, while most ML algorithms operate with the categorical data that can be converted into a numerical via the so-called OHE technique, i.e., each category is encoded with a binary number. That means that, a 1 value is placed in the binary variable for the category and 0 values for the others. For example, if data consist of the string Animals with labels dog, cat, turtle, and mouse, then, in OHE technique, each unique category value is assigned with a binary number – dog – "0100", cat is "1000", turtle – "0010", and mouse – "0001".

The number of needed binary variables depends on the number of categories. In the above example, the Animals has 4 labels, and therefore it has 4 binary variables. It is to be noted that OHE technique is applicable only with a finite set of label values. If the model has too many labels, OHE will become impractical, as it would require a high number of triggers.

f: Min-Max Scaling
Min-Max Scaling is a normalization technique for the input variables. It is applicable for the input data measured on different scales, so this might cause bias during the modeling

$$
x = \left( \frac{x - \min(x)}{\max(x) - \min(x)} \right),
$$

where $x$ is a scaled variable, $x$ is an input variable, $\min(x)$ and $\max(x)$ are the minimum and maximum values of all inputs accordingly.

To summarize the background section, we need to note that ML is widely used for prediction and management of various systems and solving different tasks. The next section presents works where ML algorithms are already applied for improving communication systems.

III. APPLICATION OF MACHINE LEARNING IN COMMUNICATION DOMAIN
This section overviews various applications of ML used to improve the network characteristics. The subsection III-F presents most relevant to this study works, while other subsections outline a broader overview of the actuality of the problematics for the next-generation networks.

It is worth noting that these works are not yet fully implemented from the industrial perspective, while researchers are looking forward to more accurate and straightforward solutions for flexible and on-the-fly data processing [27].

A. REDUCING A DELAY IN HETEROGENEOUS NETWORKS
Balevi et al. [28] proposed to implement the unsupervised soft-clustering ML model to upgrade Fog Nodes (FN) from Low Power Nodes (LPN) for shifting the computing closer to the end devices (known as Fog computing paradigm [29]). Here, some nodes are specialized as FN in a cloud-to-things architecture to control and provide services to the user. As the heterogeneous network consist of Nigh Power Nodes (HPN) and LPN, the promising idea is to upgrade LPN to the FN to improve the performance of the heterogeneous network. The successful upgrade is expected to make the network more flexible and assist in reducing the latency. The main research questions of this work are which LPN should become FN, and what should be the number of FN. Soft clustering means that in the resulting row vector, after the algorithm runs, it does not record whether the station will become FN or not (0, 1), as it happens in hard clustering, but the probabilities of each Base Station (BS) to become FN is recorded, in
order decrease. The main idea of soft clustering is to look at distance and the communication channel’s quality. Therefore, it does not give a clear answer to whether the station will become FN, but only shows the probability of this event. After testing soft and hard clustering algorithms, it was found that the proposed algorithm has a great advantage for low bandwidths. As for high bandwidths, the difference between the two algorithms is not very large. It is concluded that the delay decreases with the bandwidth up to a certain point. After that, the delay saturates and does not decrease further. However, if the nearest stations’ number tends to infinity, the algorithm’s quality will not increase much. The opposite result might cause it – with a large number of neighbors, the algorithm will be half-educated.

Khatouni et al. [30] present Supervised Learning algorithms that attempt to predict the delay of the response in the network exploiting real mobile network data of the end-user. The authors used a large-scale dataset with more than 238 million latency measurements from 3 different mobile operators to obtain the results. The work describes the k-fold cross-validation technique for model selection. The technique has the following algorithm: first, divide the dataset into k-folds; second, use k – 1 folds as a training set and 1 for the test set; third, repeat this process k all folds are used 1 time as a test set. The authors considered k = 10, and three ML classifiers: Linear Regression, Decision Trees, and SVM. As a result, Decision Trees show better cross-validation performance than the others. Despite this, the algorithms were unable to show a high prediction accuracy. Also, challenges applying algorithms are the need for data preprocessing as well as the correct feature selection.

Fiandrino et al. [17] consider to optimize the network traffic using Deep Spatio-Temporal Neural Network (D-STN). Optimization of network functions depends on the ability to classify traffic accurately. Implemented ML can extract specific flow characteristics and latency requirements and feed this information to the schedulers. The proposed solution instantiates ML in order to characterize the traffic features and to predict the future traffic demands.

The authors of [31] elaborate on real-life C-ITS transaction data prediction for communications for the public transport with the cloud by developing a set of methods to solve the Regression and Classification tasks. The paper depicts that the applicability to predict the “criticality” of the transaction as well as estimate its time with high accuracy. The authors claim that the Gradient Boosting model achieves the best results in predicting the real-time transaction time for a regression task among other analyzed works.

**B. IMPROVING NETWORK SECURITY AND PRIVACY**

As the next-generation networks undoubtedly affect human areas such as energy, banking, medicine, transportation, and others, security challenges cannot be ignored. Recent developments and existing 5G wireless security schemes [32] are presented based on related security services, including authentication, availability, data privacy, key management, and confidentiality. Artificial Intelligence (AI) has been used for intelligent detection to overcome the limitations of traditional identifiers. It classifies abnormal traffic using ML techniques.

Li et al. [33] classified traffic using ML techniques from safety perspective. In general, the authors propose a three-layer network protection system. The part that interests the research most is the intellectual part of the third layer. The following processes belong to it: feature selection, traffic classification (safe/unsafe), analysis and decision making, and a big data auxiliary center. It contains information about past attacks, which helps the classifier decide. Random Forest is used to obtain the significance of features. k-Nearest Neighbors (k-NN) – to classify traffic into two groups (malicious/safe). Adapting Boosting – to classify network attacks into four groups (Denial-of-Service (DoS), Remote-to-Local (R2L), User-to-Root (U2R), Probe). The results show that the Random Forest algorithm model has a significant impact on identifying significant features. It allows limiting the list of the three necessary traffic characteristics for its further classification. It affects not only the learning time of the k-NN and Adapting Boosting algorithms but also their classification quality. However, this approach’s disadvantage is the classifier’s inability to generalize various attacks without the first “meeting” them in the training sample.

Tran The Anh et al. [18] consider the concept of Mobile Crowd-Machine Learning (MCML) for a federated learning model. This concept addresses the data privacy challenges of traditional ML as it allows collaborative training between the mobile devices while keeping the data on the mobile devices. The authors proposed to implement Deep Q-Learning (DQL), which enables the server to find the optimal data. The energy management for the mobile devices is also enabled by the DQL scheme. In addition, this scheme reduces the training latency up to 55% compared to the random scheme.

Ahnaf Ahmad et al. [34] evaluated the application of several ML algorithms for Intrusion Detection Systems (IDS) in Software Defined Networks (SDN). SDN separates the network control and the data forwarding planes. Centralized network control in SDN improves network management but creates security challenges. The authors used ML techniques to detect abnormal traffic. Most real-world datasets in this area do not contain enough samples of intrusions and are not updated. To deal with it, the authors created a dataset with the traffic flow generator. In this dataset, User Datagram Protocol (UDP) flood attacks were considered Distributed Denial-of-Service attack (DDoS) attacks, both single and multi-host attacks were added. The authors compared the performance of SVM, Naïve-Bayes, Decision Tree, and Logistic Regression for traffic classification. SVM showed the best results.
C. NETWORK OPTIMIZATION AND MANAGEMENT

ML can also be useful in other areas of telecommunication technologies, such as serving the end-users on the network. As their number is continuously increasing and the number of data is multiplying, the latter one has to be analyzed with statistical processing methods to decrease costs and errors if the data were processed manually. Implementation of ML techniques could optimize the network’s operation and assist in predicting the network failures before they significantly degrade the QoS for the network.

Riihijarvi et al. [35] demonstrated how to predict the performance of a wireless network using ML. Radio Environment Map (REM) was taken as a data source. REM represents a coordinate plane with $X$ and $Y$ axes, which shows the distribution of a particular feature in space and its intensity using a thermal strip. Real data from a large US metropolis in the space of 10 x 10 km is considered as training and testing samples. All measurements collected relate to the 2 GHz downlink performance characteristics for a major cellular operator. The variables in the dataset are geographic coordinates and time of measurements of Received Signal Strength (RSS), Signal-to-Noise Ratio (SNR), and user data bit rate. Based on this information, the authors also enriched the dataset with information about the test unit’s speed of the mobile drive.

Xiong et al. [7] raise the problem of the need to split the network into parts for optimizing network management in connection with its increasingly complex architecture. The authors compared the Q-learning algorithm with Deep Reinforcement Learning (DRL) to optimize the slice request allocation to the network slices. The advantage of using DRL algorithm is that it does not make a preliminary hypothesis about the system, thereby, providing more practical methods for solving networks’ problems. Guaranteed Service Queue (GSQ) and Best-effort Service Queue (BeSQ) are taken as features for estimation of the proposed model’s effectiveness. GSQ is used to store user requests with the guaranteed service slices (e.g., video stream data). BeSQ is used to store the user requests with best-effort service slices (e.g., delay-tolerant data). The algorithm operates with the number of slice requests received from the GSQ and BeSQ to be served. The presented results show that the DRL model can efficiently process slice requests.

Chaudhary et al. [36] presented a new ML-based algorithm for routing in wireless networks. The authors collected a small and imbalanced dataset and used random oversampling to increase the number of samples and balance them. The proposed algorithm used supervised machine learning methods to predict the source’s network type and the destination nodes. k-NN, SVM, and Multinomial Logistic Regression (MLR) were tested on the dataset. If it were a Bluetooth network, the algorithm would have ended because the Bluetooth networks do not contain intermediate nodes. In case it was a Mobile Ad Hoc Network (MANET) or Delay-Tolerant Network (DTN), the algorithm constructed a network of nodes, choosing them depending on the battery power utilization and internal storage. Authors supposed that this algorithm could be useful in areas of disasters such as earthquakes. As these areas nodes can be partially disrupted, this algorithm could increase the probability of successful message delivery.

Sliwa et al. [37] used ML algorithms to predict datarate. They researched the concept of cars used as mobile sensors. As in these tasks, the transaction’s speed is not critical, and the authors proposed a method, which considers the current channel situation for finding an optimal time for the transaction. Analytical models and different multi-metric schemes were compared, and ML algorithm showed the best performance for this task. In this method, ML algorithm predictions are used to metric the optimal time for the transaction.

Also, some authors tried to invent new ML-based algorithms for network optimization. For instance, Sodhro et al. [38] proposed an ML-based mobility management method for Network in Box (NIB). They suggested an ML-empowered algorithm that dynamically generates the initial and master keys and then transfers them to the head nodes.

Everything above is forwarding prospects and necessary technologies in the future for this study. First, the creation of new architectural systems for obtaining performance data, and, second, the need to develop a pre-processing methodology to remove data with sufficiently severe violations and the development of ML models that can extract the maximum information from such measurements.

D. SIGNAL INTEGRITY ANALYSIS

The signal is a primary component of the telecommunication system. One of the most valuable parameters of the signal is its integrity. A signal is integral if it has fast and clear transitions, clear and stable logical levels, precise synchronization, and no transients. The Signal Integrity (SI) improving is a crucial direction in the future development of the digital systems [39].

It is fairly straightforward to see the data signal’s noise margins through the eye diagram, so it was taken as a data source for testing the model. An eye’s height and width allow critical parameters of the signal’s electrical quality to be quickly visualized and determined [40].

Lu et al. [41] suggested predicting the high-throughput channel from the trained model for various design parameters. The proposed approach is to implement the linear regression, Support Vector Regression (SVR), Deep Neural Network (DNN) models to predict eye-diagram height and width. This approach does not require sophisticated circuit modeling or significant knowledge of the subject area.

The model’s training is carried out in a reasonable time, and to reduce the cost of training, the trained model can be used again in further tasks. Once the training concludes, prediction can be performed in a highly efficient manner. As a result, linear regression showed the lowest accurate predictions of eye-diagram metrics. SVR shown good results...
as it can handle nonlinearities in data using kernel mappings. However, DNN regression outperforms SVR in terms of empirical prediction accuracy.

Chen et al. [42] also proposed to implement Hybrid Neural Network (HNN) to predict the eye-diagram metrics (height and width). Authors compare new semi-supervised learning for signal integrity analysis, which is based on a HNN, with models used in [41]. As a result, the HNN model reduces 50% labeled data for training with prediction improvement on eye-diagram height (about 32%) and width (about 34%). This model does not require substantial domain knowledge nor massive amounts of labeled data, thus it allows to eliminate complex and expensive circuit simulations.

Ma et al. [43] examined the performance of Sparse grids, SVR, and Artificial Neural Network (ANN), and applied them to a differential microstrip channel exhibiting variability in its geometric parameters (strip width, trace spacing, substrate thickness, and channel length). SVR showed the best result in terms of the prediction error. The ANN’s prediction error reduction could be achieved by adding hidden layers as its accuracy directly depends on the number of layers in the model’s architecture.

Another work by Rayas-Sánchez et al. [44] compares such ML techniques as SVM, Generalized Regression Neural Networks (GRNN), Polynomial Surrogate Modeling (PSM), and Kriging. The interaction between SI and Power Delivery Networks (PDN) plays a major role in the success of new computer products. Kriging model showed the best performance for the optimization task to find optimal sensing resistors and loading conditions.

E. ANOMALY DETECTION

Anomaly detection is the practice of identifying items or events that do not conform to expected behavior or do not correlate with other items in a dataset [45]. Anomalies can be caused by a malicious attack or inside network factors such as configuration errors and traffic congestion. Regardless of the type, these anomalies have a substantial impact on the network service. As the future generation networks are highly likely to force a strong impact on society, it becomes crucial to detect anomalies effectively [46].

Kim et al. [47] suggested using the Convolutional Long-Short Term Memory Recurrent Neural Network (C-LSTM) neural network to detect abnormal events in time series traffic. They took the dataset from Yahoo [48], which contains traffic from 67 web services. As the traffic from various services has a different distribution, it is complicated to find abnormal traffic by detecting statistical outliers. Another issue with the dataset is the data imbalance – it has only 0.02% abnormal values. To solve this problem, the authors used a sliding window algorithm and classified these windows. The created C-LSTM neural network combined Convolutional Neural Network (CNN) and Long-Short Term Memory Recurrent Neural Network (LSTM) layers that were connected in a linear structure. At first, spatial features in the traffic window were extracted by the convolutional and pooling layers of the neural network. Next, the temporal features were identified by the LSTM layer. The authors compared the results achieved by the proposed C-LSTM with other ML algorithms, and C-LSTM peaked the highest performance. They also tested the created C-LSTM neural network on another dataset [49], and C-LSTM showed good performance as well.

Zaman et al. [50] compared different machine learning algorithms for network intrusion detection. The authors analyzed K-mean, k-NN, Naïve Bayes, Fuzzy C-means, SVM, Radial Basis Function algorithms, as well as Ensemble, and combined the results of these algorithms to predict an intrusion.

Zhang et al. [45] considered a new anomaly detection method for network performance data. They concentrated on the following metrics: throughput, one-way delay, packet loss rate, and traceroute. The authors used the data collected from Worldwide LHC Computing Grid (WLCG) (a computing grid for the European Council for Nuclear Research (CERN) Large Hadron Collider (LHC) experiments) and Open Science Grid (OSG) meshes. As the network mesh size was significantly large, it was challenging to identify its anomaly source. Moreover, due to the high variance and quantity of data, it was impossible to develop normal network behavior models. Therefore, ML algorithms appeared to be the best solution to this research question. At first, the authors trained SVM, simple Neural Network (NN), and Boosted Decision Tree on the simulated annotated dataset. SVM suffered from the curse of dimensionality, so it was not tested on the real data. The real-world data is not annotated, unlike the simulated dataset, so it was an unsupervised learning task. Further, the authors conducted two experiments. The first one used Boosted Decision Tree to find anomalies in packet loss and one-way delay data. Changing the metrics threshold could achieve the desired sensitivity/false positive level for the decision tree. In the second experiment, NN detected anomalies in measured packet loss.

F. NETWORK TRAFFIC PREDICTION

The development of the Internet created a more prominent and more complex network architecture. One of the solutions of network optimization is traffic prediction, especially for cases of future wireless networks with strict latency and reliability constraints.

A.R. Mohammed et al. [51] review existing approaches for traffic classification and traffic prediction that use ML in a SDN context. This survey consists of brief explanations of traditional ML models and deep ML algorithms.

Chuang Song et al. [52] describe a dynamic traffic slice model based on ML (ML-TADS). This model allows to manage traffic in the network competently – to provide it in such a way that its distribution is uniform, there is no congestion at one BS, and, at the same time, zero traffic to another. Depending on the time of day, the amount of information transmitted over the networks increases or decreases. Also, the amount of traffic at a certain moment

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see https://creativecommons.org/licenses/by/4.0/
can be influenced by some event occurring near the user. If taking into account all conditions, it will be easier to predict traffic. Unfortunately, the authors do not mention the used data preprocessing methods. It allows one to think about other algorithms, such as compositional ones. Indeed, their learning rate can take significantly less time than a neural network with forwarding and backward propagation. In this case, it will make the system even more dynamic.

To offload the network links by predicting the traffic, a network operator can effectively predispose resource-allocation strategies to early address such challenging situations. That is the main motivation of the work by Andreoletti et al. [53]. The authors employ Diffusion Convolution Recurrent Neural Network (DCRNN) to forecast traffic load on a real backbone network’s links. Results show that DCRNN outperforms the other methods (CNN, LSTM) in terms of forecast accuracy.

Sun et al. [54] investigate the effectiveness of various ML models in terms of the prediction accuracy and computational time cost. They analyze how to identify the crucial factors limiting ML-based prediction models’ recourse to support real-time services. The dataset was collected from three traffic flow detectors every 15 minutes for evaluation. Among different algorithms, the authors are using SVR, ANN, DNN, and LSTM. The best result in the prediction accuracy for the proposed scenarios was shown by SVR. Current transport infrastructure consists of many road sections with different traffic patterns. Therefore, the authors propose to use in future reinforcement learning algorithms, which will manage the transport system more efficiently.

Diogo Clemente et al. [55] proposed a methodology to improve the accuracy of the cellular BS traffic prediction using ML. The system combines the Naive Bayes classifier and Holt-Winters ML model to improve traffic forecast on a cloud-based platform.

Yaghoubi et al. [56] compared different Supervised Learning algorithms for traffic flow estimation. They evaluate several different classical regression algorithms, including SVR, Kernel Ridge, Decision Tree, Random Forest, and LSTM on the data collected from six different locations around inner Stockholm. Results demonstrated excellent performance using both Random Forest and LSTM. One of the drawbacks of this study is that researchers have limited ground truth data as more data and more locations may have improved the algorithm’s performance.

G. SECTION SUMMARY

This section outlined the main ML techniques that are applied to signal integrity, network organization, reducing the latency, building an intelligent security block for modern networks, detection of anomalies, and traffic prediction. Table 2 presents a summary of the literature review with respect to various ML applications to the communications. It is divided into six sections according to this section’s structure.

Significantly, there are still plenty of cases for ML in the network and communications that were not mentioned in this section. For example, the Q-learning algorithm makes it possible to build an optimal route from a user to a BS in virtue of the bandwidth of network nodes and distance [57]. A different approach of the 5G BS distribution for platooning vehicles underway is shown in [58]. ML application for optimizing a waveguide slot antenna could increase the gain [59]. ML could be used to select parameters that allow customizing the user’s signal more individually [60]. Also, ML could be applied to assess the KPI of cellular communications [61]. There are many areas of application of ML in telecommunications, and each algorithm successfully copes with its task, which may indicate the success of the implementation of the ML technology. The number of directions is enormous but, following our research questions, we proceed with the cellular network traffic prediction scenario.

IV. COMPARING VARIOUS ML ALGORITHMS FOR LTE FEATURES’ PREDICTION

Figure 2 shows the general process of working with data and obtaining a traffic prediction working model.

**FIGURE 2:** Algorithm for creating a traffic predicting model

The data for this work was taken from the website kaggle.com, a repository of community-published measured or generated datasets. The collection of the dataset “Predict
| Ref. | Aim of the study & Major findings | Target variable | Pr. type | Metrics | ML model | Merits (M) / Demerits (D): Only provided if identified |
|------|----------------------------------|----------------|---------|---------|----------|-----------------------------------------------------|
| [28] | Incorporating Fog into heterogeneous networks to reduce latency in wireless networks | Locations of FN | CLU | Network delay | Soft clustering algorithm (proposed) | M: reduces the latency<br>D: latency decreases with bandwidth until it saturates |
| [30] | A study of latency prediction on commercial mobile carriers | RTT | CLS | Pr, TPR, F1 | Logistic regression | M: simplicity of the algorithm<br>D: shows low cross-validation performance |
| [17] | To present an ML-based framework to optimize the operation of 5G networks | Packet delay | CLS | N/A | Decision Tree | M: shows the best cross-validation performance |
| [33] | To propose a three-layer network protection system | Malicious /safe group | REG, CLS, FPR | Random Forest | M: has a significant impact on identifying features<br>D: the inability to generalize various attacks, without the first “meeting” them in the training sample; case of overfitting |
| [18] | To implement the DQL algorithm for the optimal data and energy management problem in the federated learning | Training latency, energy efficiency | REG | N/A | DQL | M: enables the server to find the optimal data through federated learning without any a priori knowledge of network dynamics; reduces the training latency<br>D: results are obtained with the assumption that the mobile devices in the network have the same data quantity parameters |
| [34] | To predict DoS and DDoS attacks in SDN | Attack / normal traffic | CLS | Acc, Pr, F1 | SVM | M: can learn with very limited amount of data and provide good results<br>D: small errors for long periods of time |
| [35] | To show performance prediction problems related to mobile network optimization and management as well as potential ML solutions for a real-world data set on measured mobile network performance | RSS, bit rate, SNR | REG | Gaussian process regression | M: easy to infer the reliability of the results by knowing the variance of the process<br>D: ineffective with more chaotic data (e.g. baud rates) |
| [37] | To predict the datarate using ML models | Datarate | REG | Cor, MAE, RMSE | NN, SVM, Decision Tree, Linear regression | M: decision tree outperforms other models<br>D: slow model performance |
| [39] | To design an application of DRL for 5G network slicing optimization | GSQ and BeSQ slice requests number | REG | N/A | DRL | M: quickly adjusts its policy through the learning process to serve more slice requests from GSQ to improve the utility<br>D: light average queue length of BeSQ comparing with GSQ, because the slice-allocation controller prefers to serve the slice requests from GSQ |
| [36] | To design ML-based algorithm for predicting the network type of the source and destination nodes | Type of the network | CLS | AUC, ROC, Acc | Multinominal logistic regression | M: less flexible algorithm structure (compare to Prophet algorithm)<br>D: higher reliability costs (compare to Holt-Winters algorithm) |
| [38] | To create ML-based mobility management algorithm for industrial NB | Users ‘mean opinion score’ | N/A | N/A | Proposed method | M: shows high reliability, delay-tolerant |
| [41] | To predict eye-diagram metrics of high-speed channels for signal integrity analysis using ML | Eye height and eye width | REG | RMSE, max RE | Linear regression | M: simple to implement<br>D: fails to make accurate predictions |

**TABLE 2: ML approaches used for telecommunications**
| Ref. | Aim of the study & Major findings | Target variable | Pr. type | Metrics | ML model | Merits (M) / Demerits (D) |
|------|----------------------------------|----------------|----------|---------|----------|--------------------------|
| [42] | To predict the eye-diagram metrics using fast-training semi-supervised learning method based on HNN | Eye height and eye width | REG | RMSE, max RE | HNN | M: law time cost, high prediction accuracy; a few data with labels and reasonable amounts of unlabelled data are needed for training | D: complicated architecture |
| [43] | To construct predictive model for eye opening based on channel length and interconnect cross-sectional geometry | Eye height and eye width | REG | max RE | Sparse grids | M: good performance | D: require all data available during training |
|      |                                   |                |          |         | SVR      | M: has the best performance in terms of prediction error | D: require all data available during training |
|      |                                   |                |          |         | ANN      | M: flexible since it can stream data as they become available | D: result depends on the number of hidden layers |
| [44] | Comparing ML techniques for efficient and reliable PDN and SI optimization | Eye height and eye width | REG | RE | NN, SVM, Knnig | M: Knnig shows the overall best generalization performance |
| [50] | To detect traffic anomalies (intrusions) using different ML algorithms and compare these algorithms results | Intrusion | CLS | Acc, Pr, TPR, ROC K-Means | D: a lot of false negative results |
|      |                                   |                |          |         | KNN      | M: best results on precision and accuracy metrics |
|      |                                   |                |          |         | Fuzzy C- Means | D: a lot of false negative results |
|      |                                   |                |          |         | SVM      | M: good model performance on all metrics |
|      |                                   |                |          |         | SVM      | D: middle quality model performance |
| [47] | To create a C-LSTM neural network, which can extract patterns (and find anomalies) in traffic data containing spatial-temporal information | Anomaly / normal traffic | Acc, F1 | C-LSTM | Ensemble | D: Combines the results of 6 previous algorithms and use majority voting for the prediction |
| [55] | To detect unusual events in network performance data using ML models | Traffic anomaly | CLS | Acc, AUC | Decision Tree | M: very fast execution time | D: metrics threshold must be changed manually to achieve a good performance |
|      |                                   |                |          |         | SVM      | D: problems with the curse of dimensionality |
|      |                                   |                |          |         | Neural network | D: requires GPU for processing; much slower than decision tree |
|      |                                   |                |          |         | Holt-Winters algorithm | D: assumes that all features are independent |
| [52] | To propose system that allows to predict cell usage with greater accuracy | Traffic | REG | RMSE | The Naive Bayes classifier | M: accuracy for seasonal series, simplicity and reduced computational demand |
| [53] | To describe a dynamic traffic slice model that allows to competently manage traffic in the network | Traffic | REG | MAE | ML-TADS | D: do not mention data preprocessing methods |
| [54] | To investigate the effectiveness of various ML-based prediction models by considering both the prediction accuracy and computational time cost | Traffic | REG | MSE, MAE, $R^2$ | SVR | M: relative accuracy in long and short time-series predictions, easy to build and use, overcomes vanishing gradient problem | D: the essential task for the design of SVR-based predictors is the correct deriving optimal control parameters and selecting the kernel function |
|      |                                   |                |          |         | ANN      | M: relative accuracy in long- and short time-series predictions, easy to build and use, overcomes vanishing gradient problem | D: number of hidden layers, number of neurons, as well as learning methods (i.e., the way to adjust the weights) directly affect the performance |
|      |                                   |                |          |         | LSTM     | M: provide fast traffic prediction service and has the ability to capture the non-linear components of traffic flow data | D: high training cost, not scalable |
| [56] | To present a novel method for traffic flow estimation using standardized LTE radio frequency performance measurement counters | Traffic aggregated over 15-minute intervals | REG | $R^2$ | SVR | M: can solve non-linear problems; good at high dimensional data | D: inefficient on big amount of data; requires choosing the right kernal function |
|      |                                   |                |          |         | Decision Tree | M: perform reasonably well in terms of temporal generalization | D: requires setting its internal parameters |
|      |                                   |                |          |         | Random Forest | M: the algorithm yields the highest $R^2$ score on temporal generalization | D: the algorithm does not always capture the peaks, requires setting its internal parameters |
|      |                                   |                |          |         | Kernel Ridge | D: requires setting its internal parameters |

**REG** – Regression  **CLS** – Classification  **CLU** – Clustering  **N/A** – not available in the paper
traffic of LTE network” took place for one year. A cellular network served the mobile device when a subscriber used a mobile data service. As a result, the dataset contains 497,544 strings with traffic information obtained from 57 cells. The entries contain temporal data, and this work focuses on an in-depth analysis of the ML techniques suitable for predicting cellular network traffic based on time. Notably, the dataset can not be transformed into a spatial or spatiotemporal-based one.

The paper considers linear ML models (Linear Regression, SVM, Huber Regression, Bayesian Regression) and ensembles (Bagging, Random Forest, and Gradient Boosting). It is not too complicated to justify the choice of linear models: they are interpretable and quickly training. Linear models, unlike ensembles, require data preprocessing (e.g., normalization for scaling). Also, linear models cannot recover nonlinear dependencies without data preprocessing. Therefore, ensemble models were used to eliminate the possibility of missing complex data correlations. One can greatly increase the quality with proper tuning of their hyperparameters. However, it is worth remembering that the selection of hyperparameters for ensembles is time-consuming.

The primary programming tool of this work is Python language for several reasons. First of all, it is fairly straightforward to use because it has an intuitive language syntax. Thus, it is widely applied by professionals at each level of their research/engineering careers. Second, it has a large selection of libraries and frameworks for ML models. The last reason is that it is a modern programming language that easily integrates with other languages if necessary. Python 3.7 and Jupyter notebook software were applied for modeling the prediction algorithm. The data analysis and visualization Libraries are NumPy, Pandas, Scikit-learn, Scipy, matplotlib, seaborn. The link to the Open Access repository with all the implemented tools is available before the abstract.

The following subsections describe the model development stages in detail, namely: the data preparation, the data analysis, the model applicability, and the result analysis.

A. DATA PREPARATION

The initial step of working with data is to consider the data format and structure. The used dataset contains 497,544 items, i.e., traffic on a cellular communication cell, where 57 cells are unique. Each item has three attributes: date (when traffic arrived), time (at what time traffic arrived), cell name (where the traffic arrived). Traffic is the target variable that needs to be predicted.

a: Removing unnecessary information and gaps

The collected data is often not ideal and cannot be immediately used. It might contain gaps or unnecessary information that will load the algorithm, and as a result, the algorithm will not give an accurate prediction. Various reasons cause gaps in the collected telecommunication data: system problems, packet loss, interference, etc.

The Date column is split into three: Month, Day, and Year. The original Date string is removed. Thus, the number of features increased by two. Additionally, the applied data is clean, i.e., it does not have any gaps.

b: Converting categorical features

Since machines only understand binary code, all data must be presented in numerical form. There are sophisticated algorithms, such as XGBoost or Catboost, that can handle categorical data types. In this work, algorithms are used that cannot do this, so all categories must be encoded. The OHE technique is used to change the data type from categorical to numeric. The column CellName is converted using the mentioned technique.

c: Feature scaling

The linear SVM algorithm requires normalized data for its work because linear models’ accuracy may increase if the feature is distributed similarly to the normal law. For this purpose, Month and Day features were scaled by the interval [0, 1] using min-max normalization. Initially, the traffic was distributed according to the exponential distribution, but assuming that $\text{Traffic}_j = \ln(\text{Traffic}_j)$, we obtain a normal distribution, as depicted in Figure 3.

![Observed traffic distributions](image_url)

FIGURE 3: Observed traffic distributions

Next, Figure 4 shows the distribution of the LTE traffic that was observed during one year with a start in November.
The time dependence of the average traffic for all cells was considered in detail, see Figure 5.

**FIGURE 4:** Observed traffic distributions by months

The next step is to evaluate the dependencies between features and the target variable (traffic) for the algorithm’s correct work. The correlation is shown as a heat map in Figure 7. Here, the correlation in modulus between *Month* and *Year* is very high (there are only three months from 2017 and ten months from 2018 in the observed data). This correlation has an informational nature since it does not have a strong influence on the result. In any case, it is an unnecessary correlation, so this dependency was removed by converting the *Year* with OHE technique.

**FIGURE 6:** Part of cells traffic for the entire time

**FIGURE 5:** The distribution of cumulative traffic on a specific day of the month from all subscribers

**FIGURE 7:** Correlation between features and target variable (traffic)

**B. DATA ANALYSIS**

Data analysis is an essential part of the model’s correct operation. It analyzes the graphical presentation of the data to evaluate the distributions. At this step, we studied the variable’s dependence on the features, evaluated this dependence, found out if there is a correlation between features, etc.

**a:** Three groups of cells

Cells could be divided into three groups: dark-blue – with high traffic (10% of the total number of cells), blue – with low traffic (also 10% of all cells), and light-blue (80% of cells), which have medium traffic load. Some of these cells are shown in Figure 6. For example, we can intimate about the radio transmitter’s location – high-traffic ones are most likely located in the city center or in places of large crowds, where various public events can be held. Besides, this is a clear indicator that more control is needed for the blue cells since if they go out of operation, then many users will be “cut off” from the network.

**FIGURE 6:** Part of cells traffic for the entire time

**FIGURE 7:** Correlation between features and target variable (traffic)
C. NUMERICAL RESULTS
Computational platform wise, two different approaches were utilized. The first one is the Google Colaboratory server that was used as a computational platform. The server allocates resources dynamically and does not have the exact limit. Thus, the simulation device parameters could not be specified. The second one is a laptop Intel Core i7 CPU and 8GB RAM that was used to check the wall time of learning with the specified parameters. The model’s wall time of learning is highly dependent on computing capacity, so far presented wall time results are given for the laptop scenario, i.e., when the amount of computational power is fixed.

| Aim | Traffic prediction |
|-----|-------------------|
| Problem type | Regression |
| Target variable | Traffic |
| Features | Hour, Month, Day, Year, CellName |
| Metrics | RMSE, MAE, $R^2$ |

The following describes the problem statement in terms of ML and the main information is detailed in Table 3. The formulated problem type for this work is Regression, as the aim is to get the number as the output. The chosen target variable is traffic in one specific cell. Chosen for operation features are the time (Hour), month (Month), day (Day), binary features of the cellular communication cells (CellName) and Year (2017 and 2018). Following metrics will evaluate the models performance – RMSE, MAE and $R^2$. In MAE and RMSE metrics, the ideal algorithm reaches zero, and in the determination coefficient $R^2 = 100\%$.

Table 4 summarizes the results of the work. The wall of learning is highly dependent on computer capacity and could change from simulation to simulation, so it should be considered in comparison with other models, but not like absolute values. The higher percentage of $R^2$ shows a better correlation between predicted and observed variables, while in the case of RMSE and MAE the smallest value means the best performance. In overall, the quality of the models in this research is limited by the lack of data in the dataset. The next subsection describes the main conclusions based on the results of this work.

First, linear models are slightly worse than ensembles and require additional preprocessing in the form of standardization, but their training requires a shorter time. The open question remains what is more critical for the task – good quality or the learning rate. Sometimes it is worth having a fast model and, for example, train it online, thereby correcting its errors.

Second, Gradient Boosting showed the best result of all nonlinear models. It explains the determination in the data quite well ($R^2 = 60.2\%$). Boosting is an ensemble model combining several “weak” models with low-performance accuracy. Despite the high results in prediction accuracy, this algorithm naturally requires a large amount of training time.

Third, Random forest showed the worst result, but it should not be ruled out because it depends on the number of features. The utilized dataset has data only from 57 cells, which is not enough for this algorithm. Therefore, the Random Forest can improve its metrics by generating new features.

Fourth, it is worth paying attention to the Bayesian probabilistic model. It showed comparable results with the decision tree model. One explanation for this relatively good result is that some of our data is random since we previously discussed that the distribution of traffic in a particular part of the area could be influenced by a different number of statistically difficult factors to take into account.

Fifth, the selection of hyperparameters for Bagging and Gradient Boosting is a rather laborious task. It is worth noting that the optimal parameters were selected for linear methods by using GridSearch, and it took only a few seconds. Still, ensembles cannot do this since the selection takes a long time and requires colossal computing consumption. GridSearch is an approach to parameter tuning that will methodically build and evaluate a model for each combination of algorithm parameters specified in a grid [62].

Note, the quality of the models in this work is limited by the lack of data in the dataset, e.g., $R^2 = 60.2\%$ is a good result. Also, it is worth noting that it is more important to predict the approximate value of the traffic than its exact value. Good results on other Representative metrics (MAE and RMSE) show that the models’ quality of prediction is rather high.

In reality, the data may be affected by various aspects, e.g., noise or anomaly data. To analyze this influence on predictions’ quality, we added Gaussian noise to the dataset and re-executed all the considered models on the modified data, and compared the results with the original dataset. Gaussian noise is statistical noise, having a probability density function equal to the one of normal distribution. In our case, normal distribution ranged from 0 to 1 and was used to add training (test) sample and noise. Note, adding noise is sometimes used to reduce the overfitting in ML models.

The experiment proved that noise decreases the quality of predictions on a considered dataset on all metrics, see Table 5 and Figure 8. $R^2$ of Bagging and Random Forest models degrade the most being based on the bootstrap method, where the dataset is randomly split into $N$ different smaller datasets with replacement. At the same time, Gradient Boosting (another ensemble model) shows a less significant decrease in $R^2$ in comparison with bootstrap-based models. The main reason for this difference is that Bagging models are trained on smaller datasets, where noise is higher. In contrast, Boosting algorithm is used in Gradient Boosting instead of Bagging in Random Forest or conventional Bagging models. Not-ensemble models show approximately the same decrease on $R^2$, which means that they are equally optimal for using data with noise. To conclude, if the data is expected to have a high noise level, it is better not to use models based on the bootstrap method.

To summarize, Gradient Boosting showed the best result
TABLE 4: Algorithms performance evaluation

| ML algorithm     | RMSE [Mbit] | MAE [Mbit] | $R^2$ [%] | Wall time of learning [s] | Advantages                                                                 | Disadvantages                        |
|------------------|-------------|------------|-----------|--------------------------|---------------------------------------------------------------------------|--------------------------------------|
| Bagging          | 2.59        | 1.66       | 50.8      | 58.1                     | Good quality of prediction. Minimizing model’s overfit.                   | Long time of training.               |
| Random Forest    | 3.38        | 2.19       | 34.2      | 172.0                    | Universality.                                                             | Low metrics compared to Gradient Boosting. Long time of training.            |
| Gradient Boosting| 2.18        | 1.43       | 60.2      | 24.6                     | High quality of prediction.                                              | Dependence on feature linearity and scaling.                                 |
| Linear Regression| 2.64        | 1.71       | 49.7      | 2.5                      | Simplicity of the algorithm and fast training.                           | Inference of the model can be time consuming.                                |
| Bayesian Regression| 2.25      | 1.40       | 49.7      | 1.1                      | Adapts to the data at hand.                                              | Dependence on feature linearity.                                              |
| Huber Regression | 2.43        | 1.62       | 26.8      | 4.0                      | More efficient to use on data with small number of samples.              | Needs qualified data preprocessing.                                          |
| SVM for regression | 2.68       | 1.69       | 48.8      | 7.7                      | Works well on standardized samples. Fast training.                       |                                                                                   |

TABLE 5: Comparing the impact of added noise with original dataset

| Approach          | RMSE decrease | MAE decrease | $R^2$ decrease |
|-------------------|---------------|--------------|----------------|
| Bagging           | 17.87%        | 20.65%       | 37.52%         |
| Random forest     | 8.79%         | 8.62%        | 35.29%         |
| Gradient boosting | 13.91%        | 14.77%       | 19.3%          |
| Linear regression | 10.07%        | 11.11%       | 20.32%         |
| Bayesian regression| 19.76%       | 10.75%       | 20.33%         |
| Huber regression  | 20.95%        | 10.1%        | 21.18%         |
| SVM for regression| 10.31%        | 10.1%        | 22.34%         |

from the set of all considered ML algorithms. However, it does not exactly mean that this algorithm will show the best result in real-life conditions for two reasons. This algorithm was tested on a big but still limited dataset and, as with every ML solution, we can not guarantee that this algorithm will show the best results on another dataset, for example, with data from another region where the data structure and patterns will be different. Gradient Boosting algorithm has a low tolerance to the outliers.

The main conclusion from the above is that the Gradient Boosting will perform better on those data, which distribution is similar to the data on which it was trained. Huber regression also showed good results, especially on MAE metrics. In contrast to Gradient Boosting, this algorithm better adapts to new data. It means that even though the results on metrics of this algorithm are lower in comparison with Gradient Boosting, it will show better results in cases such as the traffic prediction that constantly changes its distribution.

V. CURRENT CHALLENGES AND FUTURE PERSPECTIVE

The implementation of the ML algorithms in various aspects of human life could improve the system performance, yet, it brings many challenges. The use of ML algorithms in the network causes several problems related to the limited computing resources, poorly cleaned input data, etc. This section discusses research challenges that should be taken into consideration while solving the problems related to the ML applications. The most significant challenges are summarized in Table 6.

One of the main challenges that developers face working with data is the complexity of data processing. Most of the ML algorithms are very sensitive to the input data, so it must be presented in an accessible form suitable for the algorithm. However, we can not use raw datasets directly in many cases [30], [45], [64], [65]. Data collected from the real world are not always perfect: it might consist of gaps, categorical features, background clutter, or non-required information that could worsen the model’s performance. Therefore, before applying ML model, it is necessary to present them in the required form for processing, not to overset the algorithm’s correctness.

ML models use many features, many of which are computationally challenging [30], [66], [67]. For example, the model’s evaluation in terms of training time depends on the hardware where ML is implemented. If the number of inputs and features increases, researchers need to increase the hardware capacity to extend the processing limits. A large number of features also causes high complexity of the model and overfitting [30].

Limited storage capacity is one more challenge that appeared during the ML evolution [63]. An increase in input data and the need for programmability increase computation. ML algorithms require a data processing methodology as well as a need to have significantly massive storage for computing.

ML model’s robustness is highly dependent on the algorithm performance. The correct choice of parameters for model evaluation might give good results, but each model has its limitations. With changing data, some algorithms need other parameters or are not able to be executed efficiently [17], [18]. Traffic might be changed
considering the development of heterogeneous networks, and ML algorithms might be able to handle new inputs. Reinforcement learning has an advantage as it receives feedback from the environment and makes the decision according to the collected data [7, 35, 41]. It has a vast potential of detecting the emergence of new types of traffic. However, this is an open question for researchers.

The authors of [28] highlight another challenge of neglect of critical features. The work proposes the ML clustering algorithm significantly reducing latency. For achieving the demanding latency requirement of 5G systems in 1 ms, it might be necessary to employ open-loop communication at the expense of reliability. One of the promising solutions is applying DC-NC based on the synergistic combination of diversity coding and network coding [68].

Another challenge is related to the interface integration through which the performance data is obtained. Several incompatible drive test systems are currently in use, creating a high need for a coordinated interface and data formats. Such integration would also facilitate the portability of already trained ML models between different systems [35].

One more challenge is related to the scalability of the model. Scalability is the computational feasibility of ML models with a growing amount of data. Random Forests is well suited in this case, as it enables massive parallelization of model training and applications [35].

Regarding the research executed in this paper and the potential improvements of the analyzed algorithms for their utilization in real conditions, the following approaches could be followed:

1) Regularization could be added to the algorithms so that those fit the data more efficiently [69].
2) Introduce new optimization methods, or, e.g., develop a loss function, which will take into account both the main KPIs and improve the classic metrics (MAE, RMSE) in more detail [69].
3) Add new indicators (features) to the algorithms allowing for more knowledge about the system, e.g., the population in which the cell is located, the welfare index, which, in theory, will correlate with the ability of people to use the Internet, the number of operators in the hundredth communication, and so on. It is worth noting that the improvement of the basic metrics of the models is usually related to feature engineering and not tuning hyperparameters. By feature engineering, we understand data accounting not at a specific time but, e.g., by using a moving average [70].
4) Apply multiple approaches simultaneously, e.g., to combine clustering with a subsequent regression task. In the beginning, clusterization cellular cells according to key indicators that the model detects, and then separately for each cluster build a relatively simple model.
5) It is worth using algorithms on strictly controlled data and implementing the Data Quality process [71], i.e., to introduce additional monitoring of model indicators not only for key metrics but also for additional mathematical rules, for example, Population Stability Index (PIS), Kolmogorov-Smirnov test, Herfindahl-Hirschman Index, etc. [72].
6) Use algorithms for automatic retraining (AutoML) to provide a better reactive response to changes in data.
7) To apply the algorithms in lower-level programming

### TABLE 6: Major identified challenges

| Challenges | Group | Ref. | Potential solutions found in the literature |
|------------|-------|------|------------------------------------------|
| A need of data to be presented in a required form suitable for the algorithm | DP | [30], [45], [63]–[65] | To use data preprocessing techniques such as OHE, Min-Max Scaling, and others |
| Data quality related problems | DP | [47] | To use a sliding window algorithm solving the problem of data imbalance |
| | | [45] | To train models on artificial labeled dataset and then use trained models on real world unlabeled data |
| | | [34] | To create a dataset with the traffic flow generator |
| | | [36] | To use random oversampling technique to create more data and solve the problem of the data imbalance |
| A growing number of features that need processing | HW, SW | [30], [66], [67] | To improve the hardware |
| Overfitting | DP | [30] | To remove irrelevant features or noise |
| Potential neglection of critical features | N | [28] | To employ the Diversity Coding-Network Coding (DC-NC) in the network that improves latency but not at the expense of reliability |
| A need for a unified interface and data formats | A | [35] | Harmonization of interfaces |
| Limited storage capacity | A, HW | [63] | To move the unwanted inputs to backup storage/improve hardware |
| The emergence of new types of input data | A, DP | [7], [17], [18], [35], [41], [66] | Potential in using Unsupervised- and Reinforcement Learning |
| Limited scalability of ML models | DP, A, SW | [35] | To develop new algorithms better suited for complex systems |
| Appearance of anomalies in the network operation | DP | [45], [47] | Finding abnormal traffic patterns using machine learning algorithms |
| | | [34], [50] | Detecting attacks using data analysis algorithms |

A – Architecture N – Networking DP – Data Processing HW – Hardware specific SW – Software specific

| Challenges | Group | Ref. | Potential solutions found in the literature |
|------------|-------|------|------------------------------------------|
| Appearance of anomalies in the network operation | DP | [45], [47] | Finding abnormal traffic patterns using machine learning algorithms |
| | | [34], [50] | Detecting attacks using data analysis algorithms |
VI. DISCUSSION AND CONCLUSIONS

The development of both ICT and ML domains pace the way of the inevitable development on the edge of both multidisciplinary fields. In this work, we first executed a literature review on the applications and challenges of ML to future network problems. We identified that ML is already being applied in delay optimization, signal integrity, information security, network optimization, and network management, anomaly detection, and traffic prediction. Nonetheless, the rapid growth of the devices and generated traffic causes such challenges as limited scalability of the ML model, the emergence of new types of input data which is not familiar to the ML algorithm, a growing number of features that need processing, and others.

Notably, the most suitable models must be selected in dependence on the general optimization goal. The algorithm’s choice depends on various aspects, including the purpose, input data type, available resources, etc. Comparing chosen algorithms may not be straightforward since a peculiar one can show different results under different circumstances. The ML model robustness is highly dependent on the algorithm performance. The correct choice of parameters for model evaluation might give good results, but each model has its limitations.

Moreover, this work proved that ML could be easily applied to optimize traffic prediction the basic ML algorithms, such as Linear Regression, Huber Regression, Bayesian Regression, Gradient Boosting, Random Forest, Bootstrap Aggregation (Bagging), and SVM. It is proven to be possible to extract the information suitable for forecasting even from simple data and use ML to solve the optimization problem. The drawback of the proposed approach is that the algorithms operate separately from the real network. A possible solution for this problem could be additional overlays, where problems are solved by voting for the best result if the algorithms “argue” with each other. Finally, we have analyzed the impact of added noise on the dataset and its impact on the prediction quality. The results show that it is better not to use models based on the bootstrap method if a high level of noise or other anomalies are expected to be present in the data.

In summary, the intelligent combination and tuning of various ML techniques may significantly improve the prediction results. We have shown that the Bagging prediction quality could be improved by using GridSearch and by tuning hyperparameters. Next, the combination of several data scaling methods allowed to enhance the SVM as this algorithm works well on normally distributed data. Third, it is essential to generate new features for improving the Decision Tree algorithm execution. Next, as the majority of ML algorithms could operate only with numerical inputs, the application of new encoding approaches for converting the categories in data is crucial. Finally, the data might be classified according to the traffic load (high, medium, and low), and models could be trained on each class separately to improve the prediction results.

Overall, the ML application consists of three general parts: primary data preparation, data analysis, and predicting. The encountered problem is that many algorithms work correctly only with the preprocessed data. Data that the algorithm receives as input must be preprocessed quickly before predicting itself, or it must be taken out in a separate part of the system. This approach provides a higher level of confidence in the algorithm’s output data quality but excludes the possibility of training the model in real-time. In the second part, the data is analyzed to identify dependencies and patterns to understand which algorithm best suits this particular task, as algorithms taken from libraries of various programming languages may not
immediately provide a sufficiently good result. It is necessary to tune the hyperparameters according to the task of study for improving the prediction. The source code, as well as examples of the hyperparameters tuning, are available in Open Access (link is provided before the abstract).

The advantage of the algorithms selected in this work is, on the one hand, in the reproducibility for early-stage researchers/engineers/implementors. On the other hand, the simplicity of the algorithms is the main concept that we adhered to use fast algorithms, and not complex and more time-consuming to learn, even with a small loss in quality. That allowed the use of those for real traffic forecasting conditions (since it does not require considerable technical resources capable of performing a large number of data operations). Overall, the use of deep learning methods, possibly, will improve the quality. However, other factors should be considered here, e.g., the cost of mathematical operations, roughly speaking, the running time of the algorithm, which we attempted to minimize for a more efficient operation of the entire system. In contrast, neural network algorithms commonly require a large amount of high-quality data to operate effectively.

LIST OF ACRONYMS

| Acronym | Definition |
|---------|------------|
| 3GPP    | 3rd Generation Partnership Project |
| Aec     | Accuracy |
| AE      | Absolute Error |
| AI      | Artificial Intelligence |
| ANN     | Artificial Neural Network |
| AUC     | Area under the ROC Curve |
| BeSQ    | Best-effort Service Queue |
| BS      | Base Station |
| C-LSTM  | Convolutional Long-Short Term Memory Recurrent Neural Network |
| CAPEX   | Capital expenditure |
| CERN    | European Council for Nuclear Research |
| CNN     | Convolutional Neural Network |
| Cor     | Correlation |
| CoW     | Cell on Wheels |
| D-STM   | Deep Spatio-Temporal Neural Network |
| DBSCAN  | Density-based Spatial Clustering of Applications With Noise |
| DC-NC   | Diversity Coding-Network Coding |
| DCRNN   | Diffusion Convolution Recurrent Neural Network |
| DBoS    | Distributed Denial-of-Service attack |
| DNN     | Deep Neural Network |
| DQL     | Deep Q-Learning |
| DRL     | Deep Reinforcement Learning |
| DTN     | Delay-Tolerant Network |
| F1      | F1-score |
| FN      | False Negatives |
| FN      | Fog Nodes |
| FP      | False Positives |
| FPR     | False Positive Rate |
| GRNN    | Generalized Regression Neural Networks |
| GSQ     | Guaranteed Service Queue |
| HNN     | Hybrid Neural Network |
| HPN     | High Power Nodes |
| ICT     | Information and Communication Technologies |
| IDS     | Intrusion Detection Systems |
| IoT     | Internet of Everything |
| IoT     | Internet of Things |
| k-NN    | k-Nearest Neighbors |
| KDE     | Kernel Density Estimation |
| k-NN    | k-Nearest Neighbors |
| LHC     | Large Hadron Collider |
| LPN     | Low Power Nodes |
| LSTNM   | Long-Short Term Memory Recurrent Neural Network |
| LTE     | Long Term Evolution |
| LTT     | Long-Term Evolution |
| MAE     | Mean Absolute Error |
| MANET   | Mobile Ad Hoc Network |
| MAP     | Mean Average Precision |
| MAPAPE  | Mean Absolute Percentage Error |
| MCML    | Mobile Crowd-Machine Learning |
| MIMO    | Multiple-Input and Multiple-Output |
| ML      | Machine Learning |
| MLR     | Multinomial Logistic Regression |
| MRR     | Mean Reciprocal Rank |
| MSE     | Mean Square Error |
| NGN     | Next Generation Network |
| NIB     | Network in Box |
| NN      | Neural Network |
| OHE     | One-Hot Encoding |
| OPEX    | Operating expense |
| OSG     | Open Science Grid |
| PDN     | Power Delivery Networks |
| Pr      | Precision |
| PIS     | Population Stability Index |
| PSM     | Polynomial Surrogate Modeling |
| QoS     | Quality of Service |
| REM     | Radio Environment Map |
| RMSE    | Root Mean Square Error |
| ROC     | Receiver Operating Characteristic |
| RSS     | Received Signal Strength |
| RTT     | Round-Trip Time |
| SDN     | Software Defined Networks |
| SI      | Signal Integrity |
| SNR     | Signal-to-Noise Ratio |
| SoC     | System on Chip |
| SVM     | Support Vector Machines |
| SVR     | Support Vector Regression |
| TN      | True Negatives |
| TP      | True Positives |
| TPR     | True Positive Rate |
| UDP     | User Datagram Protocol |
| ULLC    | Ultra-Reliable Low-Latency Communication |
| WLCG    | Worldwide LHC Computing Grid |
| WSN     | Wireless Sensor Networks |

REFERENCES

[1] G. Kalfas, C. Vagionas, A. Antonopoulos, E. Kurtasli, A. Mesodiaikaki, S. Papaoaouano, P. Maniotis, J. S. Vardakas, C. Verikouskis, and N. Pleros, “Next Generation Fiber-Wireless Fronthaul for 5G mmWave Networks,” IEEE Communications Magazine, vol. 57, no. 3, pp. 138–144, 2019.

[2] I. Atov, K. C. Chen, A. Kamal, and M. Louita, “Data Science and Artificial Intelligence for Communications,” IEEE Communications Magazine, vol. 58, no. 10, pp. 56–57, 2020.

[3] B. Ji, Y. Wang, K. Song, C. Li, H. Wen, V. G. Menon, and S. Mumtaz, “A Survey of Computational Intelligence for 6G: Key Technologies, Applications and Trends,” IEEE Transactions on Industrial Informatics, 2021.

[4] Z. Ullah, F. Al-Turjman, L. Mostarda, and R. Gagliardi, “Applications of Artificial Intelligence and Machine Learning in Smart Cities,” Computer Communications, 2020.

[5] R. Pigmagedov, D. Molchanov, A. Ometov, K. Muhammad, S. Andreev, and Y. Koucheryavy, “Facilitating mmWave Mesh Reliability in PPRD Scenarios Utilizing Artificial Intelligence,” IEEE Access, vol. 7, pp. 180 700–180 712, 2019.

[6] N. Makitalo, A. Ometov, J. Kannisto, S. Andreev, Y. Koucheryavy, and T. Mikkonen, “Safe, Secure Executions at the Network Edge: Coordinating Cloud, Edge, and Fog Computing,” IEEE Software, vol. 35, no. 1, pp. 30–37, 2017.

[7] Z. Xiong, Y. Zhang, D. Niyato, R. Deng, P. Wang, and L.-C. Wang, “Deep Reinforcement Learning for Mobile 5G and Beyond: Fundamentals, Applications, and Challenges,” IEEE Vehicular Technology Magazine, vol. 14, no. 2, pp. 44–52, 2019.
[8] Q.-V. Pham, K. Dev, P. K. R. Maddikuntla, T. R. Gadekallo, T. Huyhn-The et al., “Fusion of Federated Learning and Industrial Internet of Things: A Survey,” arXiv preprint arXiv:2101.00798, 2021.

[9] A. Jindal, G. S. Aujla, N. Kumar, R. Prodan, and M. S. Obaidat, “DRUMS: Dynamic Resource Allocation and Management System for Resource Allocation and Management System (DRUMS),” in Proc. of IEEE Global Communications Conference (GLOBECOM). IEEE, 2018, pp. 1–6.

[10] S. A. Kumar and K. S. Murthy, “An Efficient Operations and Management Challenges of Next Generation Network (NGN),” Global Journal of Computer Science and Technology, vol. 14, no. 7, pp. 16–20, 2014.

[11] A. L. Beam and I. S. Kohane, “Big Data and Machine Learning in Health Care,” Jama, vol. 319, no. 13, pp. 1317–1318, 2018.

[12] A. Osseiran, F. Boccardi, V. Braun, K. Kusume, P. Marsch, M. Materna, O. Questeth, M. Schellmann, H. Schotten, H. Taska et al., “Scenarios for 5G Mobile and Wireless Communications: The Vision of the METIS Project,” IEEE Communications Magazine, vol. 52, no. 5, pp. 26–35, 2014.

[13] 3GPP TR 22.735 V16.2.0, “Study on enhancement of Ultra-Reliable Low-Latency Communication (URLLC) in support of the 5G Core network (5GC),” Rel. 16, June 2019.

[14] X. Wang, L. Zhu, F. Xiao, K. Xing, Z. Yang, Y. Liu, and C. Peng, “Spatio-Temporal Analysis and Prediction of Cellular Traffic in Metropolis,” IEEE Transactions on Mobile Computing, vol. 18, no. 9, pp. 2190–2202, 2018.

[15] M. Pasek, P. et al., “Improving the Precision of Wireless Localization Algorithms: ML Techniques for Indoor Positioning,” in Proc. of 43rd International Conference on Telecommunications and Signal Processing (TSP). IEEE, 2020, pp. 589–594.

[16] N. Richerzhagen, M. Stingl, B. Richerzhagen, A. Mauthe, and R. Steinmetz, “Adaptive monitoring for mobile networks in challenging environments,” in 2015 24th International Conference on Computer and Communications Networks (ICCCN). IEEE, 2015, pp. 1–8.

[17] C. Fiandrino, C. Zhang, P. Patras, A. Banchs, and J. Widmer, “A Machine-Learning-Based Framework for Optimizing the Operation of Future Networks,” IEEE Communications Magazine, vol. 58, no. 6, pp. 20–25, 2020.

[18] T. T. Anh, N. C. Luong, D. Niyato, D. I. Kim, and L.-C. Wang, “Efficient Machine Learning for Mobile and Wireless Communications: The Vision of the METIS Project,” IEEE Communications Magazine, vol. 52, no. 8, pp. 1348–1349, 2014.

[19] N. Stepanov, D. Alekseeva, A. Ometov, and E. S. Lohan, “Applying Machine Learning to LTE Traffic Prediction: Comparison of Bagging, Random Forest, and SVM,” in Proc. of 12th International Congress on Antennas and Propagation (APCAP). IEEE, 2017, pp. 1–2.

[20] A. S. Khatouni, F. Soro, and D. Giordano, “A Machine Learning Approach for Indoor Localization,” in Proc. of IEEE Global Communications Conference (GLOBECOM). IEEE, 2019, pp. 71–74.

[21] A. Jindal, G. S. Aujla, N. Kumar, R. Prodan, and M. S. Obaidat, “DRUMS: Dynamic Resource Allocation and Management System (DRUMS),” in Proc. of IEEE 36th International Conference on Communications and Computing Communications Conference (IPCCC). IEEE, 2017, pp. 1–2.

[22] N. Mäkitalo, et al., “Action-Oriented Programming Model: Collective Algorithms–The Numenta Anomaly Benchmark,” in Proc. of IEEE International Conference on Data Mining (ICDM). IEEE, 2019, pp. e1–e34, 2009.

[23] J. Zhang, R. Gardner, and I. Vukotic, “Anomaly Detection in Wide Area Networks for Low Latency Communications,” in Proc. of IEEE 36th International Conference on Communications and Computing Communications Conference (IPCCC). IEEE, 2017, pp. 1–2.

[24] T.-L. Wu, F. Buesink, and F. Canavero, “Overview of Signal Integrity and EMC Design Technologies on PCB: Fundamentals and Latest Progress,” IEEE Transactions on Electromagnetic Compatibility, vol. 55, no. 4, pp. 624–638, 2013.

[25] “AND9075/D Understanding Data Eye Diagram Methodology for Analyzing High Speed Digital Signals,” [Online]. https://www.onsemi.com/pub/Collateral/AND9075-D.PDF, (accessed on October 18, 2021).

[26] T. Wu, J. Sun, K. Wu, and Z. Yang, “High-Speed Channel Modeling with Machine Learning Methods for Signal Integrity Analysis,” IEEE Transactions on Electromagnetic Compatibility, vol. 60, no. 6, pp. 1957–1964, 2018.

[27] S. Chen, J. Chen, T. Zhang, and S. Wei, “Semi-supervised learning based on Hybrid Machine Learning for the Signal Integrity Analysis,” IEEE Transactions on Circuits and Systems II: Express Briefs, 2019.

[28] T. Lu, J. Sun, K. Wu, and Z. Yang, “High-Speed Channel Modeling with Machine Learning Methods for Signal Integrity Analysis,” IEEE Transactions on Electromagnetic Compatibility, vol. 157, p. 110391, 2019.

[29] A. S. Khatouni, F. Soro, and D. Giordano, “A Machine Learning Application for Latency Prediction in Operational 4G Networks,” in Proc. of IFIP/IEEE Symposium on Integrated Network and Service Management (IM). IEEE, 2019, pp. 71–74.

[30] N. Stepanov, A. Veprev, A. Sharaqova, D. Alekseeva, M. Komarov, E. S. Lohan, and A. Ometov, “On Machine Learning Applicability to Transaction Time Prediction for Time-Critical C-ITS Applications,” in Proc. of 44th International Conference on Telecommunications and Signal Processing (TSP). IEEE, 2021, pp. 408–413.

[31] D. Fang, Y. Qian, and R. Q. Hu, “Security for 5G Mobile Wireless Networks,” IEEE Access, vol. 6, pp. 4839–4874, 2017.

[32] J. Li, Z. Zhao, and R. Li, “Machine Learning-based IDS for Software-Defined 5G Network,” IET Networks, vol. 7, no. 2, pp. 53–60, 2017.

[33] A. Ahmad, E. Harjula, M. Ylianttila, and I. Ahmad, “Evaluation of Machine Learning Techniques for Security in SDN,” in Proc. IEEE Globecom 2016, pp. 1–6.

[34] J. Riihijarvi and P. Mahonen, “Machine Learning for Performance Prediction in Mobile Cellular Networks,” IEEE Computational Intelligence Magazine, vol. 13, no. 1, pp. 51–60, 2018.

[35] S. Chaudhary and R. Johari, “Oruml: Optimized Routing in Wireless Networks Using Machine Learning,” International Journal of Computer Communications and Network Systems, vol. 33, no. 11, e4394, 2020.

[36] B. Sliva, T. Liebig, R. Falkenberg, J. Pillmann, and C. Wietfeld, “Efficient Machine-Type Communication Using Multi-Metric Context-Awareness for Cars Used as Mobile Sensors in Upcoming 5G Networks,” in Proc. of IEEE 87th Vehicular Technology Conference (VTC Spring). IEEE, 2018, pp. 1–6.

[37] A. S. Khatouni, F. Soro, and D. Giordano, “A Machine Learning Approach for Indoor Localization,” in Proc. of IEEE Global Communications Conference (GLOBECOM). IEEE, 2019, pp. 71–74.
