Addis Ababa University

Addis Ababa Institute of Technology

School of Electrical and Computer Engineering

Telecommunication Engineering Graduate Program

Root Cause Analysis of Unsatisfactory User Throughput in UMTS for Bahir Dar City

By:

Yonas Alemayehu Matebie

Advisor:

Dr. -Eng Yihenew Wondie

A Thesis Submitted to the School of Graduate Studies of Addis Ababa University in Partial Fulfillment of the Requirements for the Degree of Masters of Science in Telecommunication Engineering

October 2021

Addis Ababa, Ethiopia

Addis Ababa Institute of Technology

School of Electrical and Computer Engineering

Telecommunication Engineering Graduate Program

Root Cause Analysis of Unsatisfactory User Throughput in UMTS for Bahir Dar City

By:

Yonas Alemayehu Matebie

Approval by Board of Examiners

(Chairman, School Graduate Committee)	Signature
Dr. -Eng Yihenew Wondie **Advisor**	Signature
Examiner 1	Signature
Examiner 2	Signature

Declaration

I, the undersigned, declare that this thesis is my original work, has not been presented for a degree in this or any other university, and all sources of materials used for the thesis have been fully acknowledged.

Yonas Alemayehu Matebie _____

Name Signature

Place: Addis Ababa, Ethiopia

Date of Submission: _____

This thesis has been submitted for examination with my approval as a university advisor.

Dr. -Eng Yihenew Wondie _____

Advisor Signature

Abstract

Nowadays, the increase in size and complexity of current cellular networks is complicating their operation, maintenance and optimization tasks. The user throughput has increased dramatically. However, such networks have become more susceptible to failure. Unfortunately, Bahir Dar city's mobile data is mainly supported by Universal Mobile Telecommunications System (UMTS). It is challenging to locate the root causes of unsatisfactory user throughput manually for such a system, as it requires expertise and a longtime analysis. This thesis proposes low user throughput Root Cause Analysis (RCA) using a Deep Neural Network (DNN) based on a multilayer perceptron. Local interpretable model-agnostic explanation technique has been utilized to enable DNN for RCA of low user throughput by providing feature importance. Furthermore, the impact of features on the model outcome is studied to understand the causal effect on the user throughput using a partial dependency plot.

The effectiveness of the model has been validated using a test dataset and compared with reference models. Accordingly, the proposed DNN model has performed better compared to these models. The result of the proposed model performance evaluation is an accuracy of 87%, precision of 88%, recall of 86%, F1-score of 87% and area under the curve of 95%. RCA conducted on 10 selected cells with a poor throughput of 50 samples per cell has shown the root cause is due to poor channel quality reported by users. It is recommended to optimize the network to improve the channel quality.

Keywords — RF condition, causal effect, partial dependency, feature importance, poor CQI

Acknowledgment

My first and foremost thanks go to my almighty God; without him, nothing would have been possible. He was on my way to be my strength, courage, and more of his blessing for my thesis work accomplishment. Then, I want to express my gratitude to my advisor, Dr. -Eng Yihenew Wondie. His knowledge, experience, and valuable suggestions and corrections gave me the effective support to shape and complete my research work. Special thanks to Dr. -Ing. Dereje Hailemariam and Dr. Beneyam Berehanu, who were my evaluators and gave me valuable suggestions to include in this thesis throughout my progress reports. I am also grateful to thank Mr. Gizachew Addis, Mr. Samuel Assefa Abebe, and Mr. Dawit Kebede Wassie for their great collaboration and encouragement with endless support from Ethio Telecom. Moreover, I would like to thank Ethio Telecom for the sponsorship of this thesis research with the collaboration of AAiT.

Finally, I would like to thank my deepest appreciation to my family: Marshet Alelign Demisie (my wife), Mieraf Yonas (my little daughter), and Bereket Yonas (my beloved son) for all the love and patience to complete my research.

Table of Contents

List of Figures

List of Tables

List of Acronyms

2G	Second Generation
3G	Third Generation
4G	Fourth Generation
5G	Fifth Generation
Adam	Adaptive Moment Estimation
AI	Artificial Intelligence
ANNs	Artificial Neural Networks
AS	Access Stratum
AUC	Area Under the Curve
AuC	Authentication Center
BCE	Binary Cross-entropy
BiT	Bahir Dar Institute of Technology
BLER	Block Error Rate
CC	Call Control
CCCH	Common Control Channel
CE	Channel Element
Cell_ID	Cell Identification
CN	Core Network
CQI	Channel Quality Indicator
CS	Circuit Switched
DNN	Deep Neural Network
EcNo	Chip Energy per Power Density
EIR	Equipment Identity Register
FN	False Negative
FP	False Positive
FPR	False Positive Rate
GGSN	Gateway GPRS Support Node
GMSC	Gateway MSC

GPRS	General Packet Radio Service
GSM	Global System for Mobile Communication
HLR	Home Location Register
HSDPA	High-speed Downlink Packet Access
HS-DSCH	High Speed Downlink Shared Channel
HSPA	High-speed Packet Access
HS-SCCH	High Speed Shared Control Channel
IMS	IP Multimedia Subsystem
IP	Internet Protocol
IQR	Inter Quartile Range
Kbps	Kilobits per Second
KPI	Key Performance Indicator
LIME	Local Interpretable Model-agnostic Explanations
LR	Logistic Regression
LTE	Long-Term Evolution
MAC	Medium Access Control
MAE	Mean Absolute Error
MAPE	Mean Absolute Percentage Error
Mbps	Megabits per Second
ML	Machine Learning
MLP	Multilayer Perceptron
MM	Mobility Management
MSC	Mobile Switching Center
MSE	Mean Squared Error
MSLE	Mean Square Logarithmic Error
NAS	Non-access Stratum
NSS	Network Subsystem
NMS	Network Management System
OSS	Operations Support System
PDC	Packet Data Convergence
PDP	Partial Dependence Plot

PRS	Performance Report System
PS	Packet Switched
PSTN	Public Switched Telephone Network
QoE	Quality of Experience
QoS	Quality of Service
RAN	Radio Access Network
RCA	Root Cause analysis
ReLu	Rectified Linear Unit
RLC	Radio Link Control
RNC	Radio Network Controller
ROC	Receiver Operating Characteristic
RRC	Radio Resource Control
RSCP	Received Signal Code Power
SGD	Stochastic Gradient Descent
SGSN	Serving GPRS Support Node
SINR	Signal to Interference and Noise Ratio
SM	Session Management
TCP	Transmission Control Protocol
TDD	Time Division Duplex
TD-SCDMA	Time Division Synchronous Code Division Multiple Access
TGSH	Tibebe Gion Specialized Hospital
TN	True Negative
TP	True Positive
TPR	True Positive Rate
UDP	User Datagram Protocol
UE	User Equipment
UMTS	Universal Mobile Telecommunication System
UTRAN	UMTS Terrestrial Radio Access Network
VLR	Visitor Location Register
WCDMA	Wideband Code Division Multiple Access

1. Introduction

In Ethiopia, mobile data service usage has shown considerable growth after different network expansion projects were carried out in the last years. With the advent of smartphones, more and more users are using data-intensive applications, such as WhatsApp, Instagram, Facebook, and the like. In essence, this has resulted in the evolution of new mobile data services and complexity in the management of the networks [1]. In Bahir Dar city, the UMTS network is the primary provider of mobile data services. The quality of experience (QoE), which is the measurement of the user's perception of the quality of service (QoS) supplied, is one of the most important aspects of mobile network management. A key metric used to assess service quality is throughput, measured in megabits per second (Mbps) or kbps. It's defined as the ratio of total correctly transferred bits from a source to a destination divided by the total time spent [2]. Due to the numerous contributing elements, determining low user throughput is a difficult operation. User throughput degradation can be caused by either network performance or user issues [3]–[7].

Even if High-speed Packet Access Plus (HSPA+) is enabled in the city to support good user throughput, investigating the main causes of poor user throughput during operation and maintenance is challenging. RCA is the main objective of this thesis for identifying or localizing the root cause of the low user throughput. From [8], RCA is defined as "*a systematic approach used to analyze the fundamental problems before trying to solve them*." According to [9], RCA has four important steps to go through. These are:

- Problem definition: formulate a process in the extent of "What is happened?".
- Data collection: gathering information relating to the problem defined, based on facts and data.
- Casual factors identification: a causal factor [10], [11] is any key unexpected, accidental contributor to an event (a negative event or undesirable state), that if removed would have either prevented the occurrence of it or reduced its severity or frequency.
- Solution implementation: This phase should address how to prevent the problem from happening again. Mostly this may include diagnosis strategy or recommendation.

1.1 Statement of the Problem

As introduced, several variables could contribute to poor user throughput. Ethio telecom monitors the performance of network elements and average user throughput using a performance report system. The measurement reports on user throughput include radio conditions, required resources (available code and power), and channel quality (CQI). There is research conducted to evaluate the QoE in the company that concludes UMTS data is unsatisfactory in download throughput [12]. This research was conducted using different analysis approaches. One of them is using a network management system. In this research, the root cause for throughput degradation is unanswered. Moreover, even if there is evidence of a poor signal from drive test and low user throughput in Bahir Dar city, the approach of determining the root cause is traditional. Additionally, the analysis is carried out in recurring activities that take a long time and exhausting daily. As a result, there is a research gap to study RCA for low user throughput.

The advantage of using measurement reports is that the information extracted from rich statistical data can be used to study the possible causes of unsatisfactory user throughput. Finding solutions for such multidimensional and large datasets is a challenging task. In such a case, simple linear transformation algorithms are not suitable. While decades of research have yielded a large number of algorithms and techniques for performing RCA in a variety of fields [10], usability and adaptability to the growing complexity of throughput analysis remain a limitation, where scalability and actual interaction become critical. Advanced root-cause analysis capable of precisely approximating for decision-making will be required because of the complex and dynamic behavior of user throughput interactions with input parameters (multivariate). A deep neural network [13]–[19], nowadays, is a good approximation for nonlinear transformation. As a result, RCA for low user throughput based on a deep neural network with LIME explanation is introduced for this thesis. RCA is all about causality and explanation [20], [21].

1.2 Objective

1.2.1 General Objective

The main objective of this thesis work is to study RCA in UMTS of unsatisfactory user throughput using NMS data and DNN with LIME techniques in the context of Bahir Dar city.

1.2.2 Specific Objectives

The specific objectives of this work include:

- Overview of UMTS technology.
- Literature review of important articles of RCA techniques.
- Collect, prepare and preprocess data used for RCA using python libraries.
- Discuss deep neural network (DNN), develop and evaluate an RCA model.
- Understand the causal effect of input parameters on user throughput.
- Find the root cause of low user throughput (case analysis) using the developed model with LIME.
- Generate reports with conclusions.

1.3 Literature Review

This literature review is presented in the context of state-of-the-art RCA approaches and their applicability to the problem at hand using DNN and LIME.

The basic concepts of RCA are causation and explanation, according to a survey [10]. Causality is the study of the cause and effect relationship between factors and response variables, whereas explanation is the justification for the response caused by those factors. It's important, especially in the case of explanation, because an explanation is often RCA's desired outcome. There are two types of RCA models, according to [10], deterministic and probabilistic. The known facts and inferences used in deterministic models are both certain. Neural network is one type of this family. Probabilistic models, such as Bayesian networks, on the other hand, are capable of dealing with uncertainty with a requirement for a threshold to find the most probable cause.

Automated systems for finding and diagnosing cells, not just in complete failures but also with decreasing performance, are becoming increasingly significant as commercial cellular networks become more sophisticated. Root cause investigation of observed anomalies is time-consuming and is mostly done manually, if at all; in most situations, operators just reset faulty cells. A framework was developed in a study [22] that could detect anomalies and locate the most likely root cause of not just serious problems in a cell but also service degradations. Detection is based on monitoring radio measurements and other performance indicators and comparing them to their regular behavior captured by profiles, which are likewise produced automatically without the need for threshold or manual calibration, according to [22]. The diagnosis was intervened in [22], depending on reports of previous fault-cases by identifying and learning their characteristic impact on different performance indicators.

A study [23] was conducted to handle enormous amounts of data and the complexity of new data services in mobile networks, comparable to [22]. Their studies are similar in that they rely on a dataset from monitored radio measurements and other performance indicators. The inquiry in [23] particularly, was to determine the root cause of poor data throughput in mobile networks, notably in UMTS. Deep learning techniques have gained popularity, according to [23] because they scale well with enormous data volumes and make efficient use of computational power to train models. Although it is advantageous to achieve high accuracy using deep neural networks (DNNs) for nonlinear approximation of complicated issues, such as poor data throughput with associated causal components, the decision-making process is an opaque and black box. As a result, the author in [23] employed local interpretable model-agnostic explanations (LIME) as a black box explainer for feature importance evaluation to enable DNN for root cause analysis.

The authors of [24] assessed the importance of features for anomaly identification and root cause investigation. Explainable machine learning algorithms are utilized, according to [24]. However, the strategy they favor is contingent on the isolation forest (i.e. model-specific). Other models can't use it because it's not scalable. The good side is that [24] also discusses model-agnostic interpretation approaches. By considering a model under investigation as a black box, these approaches can derive post-hoc explanations. Such methods, in addition to their adaptability, are meant for supervised learning. One of the strategies is LIME, which is particularly useful for local explanations.

Black box explanation is a new technique for revealing advanced models' decision-making strategies, such as DNNs. The explanation and applicability of LIME is discussed in the context of RCA in [11], [24], [25], [26], [27]. A survey for practitioners concerning individual explanations of machine learning models has been presented in [25]. Explainability approaches are thoroughly covered in this survey. There are two sorts of explainability: global and local. The entire dataset is assessed in the global sense, whereas individual observations are considered for explainability in the local sense. In addition to LIME, a partial dependence plot (PDP) is examined to enable causality learning as assessed in [11]. PDP is used to visualize the causality effect of input parameters on response variables, according to [11]. A key takeaway from [11] is model-agnostic causation for deep neural networks. In model-agnostic explanations, challenging tasks occur when the model parameters have more complex interactions, as stated in [11]. Post-hoc interpretability strategies are presented to use what a trained model has learned without modifying the underlying model to enhance explainability. Casual feature learning is utilized in post-hoc interpretation approaches to determine the subset of attributes that contribute causally to the models' outcomes. The attribution of weight is the most essential contribution from LIME.

An application of LIME on the root cause inference aspects of service assurance is examined in [27], to provide QoE problems with localization for service providers. According to the author of [27], LIME is used to generate rule sets for decision-making during root cause identification. The key advantage of the LIME approach, according to the author, is its short computing time, which allows it to be utilized for enormous datasets. As in [26], works on interpretation strategies for future interpretable machine learning are summarized in addition to the papers already evaluated. Local interpretable approaches, such as LIME, exhibit great fidelity and accuracy, according to [26].

This thesis will follow the methodologies in [23] because of the applicability of the solutions presented and the complexity of the problem at hand for RCA of low user throughput. The key difference is that, in this thesis work, the causal factors are limited to measurement reports of cell level and user equipment (UE), not including the core network. The objective is to narrow the scope and gain a better understanding of cell performance issues to user experience in the context of the average user throughput.

1.4 Methodology

In this thesis work, related research papers have been reviewed to see possible solutions for low user throughput root cause analysis. The purpose of the review is to see the background knowledge that is the foundation of current findings. After the literature review, state-of-the-art solutions are set for the identified problems and solutions. Appropriate tools and algorithms are to be selected. The OSS-based KPI data from PRS is considered for five months duration. The selected ways of data collection have helped to find the possible causes of low throughput by investigating correlation analysis coefficients. Feature selection based on the correlation results will be done. Data preprocessing, causal DNN model development, and final model evaluation will be done, afterward. Finally, case analysis, on selected cells will be investigated.

1.5 Scope and Limitations

1.5.1 Scope

The scope is to investigate the root cause of low user throughput in UMTS data service on selected cells targeting specific areas in Bahir Dar city. It will cover the RCA of link-level low user throughput from PRS on a per cell per hour basis.

1.5.2 Limitations

The following are the main limitations in conducting this research.

- The dataset used is aggregated values of parameters from all users on an hourly basis, and thus it may have missing information per instantaneous user throughput.
- User throughput is not only affected by RF condition, the number of users, mean CQI value, and the number of available HS-PDSCH codes, but also by other factors, like latency, licenses of resources (like CE), equipment alarms, parameter settings, UE capability, and so forth. Additionally, information may be missed during feature selection since parameters highly correlated to user throughput are selected i.e. a less-correlated feature may be a root cause.
- This thesis did not consider input parameter thresholds for decision-making. The internal DNN structure captures the interactions and makes decisions to simplify expertise efforts for threshold definition. The expert knowledge is limited to experience

and assumptions. On the other hand, the model is limited to the data at hand and the generalization is dependent on that data [28].

- The LIME-based explanation is focused on a single occurrence (per sample basis). The cost of determining the root cause of large data volume is expensive. Even though it is well known for local explanations, it lacks global decision-making.
- Previous work on Ethio Telecom's UMTS network was more towards the quality of service or QoE evaluation. As a result, there is a lack of similar papers that are directly related to RCA for low user throughput.

1.6 Contributions

The main contributions of this thesis work are:

- DNN classifier model that can classify the throughput based on selected input parameters in UMTS.
- Understanding of the impact of input parameters in depth that possibly influences or affects the user throughput to better support mitigation directions. Moreover, an explanation for root cause per instance of low user-throughput using LIME.
- The LIME does not only justify the root cause but also builds trust by the user about the causal DNN model developed for RCA.
- Better support UMTS data service maintenance and operation as well as RAN optimization in the company by using this proposed RCA approach.

1.7 Thesis Organization

The organization of this thesis work is directly related to the specific objectives presented. In Chapter 2, a detailed description of the UMTS network is presented. In Chapter 3, fundamentals of deep neural network and LIME (local interpretable model-agnostic explanation) are reviewed. In Chapter 4, the proposed RCA system model including an understanding of the dataset used for the research and procedures followed are presented. In Chapter 5, the results and discussions of RCA have been presented. In Chapter 6, the conclusion and recommendation are discussed.

2. Fundamentals of UMTS Network System

In this Chapter, the fundamental concepts of the Universal Mobile Telecommunication System (UMTS) network and its system architecture are explained. The Chapter discusses UMTS system elements, services that are delivered by the UMTS network, specifically, on data services. Finally, user throughput calculation, in general, is presented.

2.1 UMTS Network Architecture

The first UMTS network was developed as a universal infrastructure that can carry services in the past generations as well as services from future evolution. This was achieved by dividing the overall network into different technologies such as access, transport, service, and user applications. Due to such subdivisions of the network technologies in the system, the architecture of the UMTS network can be modeled in many ways. As shown in Figure 2.1, the more general architecture mainly has two parts: Core Network (CN) and UMTS Terrestrial Radio Access Network (UTRAN). Sometimes, User Equipment (UE) is also included in the architecture as a complement to the system.

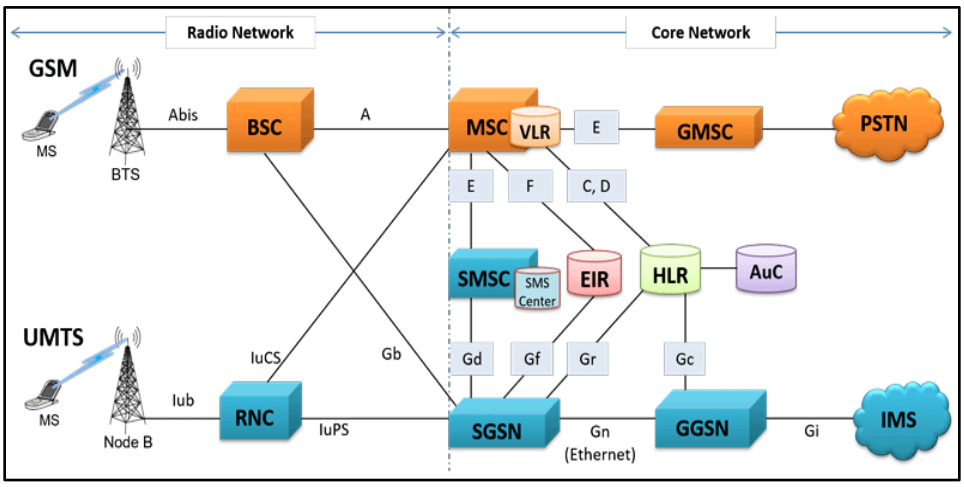


Figure 2. 1 UMTS network architecture [29].

2.1.1 UMTS Core Network

The basic functions of the core network are switching and routing calls and data connections to and from external networks. It is divided into two functional modules, which are circuit-switched (CS) and packet-switched (PS) domains. Network elements such as Mobile Switching Center (MSC), Gateway MSC (GMSC), and Visitor Location Register (VLR) are in the CS domain whereas Serving GPRS Support Node (SGSN) and Gateway GPRS Support Node (GGSN) belong to the PS domain. Equipment Identity Register (EIR), Authentication Center (AuC), and Home Location Register (HLR) are common entities that belong to both domains. The core network uses GMSC as an attachment point of the CS part to connect Public Switched Telephone Network (PSTN) and GGSN to connect its PS part to IP Multimedia Subsystem (IMS).

2.1.2 UMTS Terrestrial Radio Access Network

The UMTS Terrestrial Radio Access Network (UTRAN) is the network part that facilitates the communication between UE and other network entities over the radio. It has two parts: Radio Network Controller (RNC) and Node B.

1. RNC: some of the responsibilities of this entity are radio resource management, a part of mobility management, data encryption/decryption, and managing NodeBs that are connected to it. This network entity makes communication with the core network, NodeB under it, and with another RNC for mobility management.
2. Node B: this is a base station in the UMTS network where the air interface starts. It has transceivers for communicating user equipment. The air interface of UMTS has evolved through different releases without making any significant change on the CN part. Almost all upgrades are made on the data traffic serving section of the air interface and a summary is presented in Table 2.1.

Table 2. 1 Releases of UTRAN air interface technologies.

Release	Technology	Maximum Uplink Speed (Mbps)	Maximum Downlink Speed (Mbps)
99	WCDMA	0.384	0.384
5	HSPA	0.384	14.4
6	HSPA	5.8	14.4
7	HSPA+	12	28
8	HSPA+	12	42
9	HSPA+	24	84
10	HSPA+	24	168
11	HSPA+	72	336

The two network entities (Node B and RNC) form radio network subsystem (NSS) and the general architecture for UTRAN looks as presented in Figure 2.2.

UTRAN interfaces:

1. Uu – This is an interface through which the UE accesses the fixed part of the system and it is usually called ‘as air interface’.
2. Iub – an interface that connects NodeB with RNC
3. Iur – this interface communicates one RNC with the other. Its main function is facilitating the soft handover between NodeB’s of different RNC.
4. Iu-CS – an interface, that is in between RNC and the CS domain of the core network.
5. Iu-PS – this is also for connecting RNC with the core network but for the PS domain.

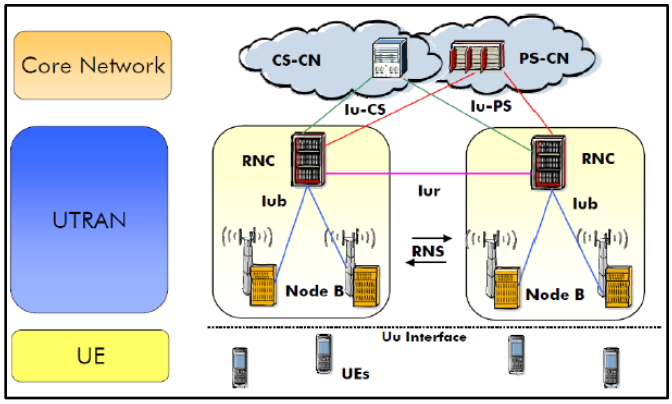


Figure 2. 2 UTRAN logical architecture [30].

2.2 UMTS Quality of Services

There are two kinds of services in Telecommunication. Teleservice is a telecommunication service provided by the network for the communication between end users whereas bearer service is a telecommunication service that is used for the transfer of user and control data between network equipment's [31]. The layered architecture for UMTS bearer service [31] is given in Figure 2.3.

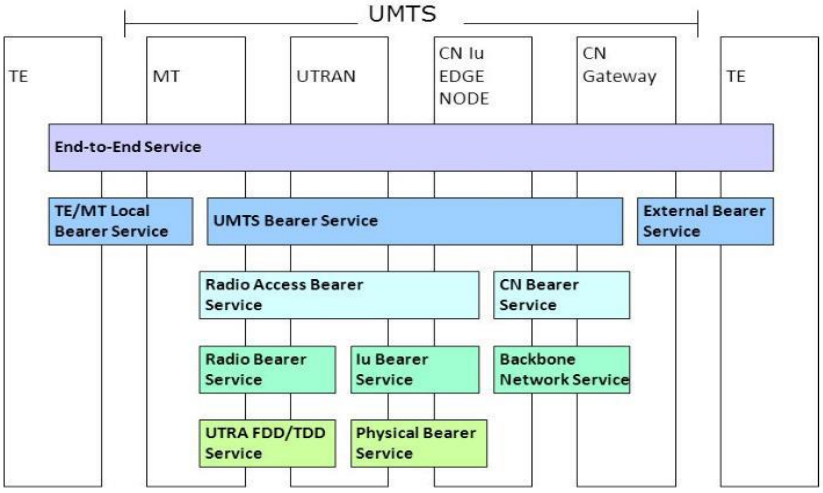


Figure 2. 3 UMTS bearer service architecture [31].

A bearer service includes all aspects to enable the provision of a contracted QoS. As defined in [31], Quality of Service (QoS) is the mechanism of ensuring that a service can be delivered to the end-user in an acceptable time frame and that the service properties are stable over time within predefined boundaries. QoS in UMTS has parameters and major ones are explained as given below in Table 2.2.

Table 2. 2 UMTS QoS parameters

Parameters	**Explanation**
Maximum bit rate	Defines the maximum bit rate when delivering information between endpoints of UMTS bearer
Guaranteed bit rate	Defines the bit rate that the UMTS bearer must carry between its endpoints
Allowed transfer delay	Set the limits of the delay
QoS negotiable	Describes the negotiability level of the service

Based on the basic and other parameters mentioned above, for UMTS bearer services there are four different QoS classes (or traffic classes). They are summarized in Table 2.3 with their properties and application services.

Table 2. 3 UMTS traffic classes' properties and application.

QoS Classes	**Properties**	**Example application**
Conversational	Minimum fixed delay, no buffering, symmetric traffic, guaranteed bit rate	Speech and video call
Streaming	Minimum variable delay, buffering allowed, asymmetric traffic, guaranteed bit rate	Real-time streaming video
Interactive	Moderate variable delay, buffering allowed, asymmetric traffic, no guaranteed bit rate	Web surfing
Background	Big variable delay, buffering allowed, asymmetric traffic, no guaranteed bit rate	File downloading, e-mail

2.3 User Throughput Calculation

Throughput, *the number of useful user data bits per unit of time delivered by the network from the source end-point to the destination end-point* [2]. Theoretically, it is calculated, Equation (2.1), from the channel capacity of using the *Shannon theorem* implemented for UMTS [3]–[6].

$$Shannon\ Limit = 5MHz * log_2\left(1 + \frac{SINR}{16}\right) \quad (2.1)$$

Where,

SINR is the signal-to-interference-and-noise ratio received by UE (HSDPA user). 16 is the spreading factor (SF) of the HSDPA channel. 5 MHz is WCDMA carrier bandwidth.

According to [5], the actual user data rate is reduced from one-third to two-thirds of the physical channel (radio link: HS-DSCH) speed, due to protocol overhead (TCP header, UDP header, etc.) and retransmitted data packets. Due to the physical properties of the HS-PDSCH, the air interface cannot be fully described by Eb/No and the BLER; therefore, the ***average HSDPA Signal-to-Interference-and-Noise Ratio (SINR)*** is used. In this scenario, HSDPA transmission uses a certain portion of the cell power denoted by $\mathbf{P_{HSDPA}}$ that depends on the resource (power) management

strategy used in the network. This power is the remaining **Node-B output power** after deduction of both Release '99 traffic power and Common Control Channel (CCCH) power. The power scheme is shown in Figure 2.4.

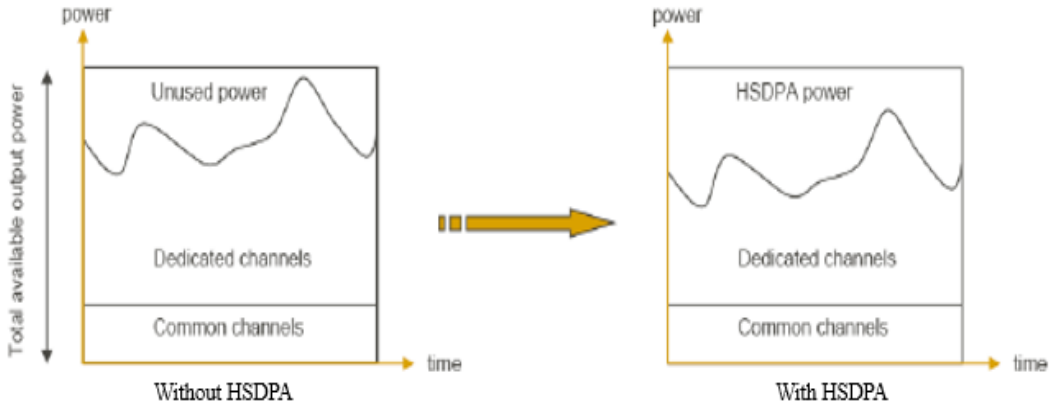


Figure 2. 4 Node-B cell power Usage.

SINR $_{\text{HS-PDSCH}}$ is calculated as in Equation (2.2).

$$SINR_{HS-PDSCH} = 16 * \frac{(P_{HSDPA} - P_{HS-SCCH})}{P_{tot} * \left(1 - \alpha + \frac{1}{G}\right)} \quad (2.2)$$

Where $P_{\text{HS-SCCH}}$ is the power of the HS-SCCH channel. α and G are orthogonality and geometry factor, respectively. P_{HSDPA} is total HSDPA power. P_{tot} is the total transmit power in the downlink including the HSDPA portion. '16' is the fixed spreading factor for HSDPA as defined by Third Generation Partnership Project (3GPP) [32] and the service processing gain for HSDPA users.

Relationship between **achievable average throughput** and the **SINR $_{\text{HS-PDSCH}}$** for **5 parallel codes** is formulated using ***second-order curve fitting*** [5] as ***User-Thrp*** *[Mbps] = 0.0039**$\boldsymbol{SINR^2}$ *+ 0.0476**$\boldsymbol{SINR}$ *+ 0.1421, -5 dB ≤ SINR ≤ 20 dB.* This equation represents either the ***throughput of one user*** having **a certain SINR** or the combined cell throughput of several users having the same average SINR value together [5] and [4]. However, the equation is limited to the number of codes used, only five. So, a general form of formula is used by the dynamic simulator platform [33]. In this formula, total average user throughput (User-Thrp) can be approximated via average HS-DSCH bit rate per user connection [kbps/user] which is the bit rate experienced by a single user connection over the radio channel. User-Thrp is calculated as in Equation (2.3).

$$User - Thrp = \frac{1}{N} \sum_{i=1}^{N} \left(\frac{B_{HS-DSCH,i}}{T_{HS-DSCH,collection,i}} \right) \quad (2.3)$$

Where;

i is the i^{th} connection during the measurement. N is the total number of connections. $B_{HS\text{-}DSCH}$ is the number of HS-DSCH bits successfully transmitted and received over the i^{th} connection during the data collection period ($T_{HS\text{-}DSCH,\ collection}$).

3. Fundamentals of Deep Neural Network and LIME

In this chapter, an overview of deep neural network and the concept of local interpretable model-agnostic explanations (LIME) is presented.

3.1 Fundamentals of Deep Neural Network

Deep neural networks (DNNs) offer a lot of value for data analysis in the case of complex problems, particularly for increasing the accuracy of a machine learning (ML) model [17]. ML is the practice of using algorithms to analyze data, learn from that data, and then decide or prediction about new data [28]. It is a subset of artificial intelligence (AI). AI consists of techniques that enable computers to figure out things from data and deliver AI applications [28]. AI allows a computer to mimic human behavior in some aspects [13], [16], [17], [34]. DNN is a type of ML that allows computers to solve tasks that are more complex [13], [15], [17], [19]. Moreover, DNNs are enabling components of AI, which is one of the most important aspects of deep learning. So there is a relationship between AI, ML, deep learning, and finally to DNN (also called deep net) [17]. This relationship is shown in Figure 3.1.

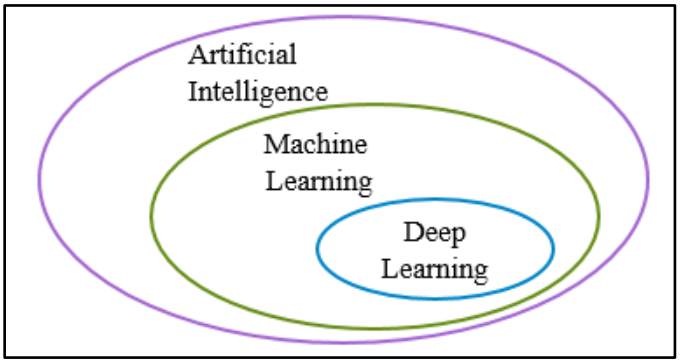


Figure 3. 1 Relation between AI, ML, and DL

As previously defined, deep learning is a branch of machine learning that employs algorithms inspired by the structure and function of the brain's neural networks [17]. However, the neural networks employed in deep learning are not biological neural networks. They are artificial neural networks (ANN) because they share some characteristics with biological neural networks [34].

ANN is built on the same principle as the biological neural network model, but it learns past experiences from data [17].

3.2 Artificial Neural Network

An artificial neural network is a computing system composed of interconnected units known as neurons [13], [17], [19], [28]. *The core of neural networks is the neuron, also known as a unit or node* [28]. The neuron multiplies one or more inputs by a parameter (also referred to as a weight), adds the weighted inputs' values with a bias value (usually 1), and feeds the result into an activation function [28]. The output is then delivered to a neuron or neurons farther down in the neural network for processing. As a result, the ANN computes a function of the inputs by propagating the computed values from the input neurons to the output neuron (s) and connecting the input neurons to build a network using the weights as intermediate parameters [28]. A signal is sent from one neuron to the next through a neuron-to-neuron connection. The receiving neuron processes the signal before sending it to the network's downstream neurons. Learning occurs by changing the weights that connect the neurons [13], [16], [28]. The way it's generally expressed is seen in Figure 3.2. In some sources, this neuron is referred to as a perceptron [16], [17].

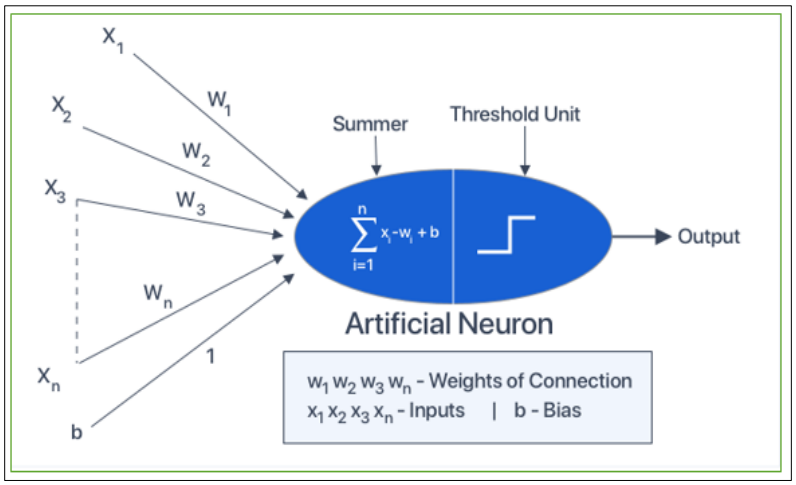


Figure 3. 2 Neuron representation [16]

According to Figure 3.2, this neuron (simple perceptron) is represented mathematically as (3.1).

$$y(x) = f(Z) = f\left(\sum_{i=1}^{n}(x_i * w_i) + b\right) \tag{3.1}$$

$$Z = \sum_{i=1}^{n} (x_i * w_i) + b \tag{3.2}$$

Where x_i is a set of inputs, w_i is a set of weights, n is the number of observations, b is the bias value, f is the activation function, and y (x) is the output. When this neuron is an output neuron and the activation function is sigmoid, y (x) is the prediction probability of class 0 or 1.

Neurons are grouped into three layers. These are the input layers, which take raw data, hidden layers, which receive input from one layer and transmit output to another, and output layers, which generate a prediction, whether it's a classification or regression [13], [14], [16], [17]. Figure 3.3 shows the design of a neural network (NN), which consists of 6 input variables (an input layer), 2 hidden layers, and an output layer. Each layer performs a kind of transformation, commonly unique, on its inputs [13]. Hidden layers, for example, L1 and L2 in Figure 3.3, determine the structure of a deep neural network and thus approximate complex problems [15]–[17].

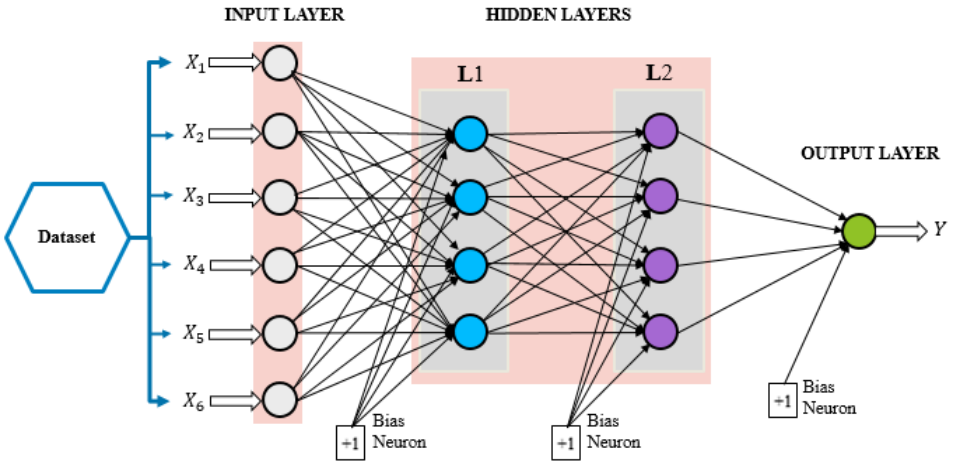


Figure 3. 3 NN architecture with one input layer, one output layer, and two hidden layers

In this figure, there are 6 neurons in the input layer and one neuron in the output layer. The layers are assumed to be used for the model proposed in this thesis work because the features are 6 in number and the output is binary classification. For each feature in the input layer, one neuron is required, and in the output layer for a class 0 or 1, one neuron is needed with a sigmoid activation function [13], [14], [34].

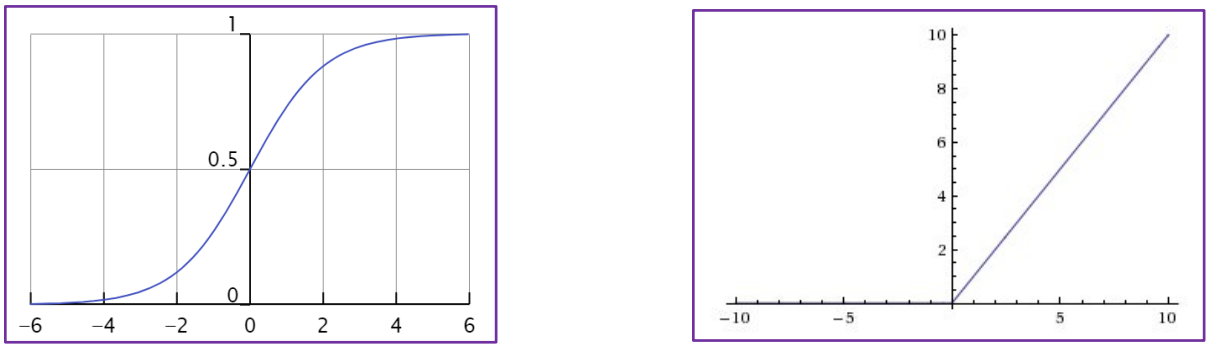
3.2.1 Activation Functions

An activation function is a function that maps a neuron's inputs to its corresponding output [19],[13], [17], [34]. It defines how the weighted sum of the input is transformed into output in a layer of the network. This transformation is often a *non-linear transformation*, [17], [19], [34]. Let's see the two recommended activation functions, [13], [16] used for this thesis work.

Sigmoid

Sigmoid will transform most negative inputs to a number that is very near to 0 [17], [35]. In addition, this sigmoid will transform most positive inputs into a number that is very close to 1 [17], [35]. Furthermore, for inputs close to zero, it will transform the input to a value between 0 and 1. Figure 3.4 depicts its graphical form (a). Mathematically, it is represented in Equation (3.3).

$$\sigma(x) = \frac{e^x}{e^x + 1} \tag{3.3}$$



(a) sigmoid (b) ReLu

Figure 3. 4 Activation functions a) Sigmoid and b) ReLu

ReLU

ReLu is one of the most commonly utilized activation functions in hidden layers [17]. Using rectified linear unit (ReLu), the input is transformed to either 0 or the input itself [17]. It will output 0 if the input is less than or equal to 0. If the input is greater than 0, it will then just output the given input. In Figure 3.4 (b), ReLu is depicted and Equation (3.4) represents its formula.

$$R(x) = max(0, x) \tag{3.4}$$

Activation functions in a hidden layer are mostly the same throughout the hidden layers. Nevertheless, the output layer functions are dependent on the problem type, such as regression or classification [13], [17], [28]. As a result, the sigmoid activation function is used for the output layer because the problem is binary classification [14], [15], [16] in our case. For hidden layers, the ReLu is chosen due to its well-known nonlinear transformation characteristics [16], [34].

3.2.2 Loss Function

The loss function is a metric that determines whether a network is learning in the correct direction [28]. It calculates the deviation from the target or the error in anticipating the actual result [28]. Several standard loss functions in deep learning are available depending on the type of model outcome [13]. Since the model outcome we investigate is binary classification, the class must be represented as a probability to be classified. As a result, a binary cross-entropy (BCE) which is a common loss function for classification use cases, particularly binary classification [14], [28] is implemented. BCE is represented in Equation (3.5).

$$BCE\ Loss = -[y * log(p) + (1 - y) * log(1 - p)] \tag{3.5}$$

Where y is the predicted class and p is the probability of a class outcome.

For class 0

$$BCE\ Los\ (0) = -log(1 - p) \tag{3.6}$$

For class 1

$$BCE\ Loss\ (1) = -log(p) \tag{3.7}$$

3.2.3 Optimizers

The optimizer function is a mathematical algorithm to understand how much change the network will see in the loss function by making a small change in the weight of the neurons [13], [28]. Usually, training would be done in batches due to memory constraints in the system [15]. A batch is a collection of training samples from the entire input [28]. The network updates its weights after processing all samples in a batch. This is called an iteration. The computing of all training samples provided in the input data with batch-by-batch weight updates is called an *epoch* [13]. In each iteration, the network leverages the optimizer function to make a small change to its weight

parameters to improve the end prediction by reducing the loss function [13], [16], [17], [28]. There are various optimization algorithms available [13]. *Adam*, which stands for adaptive moment estimation, is by far the most popular and widely used optimizer in deep learning [13], [28]. This optimization algorithm computes an adaptive learning rate for each parameter. It defines a momentum and variance of a gradient of the loss and leverages a combined effect to update the weight parameters. The momentum and variance together help smooth the learning curve and effectively improve the learning process [13], [28].

3.2.4 Hyperparameters

Hyperparameters are the variables that define a model's overall structure and, as a result, the learning process [13], [17], [18], [28], [36]–[38]. They differ from the actual parameters of a model (such as weights), which are learned during the training phase [13], [17]. Because, unlike model parameters, hyperparameters cannot be learned, they must be fine-tuned using a variety of techniques [13]. The hyperparameters used in this thesis are the number of layers, number of neurons in a layer, number of epochs, batch size, and learning rate [13].

A. Number of Neurons in a Layer

By altering the network width (i.e., neurons in a layer), DNNs are more robust for most classification and regression use-cases [13], [15]. When determining the number of neurons in the first layer, the number of input dimensions is a useful rule of thumb [28]. The output layer's neurons are determined not just by the number of features, but also by the nature of the problem [28]. For example, in this thesis, there are six attributes; hence, the input layer requires six neurons. Because the problem involves binary classification, only one neuron is needed for the output layer (0 or 1) [28].

B. Number of Layers

It's true that simply adding a few more layers will boost performance, at least marginally [13], [15]. The training time and computation grow as the number of layers increases [16]. Furthermore, a higher number of epochs is required to see good results [13], [16], [17], [28]. However, avoiding the use of deeper networks is not always possible but requires an optimum number of layers [15].

C. Number of Epochs

The number of iterations, the training process runs over the dataset is measured in epochs [13], [28]. Increasing the number of epochs for model training can improve results in some cases, but it comes at the cost of more computation and training time [13], [17].

D. Batch Size

The batch size is the number of instances that pass over the network before a weight update [28]. For small batch sizes, a model requires less memory while training networks with fewer instances each time [13]. However, because the tiny batch size makes it difficult to get a decent approximation of the gradient, fine-tuning the weight and bias can be difficult [13], [28]. One epoch equals one forward and backward pass through all of the training instances [17]. One epoch requires multiple iterations to finish [17], [19], [28], [34]. When it comes to batch size, there are no hard and fast rules. However, to save memory, it should not be too small, and it should be much smaller than the training data set [13].

E. Learning Rate

In the context of the optimization algorithm, the learning rate is the length of each step, or, to put it another way, the maximum size of each iteration's weight updates [13], [28]. In this thesis, the Adam optimizer is used, and the default value is 0.001[13].

3.3 Multilayer Perceptron

Multilayer perceptron (MLP): is an artificial neural network, also called a *feedforward neural network* used in any real-world setting [13], [15], [17], [16]. Neural networks can be dense: a series of connected layers that form a network connecting an observation’s feature values at one end, and the target value (e.g., observation’s class) at the other end, like the one shown in Figure 3.3. The name feedforward comes from the fact that an observation’s feature values are fed “forward” through the network, with each layer successively transforming the feature values with the goal that the output at the end is the same as the target’s value. Once again, neural networks with many hidden layers are considered “deep” networks and their application is called deep learning.

Neural networks are created with all parameters initialized as small random values. Once an observation (or more often a set number of observations called a batch) is fed through the network,

the outputted value is compared with the observation's true value using a loss function. This is called *forward propagation*. Next, an algorithm goes "backward" through the network identifying how much each parameter contributed to the error between the predicted and true values, a process called backpropagation. At each parameter, the optimization algorithm determines how much each weight should be adjusted to improve the output. MLP learns by repeating this process of forward propagation and backpropagation for every observation in the training data multiple times. Each time, all observations have been sent through the network is called an epoch and training typically consists of multiple epochs, iteratively updating the values of the parameters.

To build a model, matrix formalism simplifies mathematical complexity, and therefore, all matrix dimensions are needed with important notations [15].

- L: Number of hidden layers, excluding the input layer but not the output layer
- N_L: Number of neurons in layer L

For a general network, the total number of N neurons can be written (3.8) and the details can be found [15].

$$N_{neurons} = N_x + \sum_{i=1}^{L} N_i = \sum_{i=0}^{L} N_i \tag{3.8}$$

Where, by convention [15], $N_0 = N_x$, number of input features.

Each connection between two neurons will have its weight. Let's indicate the weight between neuron i in layer L and neuron j in layer $(L-1)$ with $w_{ij}^{[L]}$. In Figure 3.3 for example, all the neurons of the input layer are connected to all neurons of hidden layer 1 with the weights between each neuron in the input layer and all the others in layer 1. These weights between the input layer and layer 1 can be written as a matrix (3.9).

$$W^{[1]} = \begin{bmatrix} w_{11}^{[1]} & \cdots & w_{16}^{[1]} \\ \vdots & \ddots & \vdots \\ w_{41}^{[1]} & \cdots & w_{46}^{[1]} \end{bmatrix} \tag{3.9}$$

This means that the matrix $W^{[1]}$ has dimensions 4x6. Of course, this can be generalized between any two layers L and $(L-1)$ [15], meaning that the weight matrix between two adjacent layers L and $(L-1)$, indicated by $W^{[L]}$, will have dimensions $N_L \times (N_L - 1)$. This is generalized as (3.10).

$$W^{[L]} = \begin{bmatrix} w_{11}^{[L]} & \cdots & w_{1N_{L-1}}^{[L]} \\ \vdots & \ddots & \vdots \\ w_{N_L 1}^{[L]} & \cdots & w_{N_L N_{L-1}}^{[L]} \end{bmatrix} \tag{3.10}$$

According to [15], $N_0 = N_x$ is the number of input features (not the number of observations i.e. m). The bias (indicated by the *Bias* $_{Neuron}$ in Figure 3.3) will be a matrix this time. Each neuron that receives inputs will have its own bias, so when considering two layers, L and (L – 1), N_L *has* different values of b. Therefore, this matrix is represented by $B^{[L]}$ and has dimensions $N_L \times 1$.

Output of Neurons

By consider the i^{th} neuron of the first layer, the input layer is by definition layer 0 [15], its output is $y_i^{[1]}$ and assume that all neurons in layer l use the same activation function i.e. ReLu: indicated by $R^{[1]}$. The output is:

$$y_i^{[1]} = R^{[1]}\left(Z_i^{[1]}\right) = R^{[1]}\left(\sum_{j=1}^{N_X} w_{ij}^{[1]} x_j + B_i^{[1]}\right) \tag{3.11}$$

Where; *Zi* is as in (3.2) and rewritten as:

$$Z_i^{[1]} = \sum_{j=1}^{N_X} w_{ij}^{[1]} x_j + B_i^{[1]} \tag{3.12}$$

From (3.11), a matrix for all the output of layer 1will be:

$$Z^{[1]} = W^{[1]}X + B^{[1]} \tag{3.13}$$

Where; $Z^{[1]}$ is N_1 x 1 dimensions, and X is a matrix with all observations (rows for the features, and columns for observations). Similarly, by assuming all neurons in layer L to use the same activation function (i.e. ReLu, except for output layer, which is sigmoid) of $R^{[L]}$, the Equation (3.13) can be generalized for a layer L as:

$$Z^{[L]} = W^{[L]}Z^{[L-1]} + B^{[L]} \tag{3.14}$$

This is because *layer L receives its inputs from layer (L – 1). $Z^{[L-1]}$* replaces *X*. Dimension of $Z^{[L]}$ is N_L x 1. The output of layer L, similar to (3.11) with matrix notation, will be:

$$Y^{[L]} = R^{[L]}\left(Z^{[L]}\right) \tag{3.15}$$

However, for the output layer where *L* is the last, the activation function is sigmoid and the number of neurons is one. In this context, Equation (3.15) is valid for hidden layers. By substituting the ReLu function with sigmoid, the output of the network architecture will be:

$$y(hat) = \sigma^{[L]}\left(Z^{[L]}\right) \tag{3.16}$$

3.4 Metrics

The fundamentals of model performance metrics are critical for evaluating model performance. Metrics are the functions that are used to evaluate the model's performance on an unseen dataset, commonly known as the validation dataset. They are used to validate the test results while reporting [13]. They are used to monitor the performance of model learning during the algorithm training or model development phase. In addition, metrics are used to evaluate the model before final implementation during the final test. In this study, the DNN model must be able to distinguish low user throughput data points from normal data points. Moreover, the final goal is to make sure the unseen dataset is classified based on the input features. For classification, the most known metrics are accuracy, confusion matrix, recall, precision, F1-score, and AUC-ROC curve [13].

Confusion Matrix: is a table that shows how well a classification model (or "classifier") performs on a set of test data for which the true values are known [34]. A basic form of a confusion matrix is shown in Table 3.1.

Table 3. 1 Confusion matrix representation

	Predicted (0)	**Predicted (1)**
Actual (0)	TN	FP
Actual (1)	FN	TP

Note that the outcomes are whether class 0 or 1.

0 is a negative class (low user throughput) and 1 is a positive class (not low user throughput). The abbreviations in Table 3.1 are defined as follows.

TP is a true positive.

TN is a true negative.

FP is a false positive.

FN is a false negative.

Performance metrics are derived from these terminologies.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.17}$$

Accuracy is the number of correctly (True) predicted results out of the total prediction.

$$Recall = \frac{TP}{TP + FN} \tag{3.18}$$

$$Precision = \frac{TP}{TP + FP} \tag{3.19}$$

$$\mathrm{F1}_{\mathrm{score}} = 2 * \left(\frac{Precision * Recall}{Precision + Recall}\right) \tag{3.20}$$

AUC-ROC Curve: for various threshold levels, the curve is a performance measurement for classification tasks [39]. Two parameters are plotted. TPR (True Positive Rate) versus FPR (false positive rate) [39]. AUC represents the degree or measure of separability, while ROC (Receiver Operating Characteristics) is a probability curve. It tells how much the model is capable of distinguishing between classes. The higher the AUC, the better the model is at predicting actual class correctly [39].

3.5 LIME

For deep neural networks, which can be represented by complex mathematical equations, it is very difficult to understand why the model generalizes an outcome. This thesis employs local interpretable model-agnostic explanations (LIME) [40] for local approximation of the model outcome to explain the DNN of unknowable output (just the number of neurons utilized, matrix of weights, and hidden layers). LIME is a novel explanation technique that explains the outcomes of

any classifier in an interpretable and faithful manner by learning an interpretable model locally around the instance of interest [41].

LIME Definition

It is defined as:

$$Exp(x) = argmin\ L\ (f(x), g(x), \pi_x) + \Omega(g), g \in G \tag{3.21}$$

Where G is a category of interpretable models, such as linear models, decision trees, or otherwise [42]. A user can be presented with visual or textual artifacts when using a model g ∈ G. g has a range of $\{0, 1\}^{d}$ and indicates if it is interpretable. Ω (g) is a measure of the complexity of the explanation g ∈ G. For example, for decision trees Ω (*g*) may be the depth of the tree, while for linear models, Ω (*g*) may be the number of nonzero weights. In classification, *f*(*x*) indicates that *x* belongs to the probability of a certain classification. Π_X (z) is the distance measure from *z* to *x*, to define locality around *x*. Finally, *L* (*f, g, πx*) is a function that measures the difference between *f* and *g* in the defined range by *πx*. To ensure both interpretability and local fidelity, when Ω(*g*) is small, *L*(*f, g, πx*) must be minimized enough to be understood.

LIME Explanation

When predicting a class of user throughput (whether low user throughput or not), LIME acts on a single sample. Firstly, the slight perturbation of the selected sample is used to form a new data set (representative dataset), and the similarity between the sample in this data set and the original sample is calculated through statistical distance and a similarity matrix. The obtained data set is trained on the *classification model* to get the influence of the similarity of the samples on the prediction effect to obtain *a simple model with partial weight*. It is then constructive to make predictions on simple models using the *large contributing features*. The features that make a big contribution are selected in four steps [40], [42], [43]. In the first step, select the features that have the highest weight on the regression fit when predicting with a classification model. In the second step, select the features of the regression fit that can improve the prediction of the model. In the third step, select the regularized feature of the lasso fit with the smallest shrinkage rate through the model. Finally, build a decision tree using the number of nodes that are not more than the selected features.

4. Proposed Low User Throughput RCA System Model

In this chapter, the overall proposed model of low user-throughput RCA is presented. Additionally, the relationships between the dependent variable and the independent variables are introduced to understand the causal input-output relationships.

4.1 RCA System Model

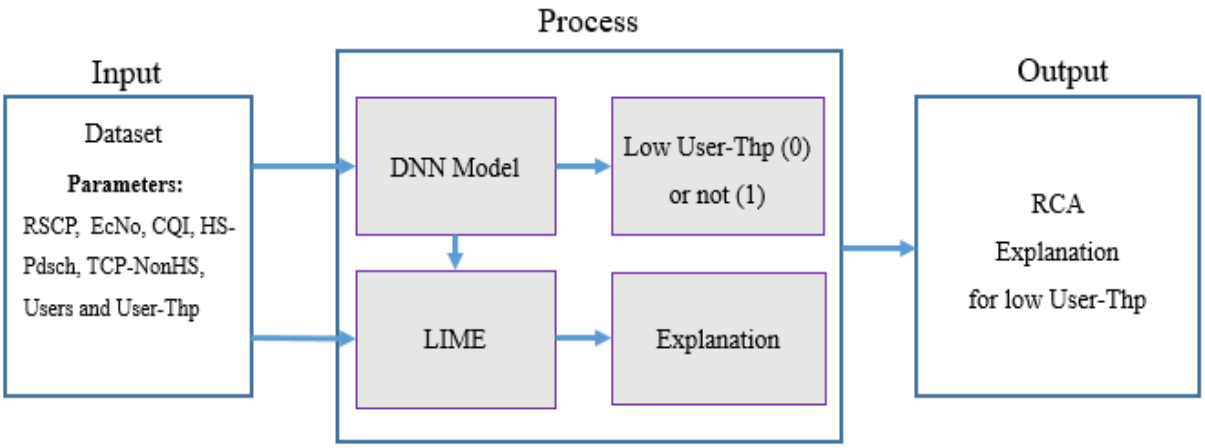


Figure 4. 1 Low user-throughput RCA system model

The RCA system model is depicted in Figure 4.1. In this figure, the dataset is the raw data collected for model training and evaluation. The DNN model is a binary classifier for user throughput of class 0 or 1 using unseen data. The DNN model is a black box that is complex and not easily interpretable. LIME takes DNN output and an unseen dataset for an explanation of why the throughput is low. The input parameters are HSDPA-Users, CQI, RSCP, EcNo, HS-PDSCH, and TCP-NonHS. The response variable is the User-Thp. Since the system model is very condensed, the end-to-end system process is depicted in Figure 4.2 with three main tasks. These are data preprocessing, model development, and root cause explanation.

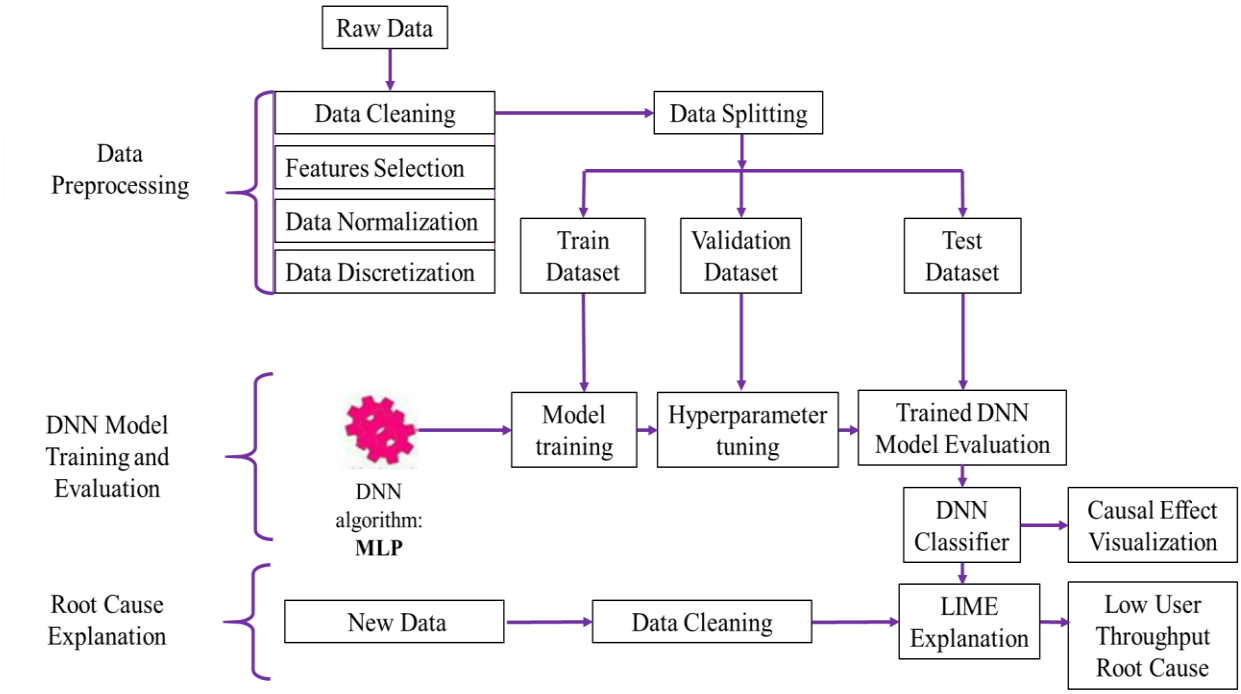


Figure 4. 2 End-to-End low user throughput RCA process

4.1.1 Dataset and Experiment Configuration

Enough data must be collected to develop the model for root cause analysis. The 1.57 million samples used in this study came from Ethio Telecom's hourly network performance data measurement report from January 1, 2021, to May 31, 2021. The data were obtained from 76 NodeBs with 445 cells in Bahir Dar city. One Lenovo laptop (8 GB RAM, 500 GB hard drive), Python, and Microsoft Excel were used. The dataset comprises six features that are causal factors for low user throughput. Correlation analysis yielded these parameters. Table 4.1 lists all of the parameters. The target (class) variable is the output parameter, while the features are input parameters. Each parameter or feature has its own set of benefits, as well as the impact it has on user throughput. The next subsections go through these features in further depth.

Table 4. 1 Parameter description

Parameters	Description	Type	Units & range
TCP-NonHS	TCP power for non-high speed services	Feature (X_1)	(0 to 45) dBm
HSDPA-Users	Number of users of HSDPA	Feature (X_2)	0 to 64
Pdsch-Avail	Available HS-PDSCH code	Feature (X_3)	0 to 15
CQI	Mean CQI value reported by UE	Feature (X_4)	0 to 30
RSCP	Mean RSCP value reported by UE	Feature (X_5)	(-115 to -25) dBm
EcNo	Mean EcNo value reported by UE	Feature (X_6)	(-25 to 0) dB
User-Thp	Average User Throughput	Target (Y)	Kbps

As a minimum requirement for good user throughput, depending on service type, suitable RF conditions are required [5]. Signal strength (RSCP) and signal quality (EcNo) at the end device (UE) are used to determine this state [7], [32].

A. RSCP

RSCP is an abbreviation for received signal code power [44]. It's the UE's measurement of the received CPICH signal's power. The term RSCP refers to a cell's coverage [6]. The RSCP range is determined by the cell's service. The range is defined in [44], and it covers from -115 dBm up to -25 dBm. However, a good RSCP threshold is to be determined by the operator.

B. EcNo

Ec/No [6] is a common pilot channel that divides the received energy per chip by the power density in the band [44]. It is CPICH RSCP/RSSI in theory [44]. RSSI, or "received signal strength indicator," is a downlink signal strength indicator that measures the wideband received power over the channel bandwidth [44]. EcNo is, however, a network quality metric [3]. Nonetheless, the radio measurements must be accurately interpreted because they might be a false indicator of network quality [3].

C. CQI

CQI, or “Channel quality indicator”, is a network quality metric like EcNo [3]. Nevertheless, CQI is not a direct radio interface measurement, for example, RSCP and EcNo [3]. Rather, the CQI value should follow the UE's Signal-to-Interference plus noise ratio (SINR), and as such, is an

important metric to consider during network optimization [3]–[5], [45]. The CQI provided by HSDPA can be a useful measurement to provide information regarding the radio conditions in the cell. CQI assists the base station scheduler in determining when it is appropriate to schedule for a specific user and in setting the proper link adaptation parameters [4], [45]. The CQI gives the UE an estimate of what kind of data rate could be reliably received under the current conditions. CQI's data indicates the size of the transport block, the number of codes, and the modulation order that the UE anticipates [3]–[6]. Aside from external considerations like interference from neighboring cells and relative strength of the own cell (serving HSDPA cell), the CQI also considers UE internal implementation issues like whether the UE has an advanced receiver or reception antenna diversity [3], [4], [6]. As a result, CQI is a UE-generated vendor-specific value.

D. HS-PDSCH Code

The HS-PDSCH is a high-speed physical downlink shared channel that serves as a channelization code for transporting or sending user-specific data [4], [5]. This code is a limited resource element whose availability has a significant impact on user throughput. It is also restricted by license, which limits its availability [7]. In general, HS-PDSCH codes are utilized to carry HSDPA services, hence a lack of HS-PDSCH codes will result in a reduction in HSDPA data rates [7]. In addition to time multiplexing, the available limited code influences average user throughput during code multiplexing [33].

E. TCP-NonHS

It is the transmitted carrier power for non-HSPA users in a cell (including CCHs) [7]. As shown in Figure 2.4 in Chapter 2, this power affects the remaining power available for HSDPA users.

F. HSDPA-Users

The number of HSDPA users in the cell affects user throughput performance [4], [5], [45]. According to [45], the per-user throughput (average user throughput in a cell) experienced depends strongly on the number of simultaneously active HSDPA users that are sharing the HS-DSCH.

Once we have a clear description of the features; let us see a sample dataset taken from a Jupyter notebook using the collected data from PRS as shown in Table 4.2. This table shows all the necessary parameters, including the date, time, and Cell-ID.

Table 4. 2 Sample of used dataset snapshot taken from Jupyter notebook

Date	Time	Cell-ID	TCP-NonHS	HSDPA-Users	Pdsch-Avail	CQI	RSCP	EcNo	User-Thp
01/16/21	0:00	39031	36.07	1.22	6.32	23.03	-86.29	-5.63	3103.89
01/16/21	0:00	33003	35.87	0.00	3.31	12.51	-115.00	-24.50	799.33
01/16/21	0:00	32151	34.62	0.22	4.92	13.36	-90.56	-6.92	685.80
01/16/21	0:00	33023	36.14	0.73	10.22	14.18	-70.94	-8.00	221.04
01/16/21	0:00	30345	36.06	0.70	5.82	15.25	-81.70	-6.44	1168.55
...	...	...	...	...	...	...	...	...	...
02/28/21	23:00	39643	35.98	0.10	5.98	10.90	-75.39	-8.41	208.19
02/28/21	23:00	31692	36.52	1.89	11.36	13.30	-83.67	-4.86	1024.19
02/28/21	23:00	33001	36.04	0.10	2.12	13.69	-91.97	-7.09	875.06
02/28/21	23:00	39281	36.44	2.16	4.44	15.23	-84.22	-5.99	970.60
02/28/21	23:00	35283	36.05	0.04	3.74	9.90	-70.66	-7.27	501.05

This sample data was taken from 01-16-2021 to 02-28-2021 on an hourly basis. Therefore, 24 samples per day per cell are shown in the table.

4.1.2 Data Preprocessing

To produce a good model, the dataset must be quality data. As a result, the raw dataset has to be cleaned from noisy data, missing values, and outliers. In data preprocessing, there are very important tasks to be performed.

1. Data cleaning: improves the quality of data by handling missing values through interpolation. Outliers are treated by the interquartile range (IQR) method.
2. Features selection: the statistical data collected has various types, so to select the most influential parameters; a correlation analysis (Pearson correlation) is used.
3. Data normalization: Normalization is a technique for shifting and rescaling values so that they fall between 0 and 1. It is the process of altering characteristics on a comparable scale. This enhances the model's performance and training stability. The dataset, shown in Tables 4.1 and 4.2, covers a wide range of units and ranges. Therefore, to enhance the performance and training stability by achieving a uniform range, a Min-Max scale is utilized.

   Mathematically:

$$MinMax\ Scaling = \frac{(X - X_{min})}{(X_{max} - X_{min})} \quad (4.1)$$

Xmax and Xmin are the maximum and the minimum values of the feature (X) respectively.

4. Data discretization: Because the throughput is a continuous value, and the goal is to separate the poor user throughput from the rest, this variable must be converted to binary. Low user throughput (class 0) and not low user throughput (class 1) make up this binary class. User throughput is low, below 1Mbps, according to a study on optimal neural networks for monitoring key performance metrics in HSDPA [46].
5. Data splitting: The prepared data has to be split into training, validation, and testing datasets [13]–[17], [19], [34]. The splitting ratio is 70%, 20%, and 10% for the training dataset, validation dataset, and test dataset, respectively.

4.1.3 DNN Model Development

To build the model, Keras is utilized [13], [47]. Keras is a powerful framework and open-source Python library for developing and evaluating deep learning models [13]. It uses an efficient numerical computation library, TensorFlow, as its backend [13]. Moreover, it allows defining, training, and evaluating neural network models. The following steps are conducted for model development.

1. Building DNN model:

   The input layer, hidden layers, output layer, number of neurons for each layer, and activation functions are defined using the sequential technique [13], [28]. The Adam optimizer, binary cross-entropy (loss function), and accuracy metric are all configured to construct the model [28]. The Fit () method is used to train a model for a specific number of epochs (iterations on a dataset) and batch size [28]. The Evaluate () method returns the training model's loss and accuracy metrics. Hyperparameters are to be tuned to improve the model performance [13]. Hyperparameter tuning is the practice of experimenting with alternative settings for hyperparameters to optimize the overall model process. The selection of the appropriate hyperparameters is a time-consuming and iterative procedure [13], [17], [28], [36]–[38]. Keras tuner [37] is used because of its scalability. Bayesian optimization[13], [28], [38] is a Keras suggested algorithm, hence it has been used for this

thesis. Various algorithms are, however, widely used nowadays, and the details can be found in [36], [37], [48].

2. DNN model performance evaluation:

   As described in Chapter 3, the accuracy, confusion matrix, recall, precision, F1-score, and AUC-ROC curve are the metrics used to evaluate the final model developed.

4.1.4 Model Performance Evaluation

There are two scenarios to incorporate in the model performance evaluation: learning and final model evaluation. After the training is finished and the hyperparameters have been fine-tuned, the final model is put to the test with a test dataset. To ensure the model's credibility and validation, the test scenario, which is the last model evaluation, is undertaken using a comparative technique. As reference models, the two most well-known binary classification techniques; logistic regression and random forest, are used.

Logistic Regression

To model dichotomous outcome variables, logistic regression (LR) [35], often known as a logit model, is utilized. The term dichotomous refers to the fact that there are only two possible classes. The log odds of the result are treated as a linear combination of features (input variables) in the logit model. It is the method of choice for binary classification issues (problems with two class values) [49]. Fundamentally, it is one of the most widely used binary-classification machine learning techniques. LR employs the sigmoid function for decision-making.

Random Forest

Random forest (RF) is a supervised learning algorithm used to solve classification and regression issues. This algorithm is an ensemble model [50], which creates a "forest" of numerous decision "trees" with a user-defined number of trees. Each decision tree is based on a subset of the original data set's attributes (columns) and observations (rows). The model's final classification is the one that most closely matches the classifications provided by the most individual decision trees.

4.1.5 Features Causal Effect on User Throughput

Because the factors are multidimensional, it is difficult to understand the impact of each one on throughput (i.e. multivariate). Furthermore, a detailed grasp of the cause-and-effect relationship for the problem at hand — RCA for low user throughput — is essential. PDPbox [51], a Python

toolbox, is employed in this investigation for such a reason. The partial dependence plot (PDP) shows the impact of a feature (s) on the output of a machine-learning model. A PDP can disclose whether a target and a feature have a linear, monotonic, or complex relationship[51], [52]. The PDP plot is created by changing feature values and observing how the outcomes vary [43]. The relationship is causal for the model since the outcome is explicitly represented as a function of the features; the dataset was acquired from a real-world network [52]. As a result, the plot helps to extract insights and ensure that the model is learning something sensible.

4.2 Root Cause Explanation Using LIME

In this section, there are two parts of the analysis. The first part shows how to use LIME. In this part, LIME is used to evaluate four randomly chosen samples in this study. The second part is executed on ten cells, each of which contains 50 low-throughput samples. Cells from various sites (areas) have been chosen. Tibebe Gion Specialized Hospital (TGSH) is the first location covered by these cells. Bahir Dar Institute of Technology (BiT) is the second area of interest. Gamby General Hospital is the last one. Figure 4.3 depicts the cell locations as determined by the satellite image.

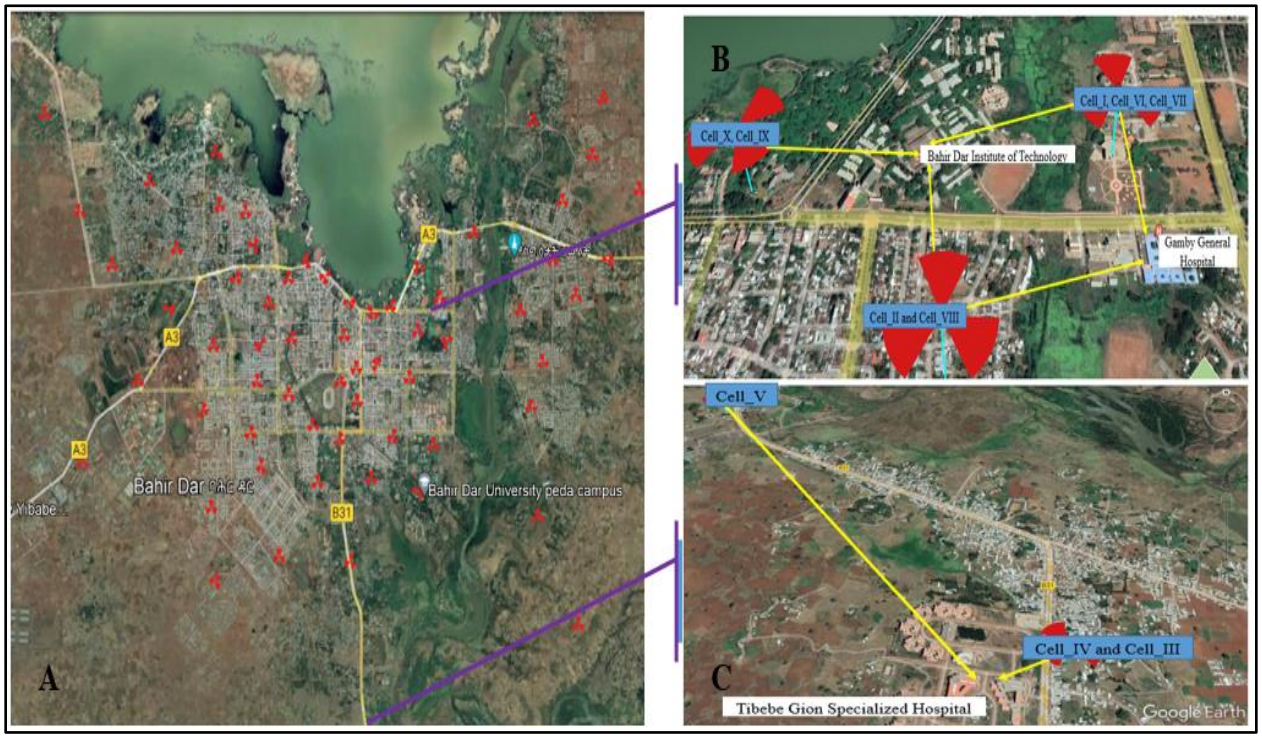


Figure 4. 3 Satellite image of selected cells located in Bahir Dar city

Figure 4.4 shows the user throughput of chosen cells for 24 hours of data. It is observed that from 19:00 to 22:00, the throughput for all the cells is less than 1 Mbps. As demonstrated in Figure 4.5, the sample distributions are shown using an empirical cumulative distribution function (ECDF). ECDF creates a cumulative distribution function for our observed data rather than theory.

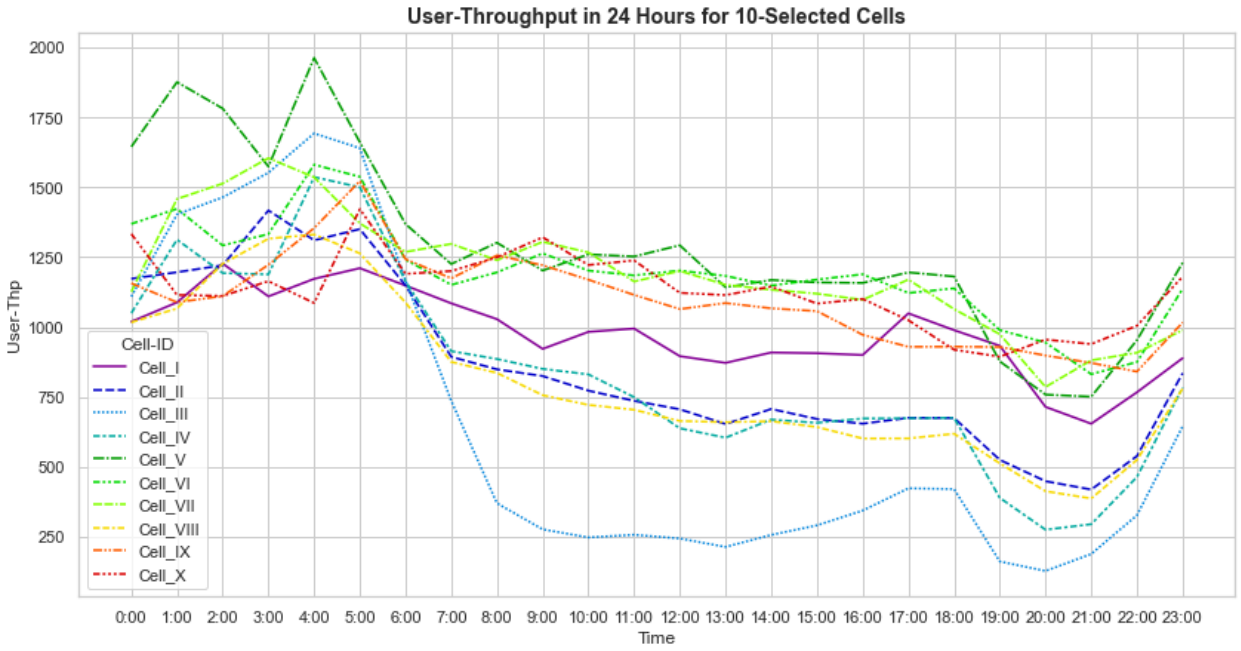


Figure 4. 4 User throughput (in Kbps) in 24 hours for 10 selected cells

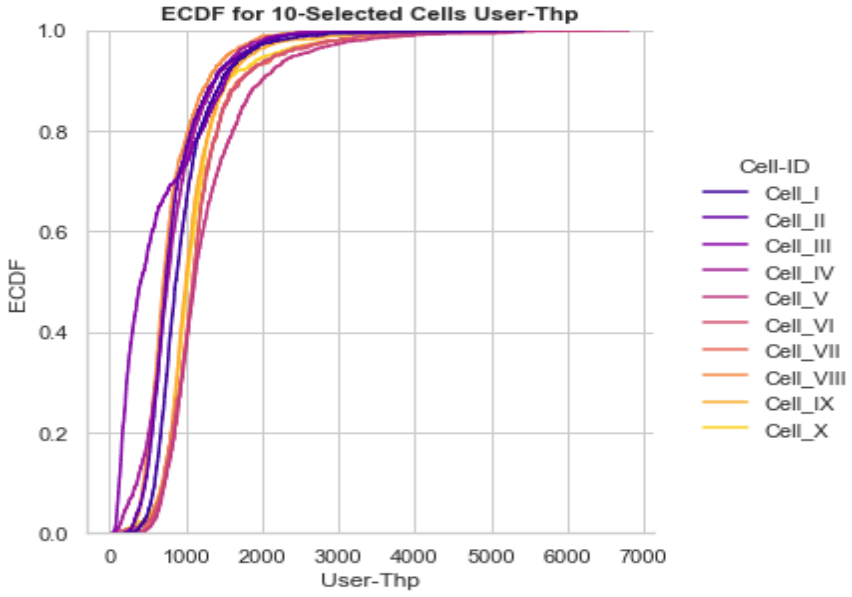


Figure 4. 5 User throughput (Kbps) ECDF per cell

5. Results and Discussions

In this chapter, the results of the proposed RCA model and data preprocessing are presented. The approach to dataset collection and the sample description are presented in the data preprocessing section. Moreover, the proposed model is optimized using hyperparameter tuning. The RCA results are discussed in two parts, in general. The first part focuses on understanding the influence of each input parameter on the proposed model. The second part addresses the root cause explanation using LIME to justify the reasons why the throughput is low for the model outcome.

5.1 Data Preprocessing and Visualization

A. Missing Data Handling

Table 5. 1 Missing data handling

Parameters	Missing Data	Missing data after Interpolation	Total data
Date	0	0	1573440
Time	0	0	1573440
Cell_ID	0	0	1573440
TCP-NonHS	90411	0	1573440
HSDPA-Users	63	0	1573440
Pdsch-Avail	111433	0	1573440
CQI	192754	0	1573440
RSCP dBm	63	0	1573440
EcNo dB	63	0	1573440
User-Thp	189763	0	1573440

Using the interpolation technique, missing data has been substituted as shown in Table 5.1. It is observed that there is no missing value after interpolation.

B. Features Correlation Matrix

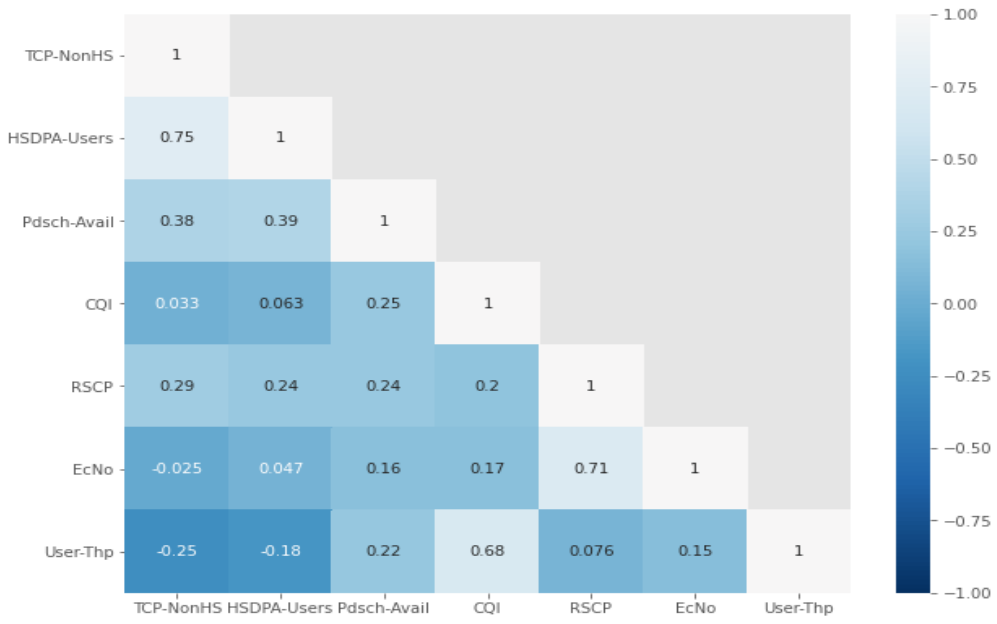


Figure 5. 1 Correlation between features or User-Thp for RCA dataset

Figure 5.1 depicts the correlation analysis result of parameters. In this figure, the most important features are selected based on their correlation coefficient via Pearson's correlation principle. It is observed that the feature with the strongest coefficient is the CQI. It has a 0.68 correlation coefficient with User-Thp. The Pdsch-Avail is the next feature with a 0.22 correlation coefficient. The feature with the weakest correlation is between RSCP and User-Thp.

C. Visualization of Raw Data

	count	mean	std	min	25%	50%	75%	max
Cell-ID	1573440.0	34268.487919	4109.551638	30001.00	31353.00	33102.500	36453.0000	59642.00
TCP-NonHS	1573440.0	37.296080	1.353649	33.79	36.21	36.990	38.0400	44.51
HSDPA-Users	1573440.0	3.396044	5.581734	0.00	0.22	1.140	4.1900	63.05
Pdsch-Avail	1573440.0	8.309718	3.595754	1.00	5.04	8.610	11.6000	14.94
CQI	1573440.0	14.050462	2.979747	1.00	12.55	14.220	15.8000	29.67
RSCP	1573440.0	-87.643770	13.667053	-115.00	-91.82	-83.990	-78.6400	-33.99
EcNo	1573440.0	-10.718783	5.623406	-24.00	-11.60	-9.140	-7.0300	0.00
User-Thp	1573440.0	1084.025687	691.234661	0.05	668.75	977.725	1337.8825	18327.20
Label	1573440.0	0.481422	0.499655	0.00	0.00	0.000	1.0000	1.00

Figure 5. 2 Key dataset statistics taken from Jupyter notebook

The basic statistics of the dataset are shown in Figure 5.2. In this figure, *50% of user throughput is 977.725 Kbps*, which is less than *1 Mbps*. This throughput distribution is plotted using histogram function and empirical ECDF shown in Figure 5.3. In Figure 5.3 (A), its histogram, and in (B) its ECDF are plotted. Moreover, the user throughput concerning individual features distribution using raw data is visualized as follows. The proportion of each feature within a specific percentile range is plotted using python toolbox, PDPbox.

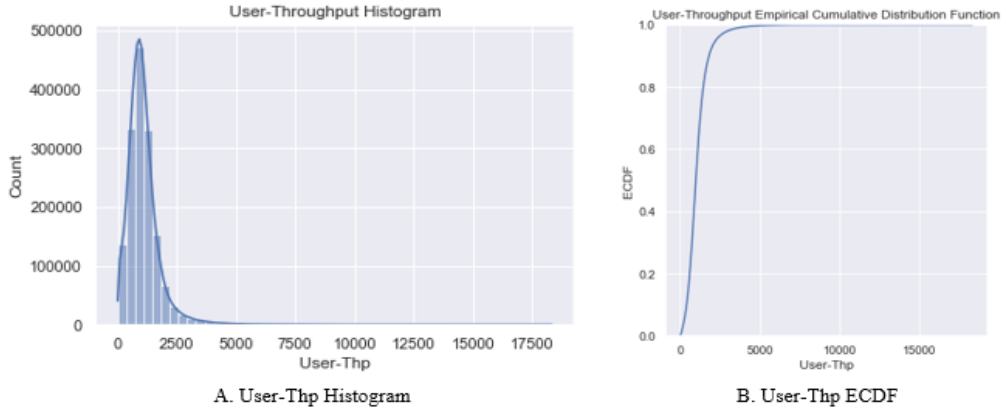
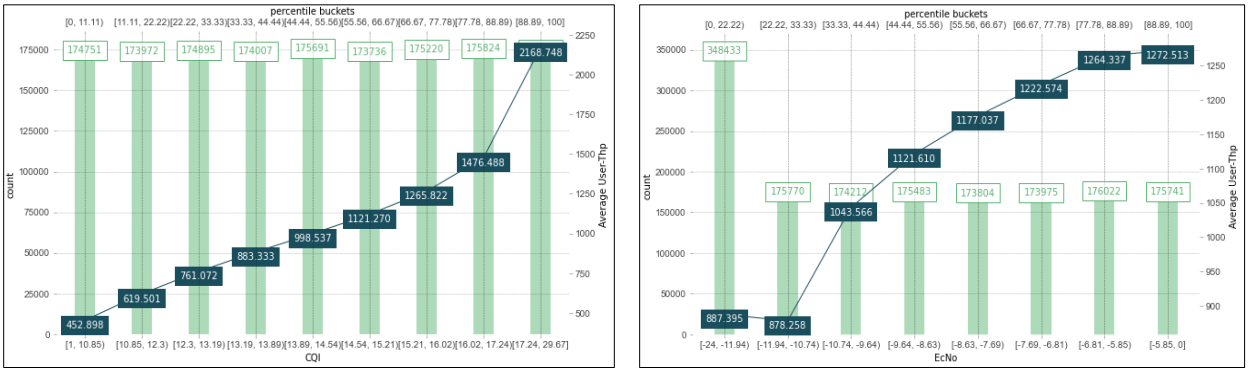
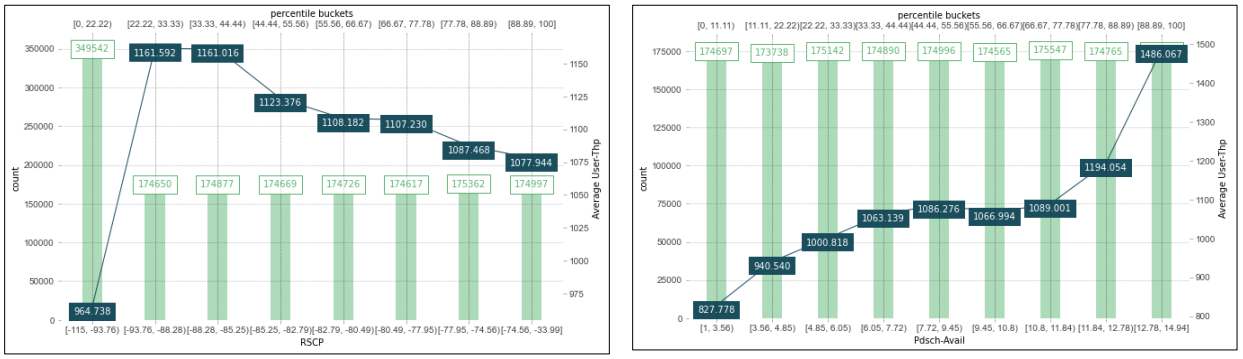


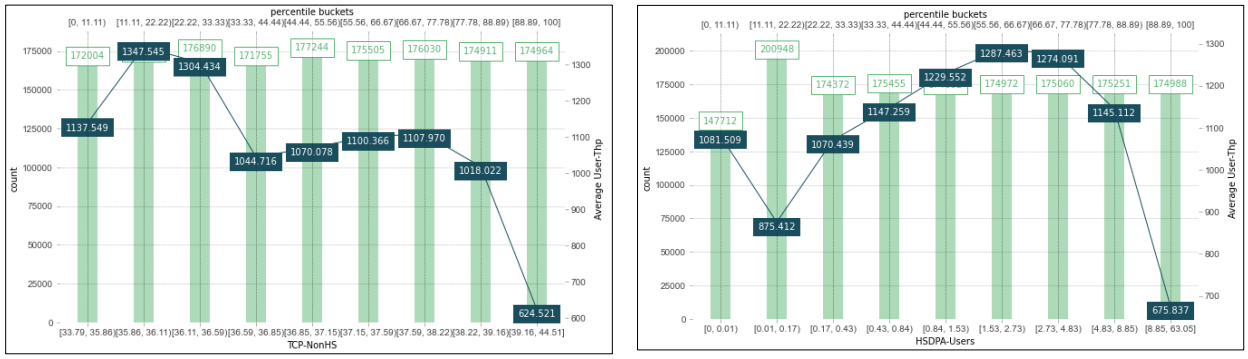
Figure 5. 3 User-Thp distribution using histogram and ECDF



A. CQI and EcNo versus average user throughput



B. RSCP and Pdsch-Avail versus average user throughput



C. TCP-NonHS and HSDPA-Users versus average user throughput

Figure 5. 4 Feature sample's distribution versus average user throughput

In Figure 5.4, each feature has its sample distribution with average user throughput. In this figure, the share of features for low user throughput is identified. *Nevertheless, it could not show the cause and effect but only data samples proportion that each feature constitutes*. Accordingly, in the plot (A), the average user throughput below 1 Mbps comes from a CQI of 14.56 and below. This CQI range has 55.56 % of the data distribution from all the data samples. For EcNo 33.33 % the dataset is below -10.74 dB contributing for 1 Mbps and below. Similarly, in the plot (B), -93.76 dBm of RSCP samples has constituted 22.22 % of the dataset. For available code, 22.22 % comes from below 4.85 HS-PDSCH. In plot (C), for TCP-NonHS, 11.11 % comes from above 39.16 dBm. As shown in plot (C), for the number of users, 22.22 % comes from two scenarios. In the first scenario when the users are high, more than and equal to 8.85 (11.11 %). In the second scenario, when the users are very small (11.11 %). The percentages in this observation are for the contribution of data points for low user throughput.

5.2 DNN Model Development

5.2.1 Baseline Model Building

The main objective for the baseline model is to define a benchmark [14] that helps to set the threshold performance whether the model is useful and acceptable. For that matter, defining model baseline accuracy is required. By observing the data distribution of the user throughput class, the *largest class percentage is the baseline accuracy* [14]. An overall accuracy anywhere below this benchmark, least accuracy, would be of practically no use. As a result, the *baseline accuracy* for the model to be developed, in this thesis, is **51.86** % shown in Figure 5.5.

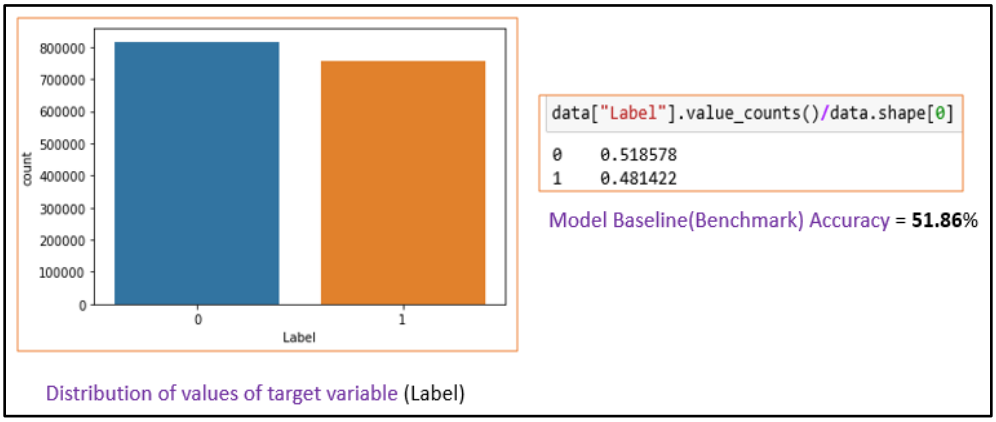


Figure 5. 5 Baseline accuracy definition and distribution of target variables

Based on the baseline accuracy, a baseline model is built and the learning curve is presented as depicted in Figure 5.6. This model was built using 2 hidden layers, 6 neurons per hidden layer, 1 output layer of a single neuron with sigmoid activation function, Adam optimizer, ReLu activation function per hidden layer, epochs = 30, and batch-size = 256.

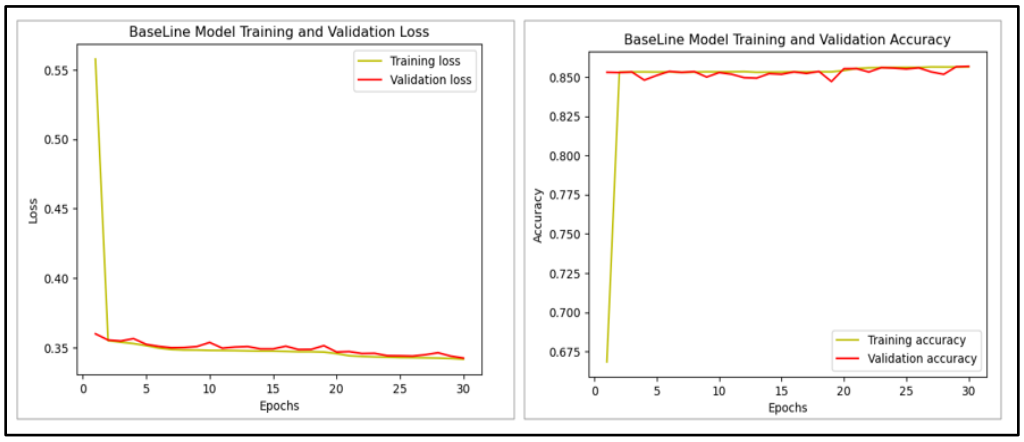


Figure 5. 6 Baseline model training loss and accuracy

As it is observed, the training accuracy and loss are 85% and 35% respectively. It is above the benchmark but it requires more improvement. The learning curve shows that it is not a good model. Therefore, to improve this performance, the hyperparameter optimization is done in the next section.

5.2.2 DNN Model Hyperparameter Tuning

There are two steps to follow to optimize hyperparameters. The first step is to find the optimum number of hidden layers, neurons per hidden layer, and learning rate for Adam optimizer. In this step, the BayesianOptimization algorithm [38] was used, discussed in Chapter 4. Table 5.2 shows

the first step result. This table summarizes the proposed DNN model architecture definition (defining and building the model). Note that the architecture (end-to-end connection) of the model is presented in Appendix 1.

Table 5. 2 First step optimized number of hyperparameters

Layer Type	Number of Neurons	Activation Function
Input Layer	6	--
Hidden Layer 1	16	ReLu
Hidden Layer 2	24	ReLu
Hidden Layer 3	28	ReLu
Hidden Layer 4	12	ReLu
Hidden Layer 5	20	ReLu
Hidden Layer 6	12	ReLu
Hidden Layer 7	32	ReLu
Hidden Layer 8	8	ReLu
Output Layer	1	Sigmoid

According to Equation (3.7), *the number of hidden layers (L-1)*, as shown in Table 5.2, is *8* with different numbers of neurons. For example, hidden layer 1 has 16 neurons, hidden layer 2 has 24 and hidden layer 8 has 8 neurons. When we see in the table, the input layer has 6 neurons which is the same as the number of features (i.e. $N_0 = 6$). Moreover, the output layer has 1 neuron (L is 9). Using Equation (3.8):

$$N_{neurons} = 6 + 16 + 24 + 28 + 12 + 20 + 12 + 32 + 8 + 1 = 159$$

The activation function used in the hidden layers is ReLu, discussed in the previous chapters. Sigmoid activation function has been used for the output layer due to its characteristic for binary classification [13]. In the second step, the optimum number of batch-size and epochs were tuned using the first step output. In this step, the batch sizes 64, 128, 256, 512, and 1024 have been experimented with. In addition, the maximum epoch was set to be 500.

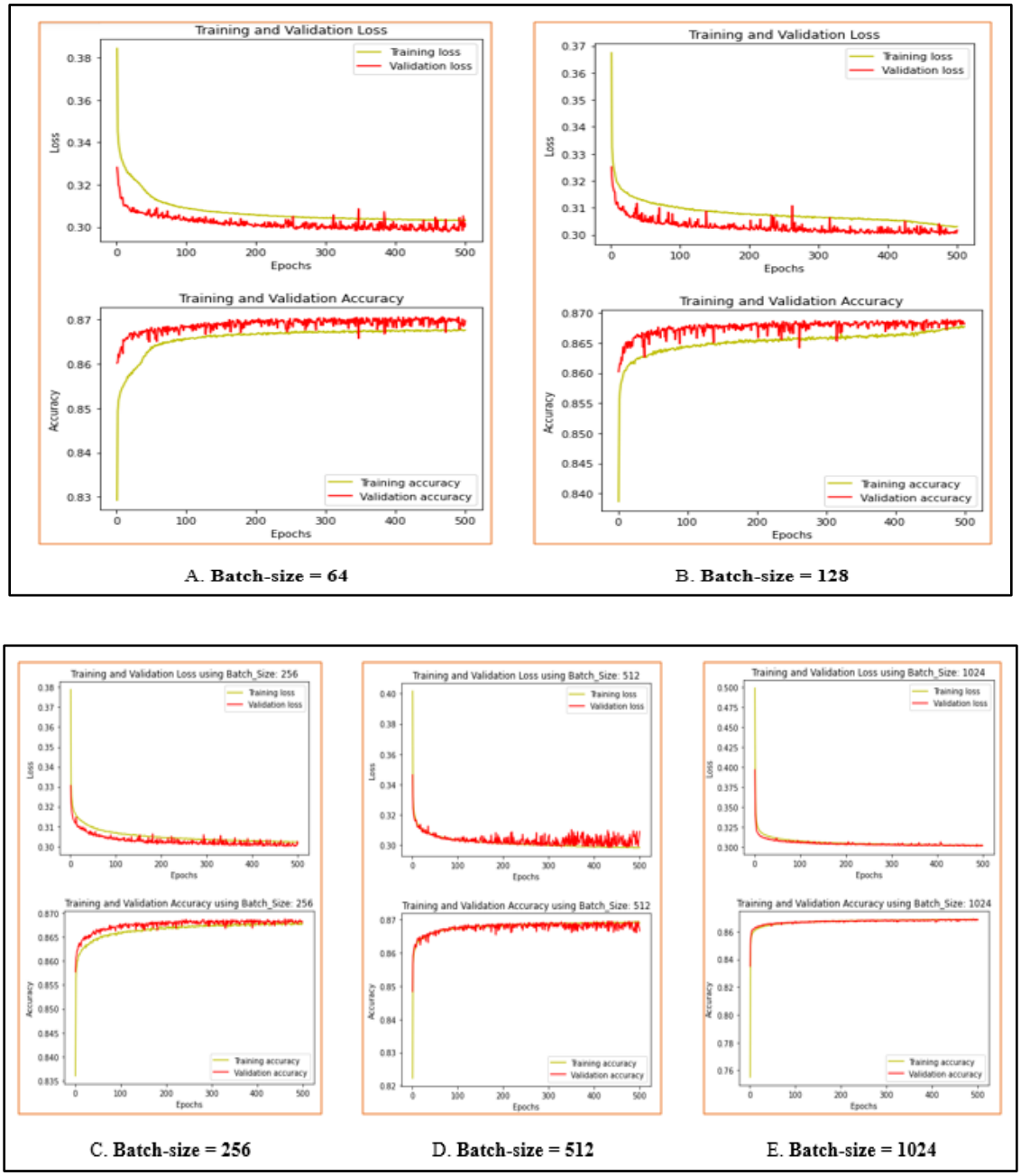


Figure 5. 7 Hyperparameter-tuning loss and accuracy learning curves

As shown in Figure 5.7 (A), for batch size 64, there is a relatively wide separation between training loss and validation loss. This separation is larger compared to other batch sizes. In addition, the training loss is higher than the validation loss. When the batch size was increased from 64 to 128, there is an improvement. By continuously increasing the batch size to 1024, the two plots were almost overlaying. Similarly, the accuracy has been improved as the number of batch sizes increased. As a result, the best model is tuned for a batch size of 1024 and an epoch of 500.

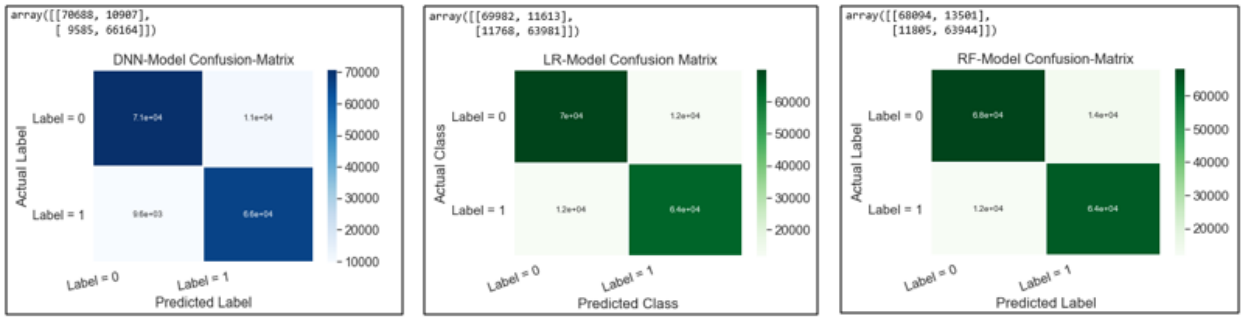
5.2.3 Final DNN Model Performance Evaluation

As observed in Figure 5.7 (E), the final model is the best model with a batch size of 1024 and a max epoch of 500. The model performance showed 86.98% accuracy. The overall hyperparameters used to compile and train the final model are shown in Table 5.3.

Table 5. 3 Proposed DNN model overall hyperparameters

Hyperparameters	Value
Optimizer	Adam
Loss	Binary cross-entropy
Metric	Accuracy
Batch-size	1024
Epochs	500
Validation split	20%
Learning rate	0.0001
Dropout rate	0.05%

Let us see the confusion matrix result to have more detailed information.



A. DNN **B. LR** **C. RF**

Figure 5. 8 Performance evaluation of models using a confusion matrix

By looking at the confusion matrix in Figure 5.8 (A), we have the predicted labels on the x-axis and the actual labels on the y-axis. The blue cells running from the top left to bottom right contain the number of samples that the model accurately predicted (i.e. 70688+66164 = 136,852). The light blue cells contain the number of samples incorrectly predicted (i.e. 9585+10907= 20492). Therefore, the model accuracy (i.e. correctly predicted / (correctly + incorrectly predicted)) is 86.98%. For **reference models,** shown in Figures 5.8 (B) for LR, and (C) for RF, the accuracy

was calculated with the same analogy of the DNN model. LR-Model was 85.14% and that of RF-Model was 83.92%. The models' performance using the AUC_ROC curve is shown in Figure 5.9.

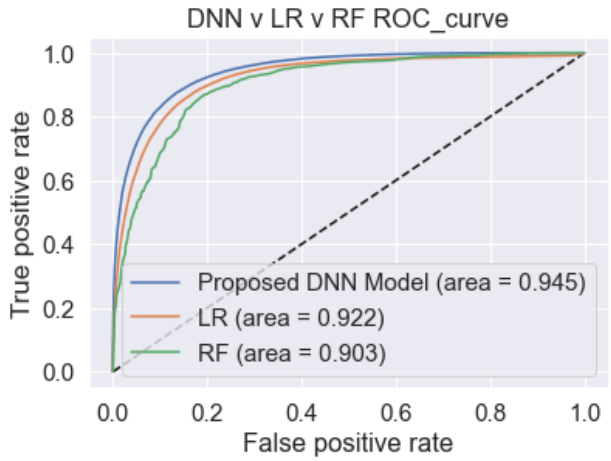


Figure 5. 9 Proposed model comparison with reference models via AUC_ROC curve

In this plot, it is observed that the proposed DNN model has higher performance, 94.5%, than the reference models. RF has a score of 90.3%, which is the least followed by LR with a score of 92.2%. The comparison of models' performance evaluation using all the metrics is shown in Table 5.4.

Table 5. 4 Comparison of models performance evaluation

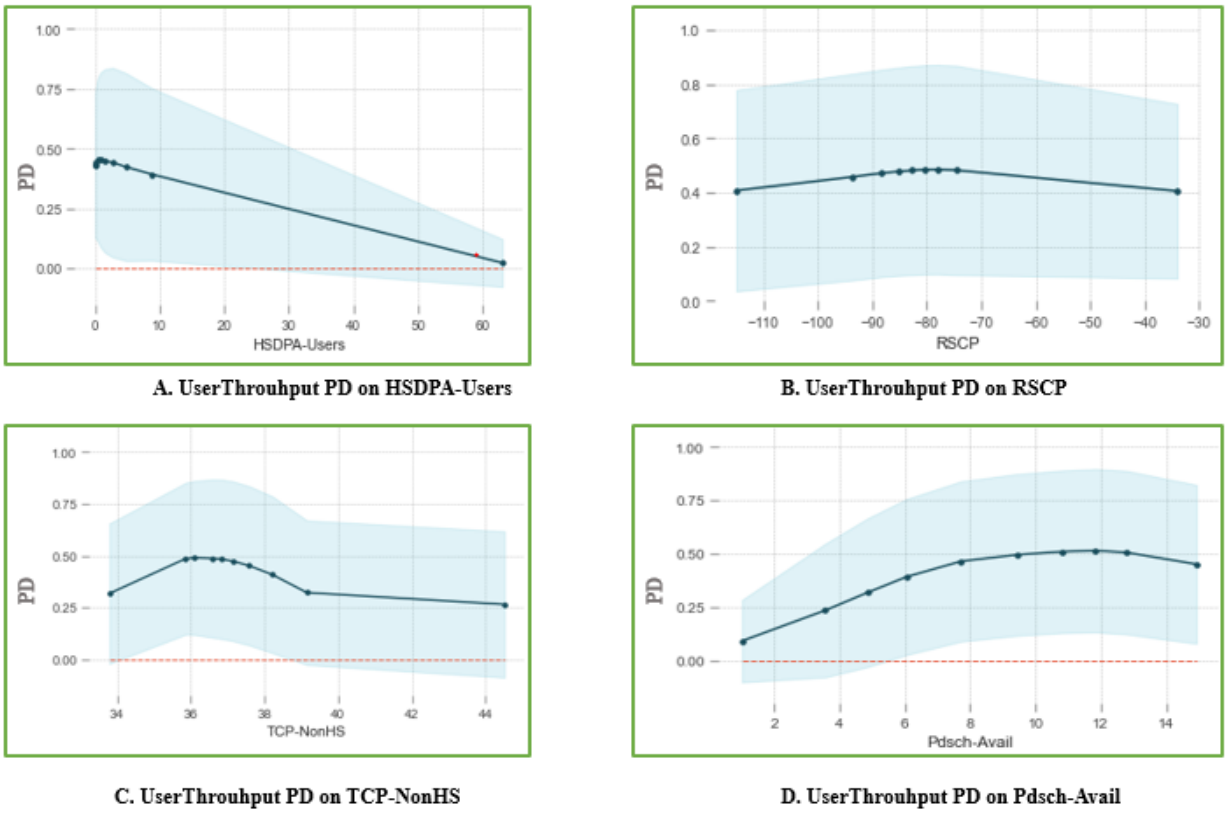
Model	Accuracy (%)	Precision (%)	Recall (%)	F1score (%)	AUC (%)
LR	85	86	86	86	92
RF	84	85	83	84	90
DNN Model	87	88	86	87	95

In this table, it is observed that the proposed DNN model is the best fit compared to the reference models in all the metrics used, except for recall, which is the same performance, recorded as LR of 86 %. The final model summary is presented in Appendix 2. The discussions in the remaining sections and chapters are using this proposed DNN model.

5.3 Causal Effect of Features on User Throughput

The main objective of the dependency plot is to have clarity about the relationship between the causes and the response variable. As clearly presented in Chapter 4, therefore, a partial dependence

plot is used to understand the causal relationship of the input parameters on the response, which is the user throughput in our case. The relationships are shown in Figures 5.10 and 5.11.



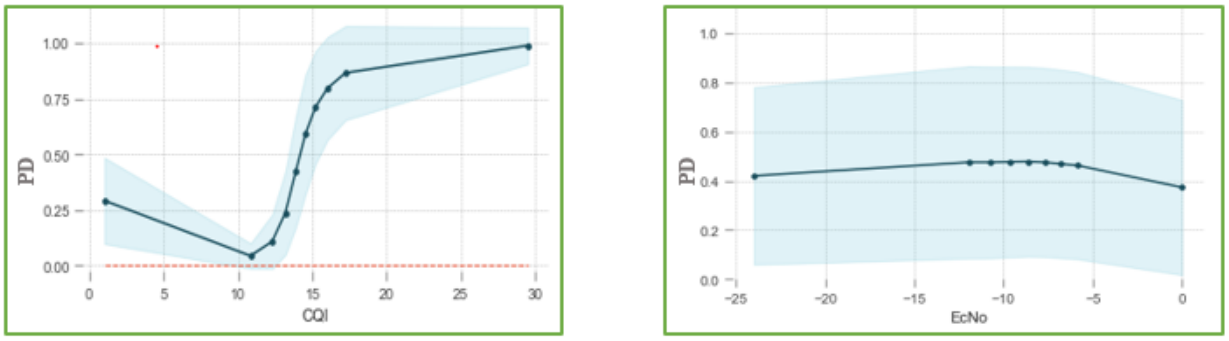
A. UserThrouhput PD on HSDPA-Users

B. UserThrouhput PD on RSCP

C. UserThrouhput PD on TCP-NonHS

D. UserThrouhput PD on Pdsch-Avail

Figure 5. 10 HSDPA-Users, RSCP, TCP-NonHS, and Pdsch-Avail PD plot



E. UserThrouhput PD of CQI

F. UserThrouhput PD of EcNo

Figure 5. 11 CQI and EcNo PD plot

Observation:

In plot (A) of Figure 5.10, the increment in the HSDPA-Users has increased the dependency of user throughput at the very beginning (for fewer users). Then, it decreases continuously to the

value of the maximum number of users supported (i.e. 64). PDP is the prediction probability of user throughput in binary classification. The less the PDP value, the more the throughput is impacted. Therefore, when the number of users was maximum, the throughput is highly impacted almost 0.01 prediction probability for class 1. That means 1 minus 0.01 equals 0.99 prediction probability for class 0.

In plot (B) of Figure 5.10, the dependency of the user throughput versus RSCP is presented. At the very beginning, the increase of RSCP improves the throughput till the RSCP value is around -80 dBm. Unfortunately, this increment could not improve the user throughput forever. The user throughput is negatively dependent on the RSCP from -80 dBm up to -35 dBm. This is possible since when the coverage has improved, more users could be added to the cell and the interference could increase, or else the available code resources are not enough to share.

In plot (C) of Figure 5.10, the impact of TCP-NonHS is presented. As the plot shows, the throughput dependency increases when the power increases till the power reaches 36 dBm. Then, the dependency negatively influences as the power remains to increase. As we know, the power used by HSDPA-Users is the remaining power used by TCP-NonHS from total NodeB transmitted carrier power. In our case, the cell power looks congested when it reached 39 dBm. Then, the PDP plot has continued decreasing steadily.

In plot (D) of Figure 5.10, as the number of simultaneous codes has increased, there was an improvement in the user throughput but it has continued up to the code value 12. Generally, decreasing the number of codes has severely affected the throughput, scares resource.

In plot (E) of Figure 5.11, at the CQI value of the range 10. 5 up to around 17, there was a sharp increase in the PDP value of user throughput. There may be a change in the modulation order from QPSK to 16 QAM or 64 QAM. At these values, mostly there is a change in the MCS (modulation and coding scheme). In plot F of figure 5.11, the impact trained looks as that of RSCP, except the values. For very poor EcNo value, the throughput dependency has shown the bottom line but progressively increases up to -10 dB. After -5.5 dB, the user throughput has again decreased while the EcNo was increasing. It did not continue getting an advantage by increasing EcNo rather it reversed. It may require other parameters like good RSCP.

5.4 Root Cause Explanation Using LIME

As we can see from the causality relationship, it is very difficult to identify the root cause for low user throughput. However, using the PDP, we can assure that each feature has a contribution to the low user throughput. The main issue, here, in the DNN model is unable to reason out why the output is class 0, which is low user throughput. Using LIME, this limitation is solved by identifying the significant features through weight assignment. The higher the weight is, the more the possibility of being the root cause of a specific outcome.

5.4.1 Case Analysis

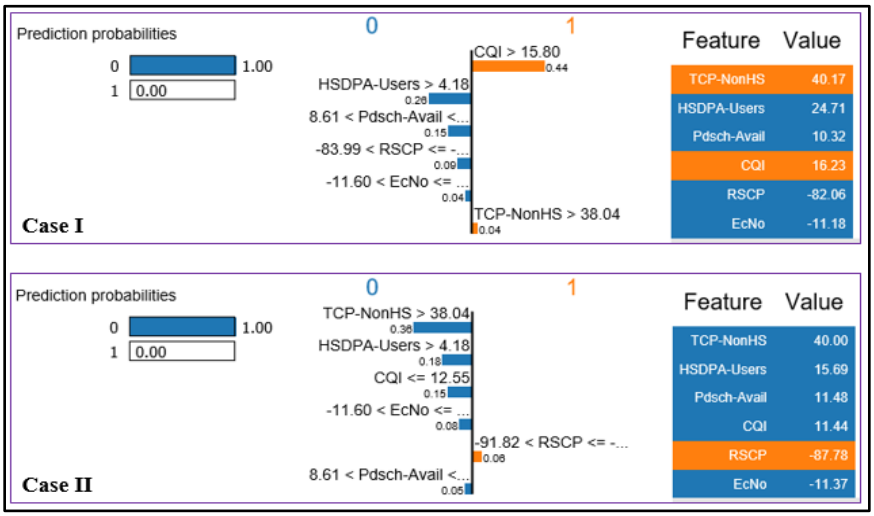


Figure 5. 12 Root cause explanation using LIME for the cases I and II

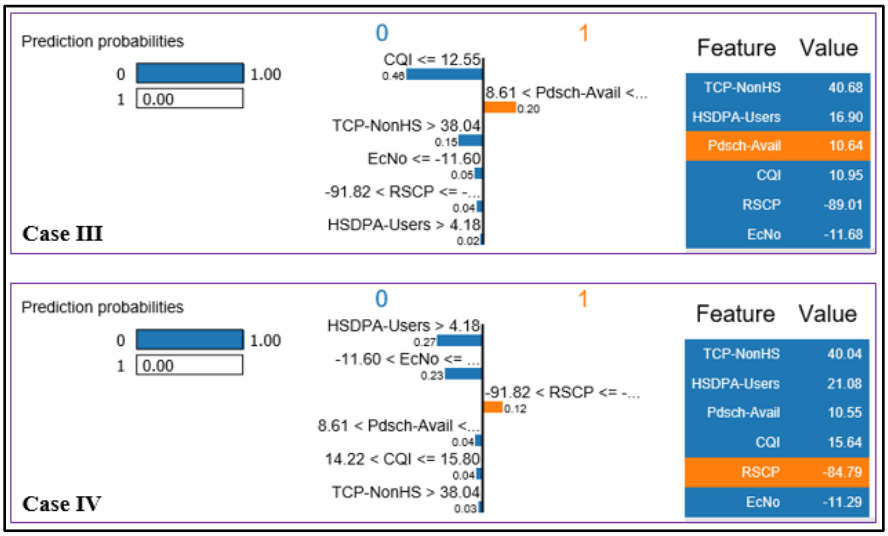


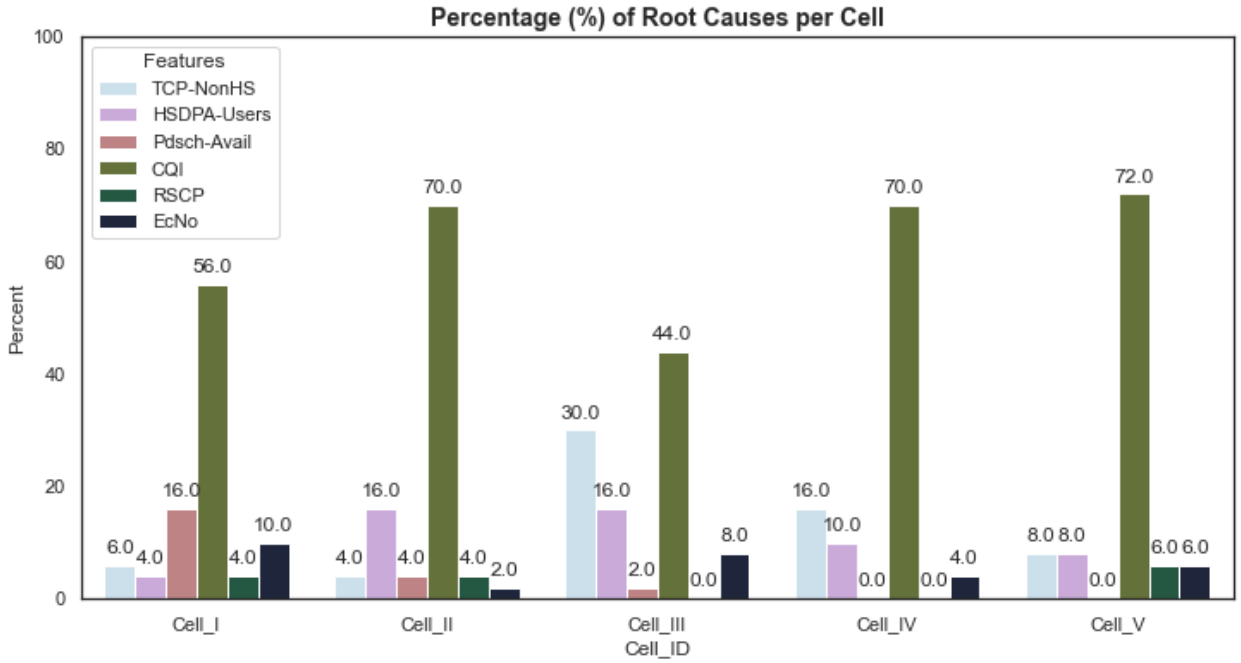
Figure 5. 13 Root cause explanation using LIME for the cases III and IV

In Figures 5.12 and 5.13, four cases of LIME-output are presented. The prediction probability in all of the cases is 100 % of class 0. As shown in the figures, the actual values for each feature are displayed on the right side of each plot. Given these values, the root cause for case I and case IV is HSDPA-Users with the weight assigned 0.26 and 0.27 respectively. The number of users supported is 24.71 for Case I and 21.08 for case IV. According to plot (A) in Figure 5.10, these numbers are significant to cause the user throughput to be poor. When we see the result in Case II, the root cause for class 0 outcome is TCP-NonHS with a weight of 0.36 that is the highest compared to other features. Unfortunately, RSCP is not the reason for low user throughput (class 0), in Case II. In Case III, the root cause is CQI because it has the highest weight contribution that is 0.46. The second root cause is TCP-NonHS with a weight of 0.15. From plot (E) in Figure 5.11, the actual CQI value is 10.95 (significant), and thus its contribution is high. Nevertheless, when we see the Pdsch-Avail code the contribution is against class 0 rather it is contributing for class 1. This is because the available code is 10.64 out of 15.

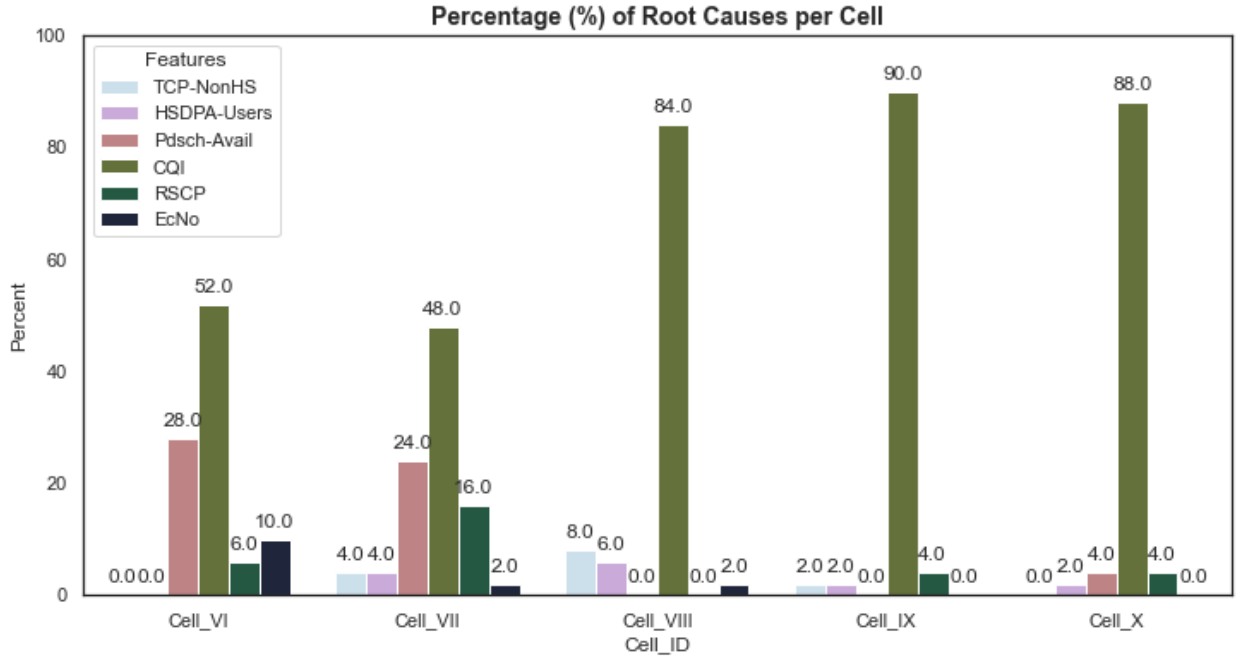
To sum up, the root causes are different for different samples. LIME is a very useful algorithm for local RCA investigation of the low user throughput. To find the root cause globally (all of the samples per cell), each instance needs to be investigated for RCA. Then calculate the percentage to determine the largest share.

5.4.2 Low User Throughput RCA on Selected Cells

The result of RCA for poor throughput performance is shown in Figure 5. 14. In this figure, the percentage of root causes has been evaluated. It is observed that, in all the cases, CQI has the largest share for low user throughput contribution. However, this share has varied from one cell to another. 90 % of the root cause has been recorded from Cell_IX. 88 % of root cause has been observed from Cell_X. Comparatively the share is followed by 84% from Cell_VIII. From the satellite image in Figure 4.3, Bahir Dar Institute of Technology and Gamby general hospital are covered by these cells where the user throughput is mostly impacted due to poor CQI values.



A. Root cause percentages of Cell_I to Cell_V



B. Root cause percentages of Cell_V to Cell_X

Figure 5. 14 Case Analysis on Selected Cells

6. Conclusion and Recommendation

6.1 Conclusion

The main objective of the study was to find the root cause of low user throughput by developing a causal model using DNN. As a result, a DNN model with optimum hyperparameters has been trained. By providing feature importance to identify the most significant feature, the LIME algorithm has enabled the DNN model for root cause analysis. The dataset was obtained from 445 cells of 76 NodeBs in Bahir Dar city as a case study to train and test the model. The LIME explainer was trained with the DNN model to explain why the model classified an output as low user throughput. This is because the DNN model is opaque in decision-making. Since the RCA analysis is based on local explanation (instance-based), the root causes of low user throughput are not unique. CQI is the most important cause detected in the city from 01-16-2021 to 02-28-2021, based on the proportion of all root causes evaluated on 10 cells (i.e. experiencing low user throughput). This RCA method is highly useful for prioritizing mitigation measures and gaining a deeper understanding of the complexities of causal factors for throughput degradation.

6.2 Recommendation

1. To better support optimization to improve channel quality (CQI), it is recommended to investigate the UE category distribution and user behavior. Moreover, the improvement of channel quality can improve the throughput as shown in Figure 5.11 (E). Nevertheless, poor CQI may come from both NodeB and UE sides.
2. The proposed model is dependent on local explanations. It is very complex to generate root causes per sample of large data sizes. As a result, it is recommended to build automating mechanism to simplify this complexity.
3. The accuracy of the DNN model using MLP is limited as the number of neurons and layers are large. So, it is recommended to further choose advanced deep neural network algorithms, such as CNN. Furthermore, MCS (modulation and coding scheme), latency, and more features can be included in future works.

References

[1] "2012 EFY (2019/20) Annual Business Performance Summary Report," Ethio Telecom, Addis Ababa, Ethiopia, 2020. [Online]. Available: https://www.ethiotelecom.et/ethio-telecom-2012-efy-2019-20-annual-business-performance-summary-report/.

[2] J. P. Romero, O. Sallent, R. Agusti, and M. A. Diaz-Guerra, "RRM Algorithms," in *Radio Resource Management Strategies in UMTS*, 1st ed., Wiley, 2005, pp. 177–301.

[3] P. Tapia, J. Liu, Y. Karimli, and M. Feuerstein, "HSPA Radio Network Planning and Optimization.," in *HSPA Performance and Evolution: A Practical Perspective*, 1st ed., USA: Wiley, 2009, pp. 71–115.

[4] H. Holma, A. Toskala, and P. Tapia, *HSPA+ Evolution to Release 12: Performance and Optimization*, 1st ed. United Kingdom: Wiley, 2014.

[5] J. Laiho, A. Wacker, and T. Novosad, *Radio Network Planning and Optimisation for UMTS*, 2nd ed. Wiley, 2006.

[6] A. R. Mishra, *Advanced Cellular Network Planning and Optimisation: 2G/2.5G/3G...Evolution to 4G*, 1st ed. UK: Wiley, 2007.

[7] "RAN17.1 Capacity Monitoring Guide (BSC6910-based)," 05, 2017. [Online]. Available: https://www.scribd.com/document/505114269/RAN17-1-Capacity-Monitoring-Guide-BSC6910-Based-Draft-A-PDF-EN.

[8] "RCATechniques with Examples," *Invensis Learning*, 2021. https://www.youtube.com/watch?v=oMZg-Q8EBek&t=676s (accessed May 18, 2021).

[9] "How to Perform RCA: 4 Easy Steps." https://imdcr.com/how-to-perform-rca-4-easy-steps/ (accessed Apr. 20, 2021).

[10] M. Solé, V. Muntés-Mulero, A. I. Rana, and G. Estrada, "Survey on Models and Techniques for Root-Cause Analysis," *CoRR*, pp. 1–18, 2017.

[11] G. Xu, T. D. Duong, Q. Li, S. Liu, and X. Wang, "Causality Learning: A New Perspective for Interpretable Machine Learning," *CoRR*, p. 8, 2020.

[12] R. Abera, "Quality of Experience Evaluation for Addis Ababa UMTS Enterprise Data Customers," M.S. thesis, Sch of ECE, AAiT, AAU Univ, A.A, Ethiopia, 2018.

[13] J. J. Moolayil, *Learn Keras for Deep Neural Networks*, 1st ed. Apress, 2019.

[14] J. Moolayil, "Deep Neural Networks for Supervised Learning: Classification," in *Learn Keras for Deep Neural Networks: A Fast-Track Approach to Modern Deep Learning with Python*, no. September, Apress, 2019, pp. 101–135.

[15] U. Michelucci, "Feedforward Neural Networks," in *Applied Deep Learning: A Case-Based Approach to Understanding Deep Neural Networks*, 1st ed., Apress, 2018, pp. 83–136.

[16] N. K. Manaswi, "Multilayer Perceptron," in *Deep Learning with Applications Using Python*, 1st ed., Apress, 2018.

[17] C. C. Aggarwal, *Neural Networks and Deep Learning*, 1st ed. NY, USA: Springe, 2018.

[18] "Deep Learning," *mathworks*. https://www.mathworks.com/discovery/deep-learning.html..html (accessed Jul. 10, 2021).

[19] "Machine Learning & Deep Learning Fundamentals," *DEEPLIZARD*, 2017. https://deeplizard.com/learn/video/hfK_dvC-avg (accessed May 09, 2021).

[20] J. Pearl, *Causality: Models, Reasoning and Inference*, 2nd ed. Cambridge University Press, 40 W. 20 St. New York, NY, United States, 2009.

[21] M. Strevens, "Causality Reunified," *Erkenn*, vol. 78, pp. 299–320, 2013.

[22] P. Szilágyi and S. Nováczki, "An Automatic Detection and Diagnosis Framework for Mobile Communication Systems," *IEEE Trans. Netw. Serv. Manag.*, vol. 9, no. 2, pp. 184–197, 2012.

[23] M. M. Mampaka and M. Sumbwanyambe, "Poor Data Throughput RCAin Mobile Networks Using Deep Neural Network," *2019 IEEE 2nd Wirel. Africa Conf.*, pp. 1–6, 2019, [Online]. Available: https://ieeexplore.ieee.org/document/8843409.

[24] M. Carletti, C. Masiero, A. Beghi, and G. A. Susto, "Explainable Machine Learning in Industry 4.0: Evaluating Feature Importance in Anomaly Detection to Enable Root Cause Analysis," *2019 IEEE Int. Conf. Syst. Man Cybern.*, pp. 21–26, 2019, [Online]. Available: https://ieeexplore.ieee.org/document/8913901.

[25] A. Carrillo, L. F. Cantú, and A. Noriega, "Individual Explanations in Machine Learning Models: A Survey for Practitioners," *CoRR*, pp. 1–13, 2021, [Online]. Available: https://arxiv.org/pdf/2104.04144.pdf.

[26] J.-X. Mi, A.-D. Li, and L.-F. Zhou, "Review Study of Interpretation Methods for Future Interpretable Machine Learning," *IEEE Access*, vol. 8, pp. 191969–191985, 2020, [Online].

Available: https://ieeexplore.ieee.org/document/9234594.

[27] J. Ahmed, A. do Nascimento, C. Kilinc, D. Pan, J. R. i Riu, and J. Gustafsson, "Using Blackbox ML Techniques to Diagnose QoE Problems for an IPTV Service," in *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, 2020, pp. 1–8, [Online]. Available: https://ieeexplore.ieee.org/document/9110375.

[28] N. K. Manaswi, *Deep Learning with Applications Using Python: Chatbots and Face, Object, and Speech Recognition With TensorFlow and Keras*. Apress, 2018.

[29] "UMTS Protocol Analyzer." https://www.gl.com/umts-network-protocol-analyzer.html (accessed Jun. 15, 2021).

[30] U. Qureshi, "UMTS system architecture, protocols & processes," 2014. [Online]. Available: https://www.slideshare.net/mushtaqyousafzai1/umts-system-architecture-protocols-processes.

[31] C. Chioariu, "QoS in UMTS," *Internetworking Res. Exp.*, pp. 1–7, 2004, [Online]. Available: https://www.academia.edu/35564504/QoS_in_UMTS.

[32] "3gpp ts 25.133 Technical Specification,Release 4," 2006.

[33] R. Kwan, P. H. J. Chong, E. Poutiainen, and M. Rinne, "The Effect of Code-multiplexing on the High Speed Downlink Packet Access (HSDPA) in a WCDMA Network," *2003 IEEE Wirel. Commun. Networking, 2003. WCNC 2003.*, vol. 3, pp. 1728–1732, 2003, [Online]. Available: https://ieeexplore.ieee.org/document/1200648.

[34] C. Albon, *Machine Learning with Python Cookbook*, 1st ed. O'Reilly, 2018.

[35] J. Brownlee, "Logistic Regression for Machine Learning," *machinelearningmastery.com*, 2016. https://machinelearningmastery.com/logistic-regression-for-machine-learning/ (accessed Mar. 16, 2021).

[36] "Optimizing Hyperparameters Using the Keras Tuner Framework," *marktechpost.com*. https://www.marktechpost.com/2021/03/20/optimizing-hyperparameters-using-the-keras-tuner-framework/ (accessed Apr. 13, 2021).

[37] "KerasTuner," *Keras*. https://keras.io/keras_tuner/ (accessed Mar. 14, 2021).

[38] "BayesianOptimization Tuner," *Keras*. https://keras.io/api/keras_tuner/tuners/bayesian/#bayesianoptimization-class (accessed Jun. 26, 2021).

[39] "Understanding AUC - ROC Curve." https://towardsdatascience.com/understanding-auc-

roc-curve-68b2303cc9c5 (accessed Feb. 27, 2021).

[40] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should i trust you?' Explaining the predictions of any classifier," *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, pp. 1135–1144, 2016, [Online]. Available: https://arxiv.org/pdf/1602.04938.pdf.

[41] "Decrypting your Machine Learning model using LIME." https://towardsdatascience.com/decrypting-your-machine-learning-model-using-lime-5adc035109b5 (accessed Apr. 18, 2021).

[42] C. Molnar, "Interpretable Models," in *Interpretable Machine Learning*, 2019, p. 318.

[43] C. Molnar, "Model-Agnostic Methods," in *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable*, 1st ed., Leanpub, 2019, pp. 110–188.

[44] ETSI TS 125.133, "Requirements for Support of Radio Resource Management (FDD),"

[45] H. Holma and A. Toskala, "High-Speed Downlink Packet Access," in *WCDMA FOR UMTS: HSPA Evolution and LTE*, Fifth., vol. 148, Nokia Siemens Networks, Finland: Wiley, 2010, pp. 353–388.

[46] L. Pierucci, A. Romoli, R. Fantacci, and D. Micheli, "An Optimized Neural Network for Monitoring Key Performance Indicators in HSDPA," *IEEE Int. Symp. Pers. Indoor Mob. Radio Commun. PIMRC*, no. June 2016, pp. 2041–2045, 2010.

[47] Fchollet, "Introduction to Keras for Researchers," *keras.io*, 2020. https://keras.io/getting_started/intro_to_keras_for_researchers/.

[48] "Automated Hyperparameter Tuning with Keras Tuner and TensorFlow 2.0," *medium.com*. https://medium.com/analytics-vidhya/automated-hyperparameter-tuning-with-keras-tuner-and-tensorflow-2-0-31ec83f08a62 (accessed Jun. 10, 2021).

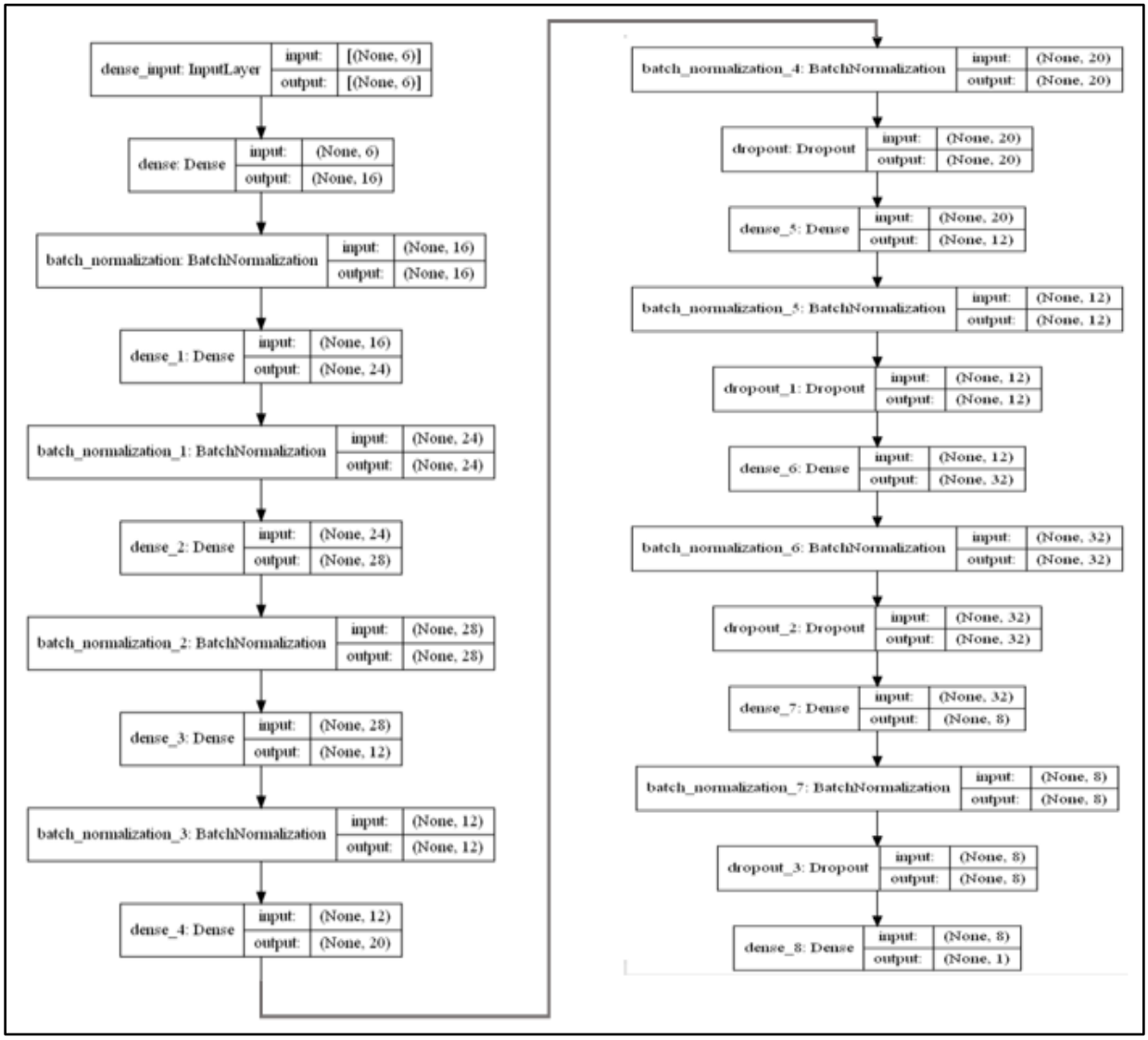
[49] "Understanding Logistic Regression in Python," *datacamp.com*. https://www.datacamp.com/community/tutorials/understanding-logistic-regression-python (accessed Jul. 20, 2021).

[50] "Ensemble Methods in Machine Learning: What are They and Why Use Them?" https://towardsdatascience.com/ensemble-methods-in-machine-learning-what-are-they-and-why-use-them-68ec3f9fef5f (accessed Feb. 07, 2021).

[51] "PDPbox." https://pdpbox.readthedocs.io/en/latest/# (accessed Jun. 09, 2021).

[52] "Partial Dependence Plot (PDP)." https://christophm.github.io/interpretable-ml-book/pdp.html (accessed May 19, 2021).

Appendix 1: Proposed DNN Model Architecture



Appendix 2: Model Summary

Layer (type)	Output Shape	Param #
dense_1 (Dense)	(None, 16)	112
batch_normalization_1	(Batch (None, 16)	64
dense_2 (Dense)	(None, 24)	408
batch_normalization_2	(Batch (None, 24)	96
dense_3 (Dense)	(None, 28)	700
batch_normalization_3	(Batch (None, 28)	112
dense_4 (Dense)	(None, 12)	348
batch_normalization_4	(Batch (None, 12)	48
dense_5 (Dense)	(None, 20)	260
batch_normalization_5	(Batch (None, 20)	80
dropout_1 (Dropout)	(None, 20)	0
dense_6 (Dense)	(None, 12)	252
batch_normalization_6	(Batch (None, 12)	48
dropout_2 (Dropout)	(None, 12)	0
dense_7 (Dense)	(None, 32)	416
batch_normalization_7	(Batch (None, 32)	128
dropout_3 (Dropout)	(None, 32)	0
dense_8 (Dense)	(None, 8)	264
batch_normalization_8	(Batch (None, 8)	32
dropout_4 (Dropout)	(None, 8)	0
dense_9 (Dense)	(None, 1)	9
Total params: 3,377		
Trainable params: 3,073		
Non-trainable params: 304		

Note: Non-trainable params are parameters used for batch_normalization; trainable params are the number of weights and bias that constitute the 3073 parameters of the overall neuron-to-neuron connection according to Equation (3.9).

Appendix 3: Publishable Manuscript

Root Cause Analysis of Unsatisfactory User Throughput in UMTS for Bahir Dar City

Yonas Alemayehu Matebie
Wireless Network Operation and Maintenance
Ethio Telecom
Addis Ababa, Ethiopia
yonasalem.12@gmail.com

Yihenew Wondie Marye
Addis Ababa Institute of Technology
Addis Ababa University
Addis Ababa, Ethiopia
yihenew.wondie@aait.edu.et

***Abstract* — Nowadays, the increase in size and complexity of current cellular networks is complicating their operation, maintenance, and optimization tasks. The end-to-end user experience in terms of throughput has increased dramatically. In the meantime, such networks have become more susceptible to failure. Bahir Dar city's data service is mainly supported by a universal mobile telecommunications system. It is challenging to locate the root causes of low user throughput manually for such a system, as it requires expertise and longtime analysis. This thesis proposes low user throughput root cause analysis (RCA) using a deep neural network (DNN) based on multilayer perceptron (MLP). The local interpretable model-agnostic explanation (LIME) technique has been used to enable DNN for RCA of low user throughput by providing the feature's importance. Furthermore, the impact of features on the model outcome is studied to understand the causal effect on the user throughput using a partial dependency plot. The effectiveness of the model has been validated using a test dataset and compared with reference models. Accordingly, the proposed DNN model has performed better compared to these models. The result of the proposed model performance evaluation is the accuracy of 87%, the precision of 88%, recall of 86%, F1-score of 87%, and area under the curve (AUC) of 95%. An RCA conducted on 10 selected cells (case study) with a poor throughput of 50 samples per cell has shown the root cause is due to the channel quality indicator (CQI) parameter, which implies poor channel quality reported by users. It is recommended to optimize the network to improve the channel quality.**

Keywords—RCA, causal effect, partial dependence, feature importance

I. INTRODUCTION

In Ethiopia, mobile data service usage has shown considerable growth after different network expansion projects carried out in the last consecutive years. With the advent of smartphones, more and more users are using data- intensive applications, such as WhatsApp, Instagram, Facebook, and the like. In essence, this has resulted in the evolution of new mobile data services and complexity in the management of the networks [1]. In Bahir Dar city, the UMTS network is the primary provider of mobile data services. The quality of experience (QoE), which is the measurement of the user's perception of the quality of service (QoS) supplied, is one of the most important aspects of mobile network management. A key metric used to assess service quality is throughput, measured in megabits per second (Mbps) or kbps. It's defined as the ratio of total correctly transferred bits from a source to a destination divided by the total time taken. Due to the numerous contributing elements, determining low user throughput is a difficult operation. User throughput degradation can be caused by either network performance or user issues.

Even if high-speed packet access plus (HSPA+) is enabled in the city to support good user throughput, investigating the main causes of poor user throughput during operation and maintenance is challenging. Root cause analysis is the main objective of this thesis for identifying or localizing the root cause (s) of the low user throughput. Ethio telecom monitors the performance of network elements and average user throughput using a network performance report system. The advantage is that the information extracted from rich statistical data can be used to study the possible causes of the low user throughput. In the city, RCA is carried out in recurring activities that take a long time and exhausting daily. The measurement reports on user throughput include radio conditions (RSCP, EcNo), required resources (available code, CE, and power), and channel quality (CQI).

Finding solutions for such multidimensional and large datasets is a challenging task. In such a case, simple linear transformation algorithms are not suitable. While decades of research have yielded a large number of algorithms and techniques for performing root cause analysis in a variety of fields [2], usability and adaptability to the growing complexity of throughput analysis remain a limitation, where scalability and actual interaction become critical. The complex and dynamic behavior of user throughput interactions with input factors (multivariate), in particular, will necessitate advanced root cause

analysis capable of precisely synthesizing to make decisions. A deep neural network [3], nowadays, is a good approximation for nonlinear transformation. As a result, root cause analysis for low user throughput based on a deep neural network with LIME [4] explanation is introduced for this study. RCA is all about causality and explanation [5], [6].

A. *Related works*

The basic concepts of root cause analysis (RCA) are causation and explanation, according to a survey [2]. Causality is the study of the cause and effect relationship between factors and response variables, whereas explanation is the justification for the response caused by those factors. It's important, especially in the case of explanation, because an explanation is often RCA's desired outcome. There are two types of RCA models, according to [2], deterministic and probabilistic. The known facts and inferences used in deterministic models are both certain. Neural networks (nets) are one type of this family. Probabilistic models, such as Bayesian networks, on the other hand, are capable of dealing with uncertainty with a requirement for a threshold to find the most probable cause.

Automated systems for finding and diagnosing cells, not just in complete failures but also with decreasing performance, are becoming increasingly significant as commercial cellular networks become more sophisticated. Root cause investigation of observed anomalies is time-consuming and is mostly done manually, if at all; in most situations, operators just reset faulty cells. A framework was developed in a study [3] that could detect anomalies and locate the most likely root cause of not just serious problems in a cell but also service degradations. Detection is based on monitoring radio measurements and other performance indicators and comparing them to their regular behavior captured by profiles, which are likewise produced automatically without the need for threshold or manual calibration, according to [3]. The diagnosis was intervened in [3], depending on reports of previous fault-cases by identifying and learning their characteristic impact on different performance indicators.

A study [8] was conducted to handle enormous amounts of data and the complexity of new data services in mobile networks, comparable to [3]. Their studies are similar in that they rely on the dataset from monitored radio measurements and other performance indicators. The inquiry in [8] particularly, was to determine the root cause of poor data throughput in mobile networks, notably in UMTS. Deep learning techniques have gained popularity, according to [8] because they scale well with enormous data volumes and make efficient use of computational power to train models. Although it is advantageous to achieve high accuracy using deep neural networks (DNNs) for nonlinear approximation of complicated issues, such as poor data throughput with associated causal components, the decision-making process is an opaque and black box. As a result, the author in [8] employed local interpretable model-agnostic explanations (LIME) as a black box explainer for feature importance evaluation to enable DNN for root cause analysis.

Due to the complexity of the problem at hand for root cause analysis of low user throughput, this article follows the same approach as in [8]. However, the causal factors are limited to measurement reports of cell level and user equipment (UE), not including the core network. The objective is to narrow the scope and gain a better understanding of cell performance issues to user experience in the context of the average user throughput. The remainder of this paper is organized as follows: Section II describes the proposed approach; section III presents results and discussions; section IV concludes the study.

II. Proposed Approach

A. *System Model*

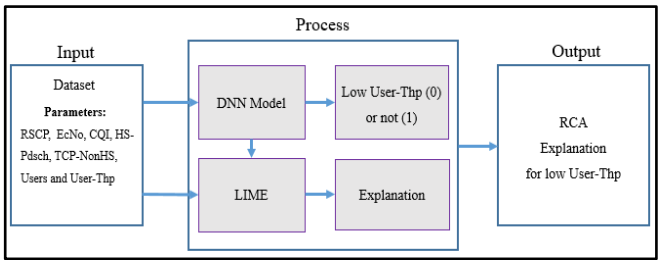


Fig. 1. Proposed RCA system model.

In Fig. 1, the proposed model has three parts: *Input*, which has a dataset with overall parameters; *Process* with DNN model and LIME; and *Output* for RCA explanation of low user throughput. DNN model has two outputs: class 0 for low user throughput and class 1 for normal user throughput. LIME generates an explanation for low user throughput to enable the DNN model for root cause analysis by providing the feature's importance. Since the system model shown in Fig. 1 is very condensed, the end-to-end system process is depicted in Fig. 2 with three main tasks: data preprocessing; DNN model building, and root cause explanation.

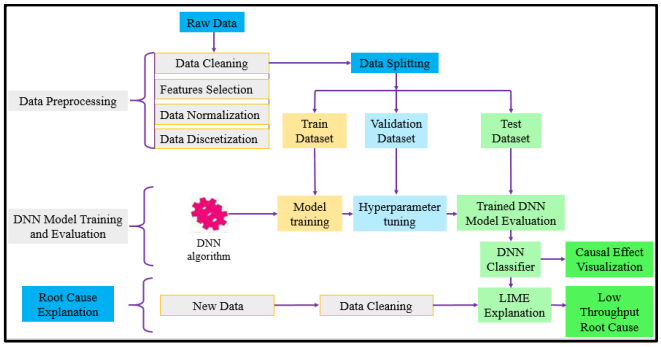


Fig. 2 End-to-end low user Throughput RCA process.

The subsections are organized based on the end-to-end RCA process shown in Fig. 2

B. *Dataset and Experiment Setup*

To build a causal model for root cause analysis of the user throughout, enough data has to be collected. The dataset used, 1.57 million samples, for this research was collected from Ethio telecom's network performance data measurement report (PRS tool) from January 1, 2021, up to May 31, 2021, on an hourly

basis per cell. This dataset was collected from 76 NodeBs of 445 cells in Bahir Dar city. The tools implemented were one Lenovo laptop (8 GB RAM, 500 GB hard disk), python, and Microsoft excel. The dataset has six input parameters called features, which are causal factors for low user throughput, which are outcomes of correlation analysis. The description of the parameters is as shown in Table 1. Except the User-Thp, all the parameters are features, which are input parameters for the proposed model. The User-Thp is the response variable or the target to train the DNN model.

TABLE 1. PARAMETERS DESCRIPTION

Parameters	Description	Units & range
TCP-NonHS	TCP power for non-high speed services	(0 to 45) dBm
HSDPA-Users	Number of users of HSDPA	0 to 64
Pdsch-Avail	Available HS-PDSCH code	0 to 15
CQI	Mean CQI value reported by UE	0 to 30
RSCP	Mean RSCP value reported by UE	(-115 to -25) dBm
EcNo	Mean EcNo value reported by UE	(-25 to 0) dB
User-Thp	Average User Throughput	Kbps

C. Data Preprocessing

To produce a good model, the dataset must be quality data. As a result, the raw dataset has to be cleaned from noisy data, missing values and outliers. In data preprocessing, there are very important tasks.

- Data cleaning: improves the quality of data by missing value handling through interpolation technique. Outliers are treated by the interquartile range (IQR) method.
- Features selection: the statistical data collected has various types, so to select the most influential parameters a correlation analysis (Pearson correlation) is used.
- Normalization: is a scaling technique in which values are shifted and rescaled so that they end up ranging between 0 and 1. The goal of normalization is to transform features to be on a similar scale. This improves the performance and training stability of the model. As it is clearly shown in Table 1, the dataset contains different units and diverse ranges. Min-Max scaling is used to have a uniform range. Mathematically:

$$\text{MinMax Scaling} = \frac{(X - X_{min})}{(X_{max} - X_{min})} \quad (1)$$

Where Xmax and Xmin are the maximum and the minimum values of the feature (X) respectively.

- Data discretization: since the throughput is continuous in value and the objective is to identify the low user throughput from the rest, converting this variable to binary is required. This binary class is composed of low user throughput (0) and not low user throughput (1). According to a study [9], on an optimized neural network for monitoring key performance indicators in HSDPA, average user throughput is low for less than 1Mbps.
- Data splitting: prepared data has to be split into training, validation, and testing datasets. The splitting ratio is 70%, 20%, and 10% for the training dataset, validation dataset, and test dataset respectively.

D. DNN Model Development

The DNN (deep neural network) refers to the deeper structure of multi-layered neural networks as an extension of the multi-layer perceptron (MLP) [10] with more than one hidden layer. A neural network (artificial) is a computing system [8] and [11], which is composed of a collection of connected units called neurons that are organized into what we call layers and thus building DNN. A neuron takes input signals, can make some sort of transformation with activation functions, and provides an output. This building block element of deep learning is formulated as in (2).

$$y(x) = f\left(\sum_{i=1}^{n}(x_i w_i) + b\right) \quad (2)$$

With x_i as a set of inputs, w_i as a set of weights, n as the number of observations, b as bias value, f as the activation function, and y(x) as the output.

Let us see a typical neural network architecture, in Fig. 3, used partially (input layer and output layer is similar to the proposed model) for this article. Deep learning techniques are required to ensure hyperparameters tuning to model complex data during the training process [12]. For this article, important hyperparameters are the number of hidden layers, neurons per hidden layer, activation function, learning rate, batch size, epochs, dropout rate, optimizer, and loss function.

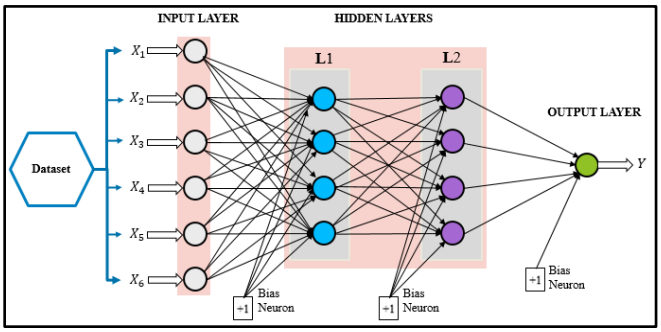


Fig. 3 One input layer, one output layer, and two hidden layers architecture.

To build the model, Keras [13] is utilized. It allows defining, training, and evaluating neural network models. The input and output layers consist of input neurons and output neuron, respectively, and in the middle of them found hidden

layers (L1 and L2) as shown in Fig. 3. Since the features are six, the input layer requires six neurons with no activation function. The problem is binary classification and therefore one neuron with a sigmoid activation function is required. However, the hidden layers are determined by hyperparameter optimization using KerasTuner [14]. There is no specified or pre-defined way of choosing these hyperparameters. The way of choosing the parameters was done in two steps. First, determine the overall network architecture by finding an optimum number of hidden layers and neurons per layer. Table 2 shows the optimum number of layers, neurons, and activation function tuned. Second, find the remaining parameters by using already known architecture. As a result, the overall hyperparameters required are depicted in Table 3.

TABLE 2. FIRST STEP OPTIMIZED NUMBER OF HYPERPARAMETERS

Layer Type	Number of Neurons	Activation Function
Input Layer	6	--
Hidden L1	16	ReLu
Hidden L2	24	ReLu
Hidden L3	28	ReLu
Hidden L4	12	ReLu
Hidden L5	20	ReLu
Hidden L6	12	ReLu
Hidden L7	32	ReLu
Hidden L8	8	ReLu
Output Layer	1	Sigmoid

TABLE 3. PROPOSED DNN MODEL OVERALL HYPERPARAMETERS

Hyperparameters	Value
Optimizer	Adam
Loss	Binary cross-entropy
Metric	Accuracy
Batch-size	1024
Epochs	500
Validation split	20%
Learning rate	0.0001
Dropout rate	0.05%

E. Model Performance Evaluation

Metrics are the functions that are used to evaluate the model's performance on an unseen dataset, commonly known as the validation dataset. They are used to validate the test results while reporting [15]. In this study, the DNN model must be able to distinguish low user throughput data points from normal data points. Moreover, the final goal is to make sure the unseen dataset is classified based on the input features. Since the problem type is binary classification and the main task of the DNN model developed is to do so, the metrics used in this study are accuracy, confusion matrix, recall, precision, F1-score, and AUC-ROC curve.

F. Features Causal Effect on User Throughput

The model outcome is user throughput from the real world live data collected and the relationship is causal. As a result, visualization using a partial dependence plot (PDP) [16] uncovers the complexity of the model developed. As introduced, RCA is all about causality and explanation. PDP helps to visualize the dependency of the model output on the input parameters considering one or two features. This helps to understand the causal-effect relationships between input and model-output.

G. Root Cause Explanation Using LIME

For deep neural networks, which can be represented by complex mathematical equations, it is very difficult to understand why the model generalizes an outcome. This article utilizes LIME (local interpretable model-agnostic explanations) [4] for local approximation of the model outcome to explain the DNN output. LIME is a novel explanation technique that explains the outcomes of any classifier in an interpretable and faithful manner by learning an interpretable model locally around the instance of interest [17]. More detailed information on LIME is found in [4].

III. RESULTS AND DISCUSSION

A. Hyperparameter tuning

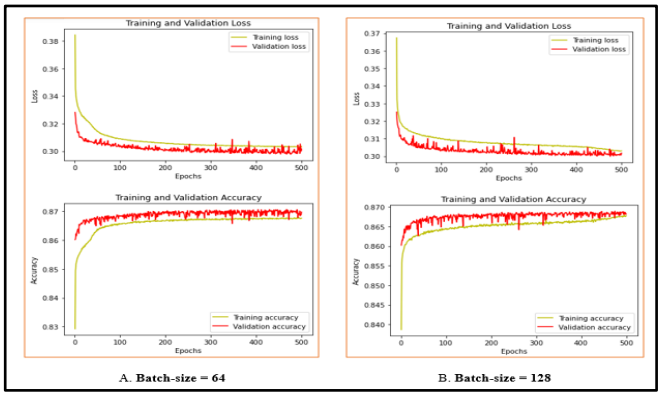


Fig. 4 Hyperparameter tuning loss and accuracy for batch-size 64 and 128

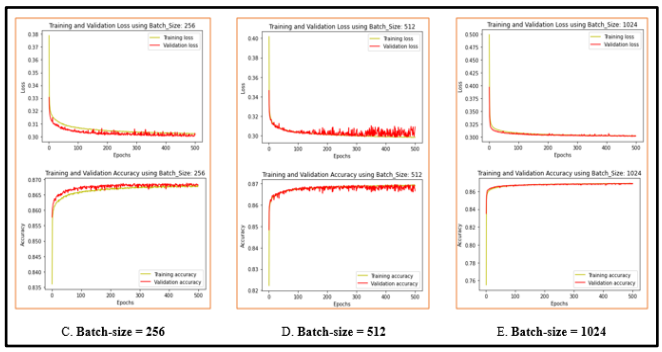


Fig. 5 Hyperparameter tuning loss and accuracy for batch-size 256, 512 and 1024

In this study, as already discussed, hyperparameter tuning has two steps. Table 2. shows the results of the first step while Fig. 4 and Fig. 5 demonstrate the second step.

As shown in Fig. 4 for batch size 64, there is a relatively wide separation between training loss and validation loss. This separation is larger compared to other batch sizes. In addition, the training loss is higher than the validation loss. When the batch size was increased from 64 to 128, there is an improvement. By continuously increasing the batch size to 1024 (shown in Fig. 5), the two plots were almost overlaying. Similarly, the accuracy has been improved as the number of batch size increased. As a result, the best model is tuned for a batch size of 1024 and an epoch of 500. In general, the overall hyperparameters tuned for the proposed DNN model are presented in Table 3.

B. Model Performance Evaluation

The effectiveness of the proposed DNN model with optimum hyperparameters is validated comparatively with well-known reference models: LR (random forest) and RF (logistic regression). Its comparative performance evaluation is shown in Table 4.

TABLE 4 COMPARISON OF PERFORMANCE EVALUATION

Model	Accuracy (%)	Precision (%)	Recall (%)	F1 (%)	AUC (%)
LR	85	86	86	86	92
RF	84	85	83	84	90
DNN	87	88	86	87	95

As depicted in Table 4, it is observed that the proposed DNN model is the best fit compared to the reference models in all the metrics used, except for recall, which is the same performance, recorded as LR of 86 %. The discussions in the remaining sections is using this proposed DNN model.

C. Causal Effect Relationship with PDP

PDP is used to understand the causal relationship of the input parameters on the response, which is the user throughput in our case. This relationship is particularly based on single features and user throughput that is shown in Fig. 6 and Fig. 7.

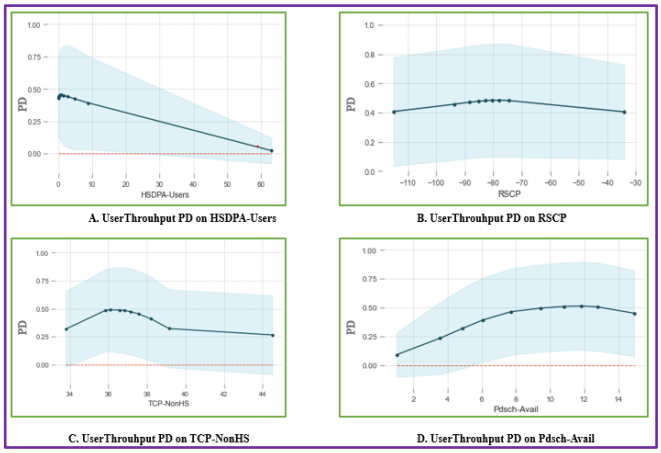


Fig. 6 HSDPA-Users, RSCP, TCP-NonHS, and Pdsch-Avail PD plot.

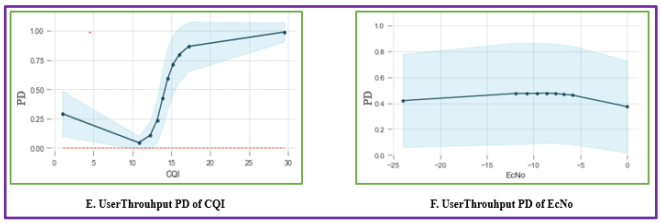


Fig. 7 CQI and EcNo PD plot.

In Fig. 6, the relationship between users and throughput is linear. However, the increase in the number of users negatively impacting the average user throughput. When we see, the RSCP in Fig. 6 and EcNo in Fig. 7, the throughput is mostly impacted for end-points in the plot. For Pdsch-Avail feature, the throughput is improving as the available code is increased. For TCP-NonHS and CQI, the relationship is not monotonic. However, for CQI less than around 15, the throughput is highly impacted.

D. RCA Explanation with LIME

1) Case analysis

Four randomly chosen samples are evaluated for feature importance to find the most impacting parameter for low user throughput. LIME, as already described, has been utilized to generate features ranking by assigning weights individually. Fig. 8 and Fig. 9 demonstrates those chosen samples' feature importance.

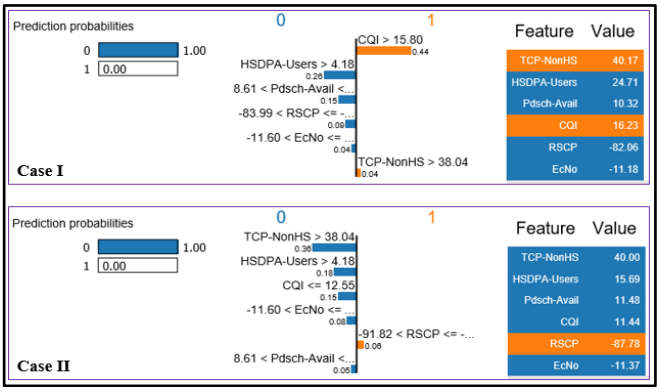


Fig. 8 Feature importance for case I and case II.

Fig. 9 Feature importance for case III and case IV.

The prediction probability in all of the cases is 100 % of class 0. The actual values for each feature are displayed on the right side of each plot. Given these values, the root cause for case I and Case IV is HSDPA-Users with the weight assigned 0.26 and 0.27 respectively. When we see the result in Case II, the root cause is TCP-NonHS with a weight of 0.36 that is the highest compared to other features. In Case III, the root cause is CQI because it has the highest weight contribution that is 0.46 as shown in Fig. 9. To sum up, the root causes are different for different samples.

2) Low User Throughput RCA on Selected Cells

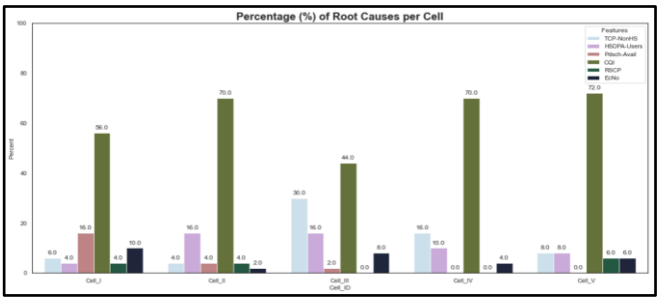


Fig. 10 Root cause percentages of Cell_I to Cell_IV.

LIME is a very useful algorithm for local RCA investigation of the low user throughput. However, to find the root cause globally (all of the samples per cell), each instance needs to be investigated. Then calculate the percentage to determine the largest share. As shown in Fig. 10 and Fig. 11, most of the root cause is due to the poor CQI value. However, the percentage is varied from cell to cell. The largest share is observed from Cell_IX by 90%. It is followed by 88% from Cell_X. This share is minimum for Cell_VI, i.e. 52%.

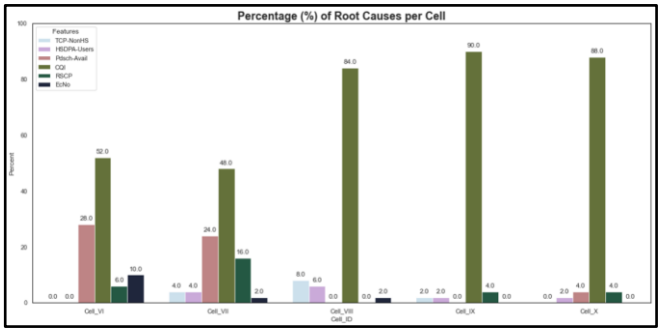


Fig. 11 Root cause percentages of Cell_VI to Cell_X.

IV. CONCLUSION

The main objective of the study was to find the root cause of low user throughput by developing a causal model using DNN. DNN model with optimum hyperparameters has been developed. The LIME algorithm has been used to enable the DNN model for root cause analysis by providing feature's importance to find the most important factor. To train and evaluate the model, the dataset has been collected from 445 cells of 76 NodeBs in Bahir Dar city as a case study. Since the DNN model is opaque in decision-making, the LIME explainer was trained with the DNN model to explain why the model classified an output as low user throughput. Because the RCA analysis is based on local explanation (*instance- based*), the root causes of low user throughput are not unique. By taking the proportion of all the root causes investigated on 10 cells (i.e. experiencing poor user throughput) dataset in the city for January 2021, CQI is the most important cause identified. This method of RCA is very helpful to prioritize mitigation solutions and to have better insight into the complexity of causal factors.

REFERENCES

[1] "Annual Business Performance Summary Report," Ethio Telecom, Addis Ababa, Ethiopia, 2020. [Online]. Available: https://www.ethiotelecom.et/ethio-telecom-2012-efy-2019-20-annual-business-performance-summary-report/.

[2] M. Solé, V. Muntés-Mulero, A. I. Rana, and G. Estrada, "Survey on Models and Techniques for Root-Cause Analysis," CoRR, pp. 1–18, 2017.

[3] M. M. Mampaka and M. Sumbwanyambe, "Poor Data Throughput Root Cause Analysis in Mobile Networks Using Deep Neural Network," 2019 IEEE 2nd Wirel. Africa Conf., pp. 1–6, 2019.

[4] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why Should I Trust You?' Explaining the Predictions of Any Classifier," Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min., pp. 1135–1144, 2016, [Online]. Available: https://arxiv.org/pdf/1602.04938.pdf.

[5] J. Pearl, Causality: Models, Reasoning and Inference, 2nd ed. Cambridge University Press, 40 W. 20 St. New York, NY, United States, 2009.

[6] M. Strevens, "Causality Reunified," Erkenn, vol. 78, pp. 299–320, 2013.

[7] M. Solé, V. Muntés-Mulero, A. I. Rana, and G. Estrada, “Survey on Models and Techniques for Root-Cause Analysis,” CoRR, pp. 1–18, 2017.

[8] C. Albon, Machine Learning with Python Cookbook, 1st ed. O’Reilly, 2018.

[9] L. Pierucci, A. Romoli, R. Fantacci, and D. Micheli, “An Optimized Neural Network for Monitoring Key Performance Indicators in HSDPA,” IEEE Int. Symp. Pers. Indoor Mob. Radio Commun. PIMRC, no. June 2016, pp. 2041–2045, 2010.

[10] N. K. Manaswi, “Multilayer Perceptron,” in Deep Learning with Applications Using Python, 1st ed., Apress, 2018.

[11] C. C. Aggarwal, Neural Networks and Deep Learning, 1st ed. NY, USA: Springer, 2018.

[12] L. F. Maimo, A. L. P. Gomez, F. J. G. Clemente, M. G. Perez, G. M. Perez, "A Self-Adaptive Deep Learning-Based System for Anomaly Detection in 5G Networks", IEEE Access, Volume: 6, pp. 7700-7712, 2018.

[13] “Keras API Reference,” *Keras*. https://keras.io/api// (accessed Mar. 5, 2021).

[14] “KerasTuner,” Keras. https://keras.io/keras_tuner/ (accessed Mar. 14, 2021).

[15] “Machine Learning & Deep Learning Fundamentals,” *DEEPLIZARD*, 2017. https://deeplizard.com/learn/video/hfK_dvC-avg (accessed May 09, 2021).

[16] “Partial Dependence Plot (PDP).” https://christophm.github.io/interpretable-ml-book/pdp.html (accessed May 19, 2021).

[17] “Decrypting Your Machine Learning Model Using LIME.” https://towardsdatascience.com/decrypting-your-machine-learning-model-using-lime-5adc035109b5 (accessed Apr. 18, 2021).