INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS, 2023, VOL. 69, NO. 4, PP. 751-762
Manuscript received July 2, 2023; revised October, 2023. DOI: 10.24425/ijet.2023.147697

Long-term Traffic Forecasting in Optical Networks Using Machine Learning

Krzysztof Walkowiak, Daniel Szostak, Adam Włodarczyk, and Andrzej Kasprzak

***Abstract*—Knowledge about future traffic in backbone optical networks may greatly improve a range of tasks that Communications Service Providers (CSPs) have to face. This work proposes a procedure for long-term traffic forecasting in optical networks. We formulate a long-terT traffic forecasting problem as an ordinal classification task. Due to the optical networks' (and other network technologies') characteristics, traffic forecasting has been realized by predicting future traffic levels rather than the exact traffic volume. We examine different machine learning (ML) algorithms and compare them with time series algorithms methods. To evaluate the developed ML models, we use a quality metric, which considers the network resource usage. Datasets used during research are based on real traffic patterns presented by Internet Exchange Point in Seattle. Our study shows that ML algorithms employed for long-term traffic forecasting problem obtain high values of quality metrics. Additionally, the final choice of the ML algorithm for the forecasting task should depend on CSPs expectations.**

***Keywords*—Traffic forecasting; Machine Learning; Classification; Regression**

I. Introduction

COMPUTER networks are an integral part of contemporary life. Quick and global development of telecommunication technologies such as VoD, a cloud computing or the Internet of things causes rapid growth of endpoint devices [1]. According to the Cisco Annual Internet Report, the number of Internet users will reach 5.3 billion by the end of 2023 [2]. Moreover, the Nokia report [3] shows that as a result of the COVID-19 pandemic, in the first weeks of lockdown, compared to pre-pandemic time, network traffic increased by 30-50%. Additionally, by September 2020, traffic has stabilized at 20-30% above pre pandemic level. To prevent the possible capacity crunch problem on the Internet, network operators constantly improve backbone networks using various optical technologies [4], [5]. However, constantly growing network traffic, the increase of which is sometimes rapid in a short time, presents new challenges to Communications Service Providers (CSPs). To improve the performance of future optical networks compared to mechanisms currently used in optical networks, the concept of a cognitive optical network [6] has been proposed. In more detail, a cognitive optical network is based on a cognitive process that monitors current network conditions and adjusts the network operation to observed conditions. The cognitive process, which often uses history to improve operation, usually applies Machine Learning (ML) algorithms [7]. ML techniques can be successfully applied to analyze and find dependencies in historical data, e.g., traffic flows. Gained knowledge can be used to forecast future traffic in the network and later as valuable information for different network optimization tasks, e.g., traffic flow control, network operational cost reduction, anomaly detection, or physical network expansion [8], [9]. In this paper, we present a procedure for long-term traffic forecasting in backbone networks.

Nowadays, the means of communication used as backbone networks, carrying voluminous, aggregated user data traffic, are optical networks [10]. They use fibers linked into one physical cable as a transmission medium. Using the wavelength division multiplexing (WDM) technique, data is transferred using optical channels transmitted at different wavelengths. Currently, in WDM one wavelength offers capacity of 100 Gbps. Optical networks are constantly being improved and developed. A next-generation of optical networks architecture called Elastic Optical Networks (EON) [11], [12] improves the network operation and management. In EONs, a single optical channel supported by a single transceiver can carry a fixed amount of data. For instance, the Ciena WaveLogic 5 transceiver depending on the selected modulation format offers the following capacity of an optical channel: 200 Gbps, 400 Gbps, 600 Gbps or 800 Gbps [13]. As a result, to establish a connection, CSPs need information about the number of optical channels required to carry the requested network traffic. Therefore, in this work, we realize the problem of traffic forecasting by predicting future traffic levels rather than the exact traffic volume.

Although the work is focused on forecasting traffic in optical networks, many network technologies are based on the approach of provisioning the network capacity in some granularities, namely, an Optical Transport Network (OTN) various versions of Ethernet (e.g., Gigabit Ethernet, 10 Gigabit Ethernet, 40 Gigabit Ethernet, 100 Gigabit Ethernet). Therefore, the methods and results reported in this article can be applied to various types of network technologies.

The main contributions of this work can be summarized as follows:

- We design and implement historical data flows preprocessing methods. Each dataset used during experiments was initially analyzed using statistical methods. The analysis included dataset values' variations

This work was supported by National Science Centre, Poland under Grant 2017/27/B/ST7/00888.

Authors are with Wroclaw University of Science and Technology, Poland (e-mail: krzysztof.walkowiak@pwr.edu.pl, daniel.szostak.94@gmail.com).

and amplitude, their relation with time, traffic flows shapes, and elements of autocorrelation. Based on the outcome, three different sets of features were proposed.

- We develop long-term traffic levels forecast strategies using ML and Time Series (TS) algorithms. Depending on the feature set, different ML and TS algorithms' approaches were proposed, namely Level Based (LB), Real Values Based (RVB) and Level Values Based (LVB). Additionally, two different strategies were proposed and tested, i.e., static prediction and dynamic prediction.
- We propose an evaluation metric suitable for the considered problem. It is based on metric described in [14] and allows to evaluate tested algorithms in terms of underpredictions and overpredictions, which can be significant for network operators. Its main characteristic is flexibility.

The rest of this paper is organized as follows. Section 2 includes a literature review of the considered problem. Sections 3, 4 and 5 present used datasets, formulate problems and describe used approaches to solve them. Section 6 shows and discusses experimental results. Section 7 concludes whole article.

II. Related Works

The network long-term traffic forecasting problem is not new in the literature and has been widely studied in many papers. Typically, it is formulated as a TS problem and solved using approaches based on ARIMA and its numerous variations, as well as ML techniques [15]. Authors of [16] compared ARIMA, Holt-Winters, and neural network algorithms for forecasting the amount of traffic in TCP/IP-based networks. The datasets were based on distinct time scales, namely 5 minutes, 1 hour, and 1 day, and different forecasting horizons were analyzed. The obtained results concluded that the neutral network achieved the best results for 5 minute and hourly data, while the Holt-Winters was the best for the daily forecast. Work [17] presents TS algorithms for traffic forecasting. ARIMA and SARMIA models were used for short-term and long-term future traffic volume forecasts. Authors propose a procedure for separating temporal and seasonal variations of traffic. Additionally, work also investigates the impact of traffic forecasting on traffic engineering. As a result of traffic management, based on forecasted traffic flows, the required bandwidth for data transmission was reduced by almost 19%. Authors of [18] presented a traffic forecasting method based on the Facebook PROPHET algorithm, procedure for time data series based on an additive model where non-linear trends are fit with yearly, weekly, and daily seasonality, plus holiday effects. The work shows that PROPHET can be well used for a 14-day horizon traffic forecasting. Three different models containing sets of additional input features to improve the forecasting quality of different ML algorithms are presented in [19]. Models were evaluated on datasets with different types of traffic. The authors evaluated number of ML algorithms. The performance of them was measured using MAPE. Experiments proved that relevant features improve the quality of ML algorithms. The obtained MAPE values varied between 1 and 10%. Authors in [20], [21] and [22] present future traffic forecasting by predicting the occurrence of future requests in the network. Each request consists of a source node, a destination node, and request volume information. The assumption is that traffic in a network can be characterized as chain traffic, i.e., it represents traffic flow between network nodes in which single virtual network functions are located. Authors employ different ML classification algorithms, namely k Nearest Neighbors, Decision Tree, Random Forest, Gaussian Naïve Bayes, Multilayer Perceptron, and Linear Discriminant Analysis. Experiments brought forecasting quality up to 94%.

Although many works have presented promising results, there is a high demand for exploring the possibility of application of ML methods to network problems [23], [24]. The majority of the related works realized the traffic forecasting task as a prediction of exact traffic bitrates. To the best of our knowledge, long-term traffic forecasting has not been addressed in the literature in the context of prediction of traffic level. Additionally, short-term traffic forecasting using traffic levels was described only in articles [14], [25] and [26]. To fill the research gap, this work introduces, formulates, and examines the long-term forecasting problem as a prediction of fixed traffic levels. Such a concept is a result of optical networks and other transport network technologies' characteristics.

III. Datasets

Datasets used for experiments are semi-synthetic and contain real and artificially generated data. This section describes the dataset generator and the generated traffic flows.

1) Traffic Generator

Datasets used for experiments contain real and semi-synthetic data. The semi-synthetic datasets were created with the use of the custom traffic generator proposed in [27], which applies the time-varying real data obtained from the Internet Exchange Point in Seattle (SIX). In more detail, the semi-synthetic traffic follows the shape of the real traffic that can be found at the SIX official website (https://www.seattleix.net/, accessed on 21 February 2021). The provided rrd files have a 5 minute granulation time and were downloaded between 27.12.2020 and 21.02.2021. The collected sets of SIX traffic were parsed and transformed for further processing.

The generator is a set of smaller request generators, each representing a different web service and having a defined share [2] and their own characteristics, such as a set of stochastic processes with assigned parameters and contribution scales for each of them. In this generator, the considered stochastic processes are Poisson process (PP), Poisson Pareto burst process (PPBP) [28], and constant traffic (CT) with uniformly distributed random offset. The web services are represented using the stochastic processes indicated above in the following manner:

- Internet video accounts for 51% of the total bitrate. Made of two different PPs, PPBP and CT.
- IP VOD accounts for 22% of the total bitrate. Made from a single PP.
- Web data accounts for 18% of the total bitrate. Made of two different PPs.

- File sharing accounts for 8% of the total bitrate. Made from a single CT.
- Gaming accounts for 1% of the total bitrate. Made from a single CT.

The primary goal of this distinction between traffic patterns for certain online services is to simulate the changing nature of Internet traffic. The overall bitrate of generated requests varies over time depending on the provided traffic characteristics.

2) Datasets

The semi-synthetic traffic created by the traffic generator described in previous Section can be characterized by its fluctuation. As a fluctuation metric, Mean Absolute Percentage Error (MAPE) is considered. It determines how values of one traffic flow differs from values of the base traffic flow. Let $A = (a_1, a_2, \dots, a_n)$ and $B = (b_1, b_2, \dots, b_n)$ contain bitrates of traffic flows. Let us consider B as base traffic flow. MAPE for A can be calculated by:

$$MAPE = \frac{100\%}{n} \sum_{i=1}^{n} \left| \frac{b_i - a_i}{b_i} \right| \quad (1)$$

To calculate the MAPE of a particular traffic dataset, as a reference traffic we use the original SIX traffic normalized to a common range of considered bitrate values. In other words, MAPE indicates how the considered semi-synthetic traffic differs from the SIX traffic.

In this paper, four traffic patterns were examined:

- dataset_1 – semi-synthetic traffic flow with MAPE equal to 2,9%.
- dataset_2 – semi-synthetic traffic flow with MAPE equal to 7%.
- dataset_3 – semi-synthetic traffic flow with MAPE equal to 11,6%.
- dataset_4 – traffic flow related to real volumes collected by SIX, MAPE is equal to 0%.

The first three datasets were chosen from a large number of generated datasets, because they are representative in terms of the MAPE value, i.e., the reported MAPE values are from a relatively small value of 2,9% to a relatively large values of 11,6%. Fig. 1 visualizes two days of data flows from 28.12.2020 to 30.12.2020 of the datasets mentioned above. Because dataset bitrates ranges differ from each other, for the sake of visualization, dataset bitrates were normalized to the same range and presented on the vertical axis. It can be clearly seen that their difference in terms of fluctuation and MAPE correctly reflects their characteristics, i.e., higher MAPE points and more frequent fluctuation.

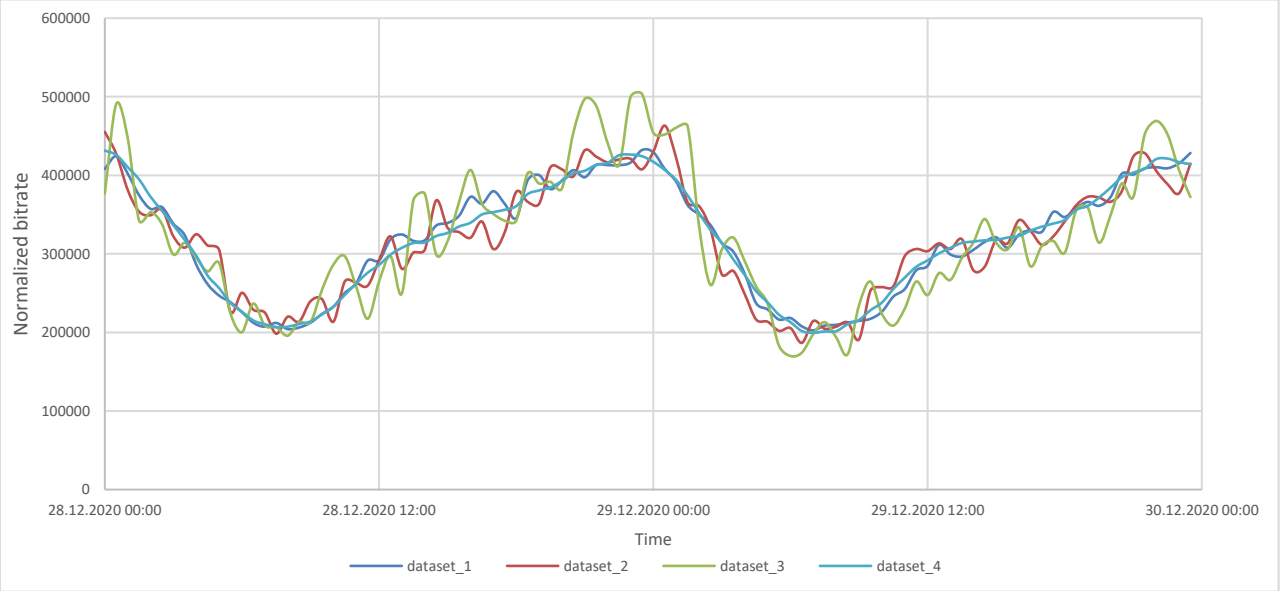

Fig. 1 - Datasets visualization

IV. Problem Formulation

This section describes a formulation of the main problem addressed in this work. It contains a description and a mathematical definition of the problem and information about the used performance metrics.

1) Network model

The problem, which is examined in this work, is a network traffic forecasting based on historical traffic data flows. The time scale of the network operation is divided into time intervals (TIs) of the same size, equal to 30 minutes. For consecutive TIs, traffic volumes (bitrates) related to the single pair of nodes or a whole traffic going through a single node create continuous and regular data flows. Depending on the transport network technology, to establish a connection, the network operator requires information about a traffic level that is sufficient to carry a transmission and allows allocation of network resources efficiently, e.g., choosing an adequate number of optical channels in optical networks. Thus, for each TI, a corresponding traffic level can be assigned.

In this work, traffic level is calculated as the maximum bitrate value within TI, however different ways of calculations can be applied, for example, the average value. Fig. 2 illustrates the process of defining traffic levels for traffic flows. The blue line

represents real bitrate values and the green line traffic levels that correspond to them. Possible traffic levels (determined by a transport network technology) are represented by grey horizontal lines. Based on such a formulation, the final network traffic forecasting outcome is information about future traffic levels in TIs, rather than the exact traffic volume.

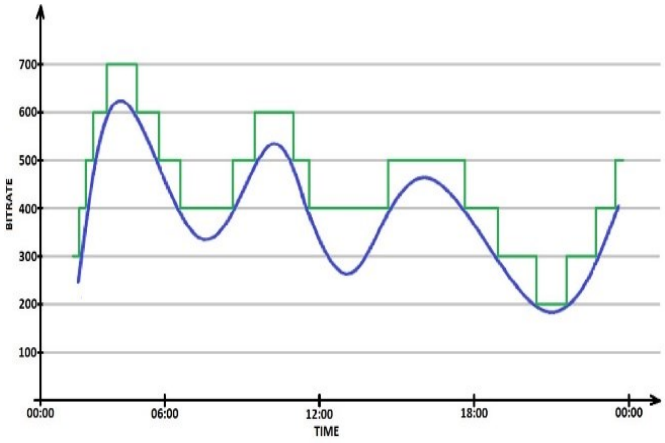

Fig. 2 - Traffic level definition

2) Classification type

As mentioned earlier, the problem considered in this work is network traffic forecasting, realized by prediction of future traffic levels. Because traffic levels have a hierarchy (they can be arranged in ascending order), a task is considered as an ordinal classification [29], also called an ordinal regression, problem. In general, it is a multiclass classification problem [30], [31] (in a specific case, when only two traffic levels are considered, it is a binary problem), where possible classes have inherent order. Labels can take any value, even numeric, e.g. "Level_1", "100Gbps", "100" However, from the classification point of view, there is no meaningful numeric difference among them [32], i.e., it does not matter if traffic levels differ by 100, 1000, or even by different granulations. Additionally, the problem is the offline learning. The whole dataset of historical data flows in the network is known in advance, and models are not updated during forecasting.

To formalize the considered problem, let us define a set of input vectors $X = (x_1, x_2, \dots, x_i)$ and a set of traffic levels (in ordinal classification called ordered labels or ordered classes) $Y = (y_1, y_2, \dots, y_j)$, where $i, j \in N$. In the network traffic levels' forecasting problem, there occurs an order among classes, i.e., $y_1 < y_2 < \cdots < y_j$. For each instance x_i, class from Y can be assigned. As a result, a set of pairs $P = (X, Y) = (x_t, y_t)$, where t points TI, is created. P can also be called a training set. The task posed to ML algorithms in classification problem is to first obtain knowledge about historical data flows and their traffic levels representation, i.e., train using a training set, and find a function $M(x_t) = y_t$, which maps X into Y [33]. Next, to forecast Y for unseen $P^* = (X, Y)$, which reflects the future. Each input vector in the set X consists of w number of features, i.e., $F = (f_1, f_2, \dots, f_w)$.

3) Quality metric

Evaluation of ordinal classification is a challenging task. The reason for that is twofold. First, there is a number of metrics that can be used for algorithms evaluation. Each metric can measure different aspects of an algorithm's performance. Additionally, because of the lack of metrics for ordinal classification, to evaluate such problems, metrics appropriate for nominal classification, i.e., classification problem, where there is no order between classes, are typically used [34]. Secondly, some errors are worse than others [32], i.e., let $Y = (y_1, y_2, y_3)$ be a set of ordinal classes and $y_1 = 100$, $y_2 = 200$ and $y_3 = 300$. Assigning y_1 to x when actual class is y_2 (some data are lost during transmission) costs network operator more than assigning y_3 to the same x (transmission occurs, however it uses more network resources than required). To evaluate the ML algorithm a confusion matrix ($ConM$) [35] can be used. It is a matrix of the size $j\ x\ j$, where j denotes the number of possible classes. Each column indicates forecasted classes and each row – real classes. Let us consider a multiclass ordinal classification problem, i.e., traffic levels forecasting, where possible classes belong to set $Y = (y_1, y_2, \dots, y_j)$. After classification task a confusion matrix $ConM$ can be created. Each element a_{ug}, where $u, g \in (1, 2, \dots, j)$ represents the number of cases when the algorithm returned y_g and actual it was y_u. For each traffic forecasting case, the interpretation matrix can be defined. It is a matrix of the size $j\ x\ j$. Fig. 3 presents confusion matrix $ConM$ and interpretation matrix $InterM$ of a classification problem described above.

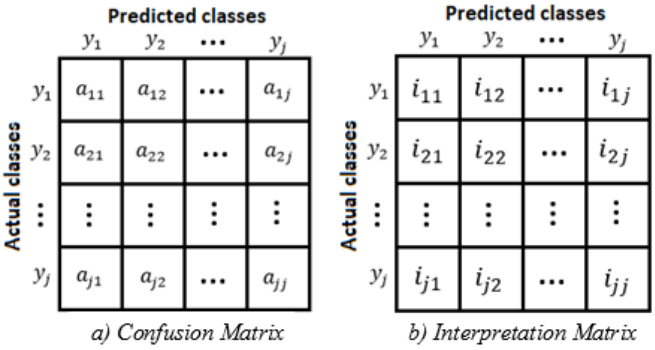

Fig. 3 - Confusion and Interpretation Matrices

Each element i_{ug}, where $u, g \in (1, 2, \dots, j)$, of $InterM$ represents the interpretation of each classification type, i.e., the importance of cases when the algorithm returned y_g and actual was y_u. i_{ug} values are in range $[-1, 1]$. A positive value of i_{ug} means that such type of classification is acceptable with specific weight, a negative values means that such type of classification is unwanted with specific weight and 0 means that such type of classification is neutral. The diagonal of $InterM$ represents cases, where correct classes have been chosen and often contain 1, i.e., correct classifications are highly desirable. Additionally, values above $InterM$ diagonal represent overestimations and those below the diagonal represent underestimations. In order to estimate the performance of algorithms, thus final traffic level forecasting quality, in this work, the metric called Traffic Level Prediction Quality (TLPQ) is defined. It is an extended version of the metric, which we presented in [14]. Compared it to its previous version, it considers more under and overpredictions variants. Each variant can have assigned a different weight. Additionally, the metric used in this article takes different range of values. It can be calculated based on the confusion matrix ($ConM$) and the interpretation matrix ($InterM$), using the following equation:

$$TLPQ = \sum_{u=1}^{j}\sum_{g=1}^{j}\frac{a_{ug}\cdot i_{ug}}{\sum_{u=0}^{j}\sum_{g=0}^{j}a_{ug}}, \quad (2)$$

where a_{ug} and i_{ug} are elements of $ConM$ and $InterM$ respectively, j denotes number of possible classes and $u, g \in (1, \dots, j)$. TLPQ is a flexible metric and can be adjusted to the specifications of any traffic level forecasting problem. It ranges from -1 to 1. It is a point metric, i.e., a greater score is better, where -1 means that the algorithm returns only unacceptable classes, 0 means that the number of acceptable and unacceptable classifications are equal, and 1 means that only acceptable classes are returned by algorithm. Defining a specific measure for an ordinal classification problem is a common practice [32], [36], [37], [38], [39]. This is due to the fact that the characteristics of each classification task are different and there is no one well-known measure that can be applied to each problem [38], [40].

To test algorithms for different network scenarios, three variants of TLPQ are calculated, namely TLPQ_1, TLPQ_2 and TLPQ_3. Let us consider the problem with five possible traffic levels. $InterM$ matrices for individual TLPQ's are presented below. $InterM_1$ can be applied for TLPQ_1, $InterM_2$ for TLPQ_2 and $InterM_3$ for TLPQ_3. For problem with a different number of possible traffic levels, $InterM$ matrices change dimension and take values of elements according to the scheme. TLPQ_1 is suitable when CSP accepts correct forecasts and overestimations by one traffic level with the same weight equal to 1. In turn, overestimations by more than one traffic level, together with underestimations, are neutral. In TLPQ_2 the highest importance have the correct forecasts. Overestimations are acceptable, but with a lower weight, equal to 0,7. Underestimations are unacceptable with a weight 0,3. Overestimations by more than one traffic level are neutral for TLPQ_2 quality metric. In TLPQ_3 the most significant are correct forecasts. Overestimations by one level are acceptable with a weight 0,5. The same weight of unacceptance has underestimations. Overestimations by more than one traffic level do not impact TLPQ_3 value.

$$InterM_1 = \begin{vmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{vmatrix} \quad (3)$$

$$InterM_2 = \begin{vmatrix} 1 & 0{,}7 & 0 & 0 & 0 \\ -0{,}3 & 1 & 0{,}7 & 0 & 0 \\ -0{,}3 & -0{,}3 & 1 & 0{,}7 & 0 \\ -0{,}3 & -0{,}3 & -0{,}3 & 1 & 0{,}7 \\ -0{,}3 & -0{,}3 & -0{,}3 & -0{,}3 & 1 \end{vmatrix} \quad (4)$$

$$InterM_3 = \begin{vmatrix} 1 & 0{,}5 & 0 & 0 & 0 \\ -0{,}5 & 1 & 0{,}5 & 0 & 0 \\ -0{,}5 & -0{,}5 & 1 & 0{,}5 & 0 \\ -0{,}5 & -0{,}5 & -0{,}5 & 1 & 0{,}5 \\ -0{,}5 & -0{,}5 & -0{,}5 & -0{,}5 & 1 \end{vmatrix} \quad (5)$$

To evaluate the performance of a ML algorithm used for the ordinal classification problem, also some error functions can be applied. Such functions measure how far are forecasts from the real value. In ordinal classification problems, their value is correlated with the numerical representation of classes, i.e., values of traffic levels, since this values are used for calculations. However, they still can give an overview of algorithm performance. One of the most widely used error function is mean absolute error (MAE) [41]]. It is used in a number of ordinal classification problems [42], [43], [44], [45], [46]. Let $Y^* = (y_1^*, y_2^*, \dots, y_n^*)$ be the set of classes returned by the algorithm in the ordinal classification task, and $Y = (y_1, y_2, \dots, y_n)$ be a set of real classes corresponding to Y^*. MAE can be calculated by:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - y_i^*| \quad (6)$$

It represents the average of the absolute differences between forecast and actual classes. MAE is a "lower is better" type of performance. Additionally, according to [32], it can be used to minimize the number of errors.

In this paper, two metrics are used as the main performance metrics of the algorithm, namely, the proposed TLPQ metric and MAE. This choice is due to the fact that TLPQ gives high flexibility, thus can be adjusted to any traffic level forecasting problem by definition of $InterM$. Its calculation is intuitive and easier comparing to classical classification metrics. A literature study shows that MAE is a widely used metric in the case of ordinal classification. Tied together, they give reliable insight on the forecast performance.

V. Methods

This work presents three different approaches for traffic levels forecasting:

- Label based (LB) – problem is treated as a pure classification task. Possible network traffic levels create a set of classes and employed algorithms that return the exact traffic level in the TI (Fig. 4).
- Real values based (RVB) – in this case, the problem is considered initially as a regression task. First, the applied algorithm returns the value of a bitrate in a particular TI. Next, based on the obtained result, traffic levels in TIs are calculated by rounding up the forecasted bitrate (Fig. 5).
- Labels values based (LVB) – it is a mix of previous two approaches. Regression algorithms are applied to forecast the values of traffic levels. Because the forecast is rarely the exact value of the traffic level, the final decision is the traffic level closest to the value returned by the algorithm (Fig. 6).

Fig. 4 to Fig. 6 illustrate the ways of forecast in the case of a particular approach. The black color represents historical traffic, which the algorithm gets as an input. The green color reflects forecasts and, for RVB and LVB approaches, the blue color symbolizes traffic levels defined based on forecasts. For each approach, different types of algorithms can be employed, i.e., ML algorithms and TS algorithms. Besides the different ways of forecasting, outcome is the same in the case of all approaches, i.e., at the end traffic levels in TIs are returned. In general,

algorithms map input vector (which describes data flows) into network traffic levels.

LB and LVB approaches can be executed as follows. Let the set $B = (b_1, b_2, \dots, b_T)$ consists of historical bitrates related to a single pair of nodes or to a single node in a network for T consecutive TIs. For each b_t, where $t = 1, \dots, T$, a class from the set Y can be assigned, based on the actual value of the bitrate. Additionally, for each b_t, a vector from X can be defined, based on instances from b_1 to b_{t-1}. Methods of defining vectors in X are presented below. Algorithms take as input pairs $P = (X, Y) = (x_t, y_t)$, find $M(x_t) = y_t$ and forecast Y for unseen X. The difference between the LB and LVB approaches is the type of algorithms used during forecasting. LB approach is pure classification, so it uses classifiers. In turn, LVB approach relies on regression, so it uses regressors.

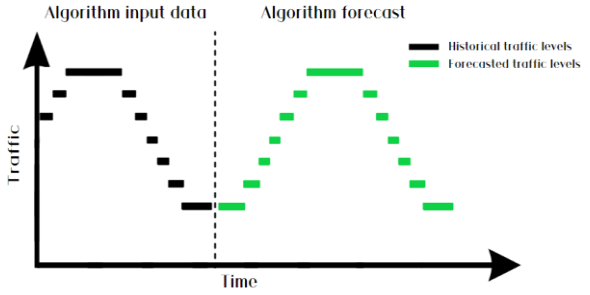

Fig. 4 - LB approach

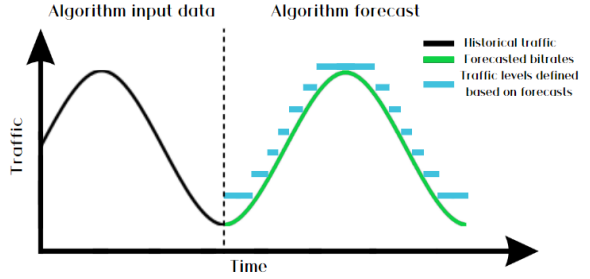

Fig. 5 - RVB approach

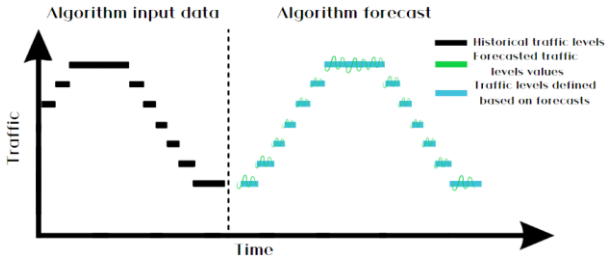

Fig. 6 - LVB approach

In the case of RVB approach, first a set of $Y^* = (y_1^*, y_2^*, \dots, y_T^*)$ has to be defined. For each b_t, y_t^*, which represents the exact bitrate value of b_t can be assigned. Vectors in set X are defined in the same way as in the case of level based approach. Algorithms first train to find a function $M(x_t) = y_t^*$, next forecast Y^* based on unseen $P^* = (X, Y^*)$ and at the end assign Y based on Y^*, by rounding up Y^* to the nearest Y.

Each input vector in set X consists of w number of features, i.e., $F = (f_1, f_2, \dots, f_w)$. Selecting a suitable set of features is crucial for the forecasting problem. Based on them, algorithms define the relevant $M(X)$. In more detail, when forecast is based on the historical information, features should allow mapping history into the future in an efficient way. In the network traffic, some daily and weekly patterns can be distinguished. Thus, the characteristics of traffic flows are correlated with time. Additionally, because the general shape of the flows repeats in time, i.e., there is a seasonality in the data, bitrates from the past can reflect future traffic. To determine which previous TIs correlate with the current TI, the autocorrelation function has to be studied Fig. 7 presents week autocorrelation of dataset 1. For the presented data, TIs length is equal to 30 min. Based on graph, strong seasonality can be noticed. A high positive autocorrelation occurs every 48 points, i.e., after 24 hours, since each single point reflects 30 minutes. Additionally, the highest autocorrelation appears for TIs close to the first TI. The blue background designates an area where autocorrelations become insignificant. Finally, a significant autocorrelation occurs for 7th day.

Considering all of this, the following sets of features F have been tested in this work:

- $F_1 = (day, minute, b_{t-1}, b_{t-2}, b_{t-3}, b_{t-24h})$.
- $F_2 = (day, minute, b_{t-24h}, b_{t-1-24h}, b_{t-2-24h}, b_{t-3-24h}, b_{t-7d})$.
- $F_3 = (b_1, b_2, \dots, b_T)$.

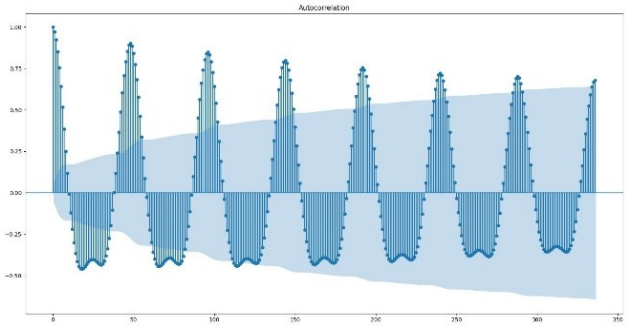

Fig. 7 - dataset_1 one week autocorrelation, TI equal to 30 min

We assume that $day \in [1, 7]$ defines the number of the day in the week and $minute \in [0, 1440]$ and also reflects the minute during the day. Different types of F were used for different models. To better understand (X, Y) pairs creating process, let us consider an example. Let set B = (95, 155, 220, 450, 390, 280, 105, 180, 240, 450, 420, 405, 395, 380, 350, 295, 250, 180, 150, 80, 90, 115, 160, 210, 280, 450, 320, 150, 200) consists of historical bitrates related to a single pair of nodes in a network for consecutive TIs (one TI reflects 1h), and set Y = (100, 200, 300, 400, 500), contains possible classes in a network traffic levels forecasting problem. For each element from B information about TI number, the day during the week, the minute during the day and a traffic level out of Y can be assigned. Based on that, a set of pairs (X, Y) can be created, where X contains F_1 features' set. Fig. 8 describes the process of creation of such pairs.

TI	day	minute	bitrate
1	1	60	95
2	1	120	155
3	1	180	220
4	1	240	450
5	1	300	390
6	1	360	280
7	1	420	105
8	1	480	180
9	1	540	240
10	1	600	450
11	1	660	420
12	1	720	405
13	1	780	395
14	1	840	380
15	1	900	350
16	1	960	295
17	1	1020	250
18	1	1080	180
19	1	1140	150
20	1	1200	80
21	1	1260	90
22	1	1320	115
23	1	1380	160
24	1	1440	210
25	2	60	280
26	2	120	450
27	2	180	320
28	2	240	150
29	2	300	200

Pair	Features (day, minute, b_{t-1}, b_{t-2}, b_{t-3}, b_{t-24})	y
(x_{25}, y_3)	(2, 60, 210, 160, 115, 95)	300
(x_{26}, y_5)	(2, 120, 280, 210, 180, 155)	500
(x_{27}, y_4)	(2, 180, 450, 280, 210, 220)	400
(x_{28}, y_2)	(2, 240, 320, 450, 280, 450)	200
(x_{29}, y_2)	(2, 300, 150, 320, 450, 390)	200

Fig. 8 - (X,Y) pairs creation

Note that B has 29 elements and only 5 pairs can be created. It is because of the fact that there is no information about b_{t-24} for first 24 elements in B.

Sets of pairs (X, Y) containing features from F_2 are created in a similar way. F_3 features' set is intended for TS. In this case, the plain dataset is given as an input for models.

During the forecasting process, algorithms take input vectors, whose features are based on the past. To forecast using F_1 features' set, algorithms have to get information about traffic in three TIs which precede the considered TI. Because of that, the forecast horizon is limited to one TI ahead (one step ahead). To forecast the future for a longer time horizon, algorithms can use their forecasts as features' values. In turn, F_2 features' set requires information about traffic from TIs distant by one day from considered TI, thus it allows to forecast one day ahead. In this paper, the strategy where real traffic levels or bitrates are used as features is called static prediction, and the strategy which uses algorithms' forecasts as features to extend the possible forecast horizon is called dynamic prediction. Note that dynamic prediction strategy is used only with F_1 features' set and F_2 features' set is designed for static prediction since it provides sufficient forecasting horizon.

The ML algorithms used in this work are as follows:

- Decision tree (DT) [31] – builds a hierarchical model in the shape of a tree, which consists of decision nodes and leafs. Decision tree decomposes a complex problem into a series of individual, simple steps.
- k – Nearest Neighbor (kNN) [46], [48] – belongs to the group of minimal distance algorithms. A final decision is made based on k-objects in the nearest neighborhood.
- Logistic regression (LR) [48] – is a linear model for classification task. Based on the dataset's statistical analysis, it estimates probabilities describing the possible outcomes of a single trial using a logistic function.
- Linear regression (LR) [49] – assumes a linear relationship between the input (features) and output. To create the model, the ordinary least squares method for coefficients' calculation is used. LR can be applied for regression.
- Multilayer perceptron (MLP) [50] – the neutral network composed of one input layer, one or more hidden layers, and one output layer. Each layer consists of a number of single perceptrons [51] also called neurons.

To achieve a better performance, single algorithms can be aggregated into ensembles [52], [53], [54]. An ensemble is a group of base algorithms (also called estimators) whose individual decisions are combined in some way, typically by unweighted or weighted voting. Ensembles often return better performance than single algorithms, which make them up. Following ensemble methods were tested during experiments:

- Extra Trees (ET) [55], [56] – uses an implementation of the decision trees ensemble provided by scikit-learn python library [57]. It trains a number of decision trees on various sub-samples of the dataset and uses averaging to improve performance. As a base estimator, DT and DTR can be applied.
- Random Forest (RF) [58] – is an ensemble of decision trees. It builds decision trees on different objects and takes their majority vote for classification and average in the case of regression. As a base estimator, DT and DTR can be applied.
- Eibe Frank and Mark Hall ensemble method (EFMH) – is an ensemble designed for ordinal classification problems and described in [59]. It transforms a multiclass problem into binary problems. Each algorithm has to answer the question if an object is higher or lower than the considered class.
- Bagging (Ba) [60] – also known as bootstrap aggregation. BR is an ensemble of regression algorithms, e.g., kNN, DT. It trains base algorithms on random subsets of the original dataset. The training set is selected with replacement, i.e., the individual objects can be chosen more than once. As a final forecast, it returns an average of base algorithm forecasts.

All ML algorithms' implementations were done using *scikit-learn* python library [57]. Each of used algorithms has a number of parameters, which influence its architecture and final performance. To select the optimal set of parameter values for each algorithm, a parameter tuning process has to be performed. For ML algorithms, it was done using a grid search procedure. TS algorithms' parameters were determined using *auto_arima()* function from *alkaline-ml* python library [61].

In case of the traffic level forecasting problem, data are related with time and have sequential order. Additionally, the specificity of the problem is that often the future is forecasted based on the nearest past. Because of that, in this work, to get reliable model performance, the dataset is divided into training and test subsets in a slightly different way than in classic k-fold cross validation. At the beginning, the whole dataset is divided into equal subsets, each containing consecutive elements. Let us assume that the dataset is divided into four subsets, based on which four different models are created. From each subset, a given number of consecutive dataset elements are taken as the

training subset and the remaining ones are taken as the test subset. At the end, as a final metric value, an average of all models' performance is taken. Note that subsets of the whole dataset can be disjoint as well as they can have partly the same elements.

VI. NUMERICAL RESULTS

This section presents and discusses numerical results obtained during experiments. Algorithms related to subsections 6.2 and 6.3 were trained on datasets containing TIs collected from 28 days and tested on datasets containing information about TI during 1 day. Experiments in those subsections were conducted 28 times, and the results were calculated as an average. The length of train datasets in subsection 6.4 was also 28 days, however train dataset length varies depending on the time horizon. Datasets used during research were described in section 4.

1) Single ML and TS algorithms

At the beginning, to find effective methods for traffic level forecast, some ML algorithms (classifiers and regressors) together with TS algorithms were tested.

Table I and Table II report TLPQ and MAE values of each tested approach in the case of static and dynamic predictions, respectively. The best results from a single dataset are bolded. TS was used only for RVB approach in static perdition, since it forecasts traffic bitrate real values. Additionally, it returns traffic levels for the whole forecast horizon, so its way of operation is similar to the static prediction. Based on TLPQ values, the general trend can be noticed. Algorithms obtained higher TLPQ values for datasets with lower MAPE, namely dataset_1 and dataset_4. In the case of static prediction, the best results had algorithms in LVB approach (Table I) For dataset_1 and dataset_4, the best TLPQ values returned LR, and for other datasets, i.e., dataset_2 and dataset_3, the best turned out to be kNN. Different situation occurs for dynamic prediction (Table II). For dataset_1 and dataset_4, the highest TLPQ values were obtained by algorithms in the LB approach, namely kNN and MLP, respectively. For datasets with higher MAPE value, i.e., dataset_2 and dataset_3, the best turns out to be kNN in RVB approach. Additionally, for dataset_2 and dataset_3, the best algorithms in RVB and LVB approaches got similar TLPQ results. In the case of other datasets, the LVB approach together with dynamic prediction did not produce good forecast results (Table II). In RVB and LVB approaches algorithms with simpler architecture, i.e., kNN and LR, turned out to return better TLPQ metrics than more complex algorithms, like MLPR. For LB approach, MLP was better than other algorithms in most cases. Algorithms in the case of static prediction get better results than in dynamic prediction. Additionally, TS algorithms have TLPQ performance lower than ML algorithms in the case of all datasets.

Looking at MAE errors, it can be observed that algorithms in the case of the RVB approach make fewer mistakes than LB and LVB approaches. Such a characteristic is true for both static and dynamic predication. For datasets with higher MAPE value, namely dataset_2, dataset_3, algorithms obtained lower MAE in the case of static prediction, when for datasets with lower MAPE values, i.e., dataset_1, and dataset_4, MAE values are similar in case of both static and dynamic predictions.

To check if differences between methods are statistically significant, the Friedman test and Nemenyi post hoc test at a significance level set to 0.05 were performed. Fig. 9 presents ranking of methods related to TLPQ_1 metric. As it can be seen, there is statistical difference between tested algorithms. The best rank obtained DT algorithm applied with LB approach. Similar results of the Friedman test and Nemenyi post hoc test were also observed for TLPQ_2 and TLPQ_3 metrics.

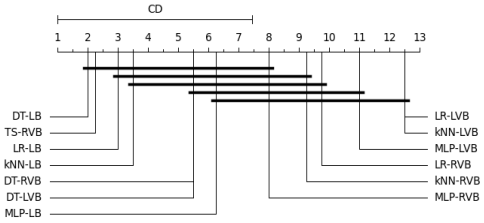

Fig. 9 - TLPQ_1 statistical tests, single algorithm

2) Ensembles

Ensembles consisting of weak algorithms often perform better than a single algorithm. To obtain better traffic levels forecasting quality, in this subsection, two different ensemble types with DT and kNN applied as base algorithms were examined, namely Ba and EFMH. Additionally, two other ensembles with DT as the base algorithm were used, i.e., RF and ET. Ba ensembles were tested with RVB and LVB approaches and EFMH ensemble with LB approach.

Table III and Table IV present results obtained during experiments. In most cases, the application of the ensemble allowed us to return better (higher) TLPQ values than in case of single algorithms. The exception was a dynamic prediction and LVB approach (Table IV), where TLPQ metrics turned out to be lower, compared to single algorithms (Table II). Ensembles during static prediction (Table III) return higher TLPQ values than ensembles during dynamic prediction (Table IV). In case of LB approach, the highest TLPQ values were achieved in ensembles with kNN as a base estimator. In turn, for RVB and LVB approaches, the best turned out to be ensembles with DT as base estimator. The ensemble which forecasted traffic with the highest TLPQ performance was ET in LVB approach and static prediction (Table III)

When analyzing MAE values, it can be noticed that ensembles during dynamic prediction make higher errors (Table IV). Additionally, ensembles in the case of RVB approach made the lowest errors among all tested approaches. Application of ensembles decreased MAE errors, compared to MAE errors of single algorithms.

TABLE I
SINGLE ALGORITHMS RESULTS, STATIC PREDICTION

		LB approach				RVB approach				LVB approach			
	Alg.	**TLPQ_1**	**TLPQ_2**	**TLPQ_3**	**MAE**	**TLPQ_1**	**TLPQ_2**	**TLPQ_3**	**MAE**	**TLPQ_1**	**TLPQ_2**	**TLPQ_3**	**MAE**
dataset_1	**DT**	0,70	0,57	0,48	162	0,75	0,62	0,53	114	0,75	0,62	0,54	156
	kNN	**0,74**	**0,62**	**0,54**	**144**	0,80	0,68	0,61	95	0,89	**0,76**	**0,67**	136
	LoR/LR	0,64	0,51	0,42	202	**0,80**	**0,69**	**0,62**	**92**	**0,90**	0,75	0,64	**133**
	MLP	0,73	0,61	0,53	158	0,77	0,65	0,57	163	0,89	0,74	0,63	135
	TS	-	-	-	-	0,69	0,58	0,47	221	-	-	-	-
dataset_2	**DT**	0,53	0,36	0,25	39	0,58	0,41	0,29	34	0,58	0,41	0,30	39
	kNN	0,55	0,38	0,26	**32**	0,67	**0,51**	**0,41**	**26**	**0,75**	**0,59**	**0,49**	**30**
	LoR/LR	0,58	0,42	0,31	38	**0,67**	0,51	0,40	27	0,73	0,58	0,48	31
	MLP	**0,59**	**0,53**	**0,33**	33	0,60	0,43	0,32	49	0,70	0,54	0,43	41
	TS	-	-	-	-	0,53	0,37	0,25	41	-	-	-	-
dataset_3	**DT**	0,56	0,39	0,28	23	0,62	0,45	0,34	19	0,61	0,45	0,34	23
	kNN	0,58	0,41	0,29	19	**0,70**	**0,55**	**0,45**	**15**	**0,76**	**0,61**	**0,50**	18
	LoR/LR	0,66	0,49	0,38	21	0,70	0,54	0,44	15	0,75	0,60	0,50	**18**
	MLP	**0,68**	**0,52**	**0,40**	**18**	0,68	0,53	0,43	16	0,73	0,57	0,46	22
	TS	-	-	-	-	0,59	0,39	0,36	25	-	-	-	-
dataset_4	**DT**	0,74	0,63	0,55	22047	0,77	0,65	0,58	15939	0,77	0,65	0,57	21782
	kNN	0,76	0,65	0,57	20897	0,81	0,71	0,65	13180	0,89	0,74	0,65	21084
	LoR/LR	0,69	0,56	0,47	31927	**0,82**	**0,72**	**0,66**	**12335**	**0,94**	**0,78**	**0,67**	**20313**
	MLP	**0,78**	**0,66**	**0,60**	**19875**	0,80	0,69	0,62	22806	0,88	0,72	0,61	31568
	TS	-	-	-	-	0,71	0,62	0,53	28254	-	-	-	-

TABLE II
SINGLE ALGORITHMS RESULTS, DYNAMIC PREDICTION

		LB approach				RVB approach				LVB approach			
	Alg.	**TLPQ_1**	**TLPQ_2**	**TLPQ_3**	**MAE**	**TLPQ_1**	**TLPQ_2**	**TLPQ_3**	**MAE**	**TLPQ_1**	**TLPQ_2**	**TLPQ_3**	**MAE**
dataset_1	**DT**	0,56	0,45	0,37	431	0,69	0,55	0,46	142	0,42	0,33	0,28	599
	kNN	**0,80**	**0,66**	**0,56**	**257**	**0,77**	**0,64**	**0,56**	**109**	**0,51**	**0,40**	**0,33**	**447**
	LR	0,41	0,32	0,26	641	0,74	0,61	0,52	125	0,15	0,11	0,09	697
	MLP	0,79	0,63	0,55	260	0,71	0,57	0,47	186	0,18	0,13	0,09	731
dataset_2	**DT**	0,47	0,33	0,25	72	0,58	0,41	0,30	35	0,54	0,40	0,31	63
	kNN	**0,65**	**0,49**	**0,39**	**36**	**0,68**	**0,52**	**0,42**	**26**	**0,65**	**0,51**	**0,41**	60
	LR	0,45	0,32	0,24	70	0,60	0,43	0,31	32	0,34	0,26	0,20	96
	MLP	0,64	0,49	0,38	42	0,65	0,48	0,37	28	0,26	0,18	0,12	133
dataset_3	**DT**	0,47	0,35	0,27	40	0,59	0,43	0,32	20	0,51	0,39	0,31	42
	kNN	0,66	0,50	0,40	20	**0,72**	**0,57**	**0,47**	**15**	**0,72**	**0,57**	**0,48**	**32**
	LR	0,56	0,42	0,33	33	0,64	0,47	0,36	18	0,30	0,23	0,18	60
	MLP	**0,68**	**0,53**	**0,43**	**19**	0,67	0,51	0,40	16	0,24	0,18	0,14	76
dataset_4	**DT**	0,28	0,23	0,20	121769	**0,73**	**0,62**	**0,55**	**17357**	0,29	0,23	0,20	126339
	kNN	**0,76**	0,59	0,47	44824	0,72	0,59	0,50	17560	**0,54**	**0,42**	**0,35**	**67895**
	LR	0,26	0,15	0,07	122496	0,58	0,41	0,29	30549	0,04	0,03	0,02	333897
	MLP	0,75	**0,63**	**0,55**	**29851**	0,71	0,61	0,53	18134	0,06	0,01	0,02	228530

TABLE III
ALGORITHMS ENSEMBLE RESULTS, STATIC PREDICTION

		LB approach				RVB approach				LVB approach			
	Alg.	**TLPQ_1**	**TLPQ_2**	**TLPQ_3**	**MAE**	**TLPQ_1**	**TLPQ_2**	**TLPQ_3**	**MAE**	**TLPQ_1**	**TLPQ_2**	**TLPQ_3**	**MAE**
dataset_1	**Ba-DT**	-	-	-	-	0,78	0,67	0,60	94	0,88	0,75	0,66	130
	Ba-kNN	-	-	-	-	0,81	0,70	0,63	92	0,91	0,76	0,66	134
	RF	0,79	0,68	0,61	136	0,81	0,71	0,64	90	0,91	0,75	0,65	136
	ET	0,79	0,68	0,61	**136**	**0,81**	**0,71**	**0,64**	**86**	**0,92**	**0,78**	**0,68**	**126**
	EFMH-DT	0,72	0,59	0,51	144	-	-	-	-	-	-	-	-
	EFMH-kNN	**0,80**	**0,70**	**0,63**	141	-	-	-	-	-	-	-	-
dataset_2	**Ba-DT**	-	-	-	-	0,67	0,51	0,41	26	0,74	0,58	0,47	30
	Ba-kNN	-	-	-	-	**0,69**	**0,54**	**0,43**	25	0,76	0,60	0,49	30
	RF	0,63	0,47	0,37	31	0,66	0,50	0,39	27	0,74	0,58	0,48	30
	ET	0,63	0,47	0,37	31	0,68	0,53	0,43	**25**	**0,77**	**0,61**	**0,50**	**29**
	EFMH-DT	0,61	0,44	0,33	35	-	-	-	-	-	-	-	-
	EFMH-kNN	**0,64**	**0,48**	**0,38**	**31**	-	-	-	-	-	-	-	-
dataset_3	**Ba-DT**	-	-	-	-	0,71	0,55	0,44	15	**0,78**	**0,63**	**0,53**	18
	Ba-kNN	-	-	-	-	0,71	0,56	**0,46**	15	0,77	0,62	0,52	**18**
	RF	**0,67**	**0,50**	**0,40**	**19**	0,71	0,55	0,45	15	0,77	0,62	0,52	18
	ET	0,66	0,50	0,39	19	**0,72**	**0,56**	0,46	**15**	0,77	0,62	0,52	18
	EFMH-DT	0,65	0,48	0,37	20	-	-	-	-	-	-	-	-
	EFMH-kNN	0,66	0,50	0,39	19	-	-	-	-	-	-	-	-
dataset_4	**Ba-DT**	-	-	-	-	0,81	0,71	0,65	12761	0,91	0,78	**0,69**	19659
	Ba-kNN	-	-	-	-	0,83	0,73	0,67	12530	0,92	0,76	0,65	20706
	RF	0,79	0,69	0,62	**19572**	**0,83**	0,74	0,67	12129	0,94	0,77	0,65	20897
	ET	0,81	0,71	0,65	19976	0,83	**0,74**	**0,67**	**11739**	**0,95**	**0,79**	0,69	**18871**
	EFMH-DT	0,77	0,65	0,58	20929	-	-	-	-	-	-	-	-
	EFMH-kNN	**0,81**	**0,72**	**0,65**	21061	-	-	-	-	-	-	-	-

TABLE IV
ALGORITHMS ENSEMBLE RESULTS, DYNAMIC PREDICTION

		LB approach				RVB approach				LVB approach			
	Alg.	**TLPQ_1**	**TLPQ_2**	**TLPQ_3**	**MAE**	**TLPQ_1**	**TLPQ_2**	**TLPQ_3**	**MAE**	**TLPQ_1**	**TLPQ_2**	**TLPQ_3**	**MAE**
dataset_1	**Ba-DT**	-	-	-	-	0,78	0,67	0,59	100	0,32	0,24	0,19	717
	Ba-kNN	-	-	-	-	0,80	0,68	0,59	102	0,43	0,32	0,25	489
	RF	0,83	0,68	0,58	283	0,67	0,52	0,42	146	**0,49**	**0,38**	**0,30**	**484**
	ET	0,77	0,64	0,55	306	**0,81**	**0,70**	**0,63**	**89**	0,38	0,29	0,22	522
	EFMH-DT	0,56	0,45	0,38	483	-	-	-	-	-	-	-	-
	EFMH-kNN	**0,84**	**0,70**	**0,60**	**249**	-	-	-	-	-	-	-	-
dataset_2	**Ba-DT**	-	-	-	-	0,68	0,52	0,42	26	0,36	0,28	0,22	109
	Ba-kNN	-	-	-	-	0,69	0,53	0,43	25	**0,66**	**0,51**	**0,41**	**62**
	RF	0,66	0,52	0,42	45	0,56	0,39	0,28	35	0,42	0,33	0,26	91
	ET	**0,71**	0,55	0,45	39	**0,70**	**0,55**	**0,44**	**24**	0,51	0,39	0,32	80
	EFMH-DT	0,49	0,37	0,28	76	-	-	-	-	-	-	-	-
	EFMH-kNN	0,71	**0,55**	**0,45**	**35**	-	-	-	-	-	-	-	-
dataset_3	**Ba-DT**	-	-	-	-	0,71	0,56	0,45	15	0,40	0,31	0,25	56
	Ba-kNN	-	-	-	-	0,72	0,57	0,46	15	**0,71**	**0,56**	**0,47**	**34**
	RF	0,69	0,55	0,46	26	0,62	0,46	0,35	18	0,43	0,34	0,27	52
	ET	0,70	0,55	0,45	24	**0,74**	**0,59**	**0,48**	**14**	0,47	0,37	0,30	48
	EFMH-DT	0,63	0,49	0,40	30	-	-	-	-	-	-	-	-
	EFMH-kNN	**0,72**	**0,57**	**0,47**	**21**	-	-	-	-	-	-	-	-
dataset_4	**Ba-DT**	-	-	-	-	0,76	0,65	0,58	17598	0,19	0,15	0,11	140616
	Ba-kNN	-	-	-	-	0,74	0,61	0,52	17214	0,43	0,33	0,26	**76328**
	RF	0,56	0,44	0,36	78180	0,67	0,52	0,42	22391	**0,49**	**0,38**	**0,30**	77869
	ET	0,68	0,53	0,43	61875	**0,83**	**0,74**	**0,67**	**11377**	0,35	0,26	0,20	89669
	EFMH-DT	0,41	0,32	0,26	105518	-	-	-	-	-	-	-	-
	EFMH-kNN	**0,78**	**0,61**	**0,50**	**44470**	-	-	-	-	-	-	-	-

Fig. 10 shows ranks related to TLPQ_1 metric obtained by ensembles during statistical test. As in case of single algorithms, ensembles applied with LB approach outranked ensembles with another approaches. The best rank got FMH-DT, ensemble which was designed for ordinal classification problems. Comparable results of the statistical tests were also viewed for TLPQ_2 and TLPQ_3 metrics.

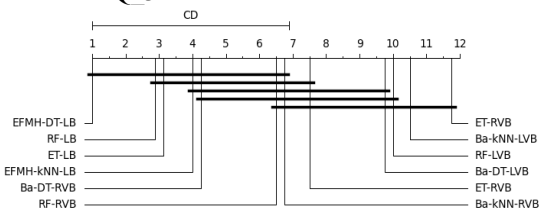

Fig. 10 - TLPQ_1 statistical tests, ensemble

CONCLUSIONS

In this paper, we proposed long-term traffic forecast strategies using ML and TS algorithms. We formulated the problem of long-term traffic forecasting in optical networks as an ordinal classification task. Due to different network technologies' characteristics, traffic forecasting has been realized by predicting traffic levels instead of exact traffic volume. Depending on the applied set of features, different ML and TS algorithms' approaches were proposed, i.e., LB approach, RVB approach, and LVB approach. Two different strategies were proposed and tested, namely, static prediction and dynamic prediction. We examined single ML algorithms, ML algorithms ensembles and compared their results to TS algorithms. To make an appropriate evaluation, we used a dedicated quality metric called TLPQ. It is a flexible metric, which can be adjusted to CSPs' expectations. Semi-synthetic datasets used during experiments are based on real traffic characteristics collected from SIX. The main conclusion is that proposed ML approaches obtained high quality metric results which are better than results returned by TS approaches, however selection of forecasting method should be preceded by an analysis of network traffic. As a future work, we plan to further explore the possibility of traffic forecasting using ML algorithms. The obtained knowledge we would like to apply to different network optimization tasks, i.e., routing, resource usage, and network planning.

REFERENCES

[1] A.Y. Grebeshkov, "Cognitive optical networks: architectures and techniques," *Optical Technologies for Telecommunications 2016*, International Society for Optics and Photonics, vol. 10342, 2017. http://doi.org/10.1117/12.2270294

[2] CISCO, "Cisco Annual Internet Report (2018–2023)," *CISCO*, 2020.

[3] Nokia. "Deepfield Network Intelligence Report Networks in 2020," *Nokia*, 2020.

[4] B. Mukherjee, I. Tomkos, M. Tornatore, P. Winzer and Y. Zhao, Springer Handbook of Optical Networks. *Springer International Publishing*, 2020.

[5] K. Walkowiak, *Modeling and Optimization of Cloud-Ready and Content-Oriented Networks*. Springer International Publishing, 2016, vol. 56, no. Decision and Control.

[6] V.W.S Chan, "Cognitive optical networks," in Proceedings of the IEEE International Conference on Communications (ICC), 2018, pp. 1-6. http://doi.org/10.1109/ICC.2018.8422787

[7] V.W.S Chan and E. Jang, "Cognitive all-optical fiber network architecture," *in Proceedings of the International Conference on Transparent Optical Networks (ICTON)*, 2017, pp. 1-4. http://doi.org/10.1109/ICTON.2017.8025063

[8] A. Knapińska, et al., "On Advantages of Traffic Prediction and Grooming for Provisioning of Time-Varying Traffic in Multilayer Networks.", *International Conference on Optical Network Design and Modeling*, IEEE, 2023.

[9] T. Panayiotou, M. Michalopoulou and G. Ellinas, "Survey on machine learning for traffic-driven service provisioning in optical networks", *IEEE Communications Surveys & Tutorials*, 2023. http://doi.org/10.1109/COMST.2023.3247842

[10] H. T. Mouftah and P. Ho, *Optical Networks Architecture and Survivability*. Springer Science & Business Media, 2003.

[11] M. Jinno, H. Takara, B. Kozicki, Y. Tsukishima, Y. Sone and S. Matsuoka, "Spectrum-efficient and scalable elastic optical path network: architecture, benefits, and enabling technologies," *IEEE Communications Magazine*, vol. 47, no. 11, pp. 66-73, 2009.
http://doi.org/10.1109/MCOM.2009.5307468

[12] L. Zong, G. N. Liu, A. Lord, Y. R. Zhou and T. Ma, "40/100/400 Gb/s mixed line rate transmission performance in flexgrid optical networks," in *Proceedings of the Optical Fiber Communication Conference (OFC)*, 2013, pp. OTu2A-2.

[13] Ciena, https://www.ciena.com/insights/data-sheets/800g-wavelogic-5-extreme-motr-module.html.

[14] D. Szostak, A. Włodarczyk and K. Walkowiak, "Machine Learning Classification and Regression Approaches for Optical Network Traffic Prediction," *Electronics*, vol. 10, no.13, 2021. https://doi.org/10.3390/electronics10131578

[15] G. Rzym, P. Boryło and P.A. Chołda, "A Time-Efficient Shrinkage Algorithm for Fourier-Based Prediction Enabling Proactive Optimization in Software Defined Networks," *International Journal of Communication Systems*, vol. 33, no. 12, 2019. https://doi.org/10.1002/dac.4448

[16] P. Cortez, M. Rio, M. Rocha and P. Sousa, "Multi-scale Internet traffic forecasting using neural networks and time series methods," *Expert Systems*, vol. 29, no. 2, pp. 143-155, 2012. https://doi.org/10.1111/j.1468-0394.2010.00568.x

[17] T. Otoshi, Y. Ohsita, M. Murata, Y. Takahashi, K. Ishibashi and K. Shiomoto, K. "Traffic prediction for dynamic traffic engineering," *Computer Networks*, vol. 85, pp. 36-50, 2015. https://doi.org/10.1016/j.comnet.2015.05.001

[18] P. Cembaluk, J Aniszewski, A Knapińska and K Walkowiak, "Forecasting the network traffic with PROPHET," in *Proceedings of the PP-RAI*, 2022.

[19] A. Knapińska, K. Półtorak, D. Poręba, J. Miszczyk, M. Daniluk, and K. Walkowiak, "On Feature Selection in Short-Term Prediction of Backbone Optical Network Traffic," in *Proceedings of the 2022 International Conference on Optical Network Design and Modeling (ONDM)*, 2022. pp. 1-6. http://doi.org/10.23919/ONDM54585.2022.9782850

[20] D. Szostak, and K. Walkowiak, "Application of Machine Learning Algorithms for Traffic Forecasting in Dynamic Optical Networks with Service Function Chains," *Foundations of Computing and Decision Sciences*, vol. 45, no. 3, pp. 217-232, 2020. http://doi.org/10.2478/fcds-2020-0012

[21] D. Szostak, and K. Walkowiak, "Influence of traffic type on traffic prediction quality in dynamic optical networks with service chains," in *Proceedings of the PP-RAI*, 2019.

[22] D. Szostak, and K. Walkowiak, „Machine learning methods for traffic prediction in dynamic optical networks with service chains," in *Proceedings of the 21st International Conference on Transparent Optical Networks (ICTON)*, 2019. p. 1-4.
http://doi.org/10.1109/ICTON.2019.8840301

[23] R. Boutaba, M.A. Salahuddin, N. Limam, S. Ayoubi, N. Shahriar, F. Estrada-Solano and O.M. Caicedo, "A comprehensive survey on machine learning for networking: evolution, applications and research opportunities," *Journal of Internet Services and Applications*, vol. 9, no. 16, pp. 1-99, 2018. http://doi.org/10.1186/s13174-018-0087-2

[24] J. Mata, I. de Miguel, R.J. Duran, N. Merayo, S.K. Singh, A. Jukan and M. Chamania, "Artificial intelligence (AI) methods in optical networks: A comprehensive survey," *Optical Switching and Networking*, vol. 28, pp. 43-57, 2018. http://doi.org/10.1016/j.osn.2017.12.006

[25] D. Szostak, “Machine Learning Ensemble Methods for Optical Network Traffic Prediction,” in *Proceedings of Computational Intelligence in Security for Information Systems Conference*, 2021. p. 105-115. http://doi.org/10.1007/978-3-030-87872-6_11

[26] D. Szostak, K. Walkowiak, and A. Włodarczyk, “Short-term traffic forecasting in optical network using linear discriminant analysis machine learning classifier,” in *Proceedings of 22nd International Conference on Transparent Optical Networks (ICTON)*, 2020, pp. 1–4. http://doi.org/10.1109/ICTON51198.2020.9203040

[27] A. Włodarczyk, P. Lechowicz, D. Szostak, K. Walkowiak, „An algorithm for provisioning of time-varying traffic in translucent SDM elastic optical networks”, *22nd International Conference on Transparent Optical Networks (ICTON)*, 2020.

http://doi.org/10.1109/ICTON51198.2020.9203045

[28] M. Zukerman, T.D. Neame and R.G. Addie, “Internet traffic modeling and future technology implications,” in *Proceedings of the IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 1, 2003, pp. 587-596. http://doi.org/10.1109/INFCOM.2003.1208709

[29] W. Kotlowski and R. Slowinski, “On nonparametric ordinal classification with monotonicity constraints,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, pp. 2576–2589, 2012. http://doi.org/10.1109/TKDE.2012.204

[30] S. Sridhar and A. Kalaivani, “A Survey on Methodologies for Handling Imbalance Problem in Multiclass Classification,” *Advances in Smart System Technologies*, 2021, pp. 775-790. http://doi.org/10.1007/978-981-15-5029-4_67

[31] J. Tanha, Y. Abdi, N. Samadi, N. Razzaghi and M. Asadpour, “Boosting methods for multi-class imbalanced data classification: an experimental review,” *Journal of Big Data*, vol. 7, no. 1, pp. 1-47, 2020. http://doi.org/10.1186/s40537-020-00349-y

[32] L. Gaudette and N. Japkowicz, “Evaluation methods for ordinal classification,” in *Proceedings of 2nd Canadian Conference on Artificial Intelligence*, Springer, 2009, pp. 207-210. http://doi.org/10.1007/978-3-642-01818-3_25

[33] C.M. Bishop and M. Nasser, *Pattern Recognition and Machine Learning*. Springer, 2006, vol. 4, no. 4.

[34] W. Waegeman, B. de Baets and L. Boullart, “ROC analysis in ordinal regression learning,” *Pattern Recognition Letters*, vol. 29, no. 1, pp. 1-8, 2008. http://doi.org/10.1016/j.patrec.2007.07.019

[35] A. Luque, A. Carrasco, A. Martín and A. de Las Heras, “The impact of class imbalance in classification performance metrics based on the binary confusion matrix,” Pattern Recognition, vol. 91, pp. 216-231, 2019. http://doi.org/10.1016/j.patcog.2019.02.023

[36] E. Amigó, J. Gonzalo, S. Mizzaro, J. Carrillo-de-Albornoz, "An effectiveness metric for ordinal classification: Formal properties and experimental results," *arXiv preprint*, 2020. http://doi.org/10.18653/v1/2020.acl-main.363

[37] S. Baccianella, A. Esuli and F. Sebastiani, “Evaluation measures for ordinal regression,” in *Proceedings of Ninth International Conference on Intelligent Systems Design and Applications*, 2009, pp. 283-287. http://doi.org/10.1109/ISDA.2009.230

[38] P. Bellmann and F. Schwenker, "Ordinal Classification: Working Definition and Detection of Ordinal Structures," *IEEE Access*, vol. 8, pp. 164380-164391, 2020. http://doi.org/10.1109/ACCESS.2020.3021596

[39] J.S. Cardoso, and R Sousa, "Measuring the performance of ordinal classification," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 25, no. 08, pp. 1173-1195, 2011. http://doi.org/10.1142/S0218001411009093

[40] M. Cruz-Ramírez, C. Hervás-Martínez, J. Sánchez-Monedero and P.A. Gutiérrez, “Metrics to guide a multi-objective evolutionary algorithm for ordinal classification,” *Neurocomputing*, vol. 135, pp. 21–31, 2014. http://doi.org/10.1016/j.neucom.2013.05.058

[41] M. Lázaro and A.R. Figueiras-Vidal, “Neural network for ordinal classification of imbalanced data by minimizing a Bayesian cost”, *Pattern Recognition*, vol. 137, 2023. http://doi.org/10.1016/j.patcog.2023.109303

[42] W. Cao, V. Mirjalili and S. Raschka, “Rank consistent ordinal regression for neural networks with application to age estimation,” *Pattern Recognition Letters*, vol. 140, pp. 325-331, 2020. http://doi.org/10.1016/j.patrec.2020.11.008

[43] J.C. Gámez, D. García, A. González and R. Pérez, “Ordinal classification based on the sequential covering strategy,” *International Journal of Approximate Reasoning*, vol. 76, pp. 96-110, 2016. http://doi.org/10.1016/j.ijar.2016.05.002

[44] M. Tang, R. Pérez-Fernández and B. De Baets, “A comparative study of machine learning methods for ordinal classification with absolute and relative information,” *Knowledge-Based Systems*, vol. 230, 2021. http://doi.org/10.1016/j.knosys.2021.107358

[45] M. Tang, R. Pérez-Fernández and B. De Baets, “Distance metric learning for augmenting the method of nearest neighbors for ordinal classification with absolute and relative information,” *Information Fusion*, vol. 65, pp. 72-83, 2021. http://doi.org/10.1016/j.inffus.2020.08.004

[46] B. Vega-Márquez, I. A. Nepomuceno-Chamorro, C. Rubio-Escudero and J. C. Riquelme, “OCEAn: Ordinal classification with an ensemble approach,” *Information Sciences*, vol. 580, pp. 221-242, 2021. http://doi.org/10.1016/j.ins.2021.08.081

[47] N. Bhatia, “Survey of nearest neighbor techniques,” *arXiv preprint*, 2010.

[48] T. Cover and P. Hart, “Nearest neighbor pattern classification,” *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967. http://doi.org/10.1109/TIT.1967.1053964 S. Dominguez-Almendros, N. Benitez-Parejo and A.R. Gonzalez-Ramirez, “Logistic regression models,” *Allergologia et immunopathologia*, vol. 39, no. 5, pp. 295-305, 2011.

[49] D.C. Montgomery, E.A. Peck and G.G. Vining, *Introduction to linear regression analysis*. John Wiley & Sons, 2021.

[50] H. Taud and J.F. Mas, “Multilayer perceptron (MLP),” *Geomatic approaches for modeling land change scenarios*, 2018, pp. 451-455.

[51] F. Rosenblatt, “Perceptron simulation experiments,” in *Proceedings of the IRE*, vol. 48, no. 3, pp. 301–309, 1960. http://doi.org/10.1109/JRPROC.1960.287598

[52] T.G. Dietterich, “Ensemble methods in machine learning,” *International Workshop on Multiple Classifier Systems*, 2000, pp. 1-15.

[53] L.I. Kuncheva, Combining Pattern Classifiers: Methods and Algorithms. John Wiley & Sons, 2014.

[54] Y. Yang, H. Lv and N. Chen, “A survey on ensemble learning under the era of deep learning.” *Artificial Intelligence Review*, 2023, vol. 56(6), pp. 5545-5589. http://doi.org/10.1007/s10462-022-10283-5

[55] Extra Trees Classifier, https://www.scikit-learn.org/stable/modules/generated/sklearn.ensemble.ExtraTreesClassifier.html

[56] Extra Trees Regressor, https://www.scikit-learn.org/stable/modules/generated/sklearn.ensemble.ExtraTreesRegressor.html

[57] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay, “Scikit-learn: Machine Learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825-2830, 2011.

[58] G. Biau and E. Scornet, “A random forest guided tour,” *Springer*, vol. 25, no. 2, pp. 197-227, 2016. http://doi.org/10.1007/s11749-016-0481-7

[59] E. Frank and M. Hall, “A Simple Approach to Ordinal Classification,” *European Conference on Machine Learning*, Springer, 2001, pp. 145-156. http://doi.org/10.1007/3-540-44795-4_13

[60] C.D. Sutton, “Classification and regression trees, bagging, and boosting,” Handbook of statistics, vol. 24, pp. 303-329, 2005.

[61] G.T. Smith, pmdarima: ARIMA estimators for Python, 2017, http://www.alkaline-ml.com/pmdarima