ML-based Fault Management Automation in Large-scale Fixed and Mobile Telecommunication Networks

Maryam Bandali, Jaume Rius I Riu, Andreas Lewitzki *Telenor Group*
Sweden
James Gross, Dibbendu Roy *KTH Royal Institute of Technology Stockholm*
Sweden

***Abstract*—Many network faults are flooding the telecommunication companies in the form of Trouble Tickets (TT). Automation in managing these TTs is vital in increasing customer satisfaction. We develop a solution to address two challenges regarding TTs generated from fixed and mobile access network domains: Prediction of resolution times and technician dispatch needs. Our study utilizes datasets from Telenor, a Swedish telecommunication operator, encompassing 35,000 access switches and 8,000 base stations. It incorporates 40,000 switch TTs and 22,000 mobile TTs during 2019-2023. None of the previous works studied multiple telecommunication domains or considered the time evolution of TTs. This work comprehensively studies several prediction models for the mentioned use cases and network domains. Our models successfully outperform the company baseline and best proposed state-of-the-art models. Within 1-hour confidence interval, our method can correctly predict shortest ranges of resolution times for 90% of switch TTs and 80% of mobile TTs. We also predict the necessity of dispatching workforce to the place with weighted F1 scores of respectively, 88% and 89% for switch and mobile TTs which shows high average accuracy of our system in prediction across both dispatch and non-dispatch TT classes to assist operation. With these scores, our model is capable of allocating resources automatically, enhancing customer satisfaction. We also studied the TTs evolution, for example, for switch TTs, within 15 minutes of creation time, prediction improves by 57% and 50%, for resolution and dispatch prediction, respectively.**

***Index Terms*—Trouble Tickets, Mobile Network, Fixed Network, Fault Management, Resolution Time Prediction, Dispatch-need Prediction, Machine Learning Models**

I. Introduction

The history of network management is characterized by a gradual progression from manual configurations (e.g., Early packet-switching networks such as ARPANET [1]) and scripting (Command-Line Interfaces (CLI)) to more sophisticated approaches involving standardized protocols (Simple Network Management Protocol (SNMP)), centralized control (Software-Defined Networking (SDN)/Network Function Virtualization (NFV)), intent-driven policies (Intent-Based Networking (IBN)), and intelligent automation technologies (Artificial Intelligence (AI) and Machine Learning (ML)) [2]. Traditional approaches to managing networks are often time and cost-consuming and prone to errors. The over-time need for continuous connectivity, optimizing the user experience, and dynamic networks such as 5G and beyond necessitates engaging automation in network management processes [3]. Such networks can autonomously monitor, analyze, and optimize their performance, minimizing human intervention, leading to zero-touch and self-driving networks [4]. However, pragmatic realization calls for incremental deployment strategies with backward compatibility to legacy systems, gradually leading to a smooth technology transition [5].

In this work, we aim to bridge the gap between the concept of self-driving networks in the literature and the reality of Telecommunication (Telco) networks by implementing an automated assistant system for operation on top of our existing network infrastructure [5]. Networks owned by Telco service providers consist of thousands of interconnected components such as access switches, radio base stations, data centers, routers, and cables. In such a complex network, it is expected that faults happen on any component from time to time and cause disturbance in the network. This potentially leads to customer dissatisfaction and complaints. The company's Network Operation Center (NOC) manages network faults and handles customers' network performance-relevant complaint cases. In such a complicated network that contains many different elements, manually controlling the cases and responding to all complaints is infeasible and a significant roadblock to scaling network functionality for NOC personnel; thus, automating the tasks plays an important role.

With ever-growing traffic data and network elements, one way to achieve automation for network fault management is to use ML techniques. ML-based automation allows learning insights from data and helps develop automated assistant systems. These systems can assist personnel in the prognosis and diagnosis of network issues. Looking at insights, NOC can proactively troubleshoot and plan tasks, resolving network issues quickly. This reduces human intervention, optimizes resource usage, and minimizes operational costs, enhancing network connectivity and performance. Ultimately, a higher Quality of Service (QoS) and a better Quality of Experience (QoE) can be ensured over time.

In the realm of Telco, particularly at Sweden's leading service provider, Telenor, network faults and resolutions are documented through Trouble Ticket (TT) records in an internal incident handling system. These TTs, categorized as Customer TTs (reported by customers) and Network TTs (originating from faulty network elements), contain various fields storing information, as depicted in Fig.1. TTs go through an evolution

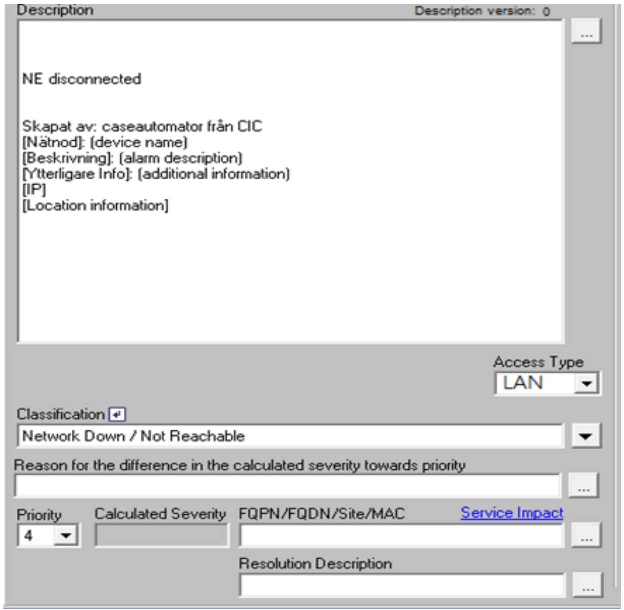


Fig. 1: Part of a network TT record

process during their life cycle. The evolution process of network TTs is rooted in updates from customer calls, NOC operators, and automatic system updates (Fig.3). Telenor's NOC employs varying degrees of automation in TTs handling tasks, primarily relying on predefined rules established by human experts. However, manual or rule-based handling of TTs may lead to challenges and potential errors due to data heterogeneity and causes resolution delays [6]. This paper facilitates NOC work of handling TTs by automating part of TT handling process. To this end, we engage TTs generated from faulty fixed access switches and radio access base stations, in both fixed and mobile network domains (referred to as ***Switch TTs*** and ***Mobile TTs***). We introduce two use cases: TT resolution time prediction and TT on-site dispatch-need prediction. Leveraging Telenor's extensive TT dataset, this study represents a pioneering effort in automating Telco network TTs by thoroughly examining their evolution over time using data-driven approaches. It aims to provide pragmatic steps towards TTs resolution automation at Telenor and similar Telco service providers.

We present some relevant works in this domain and highlight the paper's contributions.

A. Related Work

Fault management is crucial in today's heterogeneous Telco network systems to ensure effective communication. It involves detecting, diagnosing, isolating, and resolving network malfunctions [7]–[10]. Traditionally rule-based approaches were used for network fault management. These approaches worked based on rules that require expert knowledge. Relying on expert knowledge makes the fault management process time-consuming and prone to errors [7]. Recently, ML techniques have been used to learn knowledge from historical data. A large number of works managed to detect the root causes of network faults with different ML techniques such as neural networks, decision trees, Bayesian networks, and dependency graphs [11]–[18]. Some studies detected abnormality in the network using approaches such as neural networks, support vector machines, and probabilistic inference [19]–[22]. Some works predicted the faults using neural networks and support vector machines [23]–[25]. Authors in [8], [26]–[29] proposed recommender systems to automatically detect the network status using tree-based models or reinforcement learning techniques and recommended appropriate actions for fault resolution. All the mentioned works above mostly rely on data such as alarm logs, network event traces, sensor measurements, and real or simulated network topology data. In this work, we engage network TTs to automate part of NOC fault resolution tasks.

In the last few decades, several works have investigated different aspects of TTs. For example, various works [30]–[33] applied natural language processing and data mining methods to find the general trend in disturbances. Also, a wide range of works [34]–[37] focused on using ML techniques to predict TT occurrences proactively. For instance, in [34], the likelihood of TT occurrences and the best prevention actions are predicted based on the TTs studied causes.

Some works focused on managing TTs by addressing different aspects of their handling processes. For example, works presented in [38], [39] investigated applying ML models for predicting severity changes and on-site dispatch-need of TTs. An average weighted F1 score of almost 60% is achieved from [38] based on the evaluation metrics reported for predicting the on-site dispatch-need of automatically and manually generated Telco network TTs.

Some works investigated the prediction of TTs resolution time; for example, [40] used a classification method to predict the TTs resolution time spans (achieved 74.5% accuracy) and regression for predicting the value of TTs resolution time (achieved Mean Absolute Error (**MAE**) of 24.8 hours). Moreover, Haw et al. [41] predicted the resolution time of Telco customer TTs using classification and regression approaches in sequence. They implemented a classifier to predict the severity and used its outcome to predict the TT resolution times using regression. They achieved Root Mean Squared Error (**RMSE**) of 1 day and 12 hours in their respective dataset. TTs also caught attention in other domains. [42] is an example that studied TTs in electrical power companies. They investigated the prediction of resolution times of power outages.

Studies conducted in [38], [40]–[42] are the closest to ours. However, they come with shortcomings or are performed in domains other than Telco networks. The concept of TT evolution over time has been discussed in [38]; however, no results are presented about it. Furthermore, the results reported in this work regarding predicting dispatch-need for Telco network TTs are inconclusive. Firstly, classification accuracy is a poor metric of model performance to report in this setup. This is due to the unbalanced nature of the data set, which is indicated by the high percentage of false negative cases (around 63%) in their reported results [43]. Additionally, this work does not specify to which network domains their results belong. We consider the best dispatch-need prediction model proposed in [38] as a scientific baseline method to compare with the results of our second use case.

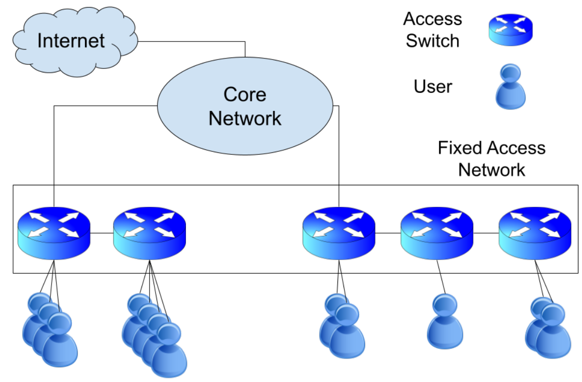
[40]–[42] address the prediction of TTs resolution time in domains other than Telco. However, we consider the best regression model reported in [40] as a scientific baseline method to compare with the results of our first use case. The comparisons have been presented and discussed in section IV

B. Contributions

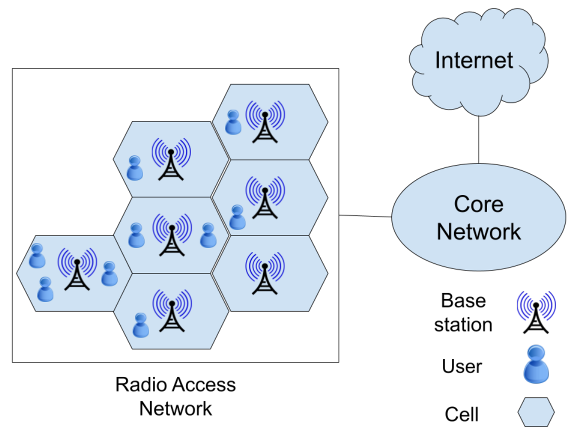
This study considers analyzing Telenor network TTs generated automatically or manually from network elements such as access switches and radio base stations. To the best of our knowledge, this is the first time this analysis has been presented for TTs from two fixed and mobile network domains, and TT evolution analysis has also been taken into account. All services offered by Telenor to the customers benefit from this solution since it helps reduce the impact of network faults on customers and optimize resource usage. Our main contributions are as follows:

- We automate part of TT handling process by predicting TTs resolution time and dispatch need to provide assistance to NOC. To this end, we extract valuable knowledge from TT by preprocessing and feature engineering of TT snapshots collected from Telenor's network encompassing 35,000 access switches and 8,000 base station cells. It incorporates 40,000 switch TTs and 22,000 mobile TTs between 2019 and 2023. Existing literature on the prediction of TTs resolution time and dispatch-need either do not use Telco datasets or uses improper evaluation metrics. To the best of our knowledge, it is the first time the resolution time has been predicted using Telco datasets of switch and mobile TTs, respectively.
- Introducing a 1-hour confidence interval results in accurate resolution time ranges in 90% and 80% of cases for switch and mobile TTs, respectively. Our models achieve an 80% and 61% reduction in MAE compared to the company baseline at TT creation time. We also predict the necessity of dispatching workforce to the site with weighted average F1 scores of respectively, 88% and 89% for switch and mobile TTs. With these scores, our model is capable of allocating resources automatically, enhancing customer satisfaction. These may serve as guidelines for other Telco providers interested in automating their TT handling process.
- In examining the evolution of switch TTs, our study represents the analysis of Telco network TTs evolution in ML-based automation for network fault management. Within 15 minutes of creation, our model exhibits a substantial 57% and 50% improvement in MAE and average F1 score for predicting, respectively, resolution times and dispatch-needs compared to the results reported at creation time.

The paper unfolds as follows: Section II provides background on network domains and the Telenor TT handling system. Section III details our data analysis methodology and the applied predictive models. Section IV presents results, comparisons, and insights for the considered models. The paper concludes in Section V, summarizing findings and outlining future directions.



(a)



(b)

Fig. 2: Topology of the network, (a) Fixed, (b) Mobile

II. Background

This section will discuss the details of the two network domains we referred to in I. Additionally, we will talk about the data sources we use to extract data. To this end, in subsection II-A, we will discuss the architectures, functionality, and fault generation processes on fixed and mobile network domains, followed by the description of the TT incident handling system and TT life cycle in Section II-B.

A. Network Domains Overview

Consumer services offered by Telenor Sweden include mobile (2.9M subscribers), broadband (700K subscribers), and TV (500K subscribers). Telenor owns a fixed and mobile network consisting of thousands of switches and radio base stations distributed over the country to provide these services.

A fixed network connects various devices to a Local Area Network (LAN) and forwards data within a building or small geographical area, providing services such as Voice Over IP (VOIP) and fixed broadband. Telenor Sweden manages around 35,000 such access switches. In contrast, a mobile network uses radio base stations to offer wireless connections to mobile devices within designated cells, which together form a Radio Access Network (RAN) linked to a core network for call routing and data transfer. Telenor Sweden operates nearly 8,000 base station cells(Fig.2).

As a result, Telenor is managing complex running networks consisting of thousands of interconnected fixed and mobile service-providing elements. Different types of faults are expected to happen in these complex networks. Faults within the network can be defined as any incidents that impede regular operation, disrupting the connection and affecting customer service. Timely response to these faults can guarantee the smooth operation of the network.

A network disturbance happens because of the faults in the elements. The component cannot respond to the controlling messages in this situation, and an alarm is raised. The source of error could be a power outage, threshold exceed, signal degradation, fiber cuts, planned maintenance, etc. [44]. The alarm might be automatically or manually converted to network TT, depending on the severity and type of the alarm (see Fig.3 (a)). In the case of automatic TT creation, the system fills out the TT's fields. Otherwise, personnel at NOC perform it manually. Once a TT is created, we consider its timestamp as T0. The timestamp indicates the evolution of a TT. As time evolves, some features in TT change due to addition of more information. All TT monitoring and management processes are performed in the incident handling system [45].

B. TTs handling System

TTs go through a journey from the moment they are created. NOC administrators work on resolving the TTs during this journey and make logs of their actions. At each time stamp of this journey, the values of TT fields might change based on the information realized from the fault up to that moment. The values can be updated automatically by systems or manually by administrators. Fig.3 (b) shows how TTs evolve during their life cycles.

Depending on the fault's causes, the TT's resolution is performed automatically or manually. In some cases, for example, if the fault happens because of a power outage, the TT usually is resolved automatically by itself and restarts responding to the monitoring message once the power is back [44]. In other cases, TTs are resolved by NOC system and field engineers either remotely or by dispatching the workforce to the place. When a TT is generated, NOC performs several remote tests. In some cases, rebooting the equipment or changing the configuration parameters resolves the problem remotely (e.g. software faults). Otherwise, the NOC will schedule on-site troubleshooting to check for the underlying causes (e.g. fiber break) [44], [45]. Fig.3 shows a diagram of TTs creation, handling, updating, and resolution over time.

When a TT is resolved, the period it has taken for resolution (resolution time) and information about the necessity of dispatching the workforce to the site (dispatch-need) are recorded in separate fields in the TT record. Currently, the company reports 480 minutes resolution time for all TTs once they are generated from access switches and radio base stations. This is an estimation based on the expertise and service level agreement. However, there is no estimation for the dispatch-need. It is usually realized by monitoring the fault symptoms during the TT handling procedures. This could take a few minutes to several hours, depending on the problem. This

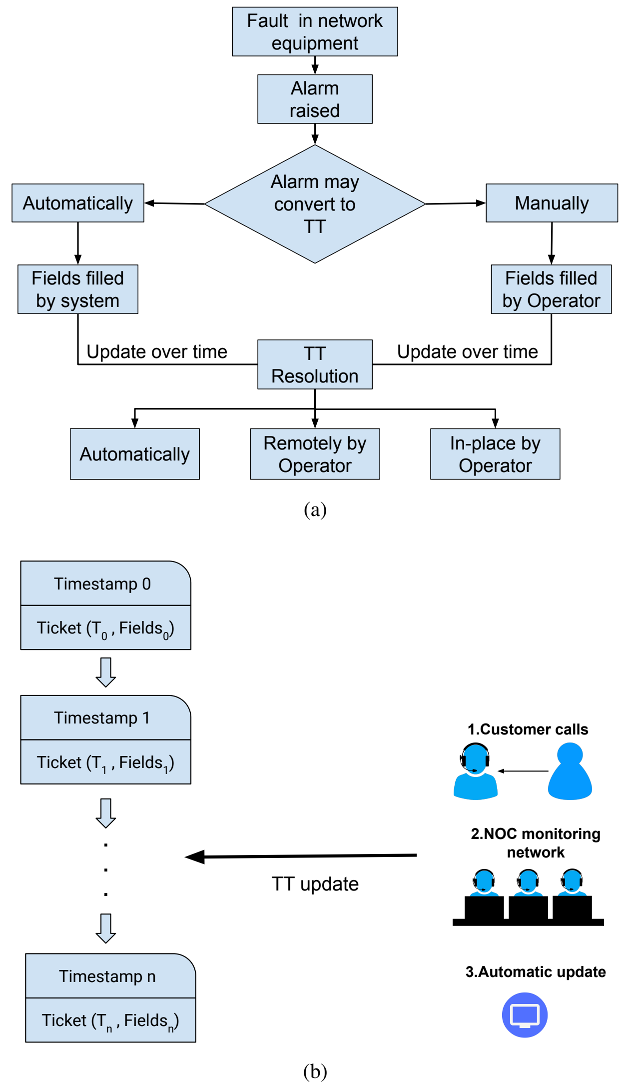


Fig. 3: (a) TTs generation process (b) Evolution of TT over time

paper investigates predicting the resolution time and dispatch-need in order to facilitate NOC operations.

III. Data Analysis and Predictive Models

This section gives an overview of the TT data sets, the feature engineering steps, and the predictive ML models we use for training. In subsection III-A, we will discuss switch and mobile TT data sets and the information available in them. In subsection III-B, we will explain the feature engineering steps on TT data sets. Finally, in subsection III-C, we will elaborate on different ML models and their tuning.

A. TT Dataset

This study uses approximately 40,000 switch TTs and 22,000 mobile TTs[1]. We analyze these two datasets separately because of their different fields (Fig.1) and characteristics. This

[1]For access to the code and datasets, please contact the corresponding author.

study only considers the fields with structured formats, excluding those filled with human natural language. This exclusion is due to inconsistency in quality, language variability (Swedish and English), and the potential inclusion of sensitive customer information. These factors make such fields unreliable for our study. Using regular expressions, we manage to mine the textual fields of the TTs and extract vital information about faults. We refer to the fields and extracted information as *features* of the TT.

Some examples of extracted features from both datasets include dates and time of TT creation, location of the element, TT severity weights, model and technology of the device, the impact of the fault on different services, alarm categories, TT resolution time, TT dispatch-need, parent TT, and the number of children. Parent TTs are the ones that have caused the generation of other TTs. Not all features are available at the time of TT creation (T_0). As discussed before, TT resolution time is added later on to the tickets (when the TT case is resolved). Information about the dispatch-need, fault impact on different services, parent TT, and the number of children are added later and updated over time as more information is received from the faults (T_1, T_2, etc.). The complete list of extracted features and their descriptions for both switch and mobile TTs can be found in Appendix A section.

B. *Feature Engineering and Visualization*

We keep 80% of the data for training the models and 20% for testing their performance on unseen data. There are many blank fields in both datasets. To prepare data for model building, we fill out the categorical blank fields with a constant value and infer blank numerical fields from known parts of the data using the iterative imputation technique [46]. Furthermore, we encode the categorical features so that ‘1’ stands for the category that appears in the field and ‘0’ stands for the remaining categories. We also scale the numerical features between 0 and 1 to speed up the optimization process of the models’ cost function.

Fig.4a and 4b show the histogram of the switch and mobile TTs resolution time distribution. There are peaks in the first bins of both figures, suggesting that many of TTs are resolved within first few hours of their creation. That is because the most common causes of TTs generation in our system are power outages and fiber breaks. TTs that are created because of power outages are usually automatically resolved within a few hours. However, TTs that are generated because of fiber breaks need more time usually up to 3 days for recovery. TTs resolved above 3 days might happen because of for example, planned construction, equipment dismantling, replacement or other rare causes. Hence, the range of resolution time that we consider for this study is 1 to 4,320 minutes, equivalent to 3 days. Finally, after taking all feature engineering and cleaning steps, we are left with 37,807 switch and 19,554 mobile TTs.

Fig.4c and 4d show the distribution of dispatch-need for switch and mobile TTs. In this case, the target feature is binary (Yes or No). In both data sets, almost 11% to 12% of the TTs need on-site work. This indicates an imbalanced distribution in the data sets for predicting dispatch-needs.

C. *Applied ML Models and Model Tuning*

We want to build models to predict the resolution time and dispatch-need for switch and mobile TTs. We train the models on the set of TTs’ features available at the time of ticket creation; the target features for prediction are TTs’ resolution time and TTs’ dispatch-need.

Since the target feature is continuous, we use ML regression models in the resolution time prediction problem. The regression models that we use for this prediction are as follows:

- LINear Regression (**LINR**) [47]
- K Nearest Neighbor (**KNN**) [48]
- Decision Tree (**DT**) [49]
- Random Forests (**RF**) [50]
- eXtreme Gradient Boosting (**XGB**) [51]
- Fully connected Neural Network (**NN**) with 3-hidden dense layers [52]

In the case of dispatch-need prediction, we apply ML classification models. The classifiers that we use for this task are as follows:

- LOGistic Regression (**LOGR**) [53]
- Random Forest (**RF**)
- Extreme Gradient Boosting (**XGB**)
- Fully connected Neural Network (**NN**) with 3-hidden dense layers

We use three-fold cross-validation with Bayesian search CV [2] [54] (on training set) for tuning the hyper-parameters and optimizing the results on Mean Absolute Error (**MAE**) metric (Eq. 1) for regression and Area Under the Curve (AUC) score (Eq. 2) for classification [55], that can be computed according to:

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i| \tag{1}$$

Where n is the number of samples, y_i is the actual value, $\hat{y}_i$ is the predicted value.

$$\mathrm{AUC} = \int_{-\infty}^{\infty} \mathrm{ROC}(t)\, dt \tag{2}$$

Where the Receiver Operating Characteristic (ROC) curve is the plot that represents the True Positive Rate (TPR) against the False Positive Rate (FPR) at various threshold settings. Using Bayesian search CV, we can efficiently explore the search space of the parameters. To find the best parameter set, we iterate on the parameter space of each model 50 times on three folds (default values of the scikit-learn library). Table I shows the parameter space we consider for tuning each regression and classification model [51], [56].

IV. Numerical Evaluation

This section discusses the numerical results obtained after training and evaluating the models. To this end, in subsection IV-A and IV-B, we present and discuss the results of, respectively, resolution time and dispatch-need prediction tasks at TT creation time. In subsection IV-C, we analyze if adding more

[2] https://scikit-optimize.github.io/

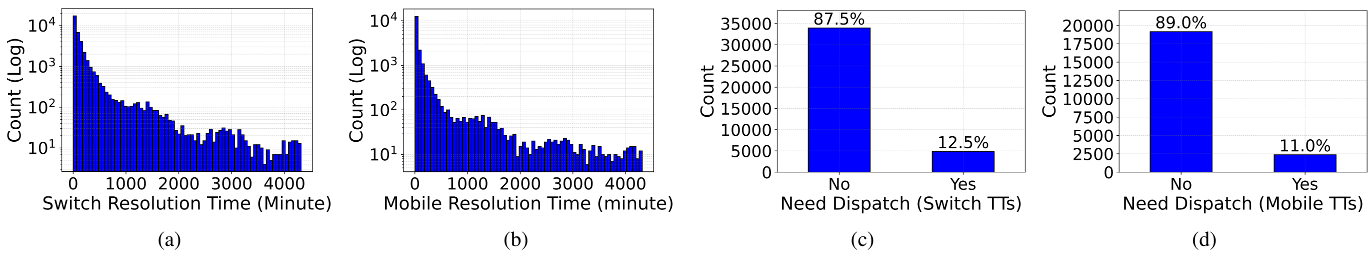


Fig. 4: Distribution of TTs resolution time, (a) Switch, (b) Mobile. Distribution of TTs' dispatch-need, (c) Switch, (d) Mobile

TABLE I: Models' Search Parameter Space

MODEL	Parameter Space
KNN	n_neighbors: [2,15]
LINR	fit_intercept:[True, False]
LOGR	Inverse_regularization_strength:[10^{-4},10^4], Penalty:{l1, l2}
DT	max_depth:[10,800], max_features:[0.5,1]
RF	max_depth:[1,100], n_estimators:[100,1000], max_features:[0.1,1], max_samples: [0.1,0.99],
XGB	learning_rate: [0.01,0.1], max_depth: [1,100], n_estimators: [100,1000], colsample_bytree: [0.1,1] subsample: [0.1,1]
NN	nodes:[16, 256], batch_size: [4, 64], Learning_rate: [10^{-5},10^{-1}], optimizer: (rmsprop, adam, nadam, sgd), activation_func:(relu, tanh)

data samples will improve the performance of the models. In subsection IV-D, we will present an approach in order to put the results into practice. Finally, in subsection IV-E, we will explain the results of predictions at different time stamps of TTs' life cycle.

A. Results of Resolution Time Prediction at T_0

First, we build the models based on all information available at the creation time of TTs (T_0). We do this to inform NOC administrators and customers about the probable time ranges within which the network faults will be resolved. In switch TTs, information such as mother TT and the number of children are updated over time. In mobile TTs, the way the data is collected about faults differs from switch TTs, and there is no such updating of information over time. We train the models on the train set, tune their parameters, and evaluate them on the test set. Fig. 5 shows the models' MAE results on switch and mobile train, validation, and test sets at T_0 [55], [57].

We compare the MAE of our models with MAE of the Company Baseline (**CB**), a Naive Baseline (**NB**), and the best models in the State Of The Art (**SOTA**) [40] (Fig. 5). We consider naive baseline as reporting the mean of the distribution for all TTs. For switch TTs, all models perform better than the company and naive baselines with MAEs of respectively about 430 and 245 minutes on the test set. LR and KNN with MAE of almost 240 minutes on the test set show weak performance in predicting the target. Instead, tree-based models such as DT, RF, XGB, and NN perform well for this prediction. XGB and NN with MAE of around 85 minutes on the test set perform almost the same, outperforming other models and SOTA with MAE of 188 minutes. For mobile TTs, all models perform better than the baselines with MAE of respectively, 480 and 295 minutes on test set. KNN performs better than LR and DT. XGB and NN, with MAE of almost 185 minutes on test, perform the best among other algorithms.

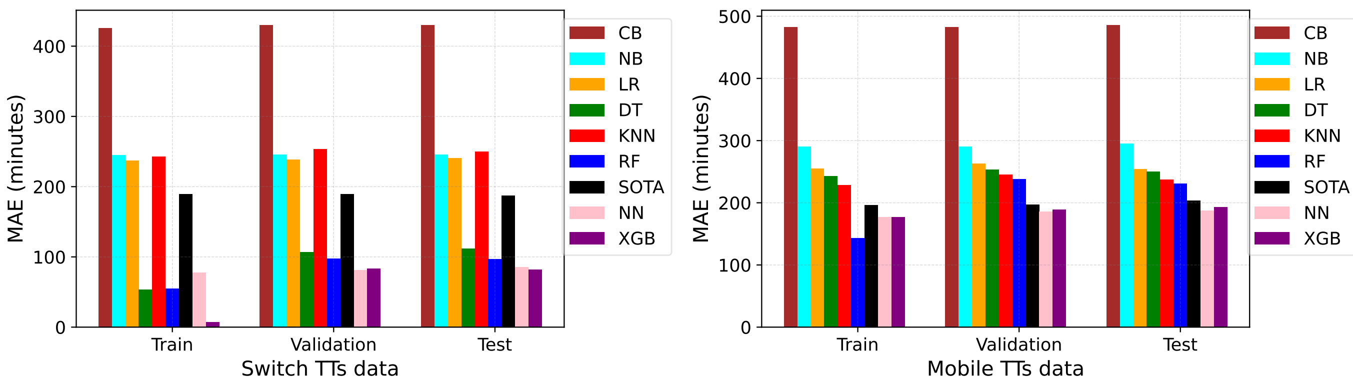
Fig. 6 shows the residual plots of XGB models for switch and mobile TTs. The model on switch TTs perform better than the mobile TTs, which can be also observed in Fig. 5. On the switch dataset, the model predicts about 82% of the resolution times with absolute value of less than 100 minutes. On the mobile dataset, the model predicts about 78% of tickets with absolute residual values less than 100 minutes. Also, it can be observed that on switch dataset, model performs well on TTs that have long resolution time. It should be noted that the number of tickets that have residual values above 1000 minutes in Fig. 6 in switch and mobile TT data sets are respectively, around 1% and 6% of test set.

Based on the results from switch and mobile TTs resolution time prediction, we can draw several conclusions. First, various characteristics of access switch and radio base station faults, which are reflected in both data sets, result in different predictive performance of the models. Second, we have improved the MAEs compared to the company baseline by 80% for switch TTs and 61% for mobile TTs. This is a proof of concept for the usefulness of these predictive models in assisting NOC personnel in their daily fault-handling processes.

B. Results of Dispatch-Need Prediction at T_0

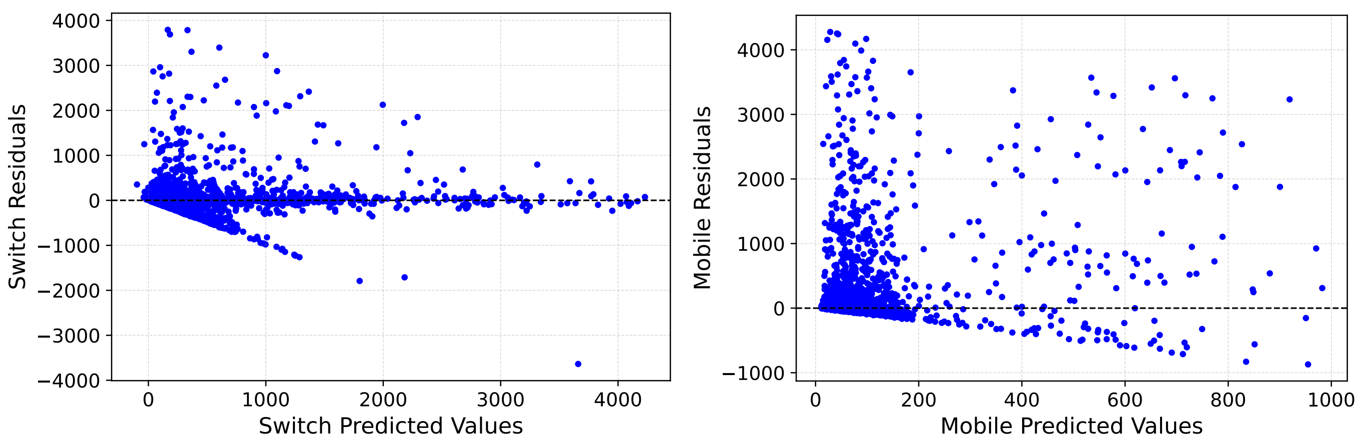
Table II and III show the results of whether dispatch is needed or not on switch and mobile TT test sets at T_0. We report Precision and Recall of both classes (Eq. 3 and 4), along with weighted average F1 (Eq. 5 and 6) and AUC scores [55], [57]. These metrics can be computed according to:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{3}$$



(a) (b)

Fig. 5: MAE comparison for resolution time prediction of TTs, (a) Switch, (b) Mobile



(a) (b)

Fig. 6: Residual plots for resolution time prediction of TTs, (a) Switch, (b) Mobile

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{4}$$

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{5}$$

$$\text{Weighted Average F1 Score} = \frac{\sum_{c=1}^{C} n_c \times \text{F1}_c}{\sum_{c=1}^{C} n_c} \tag{6}$$

Where C is the total number of classes, n_c is the number of true instances in class c, $F1_c$ is the F1 score for class c.

The recall and precision of each class shows how well the model performs in predicting the TTs of that specific class, avoiding false prediction. It is worth mentioning that unnecessary dispatches incur costs and waste workforce time, while not predicting the correct dispatches results in customer dissatisfaction. Furthermore, we also report weighted F1 score which provides a combination of precision and recall across both classes and is an indicator of overall performance of the model. The AUC score is another metric that describes the model capacity in recognizing classes, independent of a threshold.

For switch TTs, XGB achieves the best W_F1 and AUC results of almost 88% and 80%, respectively. For mobile TTs, XGB achieves W_F1 and AUC scores of respectively, 89% and 80%. This means the models predict well across both the majority (no dispatch need) and minority (dispatch needed) classes, while showing high discriminative power across all possible classification thresholds.

For switch TTs, XGB model predicts almost 99% of the TTs that do not need dispatch (‘No’ label) with a precision of 90%. For mobile TTs, 98% of TTs that do not need dispatch are classified correctly with precision of 92%. High Recall indicates that most of the no-dispatch TTs are identified correctly. This reduces the number of false alarms, saving time and resources by avoiding unnecessary dispatches. High precision for both switch and mobile TTs indicates that most of the no-dispatch predictions are accurate, leading to fewer missed no-dispatch cases. This efficiency is crucial for maintaining operational reliability and minimizing unnecessary technician dispatches.

XGB model also predicts almost 26% of the switch TTs that need dispatch (‘Yes’ label) with a precision of 82%. For mobile TTs, almost 30% of TTs that need dispatch are correctly classified by XGB model with precision of 65%. Although, we have low recall rate for both switch and mobile TTs, high precision rate indicates that the model is efficient in predicting cases where a technician dispatch is truly needed. This efficiency directly translates to cost savings by avoiding false positives (unnecessary dispatches), which are resource-intensive. High precision ensures that the resources are optimally utilized, and technicians are dispatched only when necessary, thereby improving operational efficiency.

The low recall rates of all models accross need-dispatch class are due to imbalanced data set, in which, from every 10 generated TTs almost only one TT need dispatch. This imbalance makes the models heavily biased toward not-need-dispatch TTs (see Fig. 4c and 4d. We apply oversampling techniques such as Synthetic Minority Oversampling Technique (SMOTE) [58] to tackle this problem; however, using balancing techniques, we observe the trade-off between precision and recall metrics rates, and the result could not be improved. Handling the imbalanced nature of the problem could be left as a reference point for future works.

It is worth noting that although LOGR has the highest recall score on no-dispatch-need TTs, it does not perform well in predicting dispatch-need TTs, classifying all samples as not needing dispatch. DT also has the highest recall rate on dispatch-need TTs, however; it has lowest precision on dispatch need samples among all models.

This approach brings considerable value from an industrial point of view. The NOC routine for handling TTs involves administrators addressing TTs as soon as they are created, often engaging in extensive manual handling and monitoring to determine the network issue and whether dispatch is required. This process is time-consuming. Our system assists NOC in two significant ways:

1) Efficiency in non-dispatch Cases: By predicting a high percentage of TTs that do not require dispatch with high precision, the system allows NOC administrators to confidently exclude many cases from needing further manual investigation, thereby saving time and resources.
2) Timely Identification of Dispatch Needs: Although the portion of predicted dispatch-required cases is low for both mobile and switch TTs, this prediction is still valuable since it is made at the moment of TT creation when there is limited information available about the network issue, thus providing early insights.

Even with a relatively lower portion of predicted dispatch-needed cases, these early predictions are crucial as they facilitate the process and prioritize the workflow for the NOC administrators, enhancing overall operational efficiency.

C. Performance by Increasing Training Samples

We aim to improve the results to provide the most reliable information to NOC and customers. To do so, we investigate if having more data can improve the performance of our predictive models. In other words, will collecting more data over the years result in better predictive performance? To come up with an answer, we train the models on exponentially growing numbers of switch and mobile training samples and evaluate the obtained models on the unchanged sets of test set. Fig. 7 shows the MAE convergence of the models. According to this plot, the MAEs are decreasing and flattening to some points, making the changes insignificant. As a result, increasing the number of training samples will not help improve the models' performance. We need to search for other solutions to boost the predictive performance of our models.

TABLE II: Percentage of metrics on switch test set, PR: Precision, REC: Recall, W_F1: Weighted F1 score, AUC: Area Under the Curve, SOTA: State-Of-The-Art

Model	Dispatch?	PR	RCL	W_F1	AUC
DT (SOTA)	No	90.38	90.35	83.15	61.46
	Yes	32.54	**32.61**		
LOGR	No	87.53	**99.99**	81.74	65.85
	Yes	66.67	0.002		
RF	No	90.07	99.32	87.22	78.02
	Yes	83.09	23.32		
XGB	No	**90.39**	99.19	**87.72**	**80.28**
	Yes	**82.14**	26.11		
NN	No	88.85	96.32	83.59	67.10
	Yes	37.19	15.27		

TABLE III: Percentage of metrics on mobile test set, PR: Precision, REC: Recall, W_F1: Weighted F1 score, AUC: Area Under the Curve, SOTA: State-Of-The-Art

Model	Dispatch?	PR	RCL	W_F1	AUC
DT (SOTA)	No	91.19	92.01	87.75	63.53
	Yes	35.17	**35.10**		
LOGR	No	90.13	**99.40**	86.38	76.50
	Yes	**70.89**	11.84		
RF	No	**92.04**	97.78	**89.03**	79.41
	Yes	63.68	31.50		
XGB	No	91.98	97.91	89.02	**79.77**
	Yes	64.60	30.87		
NN	No	91.83	97.96	88.81	76.61
	Yes	64.06	29.39		

D. Resolution time confidence interval for customer satisfaction

Regarding resolution time prediction for the switch and mobile TTs, we devise an approach to propose approximate resolution time ranges within which the TTs will be resolved with a high probability. In other words, to understand the predictive performance of our best regression model (XGB), we investigate what percentages of the test sets are underestimated and overestimated by the models and if marginalizing the predicted values increases the chance of a more reliable prediction.

Marginalizing the predicted time refers to increasing it with different values. This can be done in two ways: 1. Adding a

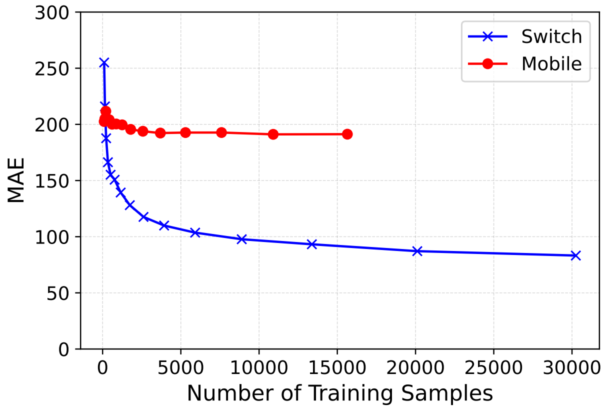


Fig. 7: MAE changes on the switch and mobile test sets as training sets are growing exponentially

fixed value (ex. 25 minutes, 50 minutes, etc.) to the predicted time, or 2. increasing the predicted time by a percentage (ex. 25%, 50%, etc.). Fig. 8a shows the percentages of switch and mobile TT test samples resolved within their marginalized predicted time. Based on this plot, there is an insignificant difference between resolved percentages of fixed and relative margins for the switch TTs. However, more test samples are resolved for mobile TTs within the predicted time shifted by fixed margins. According to Fig. 8b, the MAE changes for mobile TT test samples are almost the same in both fixed and relative marginalization cases; however, for switch TT test samples, the changes in MAE resulted by relatively marginalizing the predicted values are much more severe.

Based on Fig. 8, we choose 60 minutes as the confidence interval among different fixed margin values. That is because of two reasons: 1. increasing the predicted times by 60 minutes results in the resolution of almost 90% of switch TT test samples and 80% of mobile TT test samples within the estimated ranges (Fig. 8a), 2. It also causes the lowest possible increase in MAEs of XGB predictors (Fig. 8b). This means reporting a 60-minute marginalized predicted resolution time to the users at the time of TT occurrence will guarantee QoE. That is because, with a probability of 90% for switch TTs and 80% for mobile TTs, the network fault will be resolved within the estimated ranges.

This analysis provides significant advantages over the company's current baseline of reporting an 8-hour resolution time for all TTs due to the following reasons:

- Enhance Efficiency: By predicting closest best time ranges for TT resolutions, NOC operators can prioritize their work more effectively, focusing on TTs that are likely to require manual intervention.
- Automate Decision-Making: This analysis provides automated and customized time estimates at the time of TT creation, reducing the need for extensive manual investigation.
- Improve Customer Satisfaction: By providing more accurate time ranges, we can manage customer expectations better and ensure that they are informed of the most probable resolution times.

E. Results at Different Timestamps

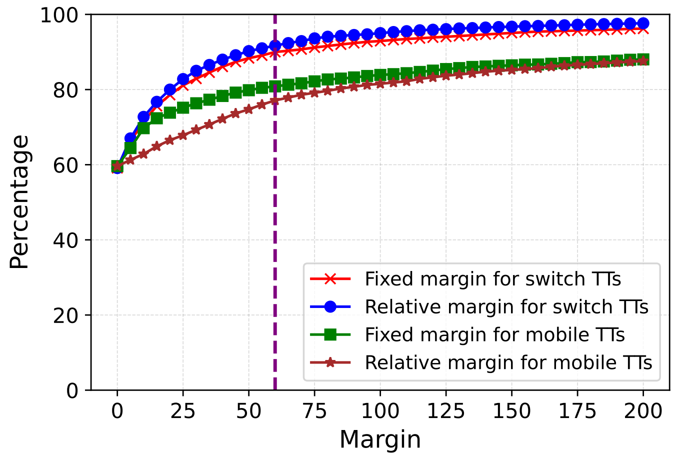
As mentioned before, the values of some features are updated over time in switch TTs. We would like to understand if this update will result in a better prediction. Thus, we build the models every 15 minutes on the updated set of features. As time passes, more TTs are resolved, specifically the ones that are resolved automatically. Fig. 9 (a) shows the over-time changes in the number of unresolved TTs and MAE of the XGB model for the prediction of resolution time. According to this figure, we observe a drop in MAE value at T_1. We also have a low portion of test samples resolved at this time (almost 2%). This means because of adding more information to the system within 15 minutes after TT creation, we get a 57% improvement in resolution time prediction with the least possible data loss. Additionally, the increase in MAE for the XGB model after 15 minutes can be attributed to the fact that TTs with shorter resolution times are progressively resolved over time, leaving behind those with longer resolution times. These remaining tickets are more challenging for the models to predict accurately, which results in the observed increase in error.

Fig. 9 (b) shows the over-time changes in the number of unresolved TTs and macro average F1 score for dispatch-need prediction of switch TTs. In this case, at T_1, we have the maximum value of the macro F1 metric. This indicates that only after 15 minutes of TT creation, we can predict 95% of TTs that need dispatch and almost 100% of TTs that do not need dispatch with significantly the lowest rate of FP and FN rates.

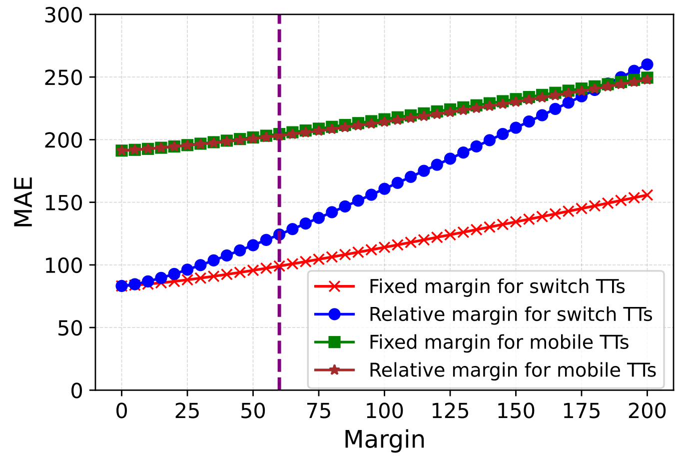
V. Conclusion and Future Work

In this work, we presented a proof of concept for an assistant automated system in the Telco network fault management domain. Our approach was a pragmatic step towards realizing autonomous networks, benefiting Telcos' customer services delivered through fixed and mobile access networks. We used thousands of historical TTs coming from fixed and mobile network domains. Mainly, we investigated the impact of different TTs' features on two target variables (resolution time and dispatch-need), considering their life-cycle evolution. For the first use case, we came up with a 1-hour confidence interval and succeeded in predicting the correct resolution time ranges in 90% and 80% of the cases at ticket creation time for, respectively, switch and mobile TTs. In terms of performance, we outperformed the best methods in the literature and had an 80% and 61% improvement over the company baseline. On prediction of the required dispatching workforce, with weighted F1 scores of respectively, 88% and 89% for switch and mobile TTs, our model outperformed the best method in the state of the art. With these scores, our model is capable of allocating resources automatically, enhancing customer satisfaction.

We studied the evolution of the switch TTs over time as more information was added to them. We realized that adding more data to TTs within 15 minutes of their creation can improve their resolution time and dispatch-need prediction by respectively, 57% and 50%. In the resolution time prediction

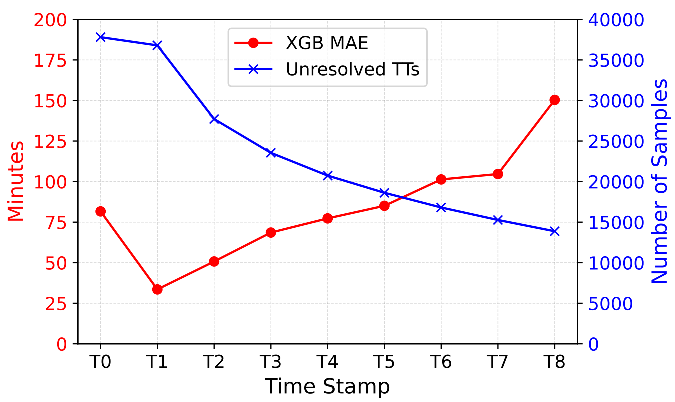


(a)

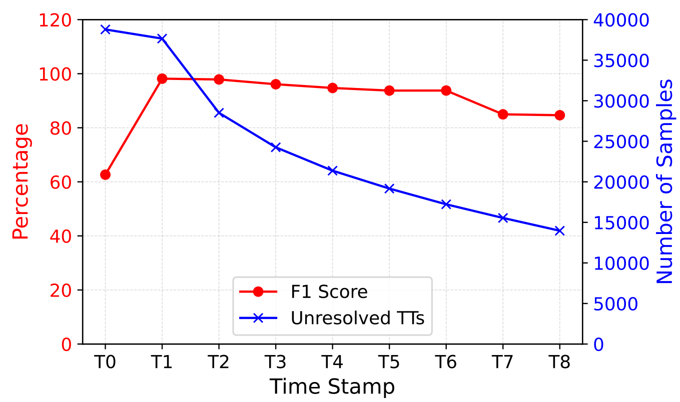


(b)

Fig. 8: (a) Percentage of covered switch and mobile TTs' resolution time by adding different values of relative and fixed margins to the prediction, (b) MAE changes after adding margins to the predictions



(a)



(b)

Fig. 9: (a) Changes of best model MAE and number of unresolved switch TTs over time for resolution time prediction (b) Changes of best model average macro F1 score and number of unresolved switch TTs over time for dispatch-need prediction (The time difference between T_n and T_{n+1} is 15 minutes

case, we devised a solution to increase customer satisfaction by choosing a confidence interval. To address future works, we consider studying the root causes of network faults based on performance data and the impacts on target variables. A thorough research can also be performed to address the imbalanced nature of the data. We can develop approaches for correlating performance data with TTs and draw insights for more autonomous network management.

Acknowledgment

This project has received funding from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No 861165. This work is also partially a result of the CELTIC-NEXT project A5GARD (project ID C2019/2-8)

References

[1] P. J. Denning, "The science of computing: The arpanet after twenty years," *American Scientist*, vol. 77, no. 6, pp. 530–534, 1989.

[2] A. Arlegui, G. Page, and J. J. Fonseca, "How automation is driving network engineer skills transformation," *CISCO White Paper*, 2022.

[3] Semantic 2020. [Online]. Available: https://www.semantic2020.eu/

[4] M. Liyanage, Q.-V. Pham, K. Dev, S. Bhattacharya, P. K. R. Maddikunta, T. R. Gadekallu, and G. Yenduri, "A survey on zero touch network and service management (zsm) for 5g and beyond networks," *Journal of Network and Computer Applications*, vol. 203, p. 103362, 2022.

[5] J. Rexford, "Evolving toward a self-managing network."

[6] W. Sun and A. Isac, "How to automatically resolve trouble tickets with machine learning," 2020. [Online]. Available: https://www.ericsson.com/

[7] M. Nouioua, P. Fournier-Viger, G. He, F. Nouioua, and Z. Min, "A survey of machine learning for network fault management," *Machine Learning and Data Mining for Emerging Trend in Cyber Dynamics: Theories and Applications*, pp. 1–27, 2021.

[8] S. A. Mohammed, A. R. Mohammed, D. Côté, and S. Shirmohammadi, "A machine-learning-based action recommender for network operation centers," *IEEE Transactions on Network and Service Management*, vol. 18, no. 3, pp. 2702–2713, 2021.

[9] K. Murphy, A. Lavignotte, and C. Lepers, "Fault prediction for heterogeneous telecommunication networks using machine learning: a survey," *IEEE Transactions on Network and Service Management*, 2023.

[10] D. Mulvey, C. H. Foh, M. A. Imran, and R. Tafazolli, "Cell fault management using machine learning techniques," *IEEE access*, vol. 7, pp. 124 514–124 539, 2019.

[11] A. K. Arhouma and S. M. Amaitik, "Decision support system for alarm correlation in gsm networks based on artificial neural networks," in *Conference Papers in Science*, vol. 2013, no. 1. Wiley Online Library, 2013, p. 958926.

[12] A. Bouillard, A. Junier, and B. Ronot, "Alarms correlation in telecommunication networks," Ph.D. dissertation, INRIA, 2013.

[13] R. M. Khanafer, B. Solana, J. Triola, R. Barco, L. Moltsen, Z. Altman, and P. Lazaro, "Automated diagnosis for umts networks using bayesian network approach," *IEEE Transactions on vehicular technology*, vol. 57, no. 4, pp. 2451–2461, 2008.

[14] T. Reidemeister, M. A. Munawar, M. Jiang, and P. A. Ward, "Diagnosis of recurrent faults using log files," in *Proceedings of the 2009 Conference of the Center for Advanced Studies on Collaborative Research*, 2009, pp. 12–23.

[15] H. Wietgrefe, "Investigation and practical assessment of alarm correlation methods for the use in gsm access networks," in *NOMS 2002. IEEE/IFIP Network Operations and Management Symposium.'Management Solutions for the New Communications World'(Cat. No. 02CH37327)*. IEEE, 2002, pp. 391–403.

[16] S. Kobayashi, K. Otomo, K. Fukuda, and H. Esaki, "Mining causality of network events in log data," *IEEE Transactions on Network and Service Management*, vol. 15, no. 1, pp. 53–67, 2017.

[17] C.-C. Chen, H.-H. Shuai, and M.-S. Chen, "Distributed and scalable sequential pattern mining through stream processing," *Knowledge and Information Systems*, vol. 53, pp. 365–390, 2017.

[18] E. Kiciman and A. Fox, "Detecting application-level failures in component-based internet services," *IEEE transactions on neural networks*, vol. 16, no. 5, pp. 1027–1041, 2005.

[19] G. A. Barreto, J. C. M. Mota, L. G. M. Souza, R. A. Frota, and L. Aguayo, "Condition monitoring of 3g cellular networks through competitive neural models," *IEEE Transactions on Neural Networks*, vol. 16, no. 5, pp. 1064–1075, 2005.

[20] S. Zidi, T. Moulahi, and B. Alaya, "Fault detection in wireless sensor networks through svm classifier," *IEEE Sensors Journal*, vol. 18, no. 1, pp. 340–347, 2017.

[21] A. Johnsson and C. Meirosu, "Towards automatic network fault localization in real time using probabilistic inference," in *2013 IFIP/IEEE International Symposium on Integrated Network Management (IM 2013)*. IEEE, 2013, pp. 1393–1398.

[22] A. Dridi, C. Boucetta, S. E. Hammami, H. Afifi, and H. Moungla, "Stad: Spatio-temporal anomaly detection mechanism for mobile network management," *IEEE Transactions on Network and Service Management*, vol. 18, no. 1, pp. 894–906, 2020.

[23] H. Mannila, H. Toivonen, and A. Inkeri Verkamo, "Discovery of frequent episodes in event sequences," *Data mining and knowledge discovery*, vol. 1, pp. 259–289, 1997.

[24] Z. Wang, M. Zhang, D. Wang, C. Song, M. Liu, J. Li, L. Lou, and Z. Liu, "Failure prediction using machine learning and time series in optical network," *Optics Express*, vol. 25, no. 16, pp. 18 553–18 565, 2017.

[25] C. Yuan, N. Lao, J.-R. Wen, J. Li, Z. Zhang, Y.-M. Wang, and W.-Y. Ma, "Automated known problem diagnosis with event traces," *ACM SIGOPS Operating Systems Review*, vol. 40, no. 4, pp. 375–388, 2006.

[26] A. R. Mohammed, S. A. Mohammed, D. Côté, and S. Shirmohammadi, "Stateless are: Action recommendation engine without network state measurement," in *2022 IEEE International Symposium on Measurements & Networking (M&N)*. IEEE, 2022, pp. 1–6.

[27] B. Altamimi, D. Côté, S. Shirmohammadi *et al.*, "Toward a super-intelligent action recommender for network operation centers using reinforcement learning," *IEEE Access*, vol. 11, pp. 20 216–20 229, 2023.

[28] K. Guo, J. Chen, P. Dong, S. Liu, and D. Gao, "Fullsight: A feasible intelligent and collaborative framework for service function chains failure detection," *IEEE Transactions on Network and Service Management*, vol. 19, no. 4, pp. 4546–4565, 2022.

[29] M. F. Silva, A. Pacini, A. Sgambelluri, and L. Valcarenghi, "Learning long-and short-term temporal patterns for ml-driven fault management in optical communication networks," *IEEE Transactions on Network and Service Management*, vol. 19, no. 3, pp. 2195–2206, 2022.

[30] A. Medem, M.-I. Akodjenou, and R. Teixeira, "Troubleminer: Mining network trouble tickets," in *2009 IFIP/IEEE International Symposium on Integrated Network Management-Workshops*. IEEE, 2009, pp. 113–119.

[31] S. Symonenko, S. Rowe, and E. D. Liddy, "Illuminating trouble tickets with sublanguage theory," in *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, 2006, pp. 169–172.

[32] R. Potharaju, N. Jain, and C. Nita-Rotaru, "Juggling the jigsaw: Towards automated problem inference from network trouble tickets," in *10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13)*, 2013, pp. 127–141.

[33] E. D. Liddy, S. Symonenko, and S. Rowe, "Sublanguage analysis applied to trouble tickets." in *FLAIRS Conference*, 2006, pp. 752–757.

[34] A. F. Fahmy, A. H. Yousef, and H. K. Mohamed, "The application of data mining for the trouble ticket prediction in telecom operators," in *2017 12th International Conference on Computer Engineering and Systems (ICCES)*. IEEE, 2017, pp. 227–232.

[35] F. C. Yayah, K. I. Ghauth, and C.-Y. Ting, "The performance of classification method in telco customer trouble ticket dataset." *IAENG International Journal of Computer Science*, vol. 49, no. 2, 2022.

[36] J. S. Tan, C. K. Ho, A. H. L. Lim, and M. Ramly, "Predicting network faults using random forest and c5. 0," *Int J Eng Technol*, vol. 7, no. 2.14, pp. 93–96, 2018.

[37] M. W. Asres, M. A. Mengistu, P. Castrogiovanni, L. Bottaccioli, E. Macii, E. Patti, and A. Acquaviva, "Supporting telecommunication alarm management system with trouble ticket prediction," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 2, pp. 1459–1469, 2020.

[38] Y. Temprado, F. J. Molinero, C. Garcia, and J. Gomez, "Knowledge discovery from trouble ticketing reports in a large telecommunication company," in *2008 International Conference on Computational Intelligence for Modelling Control & Automation*. IEEE, 2008, pp. 37–42.

[39] J. Gómez, Y. Temprado, M. Gallardo, C. García, and F. J. Molinero, "Application of neural network to predict adverse situations in trouble ticketing reports," *INTELLIGENT SYSTEMS AND AGENTS 2009*, p. 204, 2009.

[40] K. R. Sample, A. C. Lin, B. J. Borghetti, and G. L. Peterson, "Predicting trouble ticket resolution," in *The Thirty-First International Flairs Conference*, 2018.

[41] S. C. Haw, K. Ong, L. J. Chew, K. W. Ng, P. Naveen, and E. A. Anaam, "Improving the prediction resolution time for customer support ticket system," *Journal of System and Management Sciences*, vol. 12, no. 6, pp. 1–16, 2022.

[42] A. Arif and Z. Wang, "Distribution network outage data analysis and repair time prediction using deep learning," in *2018 IEEE International Conference on Probabilistic Methods Applied to Power Systems (PMAPS)*. IEEE, 2018, pp. 1–6.

[43] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on knowledge and data engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

[44] A. Björling, "Telecommunications trouble ticket resolution time modelling with machine learning," 2021.

[45] R. Colella, "Trouble tickets resolution time estimation: The design of a solution for a real case scenario," 2021.

[46] N. L. Crookston and A. O. Finley, "yaimpute: an r package for knn imputation," *Journal of Statistical Software*, vol. 23, pp. 1–16, 2008.

[47] D. A. Freedman, *Statistical models: theory and practice*. cambridge university press, 2009.

[48] L. E. Peterson, "K-nearest neighbor," *Scholarpedia*, vol. 4, no. 2, p. 1883, 2009.

[49] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and regression trees*. Routledge, 2017.

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

[51] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794. [Online]. Available: https://xgboost.readthedocs.io/

[52] J. Heaton, "Ian goodfellow, yoshua bengio, and aaron courville: Deep learning: The mit press, 2016, 800 pp, isbn: 0262035618," *Genetic programming and evolvable machines*, vol. 19, no. 1-2, pp. 305–307, 2018.

[53] G. James, D. Witten, T. Hastie, R. Tibshirani *et al.*, *An introduction to statistical learning*. Springer, 2013, vol. 112.

[54] D. J. MacKay, "Bayesian interpolation," *Neural computation*, vol. 4, no. 3, pp. 415–447, 1992.

[55] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning*. Springer, 2006, vol. 4, no. 4.

[56] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011. [Online]. Available: https://scikit-learn.org/

[57] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman, *The elements of statistical learning: data mining, inference, and prediction*. Springer, 2009, vol. 2.

[58] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

APPENDIX

TABLE A1: Switch TTs Features

Feature Name	Description	Data type
CREATED_YEAR	TT created year	2019, 2020
CREATED_DAY_NR	TT created day	21, 22, 23
CREATED_DAY_NAME	TT created day name	Monday, Tuesday
CREATED_MONTH	TT created month number	January, February
CREATED_HOUR	TT created hour	7, 8, 13
CHILD	Number of TT children	1, 2, 4
MOTHER	TT having any child	0, 1
DISPATCH	TT needing technician dispatch	0, 1
POSTCODE	Switch location postcode	11111, 22222
LATITUDE	Latitude of switch location	11.123
LONGITUDE	Longitude of switch location	11.123
LOCATION	Switch city	STOCKHOLM
SITE_ID	Switch site ID	222222, 333333
DHCP_CLIENTS	Number of effected DHCP clients	120, 1232
BROADBAND	Number of affected broadband connection	20, 140
VOIP_CONNECTIONS	Number of affected VOIP connection	110, 450
LAN_SME	Number of affected enterprises	10, 103
WEIGHT6_18	Weight of TT created during time 06:00 to 18:00	10, 1004
WEIGHT0_6	Weight of TT created during time 00:00 to 06:00	10, 1004
WEIGHT18_0	Weight of TT created during time 18:00 to 00:00	10, 1004
TECH	Switch technology	LAN
MODEL	Switch model	Model 1, Model 2
AUTO_CREATION	Automatic created TT	0, 1
RESOLUTION_TIME_IN_MINUTES	TT resolution time	24, 328, 1133

TABLE A2: Mobile TTs Features

Feature Name	Description	Examples
CREATED_YEAR	TT created year	2019, 2020
CREATED_DAY_NR	TT created day	1, 2, 3
CREATED_DAY_NAME	TT created day name	Monday, Tuesday
CREATED_MONTH	TT created month number	January, February
CREATED_HOUR	TT created hour	7, 8, 13
IMPACT	Number of affected enterprises on different 2G, 3G, 4G, 5G frequency bands with different priorities	1, 2, 3
NR_AFFECTED_SITE	Number of affected sites	0, 1, 2
POSTCODE	Switch location postcode	11111, 222222
LATITUDE	Latitude of switch location	11.120
LONGITUDE	Longitude of switch location	17.230
LOCATION	Base station city	STOCKHOLM
SITE_ID	Base station site ID	ID 1, ID 2
AFFECTED_NODE	Affected node ID	NODE 1, NODE 2
WEIGHT6_18	Weight of TT created during time 06:00 to 18:00	10, 1004
WEIGHT0_6	Weight of TT created during time 00:00 to 06:00	10, 1004
WEIGHT18_0	Weight of TT created during time 18:00 to 00:00	10, 1004
AUTO_CREATION	Automatic created TT	0, 1
DISTRICT_ID	ID of the district	ID 1, ID 2
PLANNING_REGION	ID of region planned for maintenance	Region 1, Region 2
PROVIDER	Hardware provider	NOKIA
HARDWARE	Broken part of the hardware	Rectifier, chasi
ALARM	Type of the alarm caused TT creation	Threshold exceeds limit
DISPATCH	TT needing technician dispatch	0,1
RESOLUTION_TIME_IN_MINUTES	TT resolution time	24, 328, 1133