Addis Ababa Institute of Technology

School of Electrical and Computer Engineering

Resource Allocation Model for IP-Backhaul Maintenance Using Risk Analysis and Convolutional Neural Network

In the case of: ethio-telecom

By:

Tamrat Demena

Supervisor:

Dr. Rosa Tsegaye

A Thesis submitted to the School of Electrical and Computer Engineering in partial fulfillment of the requirements for the Degree of Master Science in Telecommunications Engineering.

November 2021, Addis Ababa, Ethiopia

Addis Ababa Institute of Technology

School of Electrical and Computer Engineering

Telecommunication Engineering Graduate Program

Resource Allocation Model for IP-Backhaul Maintenance Using Risk Analysis and Convolutional Neural Network

In the case of: ethio-telecom

By:

Tamrat Demena

Dr. Rosa Tsegaye	_____
Supervisor	Signature
Dr. Yalemzewud Negash	_____
Evaluator	Signature
Dr. Murad Redwan _____	_____
Evaluator	Signature
_____	_____
Evaluator	Signature

Declaration

I declare that this thesis is my original work and 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.

Tamrat Demena _____

Name Signature

Place: Addis Ababa, Ethiopia

Date of Submission: _____

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

Dr. Rosa Tsegaye _____

Advisor Name Signature

Acknowledgment

First, I would like to thank the Almighty God for my strength in completing this research and then my mentor, Dr. Rosa Tsegaye, for her continuous dedication and support during my work. Without your time, feedback, and follow-up, this thesis would not have happened. I would also like to thank Dr. Yalemzewud Negash and Dr. Murad Redwan for their constructive comments. Finally, I would like to express my gratitude to the NNOC, O & M, and SMC teams of Ethiotelecom for their practical support and to others who helped me conduct this research.

Abstract

For telecommunication business sustainability and efficiency, effective allocation of maintenance resources is vital. Over- or under-used maintenance resources such as automotive, tool, technician, and others are the main causes of extended maintenance times and dissatisfied customers. Ethiotelecom uses IP backhaul transport network technology mainly using IP-MPLS enabled routers and power systems to deliver Fourth Generation (4G) cellular and Fixed Mobile Convergence (FMC) services. Due to a lack of a failure risk analysis as a metric in the maintenance of the IP backhaul and unavailability of the services, Ethiotelecom loses 2.073 billion ETB every month. In this study, to assist the decision-making framework in daily resource sharing between IP and Power maintenance teams optimally at Ethiotelecom, we developed a knowledge-based optimal failure mode classifier model by fusing IP backhaul network alarm and Key Performance Indicators (KPIs) data obtained from Network Management Systems (NMS) at the network (topology) and feature level by using a design matrix and sporadic data aggregation process (DAP). To create a labeled design matrix based on a rational decision, we use weak-supervision data programming based on user-defined criteria functions obtained from reliability analysis using Failure Mode and Effect Analysis (FMEA) and the Decision-Making and Trial Evaluation Laboratory (DEMATEL) on the network failure history data with the help of experts from the two sections. In addition, we adopted a Windows slicing method in the time domain to obtain daily sliced grayscale images of the two failure modes on Python. We trained the two-dimensional convolutional neural networks (2D-CNN) to classify the images using grid search and k-fold cross-validation and tested the model using F1-Score and ROC-AUC. We have a 97 percent accurate baseline model. Besides, we test the model separately by selecting the alarm and KPI frames to capture the confidence of the model and to avoid uncertainty in the decision-making framework in IP-backhaul maintenance resource allocation. We then have a 98 percent and a 46 percent fair model, respectively. The study then concludes with some validation challenges and pitfalls.

Keywords—IP-backhaul, Maintenance, Resource allocation, Decision support, 2D CNN, KPI, Alarm, Domain Knowledge, FMEA, DEMATEL.

Table of Contents

List of Tables

List of Figures

List of Acronyms

Abbreviation	Description
APD	Alternating Power Distribution
ASG	Access Site Gateway router
ATM	Asynchronous Transmission Mode
ATN	Augmented Transition Network
BBU	Base Band Unit
BGP	Border Gateway Protocol
BSC	Base Station Controller
BTS	Base Transceiver Station
BWU	Band Width Utilization
CM	Corrective Maintenance
CNN	Convolution Neural Network
CSG	Cell Site Gateway
CSU	Controller System Unit
G-CV	Grid search Cross-Validation
DB	Distribution Board
DEMATEL	Decision Making Trial and Evaluation Laboratory
DNN	Deep Neural Network
DSP	Digital Signal Processing
DWT	Discrete Wavelet Transform
FM	Failure Mode
GRU	Gated Recurrent Unit
HVAC	Heating, Ventilation, and Air Conditioning
IDU	Indoor Unit
IP	Internet Protocol
KPI	Key Performance Indicator
LSR	Lable Switching Router
LSTM	Long Short-Term Memory
LTI	Linear Time Invariant

MPLS	Multiprotocol Label Switching
MSAG	Multi-Service Access Gateway
MTBF	Mean Time Before Failure
MW	Microwave
NMS	Network Management Server
NP	Nondeterministic Polynomial
ODU	Outdoor Unit
OMPS	Operation and Maintenance Performance Enhancing Solution
OSPF	Open Shortest Path First
PCS	Probable Causes
PRTG	Paessler Router Traffic Grapher
RPN	Risk Priory Number
RSG	Radio Service Gateway router
RTN	Radio Transmission Node
SNMP	Simple Network Management Protocol
SQL	Structured Query Language
TDM	Time Division Multiplexing
UDP	User Datagram Protocol
VPLS	Virtual Private LAN Services
VPN	Virtual Private Network
VRF	Virtual Route Reflector

1 Introduction

1.1 Background

Optimal allocation of maintenance resources includes modeling techniques, maintenance environments, data sources, system configurations, optimization criteria, and planning horizons[1]. To this end, mathematical and deep neural network models have determined that optimal allocation of maintenance resources is the greatest contributor to decision-making and business growth in telecommunications and other fields. However, there are still research gaps in modeling network performance and alarm data using domain knowledge on neural networks in a dynamic telecommunications network.

In telecommunication maintenance, resource allocation includes technicians, tools and cars, site visit funds, and other limited resources. For example, Ethiotelecom, the only telecommunications operator and service provider in Ethiopia, has set up the HUAWEI IP-backhaul network in Addis Ababa, which includes power and IP subsystems, mainly to provide Second Generation (2G), Third Generation (3G), and Long-Term Evolution-Advanced (LTE-A) cellular services over Multiprotocol Label Switching (MPLS) provide router.

In the Network Management System (NMS), however, 9,500 alarms per day occurred in connection with the services. The company also uses maintenance support systems only to manage operations without considering the risk of failure for resource allocation. And these approaches make excessive use of scarce resources. As a result, the company was giving up ETB 2.073 billion per month due to inadequate or inaccurate maintenance and unavailability of the services [2].

1.2 Motivation

The main motivation of this thesis arises from a challenge that Ethiotelecom is facing; which is on the verge of privatization and its current mission is, to provide a better telecommunication solution and create customer satisfaction. One of the challenges is making decision optimally to assign resources between power and Internet Protocol (IP) sections. Based on the failure prioritization and classification in the network, the problem belongs to the classification in the context of a neural network.

This work specifically focuses on the classification of failure modes based on domain knowledge and the network data behavior (pattern). Successful classification in advance is important in many ways. Failures can be actively resolved with fewer resources, which leads to less downtime and satisfied customers.

1.3 Statement of the Problem

The Ethiotelecom maintenance system did not have a resource allocation plan until recently. The company started using the automated framework of Operation and Maintenance Performance Enhancing Solution (OMPS), NMS, email, and Excel to manage maintenance activities. However, the current resource allocation system is not optimal as the framework focuses on managing activities to reduce network downtime and improve device health / security by using operational and maintenance metrics to meet the preventive maintenance compliance percentage.

In addition, the framework lacks planning and resource management using the reliability data analysis in its KPI measurement benchmark. There is also an integration problem between OMPS and NMS. Because of this, there is a gap between operations and fleet management, for example, there is an over allocation problem on the fleet management side when a fixed pooling strategy is used to meet vehicle requirements without considering the feedback from the operational failure risk analysis. In addition, the company takes an authority / top-down approach to decisions about the allocation of maintenance resources and is subject to a lack of intelligent systems between the operating units. Thus, there are manual and peer-to-peer resource calls for daily maintenance operations.

Moreover, even though the company designed the preventive maintenance policy for a week, its implementation is random. Thus, these approaches, together with the NMS's limited policy of filtering critical alarms, have a severe impact on limited resources and sometimes cause conflicts between the IP and the power units. For example, it takes between 1 and 8 hours to reach a site and restore services depending on the traffic in the city. The weight of the problem also depends on the accessibility of maintenance resources, which affects the maintenance time and customer satisfaction.

As far as this study is concerned, there is little research effort on this domain to reduce the problem. Therefore, the study aims to address the problem by examining the IP-backhaul network power

and the IP section by using the network alarm and performance data with a risk of failure analysis to build a knowledge-based optimal decision support system model using a convolutional neural network.

As mentioned above, this work aims to solve the problem of detection of failure modes (i.e., failure classification) in Ethiotelecom networks, in order to assist the system in making optimal resource allocation decisions. The main research problem will be:

How can the optimal failure mode classification model be implemented using domain knowledge of IP-backhaul network alarm and performance data.

1.4 Objective

- *General Objective*

The main aim of this study is to identify and classify the failure modes in the network in order to assign the priority for resources requirement between the two maintenance sections in an optimal way, based on the reliability-centered maintenance tools, network data and neural network model to help the decision makers in the company.

- *Specific Objective*

The specific objectives of the study are to:

- Select links from network topology for data collection and preprocessing;
- Assess IP-backhaul failure modes using reliability tools for data labeling;
- Train the 2D CNN model, using the data to classify the failure modes;
- Analyze and verify the result for model optimality and practicability;

1.5 Scope and Limitation

The main scope of this research work is to classify failure modes in the Ethiotelecom IP-backhaul network in specific links in Addis Ababa that only cover aggregation and access layer, using only available alarm and KPI logs to aid decision-making in allocation of maintenance resource. Even if there are many reliability tools and classifier models, the study only uses Failure Mode and Effect Analysis (FMEA) and Decision-Making Trial and Evaluation Laboratory (DEMATEL)

along with Convolutional Neural Network (CNN). It also goes beyond this study to examine areas other than IP and power.

1.6 Contribution

This study provides the company and researchers with insight into optimal resource sharing based on network reliability study through decision-making tools and system insight by learning relevant variables from network metrics and devices logs to classify failure modes. In addition, this research has made a significant contribution to automate the maintenance framework to manage tasks, improve the critical alert filtering policy by taking into account the priority of failures, synthesis information about systems, data aggregation methods in complex network and enhance neural network applications with ground truth labels. Overall, the study enables domain knowledge to be leveraged for deeper risk assessments to improve operational performance and revenue while reducing maintenance costs and time.

1.7 Thesis Organization

The study starts with the research question statement and its motivation. Chapter 2 summarizes the literature review of relevant topics. Chapter 3 states the theoretical development on the bases of which the entire knowledge, data, and convolutional neural network are carried out. Chapter 4 shows the method of the study. Chapter 5 shows the implementation of the model and provides the answer to the research question. Chapter 6 provides the result gained from theory and practical experiments, and chapter 7 concludes by laying out the limitation and future work of the study.

2 Literature Reviews

This Section reviewed relevant and current insight into Reliability-Centered Maintenance (RCM), decision-making tools, data preprocessing techniques, resource allocation models and classification algorithms in telecom and other industries.

2.1 Reliability-Centered Maintenance (RCM)

A comprehensive and up-to-date study [3] attempted to close the data/knowledge gap to optimize railway expert systems. With the FMEA and the Fault Tree Analysis (FTA), the system behavior was analyzed using sensor and failure data. The experiment was conducted with hourly data collected in the same location from different trains, and at one point, the Apache Kafka platform was used to merge the data. The semantic rule mining and windowing algorithm was applied to capture contexts from data using the Folio ontology for rule mapping with user feedback. The accuracy of the model recognition has increased by 50%.

In addition, [4] studied the FMEA to enhance the High-Speed Downlink Packet Access (HSDPA) quality and satisfy clients. Moreover, Decision Making Trial and Evaluation Laboratory (DEMATEL) was examined in[5] using a weighted digraph to represent two complex systems and experts' knowledge, and then to prioritize failures based on direct /indirect influence relationships. The research results have shown that DEMATEL can assign unique levels and group failure modes with equivalent effects and relationships. Furthermore, along with the classical DEMATEL technique fuzzy, grey, Analytical Network Process (ANP), and other techniques have been developed for dealing with group decision making problems[6].

2.2 Data Preprocessing for Neural Networks

Many studies have engaged in a network data aggregation, data labeling and data segmentation for localization, classification, and detection of network faults. For instance, in[7], user-defined functions such as; complementary, redundant, and cooperative data integration based on input source relation in a complex network, to supplement data-driven decisions. In addition, [8] proposed features level abstraction on different sensors' data. Further, another pioneer work[9] provided real-time data aggregation taxonomy to decision support systems in a network and automotive companies. And the study has set out design constraints that should be satisfied in

hard, firm, soft-real time scenarios. The study examined both Structured Query Language (SQL) standards and user-defined aggregator functions. Prior to data aggregation, the study implemented periodic features selection, worst-case scenarios selections, replacement of sporadic with aperiodic features. Finally, they evaluated the process using the 'SAFERE tool' to guarantee the bleach of design constraints.

Studies [10] and [11] that applied data programming to train neural network models have developed weak-supervision labeling methods based on domain knowledge to extract text relation patterns using the Long Short-Term Memory (LSTM) algorithm on three domains datasets. Based on the F1-score result, weak supervision labeling by data programming was 10 times better than the baseline labeling techniques. In contrast, [12] weak supervision models without the programming skills of experts proposed for text mining. The study found that this approach was far better than the other methods for reducing time and cost. In addition, [13] proposed rule-based and weak supervision data programming for chemical domain data, using CNN and Recurrent Neural Network (RNN), and achieved an equal 93% predictive accuracy.

To transform fused data from different sources and train CNN, Gated Recurrent Unit (GRU), and LSTM models, signal processing in the time domain[14] and frequency domains [15] was studied to solve network and power failure problems using simulated logs on Python and math lab, correspondingly. The studies have achieved 74.6% and 73.53% detection accuracies, respectively. In addition, another study[16] proposed an ensemble simulation method to train the CNN model for a power failure category generation (labeling) on MATLAB to utilize four (4) types of failure data. They compared sliding windows, Discrete wavelet transform (DWT), and Gramian Angular Field (GAF) techniques. The analysis result showed that CNN accuracy reaches 76%, 87%, and 76% for all three methods.

Moreover, a recent study[17] solved the footstep detection problem by encoding temporal 2-D pressure sensor data to the visual domain (pixels), using histogram and windowing techniques for binary frame preparation on a pre-trained CNN model. Based on the result, average frames outperform the SVM classifier with DWT by 10%. In addition, [18] investigated the impact of overlapping and non-overlapping windows approaches for human activity detection. The study used a 10 C-V on the 'HAR' dataset on stacked and not stacked windows to train three-machine learning and one neural network (NN) model. From the results, the overlapping window improves

the model F1-score by 10% for NN models with subject-independent cases. However, within machine learning classifiers, the F1 score even declined by 4%.

Overall, in all signal-processing techniques, the accuracy of the CNN depends on the size of training data, class size, feature selection, window parameters, and the number of epochs. Besides, data collection technique, the pattern of features, and the noise in data affect the models' performance.

2.3 Maintenance Resource Allocation Models

Most resource allocation studies deal with mathematical models to optimize different objectives, as described in [1]. However, neural network models have emerged in telecommunication and other industries. For example, in [19] to solve the resource allocation decision problem for an emergency fleet management system to minimize the time gap between incidence and response, using risk analysis and CNN-LSTM on spatial-temporal incident data. In addition, recently in[20], deep learning models were applied to minimize the total transmission time and energy in real-time for a congested content delivery network. By preparing channel matrix from mobile terminals data as input to CNN and DNN algorithms, as a result, they were found to be optimal to time and energy-efficient allocation to NP-hard problems with more than 95% accuracy.

Moreover, in[21], feature extraction and classification capability of deep learning (DL) algorithms were investigated. The author has recommended machine learning to resource allocation problems when either a model deficit or algorithm deficit occurs in the study domain. This view is supported by[22] that models or knowledge learned from one domain can be transferred to a new domain for different tasks for domain adaptation when both the features and the labels are different in the source and target domains. Among other requirements. Transfer learning by using a pre-trained 2D-CNN model has been discussed in detail in[23].

Based on the previous work, we can understand that traditional machine learning techniques can classify failures in a network based on fault history data from different NMS. However, combining reliability study and data analysis with neural network models can close the gaps in current mathematical models to make optimal resource allocation decisions in complex network maintenance.

3 Theoretical Development

3.1 Reliability Maintenance and Decision-Making

3.1.1 Failure Mode and Effect Analysis (FMEA)

FMEA is one of the five processes in six-sigma as a reliability-centered maintenance tool standardized by IEC 60812[24] and it is a systematic approach to analyze a system, design, process, and service to identify and reduce the potential failure modes' cause and effects. Besides, it is a structured, logical, and engineering technique to define, rank, and prevent the risk of failure modes(FM)[25].

There is a functional and hardware approach to FMEA, based on a study objective, scope, and outcome, both of them can be applied[26]. **The functional approach** is considered when the system's detail function is known and applied in a top-down manner, to define the most valid maintenance. Though the **hardware approach** is usually implemented before and after a design process, via using a schematic diagram, drawing, and bill of materials in a bottom up- fashion.

Risk is a measure of the possible severity of failures arising due to natural or artificial events. To estimate failure probability and severity, we can analyze a risk qualitatively (without data) or quantitatively (with data) depending on the accessibility of failure rate data. **Qualitative risk analysis** is the most widely used subjective technique to estimate risks using linguistic scales such as low, medium, and high. also, it hardly relies on actual data and probability[27]. Yet, **quantitative risk analysis** is applied using a system engineering approach for electronics components while reliability and ample field data are available.

- *Semi-Quantitative analysis*

Semi-quantitative analysis fills the gaps in both approaches to make decisions using the available domain knowledge and data. Such that, **Risk** is a subjective measure of the severity(S) of the effect and an estimate of the expected probability of its occurrence (P).

$$Risk(R) = S * P \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (3.1)$$

Risk priority number (RPN): is a numerical risk ranking of potential failure modes using quantitative and qualitative risk analysis. In addition, the Value of S, O, and D are assigned from 1-10 in ascending order, except for D that is in reverse order.

$$RPN = R * D = O * S * D \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (3.2)$$

Where **S** is the severity, an impact approximation of the effect of the failure mode in the system. **O** is the probability of the failure mode occurred when P < 0.2, P ~ C (criticality number). **D** is an estimation to identify and eliminate the failure mode before the system is affected. RPN has three Tables. For example, Tables 3.1, 3.2, and 3.3 show the qualitative scales commonly used for severity, occurrence, and detectability indices.

For example, if FM_1, FM_2 are subsystem failures in a system, then FM_{11} & FM_{12} will be failure modes of a subsystem FM_1. Similarly, FM_{21} and FM_{22} would be failure modes in subsystem FM_2. Therefore, the S, O, and D failure mode ranking would be gradually calculated for each subsystem.

Table 3.1 Severity ranking index of the failure mode FM1

Failure mode	Severity	Criteria	Ranking, S
FM_{11}	Low	From domain/ system expert	1
FM_{12}	Medium	From domain/ system expert	2
FM_{13}	High	From domain/ system expert	3
FM_{14}	Very high	From domain/system expert	4

Table 3.2 Occurrence Frequency ranking of the failure mode FM1

Failure mode	Occurrence	Ranking	Frequency	Probability
FM_{11}	Unlikely	1	≤ 0.001/thousands	$< 1x10^{-5}$
FM_{12}	Low	2, 3	0.1, 0.5	$1x10^{-4}$, $5x10^{-5}$
FM_{13}	Moderate	4,5, 6	1,2 ,5	1x10-5, $2x10^{-3}$, $2x10^{-5}$
FM_{14}	High	7 ,8	10 ,20	$1x10^{-2}$, $2x10^{-2}$
FM_{15}	Very high	9 ,10	50 ,100	$5x10^{-5}$, $> 1x10^{-1}$

Table 3.3 Detection ranking of the failure mode FM1

Failure mode	Detection	Criteria	Likelihood	Ranking
FM11	Certain	From domain expert	1	1
FM12	Very High	From domain expert	2	2
FM13	High	From domain expert	3	3
FM14	Moderate	From domain expert	5	4

❖ *Evaluation of RPN*

After the RPN has been calculated and evaluated, the risk severity ranking process will follow on the root causes. Table 3.4 shows FM1 RPN evaluation as an example, at this stage, the root cause of the system failure is taken as failure modes, to calculate their overall risk and rank (prioritize) them for further decision-making.

Table 3.4 RPN matrix

Subsystem	S	O	D	RPN	Rank
FM_{11}	10	9	5	S*O*D	2
FM_{12}	8	8	4	S*O*D	4
FM_{13}	7	7	6	S*O*D	3
FM_{14}	5	5	4	S*O*D	1

In the RPN ranking, the high severity failure mode (9 or 10) is mitigated first despite the value of other parameters. A good assessment by domain experts is therefore imperative. However, the RPN calculation is noticeably criticized for its precision because of the three risk factors the linguistic metrics and equal treatment with different risks and that usually leads to uncertain or vague results and contradicts the FMEA team [27].

3.1.2 Multi-Criteria Decision-Making (MCDM)

Studies are trying to deal with the weakness of the FMEA, based on uncertainty theories (Fuzzy logic) and MCDM (Euclidian geometry) methods [27]. From MCDM methods, Decision Making Trial and Evaluation Laboratory (DEMATEL) is an effective technique for complex systems to analyze the severity impact of failure modes using pairwise comparison and assigning unique criteria taking the FMEA's uncertain results for better decision-making [5]. Mathematically we can express it easily as linear algebra, digraph, and matrix form [28].

❖ *DEMATEL implementation procedures*

Dematel analysis depends on the volume of data, familiarity with tools and interest [29], [30], and can be carried out using with the Ms-Excel worksheet, with Python, MATLAB, or online platforms as follows. In addition, usually the words factor and criterion are used interchangeably to refer to causes of failure.

Step 1: - Generate (n x n) direct relation matrix among n factors

The effect of the element in each row is exerted on the element in each column of this matrix. If several expert opinions are used, all experts must fill in the matrix at least 1 and at most 4 weight values.

The arithmetic mean of all expert weight is used and then a direct relationship or influence matrix M is generated as follows.

$$M = \begin{bmatrix} 0 & \cdots & x_{n1} \\ \vdots & \ddots & \vdots \\ x_{1n} & \cdots & 0 \end{bmatrix} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \text{(3.3)}$$

For example, the following Table shows the direct relationship matrix (M), which is identical for five factors to the paired comparison matrix of the experts of the experts.

Table 3.5 Direct relation matrix

M	Factor 1	Factor 2	Factor 3	Factor 4	Factor 5
Factor 1	0	1.75	3	3	3.25
Factor 2	3.25	0	1.75	3	3
Factor 3	1	3	0	2.5	2.5
Factor 4	3	2	2	0	2.875
Factor 5	2.75	4	3	2.5	0

Step 2: Compute the normalized direct-relation matrix

By normalizing the influence matrix, will get the indirect relationship and the severity influences of one factor on the other factors. To normalize, the sum of all rows(R) and columns (C) of the matrix is calculated directly. The largest number of the row and column sums can be represented by k. To normalize, each element of the direct relationship matrix must be divided by k.

$$k = max\left\{max \sum_{j=1}^{n} M_{ij}, \sum_{i=1}^{n} M_{ij}\right\}, and\ N = \frac{1}{k} * M \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots.. \text{(3.4)}$$

Table 3.6 The normalized direct-relation matrix

M	Factor1	Factor2	Factor3	Factor4	Factor5
Factor1	0	0.143	0.245	0.245	0.265
Factor2	0.265	0	0.143	0.245	0.245
Factor3	0.082	0.245	0	0.204	0.204
Factor4	0.245	0.163	0.163	0	0.235
Factor5	0.224	0.327	0.245	0.204	0

Step 3: Compute the total relation matrix

After calculating the normalized matrix, the overall fuzzy relationship matrix can be computed to get an indirect relationship and a severity influence as follows.

$$T = \lim_{k \to +\infty} (N^1 + N^2 + \cdots + N^k) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3.5)$$

In other words, an n × n identity matrix is first generated, then this identity matrix is subtracted from the normalized matrix and the resulting matrix is reversed. The normalized matrix is multiplied by the resulting matrix to get the overall relationship matrix.

$$\mathrm{T} = \mathrm{N} \times (I - N)^{-1} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3.6)$$

Table 3.7 The total relation matrix

T	Factor 1	Factor 2	Factor 3	Factor 4	Factor 5
Factor 1	1.174	1.376	1.341	1.455	1.529
Factor 2	1.408	1.257	1.283	1.473	1.534
Factor 3	1.086	1.258	0.963	1.238	1.287
Factor 4	1.286	1.29	1.198	1.163	1.409
Factor 5	1.476	1.617	1.445	1.552	1.447

Step 4: Set the threshold value

The threshold value must be determined to calculate the internal relationship matrix. Correspondingly, partial relationships are neglected and the network relationship map (NRM) is applied. In the NRM, only relations are mapped whose values in matrix T are greater than the threshold value.

To compute the threshold value for relations, it is sufficient to calculate the mean values of the matrix T. After the threshold intensity has been determined, all values in matrix T that are smaller than the threshold value are set to zero, that is T, the causal relationship mentioned above is not taken into account. For demonstration purposes, the threshold value is 1.342. All the values in matrix T that are smaller than 1.342 are set to zero. The significant relationships model is shown in the Table below.

Table 3.8 The total- relationships matrix by considering the threshold value

T	Factor1	Factor 2	Factor 3	Factor 4	Factor 5
Factor 1	0	1.376	0	1.455	1.529
Factor 2	1.408	0	0	1.473	1.534
Factor 3	0	0	0	0	0
Factor 4	0	0	0	0	1.409
Factor 5	1.476	1.617	1.445	1.552	1.447

Step 5: Final output and create a causal diagram

The next step is to find out the sum of each row and each column of T (in step 3). The sum of rows (R) and columns (C) can be calculated as follows:

$$R = \sum_{j=1}^{n} T_{ij} \text{ and } C = \sum_{i=1}^{n} T_{ij} \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (3.7)$$

Table 3.9 The final output

	R	C	R+C	R-C
Factor 1	6.431	6.874	13.305	0.443
Factor 2	6.798	6.955	13.754	0.157
Factor 3	6.23	5.833	12.063	-0.398
Factor 4	6.88	6.346	13.226	-0.534
Factor 5	7.206	7.538	14.744	0.332

The final output in the above Table can be found by calculating the sum of R-C and R+C for factors (causes of failure) and identify their relationship to each other. Then the values of R + C and R - C can be calculated by R and C, where R+C represents the degree of relationship of factor I throughout the system and C-R represent net effects (severity influence) that factor i contributes to the system.

The following Figure 3-1 shows the criteria relationships model. This model can be represented as a diagram with the values of (R + C) placed on the horizontal axis and the values of (R - C) placed on the vertical axis. The position and interaction of each factor with a point in the coordinates (R + C, R - C) are determined by the coordinate system.

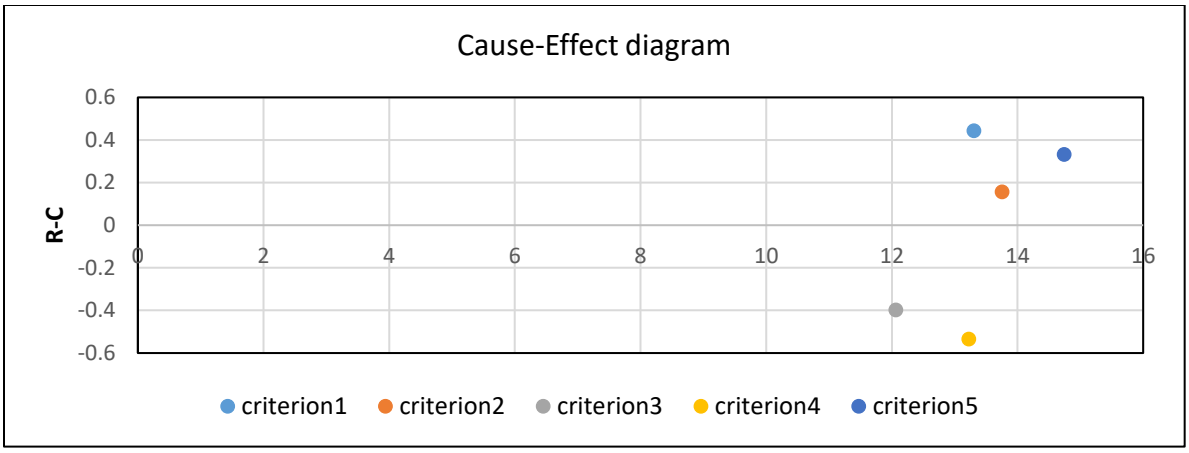


Figure 3-1 Criteria/ factors /cause-effect diagram.

Step 6: Interpret the results

Using the graph and Table above, each factor can be rated based on the following:

The horizontal vector (R + C) represents the degree of relationship between each factor in the system. When it comes to the degree of relationship, Factor 5 comes first and Factor 2, Factor 1, Factor 4, and Factor 3 come next.

The vertical vector (R - C) represents the degree of influence factors on the system. In general, the positive value of R-C represents a causal variable, and the negative value of R - C represents an effect. Factor 1, 2, and 5 are causal variables, Factors 3 and 4 are an effect.

3.2 Convolutional Neural Network (CNN)

CNN is one of the deep learning universal approximator algorithms that imitate the real brain and should solve the problem of simple machine learning algorithms such as, learning and mapping complex functions in high-dimensional spaces to solve intelligent tasks[31].

There are more than ten CNN architectures with sparse interactions, parameter-sharing capability, and transitional invariance as feature extractor for multilayer neural networks. and because of the hardware acceleration, in addition to its statistical efficiency in training big data, it is more efficient in terms of storage and runs time requirements. Furthermore, Two-dimensional (2D)-CNN in particular has proven to be more efficient than 1D-CNN in order to classify images from a feature vector of fixed size (image). In addition, its classification performance in computer vision [32].

3.2.1 Mathematical Model of CNN

Convolution is a mathematical operation used to combine two signals to form a third signal. Systems are described by a signal called an impulse response. The convolution relates the input, output, and impulse response signals in relation. The input signal can be broken down into a series of pulses, each of which can be viewed as a scaled and shifted delta function. Second, the output resulting from each impulse is a scaled and shifted version of the impulse response. Third, the total output signal can be determined by adding these scaled and shifted impulse responses. In other words, if we know the impulse response of a system's, then we can determine the output for each possible input signal. Mathematically represented as follows in Equation 3.8. [33]

$$x[n] * h[n] = y[n] \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (3.8)$$

where n, is the number of samples space, x[n] is input and h[n] is impulse response, y[n] is the output of the system, convolution operator [*] yields y equals x+h-1, scaled and shifted version of the input samples. Convolution is commutative: a[n]*b[n] = b[n]*a[n].

This first viewpoint of convolution is based on the fundamental concept of digital signal processing (DSP) for low-pass, high-pass filtering, and attenuator in DSP[34] , combining this with image processing; images can be thought of as 2D versions of LTI systems. In this case, we are talking about Linear Space-Invariant (LSI) systems. In the bi-dimensional discrete case, the convolution operation is defined as follows in Equation.3.9.

$$O(i,j) = \sum_{u=-\propto}^{\propto} \sum_{v=-\propto}^{\propto} F(u,v)I(i-u, j-v) \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (3.9)$$

Images are finite dimension signals with a well-defined spatial extent. Therefore, the previous formula becomes the following:

$$O(i,j) = \sum_{u=-k}^{k} \sum_{v=-k}^{k} F(u,v)I(i-u, j-v) \dots\dots\dots\dots\dots\dots (3.10)$$

Where: I is the input image, F is a convolutional filter (kernel) itself and 2K is its sides, O (i, j) is the output pixels, in the (i, j) positions. Different Convolutional filters extract different features of the input image.

Two additional parameters control how the convolution operation is performed, and tell the operation how many pixels to skip when moving the kernel horizontally and vertically across the input image, and these are the horizontal and vertical **stride**s. Usually they are the same and marked with the letter **S**. If the input image has one side, then the resolution of the output signal resulting from the convolution with a **kernel** of size *k* can be calculated as follows:

$$O_w = O_h = \frac{I_w - 2k}{S} + 1 \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (3.11)$$

Where; *w* is the width and *h* is the height of an image, When we think of color images, they have (RGB) instead of grayscale, they increase *D*-depth next to w and h. when we treat images like stacks of 2D signals; we call these volumes. The result of this convolution operation is called a **feature map**.

$$O(i,j) = \sum_{d=1}^{D} \sum_{u=-k}^{k} \sum_{v=-k}^{k} F_d(u,v) I_d(i-u, j-v) \dots\dots\dots\dots (3.12)$$

3.2.2 CNN Model Parameters

Model parameters contain components, such as the number of layers, activation function, and optimization algorithm variables (kernel/filter) that are learnable parameters directly from the data. In addition, hyper-parameters are tunable parameters of a model, such as a dropout, learning rate, and batch size. CNN is a feed-forward learning algorithm, and its architecture has the following four major building blocks to extract and learn features from the encoded data [35].

Convolution Layer: is the first layer for extracting features from an input image that preserves the relationship between pixels by learning image features, using a small squares filter (kernel) on the input image to create the feature map matrix.

Activation functions: are non-linear activation functions with a smooth curve that is easy to learn and provides probabilistic support for complex real-world problems. A widely used activation function is the Rectified Linear Unit (ReLu) and its output is (x) = max (0, x). The real data needed such a smooth non-lean activation function to learn non-negative linear values. Figure 3.2 shows widespread activation functions in deep networks.

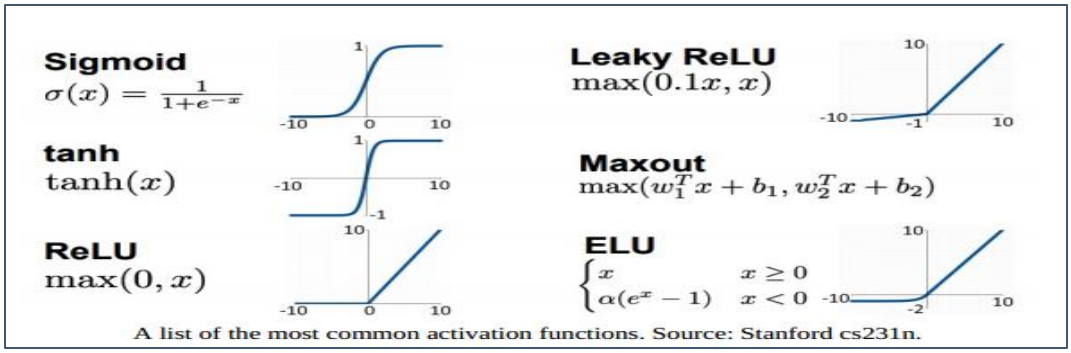


Figure 3-2 Activation functions in neural network.

Pooling Layers: they are windows that move in the horizontal and vertical direction of the input without a learnable kernel to reduce the dimension of each feature map, but they retain the required information by using Max Pooling, Average Pooling, and sum pooling on the previous convolution layer.

Fully Connected Layer: is the final layer that combines features for modeling after flattening the feature map matrix into a vector (X_1, X_2, and X_n) and feeds them into this layer to classify outputs using SoftMax or sigmoid activation function as shown in Figure 3.3 below.

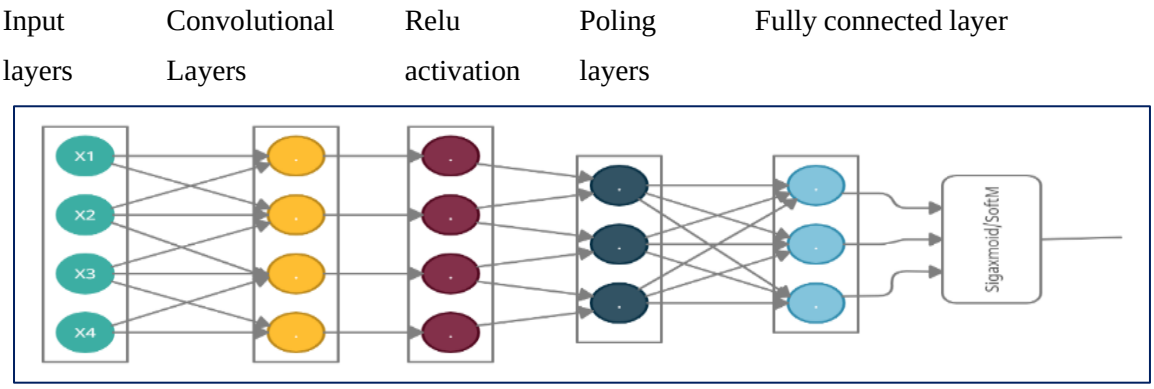


Figure 3-3 CNN sequential architecture with building blocks.

- ***Backpropagation /Optimization algorithms***

Deep neural network models optimize their performance based on an objective function J (θ) that computes the model's loss by using categorical cross-entropy with SoftMax for multi-class classification or binary cross-entropy with sigmoid for binary classification problems. Since CNN is a feed-forward learning algorithm, it uses backpropagation algorithms to optimize the loss functions.

Gradient descent algorithms are the most widely used optimization algorithms in deep networks [36]. Gradient descent (GD) is a way of minimizing an objective function J(θ) parameterized by a model's Parameters θ ∈ Rd by updating the parameters in the opposite direction of the gradient of the objective function ∇θ J(θ) w.r.t. to the parameters. The learning rate (η) determines the size of the steps taken to reach a (local) minimum. Expressed mathematically as:

$$(GD) \qquad \vartheta := \vartheta - \eta . \nabla\theta J(\theta) \dots\dots\dots\dots\dots\dots (3.13)$$

There are three (3) variants of GD to use, depending on the amount of data and the tradeoff between accuracy and time for parameter update; they are batch GD, mini-batch GD, and stochastic GD. However, the mini-batch gradient descent is more efficient than the others in terms of convergence stability by iterating over mini-batches of the usual size 32 to 256 to reduce computational costs.

$$\theta = \theta - \eta . \nabla\theta J\big(\theta;\ x(i{:}\, i+n);\ y(i{:}\, i+n)\big) \dots\dots\dots (3.14)$$

GD algorithms can get trapped in their numerous sub-optimal local minima. Therefore, the choice of the learning rate for spares data is difficult. Studies have developed various update rules (flavors of GD), such as Momentum, Nesterov Accelerated Gradient, Adagrad, Adadelta, RMS prop, Adam, AdaMax, and Nadam to address the local minima convergence problem and enhance GD algorithms. Further details can be found in [36].

- *Hyperparameters*

Hyperparameters are all the training variables that are manually set to a pre-determined value before training begins, and the search for hyperparameters is an iterative process that is limited by time, money, and energy[37]. However, finding optimal parameters is used to reduce the over- and under-fitting of the model.

Learning rate (η): determines the size of the steps that the GD optimization algorithm needs to reach a (local) minimum (usually in the ranges from 10^{-1} to 10^{-6}). This is a key parameter to tune deep networks.

Batch size: is the number of samples processed before the model is updated, usually 2^N = 32-512 data points used for the computational efficiency of the model. A batch size of 32 means that 32 samples from the training dataset are used to estimate the error gradient before the model weights

are updated. Also, the optimization algorithms to be used determine the batch. Popular batch sizes are 32, 64, and 128 samples.

The number of epochs: is the number of complete passes through the training data set. An epoch means that the learning algorithm has run through the training data set once, an integer value between one and infinity can be used to execute the algorithm, and even stop it by using stop criteria.

Dropout rate: is a regularization algorithm and widespread hyperparameter at neuron level in CNN models [38] to prevent overfitting and improve generalization ability. During training, it randomly turns off the neurons at every iteration with a probability of 0.5, and the remaining kernels are trained by backpropagation during the prediction. In addition, early stopping, weight decay, data augmentation, and batch normalization techniques can be applied depending on the model complexity.

- *Model Parameter selection and tuning*

Parameter selection and tuning are used to prevent both overfitting and underfitting of a model. The **over-fit model** is an over-trained model for the data, so that, it even learns the noise and misclassifies unseen data with a perfect training result and poor test result. While the **Under-fit model** cannot properly learn the patterns in the data. In such cases, we see a low score on both cases.

Searching for the best parameter configuration that gives the best score on validation or test set manually in deep learning models is, computationally intensive. Such that, there are semi/ fully automatic random search algorithms that are used for large data. However, integration of Keras

models with the **Grid-search** algorithm(brutal-force) is effective for smaller datasets along with **cross-validation** techniques to reduce the model bias-variance trade-off problems [35].

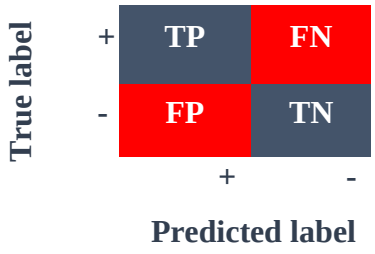
3.2.3 Classification Models and Evaluation Metrics

From supervised learning, classification models approximate a mapping function from input data to discrete output labels and are used to classify spam email, credit card fraud, disease, network fault, and to segment customers [39].

In addition, to assess their performance accuracy, the ROC curve (AUC-ROC), F1 score, and others calculated from the confusion matrix can be used. And the choice of these metrics depends on the objective, the level of accuracy, the runtime/ memory requirements, the training instances and the class balance of the model [40].

❖ *Confusion Matrix:*

Confusion matrix maps the expected to the ground truth labels of the binary classification problem, and also helps to check the model prediction ability of classes. The confusion matrix is a contingency Table that consists of four cells and shows the labeled class TP, FP, FN, and TN. If there are two classes, it shows positive and negative (or just + and −).



- True Positive (TP): The number of positive instances that are classified as positive;
- False Positive (FP): The number of negative instances that are classified as positive;
- False Negative (FN): The number of positive instances that are classified as negative;
- True Negative (TN): The number of negative instances that are classified as negative;
- P = TP + FN is the total number of positive instances and;
- N = FP + TN is the total number of negative instances.
- FP (Type1) and FN (Type2) are errors, and their meaning depends on the application.

◆ ***Classification Accuracy:***

It is the number of correct predictions divided by the total population that shows the overall performance of a model's correctness. Inversely, the classification error rate is defined as the proportion of observations that have been misclassified and expressed as the error rate (loss) i.e., 1-accuracy.

$$Acuracy = \frac{TP + TN}{TP + TN + FP + FN} \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (3.15)$$

- ***Recall***

A recall is the number of correctly classified positive results divided by the number of all positive results and it shows how the model correctly detects all positive classes i.e., all positive correct detection rates.

$$Recall(Sensitivity)(TPR) = \frac{TP}{TP + FN} \dots\dots\dots\dots\dots\dots\dots\dots\dots (3.16)$$

- ***Precision:***

The precision is the number of correctly classified positive results divided by the number of all classified positive results and shows the model confidence, ranking, and suggestion. The number of times the model correctly classifies the positive class.

$$Precision = \frac{TP}{TP + FP} \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (3.17)$$

- ***F β – Score***

F $_\beta$ -score is the harmonic mean of precision and recall; therefore, it indicates well the performance of the optimal classifier, especially in the case of the imbalance class, where β is 1 for binary classification.

$$F_\beta = (1 + \beta^2)\frac{(Precision \times Recall)}{\beta^2(Precision + Recall)} \dots\dots\dots\dots\dots\dots\dots\dots (3.18)$$

The F1 value is considered perfect when it is 1, while the model is considered a total failure when it's 0. For instance. For instance, from the confusion matrix TP, FP, TN, FN are 0, 0, 80, 20 respectively, the F1-score will have a value of 0, thus F1-score penalizes the classifier for failing to detect any of positive class, even if the classifier has 80% accuracy. Therefore, F1-score is the major evaluator to test a model's fairness.

- ***Receiver Operating Characteristic Curve -the Area Under the Curve (AUC -ROC)***

The TP rate and FP rate values of different classifiers on the same test set are often represented with a ROC graph that reflects its original uses in signal processing applications. The value of TP rate versus the FP rate is plotted on the Y-axis and the X-axis for different discrimination

thresholds. In other words, it shows the performance of binary classifiers under various circumstances.

$$ROC: TPR\ Vs(FPR)\ or\ (1 - TNR) \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (4.19)$$

The area under the curve (AUC): is the most common measure from ROC analysis. AUC indicates the skill of a classifier to separate labels between positive and negative.

3.3 Overview of Ethiotelecom IP-backhaul Network

Any study should provide details of the underling system to carry out reliability-centered maintenance. Hence, the study has investigated IP-backhaul network in Ethiotelecom in advance from power and IP systems failure perspective, referring to recent studies, device vendor documentation, and using the domain experts' experience.

Ethiotelcom's IP backhaul HUAWEI network in Addis Ababa is FMC oriented that uses Internet protocol and multi-protocol label switching (IP-MPLS) tunnel technology to carries the dynamic Layer 2 and Layer 3 service packet from base stations to base station controller using the open mechanism. The network provides a virtual private network (VPN), MPLS traffic engineering (MPLS-TE), and quality of services (QoS) by using label switching routers(LSRs), that found in BTS shelters and data centers in the access and aggregators layer of the network, these routers also use DC power source and AC powered cooling system for their proper function[41].Figure 3.4 below, shows the simplified functional block diagram of the network (system), the red line shows the power input to the network devices and the green one indicates the service signals.

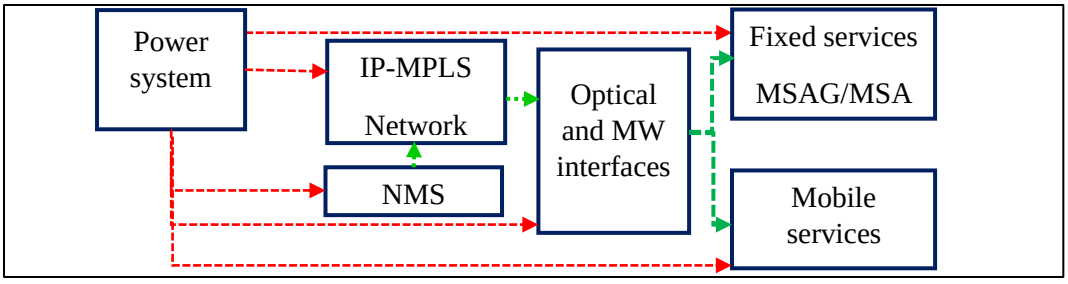


Figure 3-4 Functional block diagram of Ethiotelecom IP-backhaul network.

Moreover, the MPLS network and the power system are inputs to the mobile and fixed service network devices, and they are directly or indirectly responsible for the network failure. This is why the study chooses a system approach. Besides, the network has the following subsystems:

3.3.1 IP-MPLS Network Subsystems

IP-MPLS is a subsystem of the IP-backhaul system that contains components such as layer one devices (MAC), layer 2(Link), and layer 2.5 (MPLS-IP) services listed as follows.

- *L1 (physical layer) subsystem*

In the physical layer of the network, topology, fiber connection, and Media Access Control (MAC) layer protocols are found. In addition, ports, power supply, fan modules, cards, and motherboard of the routers lie in this subsystem. The bandwidth in the access and aggregation layer is 10GE and 40GE respectively, and using this link capacity, multiple physical interfaces are bundled into a logical Ethernet interface (i.e., Eth-Trunk) using link aggregation, just like a smart group. In addition, to increase the reliability of the hub sites, LSR, BBU, and RTN use 1+1 redundancy.

The topology in Figure 3.5 below contains fifteen (15) physical direct links and twelve (12) LSRs that are connected in a hierarchical ring architecture. In addition, the routers operate using the Intermediate System to Intermediate System (IS-IS) protocol in the link layer for data flow. Hence, access layer LSRs capture the required raw data and forward it to their respective aggregators. This study uses this topology for data collection.

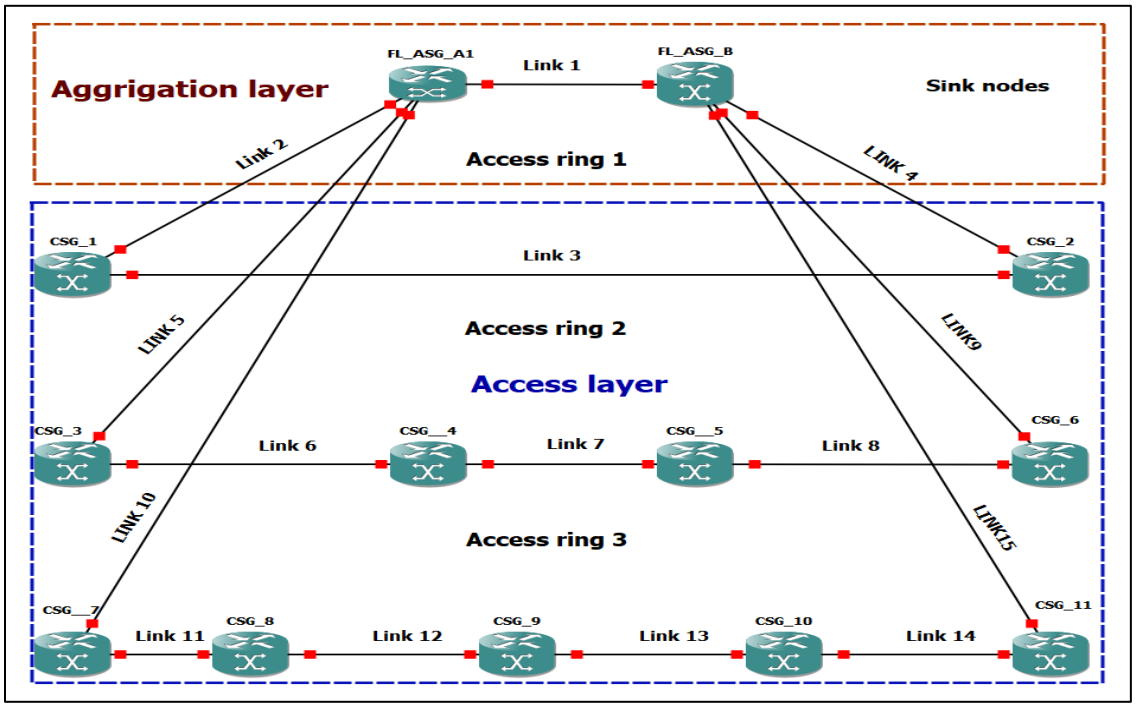


Figure 3-5 The selected IP-backhaul hierarchical ring topology.

❖ *L2 (Link layer) subsystem*

The link layer (L2) subsystem is responsible for packet transportation and congestion control in the network and is composed of link-layer devices and protocols (components). For this purpose, the IS-IS protocol is implemented and directly runs on L2 devices, for logical Interfacing, memory buffering, CPU, Pseudo Wire(PW), clock synchronization. In addition, to schedule service and avoid congestion the combination of Strict Priority Queue (PQ) and Weighted Fair Queuing (PQ+WFQ) was implemented. On top of these, access and aggregation ring isolation at the protocol layer is implemented for failure propagation prevention.

❖ *IP/ MPLS layer (2.5 layer) subsystem*

The IP-MPLS layer (subsystem) is responsible for labeling and forwarding IP services such as MS-PWs to transmit 2G TDM and 3G ATM services, and E2E L3-VPN to transmit Ethernet services, for fault rectification and quick service adjustment in a flexible manner. To carry these services, IP layer devices (LSRs) such RSGs and ASGs route reflectors are deployed, which reflect the private network route from CSG to RSG. Besides, all services are delivered using end-to-end using Hierarchy VPN (H-VPN) protocol and service components.

Moreover, the logical topology relies on the physical environment to save the optical fiber resource. Further, the interconnected interfaces at the aggregation and core layer reside on different boards for service protection by using master/slave interfaces at the core layer. Figure 3.6 below shows the service-level architecture of the network.

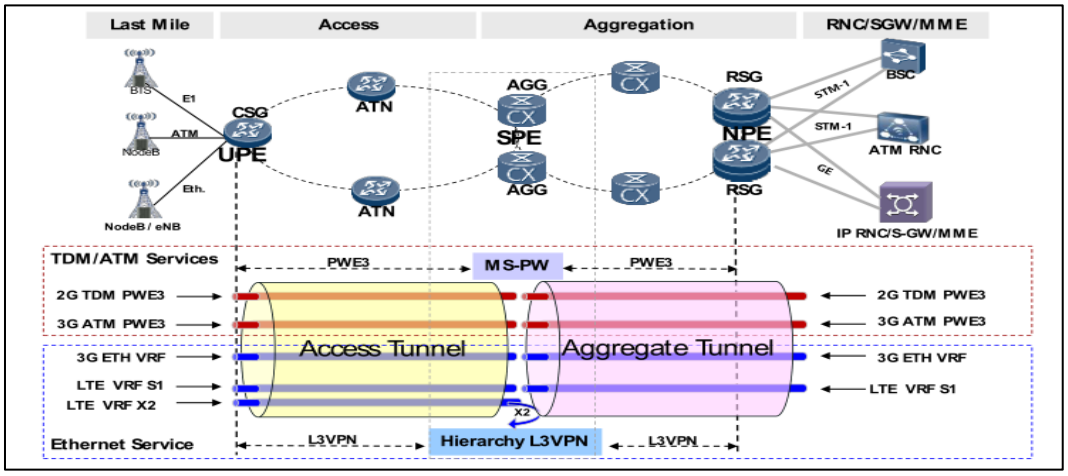


Figure 3-6 Ethiotelecom IP-backhaul architecture adapted from[42].

3.3.2 Power and Environment Subsystems

The power system is composed of commercial power (mains), diesel generators, and battery as power sources. When the commercial power source fails to supply alternating current (AC) power, the generator starts automatically and the automatic transfer switch (ATS) swap all loads to the generators. In addition, the rectifiers get AC power from a commercial source or generator and convert 220ACV to 48DCV to provide power to direct current (DC) loads and to charge the battery banks [43].

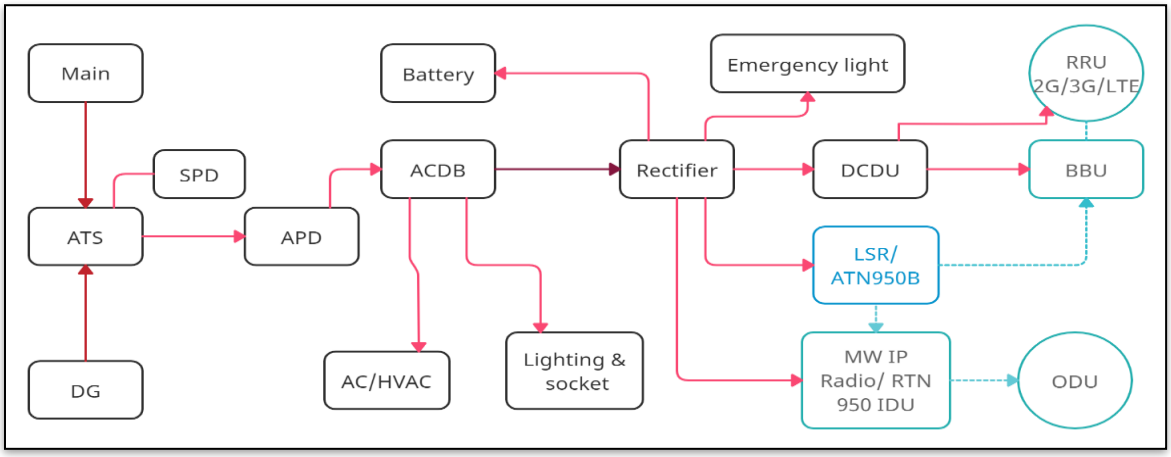


Figure 3-7 A typical BTS power system architecture in Ethiotelecom

As can be seen in Figure 3.7 above, the power distribution layout of the existing design confirms that the ATN routers (LSRs) are the input to all loads in the service layer. It is a suitable scenario for reliability maintenance analysis. Such that, the following system knowledge have gained from the power system.

- *AC power sources*

Commercial power is a primary power supplier while the generator and batteries are standby power sources. There are 40 to 50 KVA\3P five-wire transformers on most sites, which means, the rated input voltage is 220VAC in single-phase or 380VAC line voltage in three-phase. Therefore, the main AC power is guaranteed to provide power for current and future loads.

However, According to AC power specialists even if most hub sites have a generator and some of them use only AC and battery, there are AC power failures including blackout, phase loss, under-

voltage, and overvoltage. Moreover, there is approximately 20 % commercial power outage per week for 1 to 2 hours.

- *Automatic Transfer Switch (ATS)*

ATS is an electromechanical switch, which is used to swap the loads from commercial power to the generator or vise-versa without interruption [43], to ensure the reliability of the network by harmonizing the AC supply voltage from different sources.

Such that, Ethiotelecom installed power frame switches type ATSs using two-step stored energy technology that permits semi-automatic operation under load up to 5,000 amps capacity, utilizes sensitive controls to monitor the minimum threshold power[44]. Moreover, the ATS matches the power design.

- *Diesel Generator (DG)*

A diesel generator converts mechanical energy into electrical energy using the engine, alternator, controller parts and using deasil as a fuel. In addition, in Ethiotelecom the generator serves a standby power supply to AC loads when main power fails, the voltage level is below the thresholds. Further, almost all sites satisfy the minimum requirement to have 50KVA- 40KVA generators capacity except one site.

However, the generators mostly fail to start automatically that drains the battery power, and exhausts their energy-storing capacity.

- *Cooling system*

The cooling system is the air conditioner (AC) and fans that provide suitable temperature to all loads including the MPLS routers for their proper function. In Ethiotelecom, two split-type ACs are found in most shelters. Also, the maximum power consumption of an AC is 4.8 KW, and rarely temperature or humidity threshold alarms are generated.

- *AC Protection Devices*

To protect AC loads from a short circuit or overvoltage, an unconditional voltage spike, Surge Protective Device (SPD), circuit breakers, grounds, fuses, and rectifiers are installed in

Ethiotelecom BTSs. In most sites, Type 2 SPD is used between the main ATS and the earthling bar, in parallel to a load. However, some sites have the wrong circuit breaker size.

- ❖ *DC Power Sources Subsystems*
- ◆ *Rectifier Unit*

A rectifier is a critical component in the IP-backhaul DC power system to convert AC voltage into DC voltage besides charging the battery banks by using a monitoring system in control interface from both main and generator sources. In addition, it is used to filter harmonics while DC to DC voltage conversion, to protect against spikes in voltage or current, to feed critical loads such as; MPLS-routers, MW IP radio(or RTN), aviation light, and the battery banks, by using two types of DC distribution units (DCDU) are installed by HUAWEI and ZTE, which differ in capacity and efficiency[45].

The ZTE rectifier can support up to 12 rectifier modules for their full capacity. Moreover, most Huawei CSG (LSR) use ZTE rectifiers to get 30A and a -48 VDC power, by using 220V Single-phase rated voltage, with 40A maximum input current, and its reliability (MTBF) is $\geq 2.2\times105$ h.

- ◆ *Battery Banks*

The battery banks are on standby to supply DC power for the critical loads including MPLS routers until the generator starts and swap the loads in case the commercial power fails or vice-versa. There are two battery banks connected in a series-parallel configuration, and in each bank, there are four 12V batteries to achieve the 48DC V[45].

Moreover, there are first and second shutdown modes in the battery system, to separate non-critical and critical loads from rectifies, also it uses a remote sensing unit to control the batteries' normal state and prevent floating charge. However, because of corrosion, temperature, overcharging, and deep discharging, their present load carry capacity is only six (6) hours, i.e., two (2) hours below their maximum capacity.

- ◆ *DC Protection Devices*

To protect the DC loads and to ensure that the batteries neither are under nor over-discharge, the batteries, rectifiers, and other DC loads are connected with the Load Low Voltage Disconnect

(LLVD) and Battery Low Voltage Disconnect (BLVD) intelligent contractors, using the Bus-Bar Under Voltage (BBUV) threshold. Figure 3.8 below shows the block diagram of the subsystem.

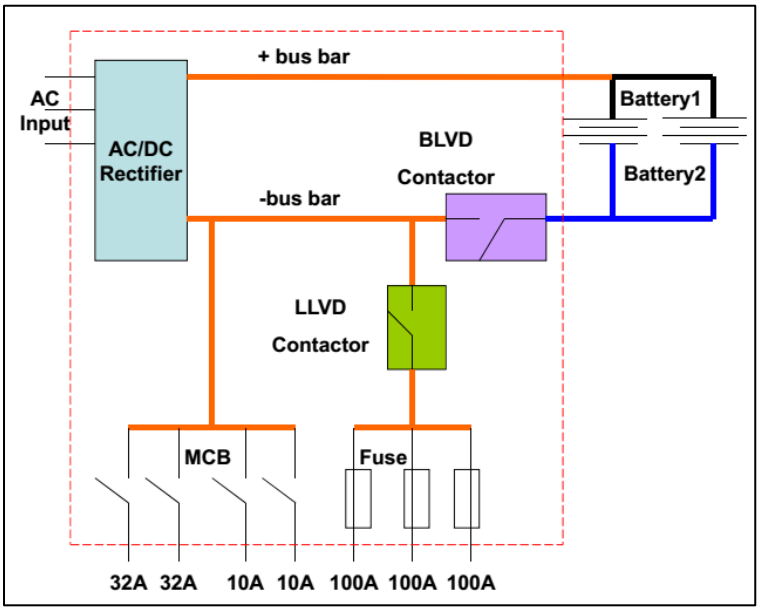


Figure 3-8 DC protection circuit block diagram.

- *Power Distribution Unit (PDU)*

The PDU integrates AC and DC power distribution. The ACDU comprises the mains protection, generator, ATS, and AC power supply to AC, Light, and Socket outlets. The DCDU completes the DC output, access of batteries, rectifiers, and load protection. Overall, this study uses this domain knowledge to prepare the failure and performance data for the proposed optimal maintenance resource allocation model.

4 Methodology

4.1 Experimental Protocol

Knowledge-driven studies require knowledge acquisition and transformation. Based on the risk analysis results for the selected domain, a knowledge base optimal maintenance resource allocation model has been proposed. To fulfill the optimal model criteria and to achieve the study objective, the methodology applied in this thesis is illustrated in Figure 4.1 below.

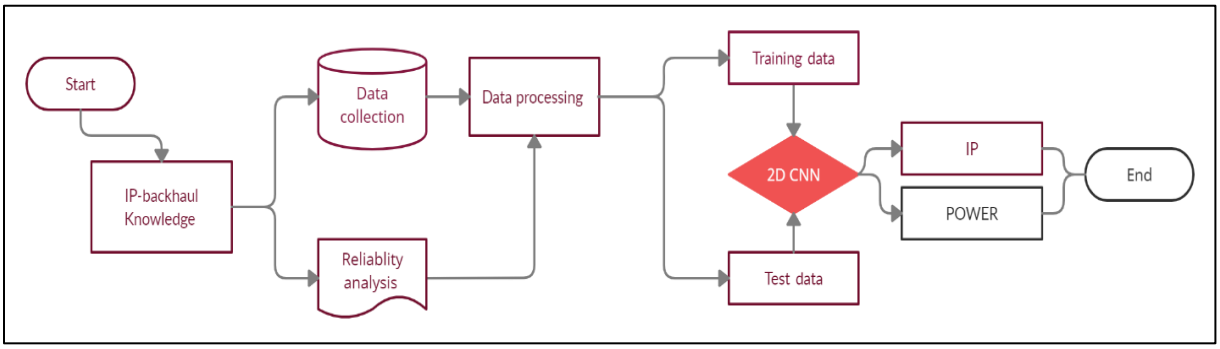


Figure 4-1 Overview of the proposed model framework.

In the next sub-Sections, the study will show how the data was collected from the network topology; how the features were selected, cleaned, fused, labeled, and sliced to train the 2D-CNN algorithm for power and IP failure mode classification. Also, the assumptions of data aggregation and labeling are demonstrated.

4.2 Data Preparation

Before data collection, feature selection, data aggregation, data transformation, and exposing the data for a neural network, it is vital to understand the aim of the study clearly to extract meaningful information from domain specific features using domain knowledge, and following a machine-learning pipeline[46].

4.2.1 Data Collection

Data collection from a complex network topology is possible to solve NP-hard problems using causality graph, domain knowledge, and statistical data analysis on KPI and alarm data [28], [47]. Therefore, in this study to collect data and to select important features to use as inputs for the model, discussing the study aim with the network experts was the first step.

We collected the data sets between August 2019 and June 2021 for 23 months from access and aggregator layer routers that are more susceptible to failure and suitable for data collection and aggregation in the IS-IS ring topology. We used the U2000 NMS to collect actively the alarms and link performance in the network. In addition, to avoid missing data while collecting the KPI data, the PRTG tool, via passive packet monitoring on the 10Gib access routers interfaces configured.

The collected KPI data contains average values of 15 links and 12 devices' performance extracted at 1-hour intervals using peer-to-peer interfaces measurement intervals of 1 sec on all links at a time. In addition, the KPI data includes traffic utilization and availability metrics, such as packet delay, packet loss ratio, and jitter. Moreover, CPU usage, memory usage, and device availability found in LSR performance data. Moreover, the alarm data contains random faults from 12 devices with a 1-hour interval.

Overall, we gathered 16,170 hourly instances with nine (9) features from each link, 16180 device performance instances with four (4) features and 17,000 alarm instances with nineteen (19) attributes from twelve devices.

4.2.2 Data Preprocessing

Following supervised data preprocessing pipelines to get structured data, to fit and evaluate the CNN algorithm. In this study, data preprocessing includes feature selection, data cleaning, data aggregation, and data labeling, feature scaling, and data segmentation using domain and system knowledge on the Python environments.

4.2.2.1 Feature Selection

The main idea behind the feature selection was, to identify the most relevant variables to the study and understand how the link, device performances, and the alarm influence the tendency of assigning the right resource at the right time and finally to support the decision-making process. Hence, we have used domain knowledge to select important features from the three datasets as follows.

4.2.2.1.1 Alarm data feature selection

The devices' alarm log is an external manifestation of the network; it provides contextual information and temporal relation under which the network is working. The raw alarm data

contains 17 categorical features, such as alarm types, alarm names, alarm IDs, alarms sources, generated time, and frequency. Besides the time and the alarm IDs, it contains the device's IP address and redundancy information, such as alarm status and states that were not relevant to the study. Consecutively, this study has selected the following four (4) categorical variables listed in Table 4.1 as input for modeling.

Table 4.1 Selected alarm data features

Name	ID	Severity	PCS	S. Time	E. Time
Port	1111115	Critical	Correlative	02/07/19	12/07/19
Link	3	Major	Root alarm	11/22/20	4/22/21
Power	300909	Minor	Correlative	12/21/21	4/22/21
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
MPLS	261145	Warning	Correlative	8/22/21	8/30/21

- **Name:** indicates the alarm types that correspond with 59 kinds of alarm IDs.
- **Probable cause (PCS):** indicates whether it is a root or correlative alarm.
- **Time:** indicates the start (S) and end (E) time of the generated alarm.
- **Severity:** indicates the alarm hazard level (filtering policy).

4.2.2.1.2 Devices' performance feature selection

From the router performance metrics, the selected device performance indicators were only three, such as CPU, memory utilization, and device availability, because the device software version cannot allow extracting power-related indicators. Table 4.2 shows the minimum and maximum values and, based on ITU-R as stated in [48][42], they were in their optimum threshold.

Table 4.2 Running status of the twelve (12) devices and their corresponding values per hour

LSRs	Indicators			**Granularity**
	CPU usage	Memory Usage	Availability	
Aggregator	13.25-18.75%	16.75 – 17%	100 %	1 hour
Access	35.75 -36.75%	33 - 33.35 %	91%	1 hour

- **CPU utilization**: indicate devices load execution time and related to service delay failure.
- **Memory utilization**: indicates available memory and relates to network congestion failures.
- **Device's availability**: indicates the device states either it is offline or online.

4.2.2.1.3 *Link performance data feature selection*

To select information from the link metrics, interpretation is important for monitoring them. Hence, the selected link performance numerical features and their standard suggested by [49] [41], are listed in Table 4.3 below.

Table 4.3 Standards of the link performance

Metrics		BWU (SNMP)	Delay Jitter	UDP Jitter	Packet loss ratio	Granularity
Thresholds	Non-Real time	≤70%	<60ms	<20ms	<1.0%	1hr
	Real time	≤50 %	<40ms	<15ms	<0.1%	1hr

Even if the indications state was in their optimal threshold, the study has believed that they might be useful to reinforce the failure symptoms, to create a robust model. In such, the study first selected the three key link metrics: availability, utilization, and latency of the IP-MPLS network.

- **Availability:** metrics indicate delay and related to reliability problem in the network.
- **Packet loss:** indicate **reachability** of nodes and caused by network congestion, HW/SW.
- **Jitter:** indicates the average **packet delay** and relates to packet service latency problems.

We have selected the average values from the link data. In addition, we excluded the Bandwidth utilization (BWU) from the study since it shows the capacity of the link and the efficiency of the devices' interface. In addition, this metric primarily links to the throughput of a specific service, and often performance optimization models used the BWU data. Second, all LSRs interface BW capacity is 10 GB. However, the collected link BW data was below the threshold that is <70%.

Thus, it is difficult to model the problem using a neural network that learns features from raw data. Eventually, the study has selected ten (10) features, including reachability (Ping), packet loss ratio, packet delay, jitter, alarm severity, alarm name (ID), alarm probable cause (PCS), CPU, memory, and device availability from the three data sets for data cleaning.

4.2.2.2 Data Cleaning

Data quality is the crucial factor for the accuracy of data analytics results and business decisions making problems. Therefore, after selecting ten (10) relevant features from the data, we were able to identify and remove outliers, to increase the model's computational efficacy and improve the data consistencies.

We cleaned the dataset by modifying long words, deleting blank space, removing units and redundant columns in the data before aggregation using Excel and Python. In addition, we applied Visual basic codes in MS excel to get separated CSV files from each test link and to bring the different formats of the datasets collected from two monitoring tools before cleaning. Likewise, we treated the other two datasets. Then, from each link, we were able to obtain 16,170 rows and over four (4) columns performance data, besides delay, Ping, Jitter, and packet loss-ratio feature as we can see from a week sample data below in Figure 4.2 below.

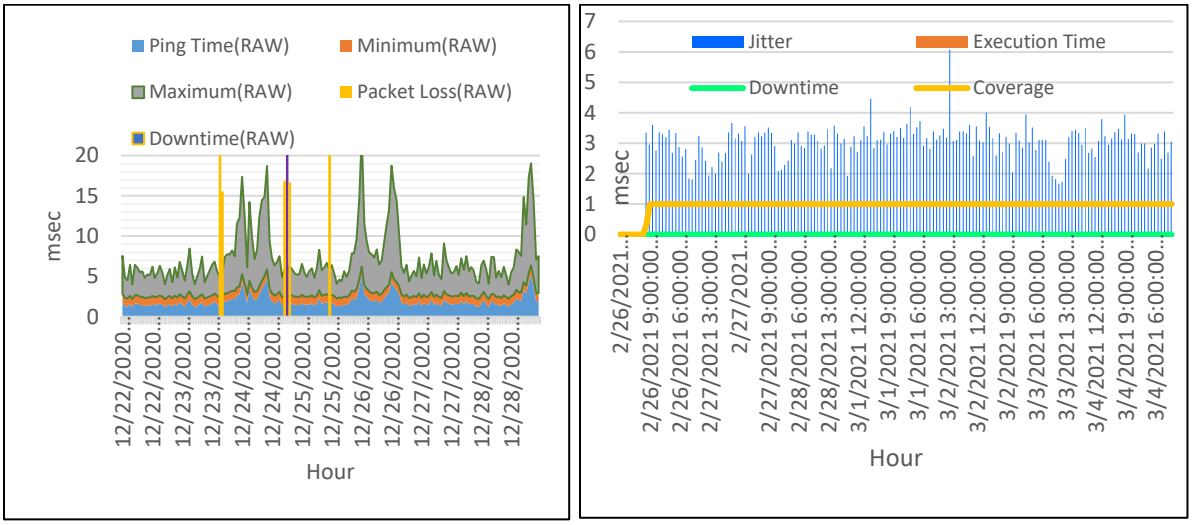


Figure 4-2 A link availability (right) and jitter (left) raw sample data

To clean the data, for instance, first, we removed execution time and coverage columns from jitter data because we use them to monitor the performance of the tools. Then, we have removed redundant columns, such as downtime and coverage features.

In addition, as we can notice from the delay features correlation matrix in Figure 4.3 below, maximum-delay, average-delay, and delay-jitter variables have shown a significant correlation between them. Moreover, the minimum delay feature has shown a less correlation with the other variables. However, we have selected the average-delay variable for analysis by dropping the rest features as noise because; it shows the network status more.

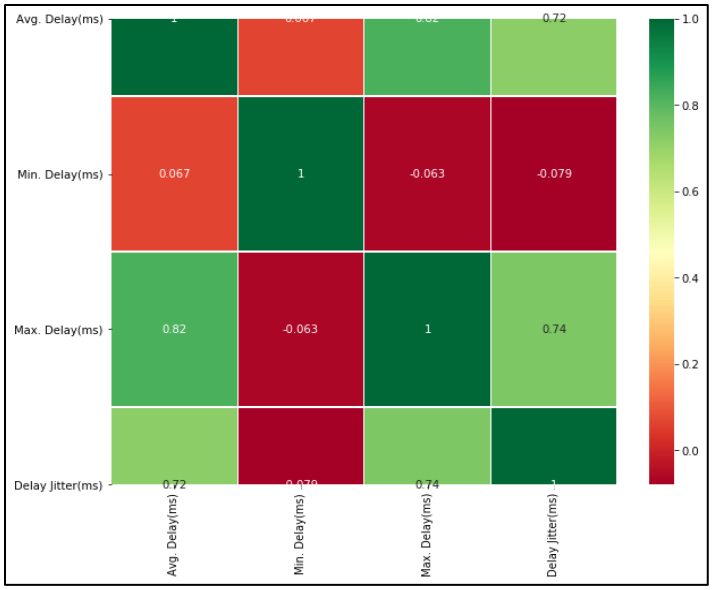


Figure 4-3 The delay features' correlation matrix.

Subsequently, we have compared the data before and after filtering the noises to get an insight into the features. For example, Figures 4.4 and 4.5 below show the delay and packet loss ratio.

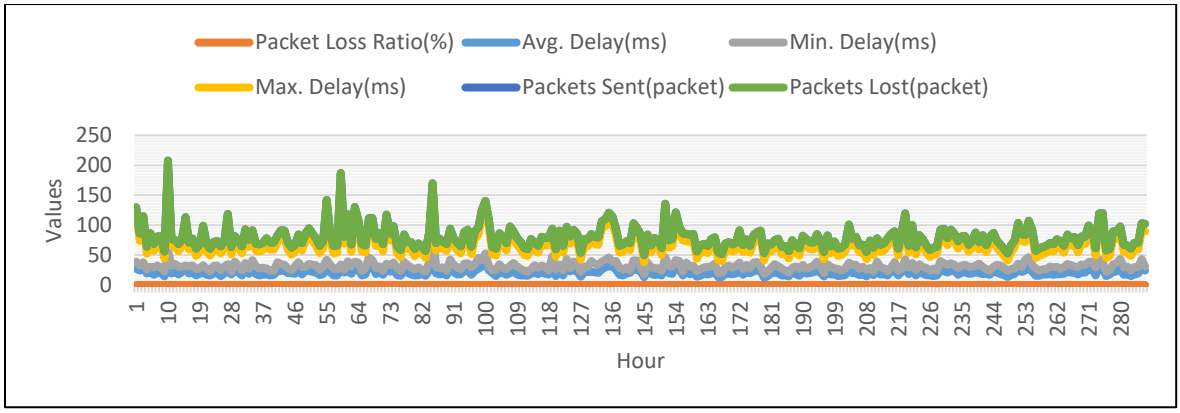


Figure 4-4 Raw delay and packet loss ratio data.

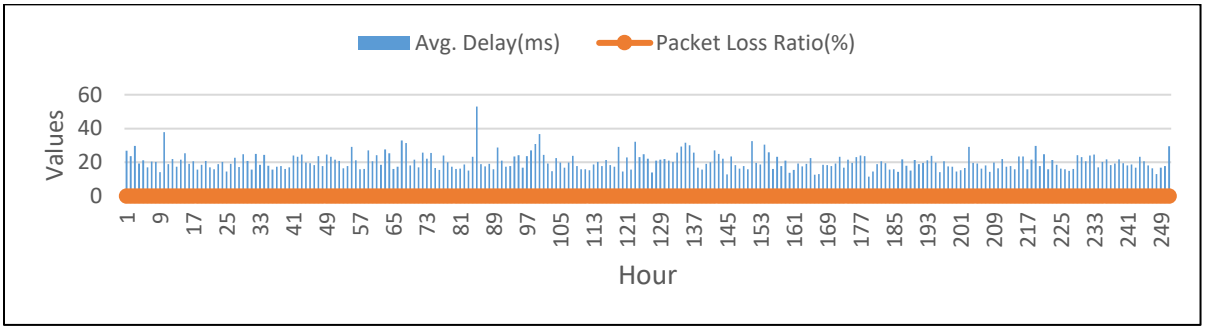


Figure 4-5 Filtered delay and packet loss ratio data

As depicted in Figure 4.6 below, we have got 16,180 instances and three (3) numerical columns from the device performance dataset after dropping the granularity column that was not directly applicable for the analyses. We can see that the data sets were free from noise.

However, the CPU, Memory, and availability indicators were below their maximum threshold values. These show that the network was stable and 100% available from the device performance perspective. Fortunately, there were few missing values as compared to the other two datasets.

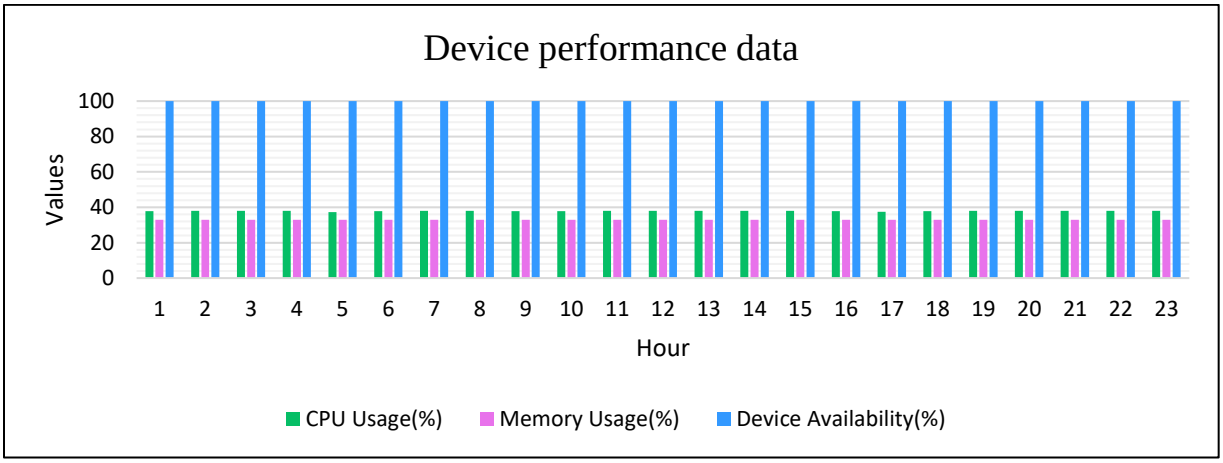


Figure 4-6 A specific device performance data per hour

In addition, to create a uniform format, we have prepared a new data frame using Python. Then the data aggregation process has filled the devices' missing values with each other. Further, using two monitoring tools has aided the study to fill the missing values and improve the consistency between the KPI data datasets.

Eventually, we could get seven (7) features with 16,125 hourly interval instances from each link and all device performance data.

There were also 14 extra columns with16, 607 hourly instances in the alarm dataset. Besides, from the instances, 5% of them were empty and 25% of them lacked time consistency. Further, 5% of the alarm records occurred more or less while reports the same fault.

However, we have grouped incidents that happened within the same time interval to filter the noise using the tupling method in Python. When we look at the raw alarm log distribution from all routers in Figure 4.7 below, the values in the selected categorical variables were as text.

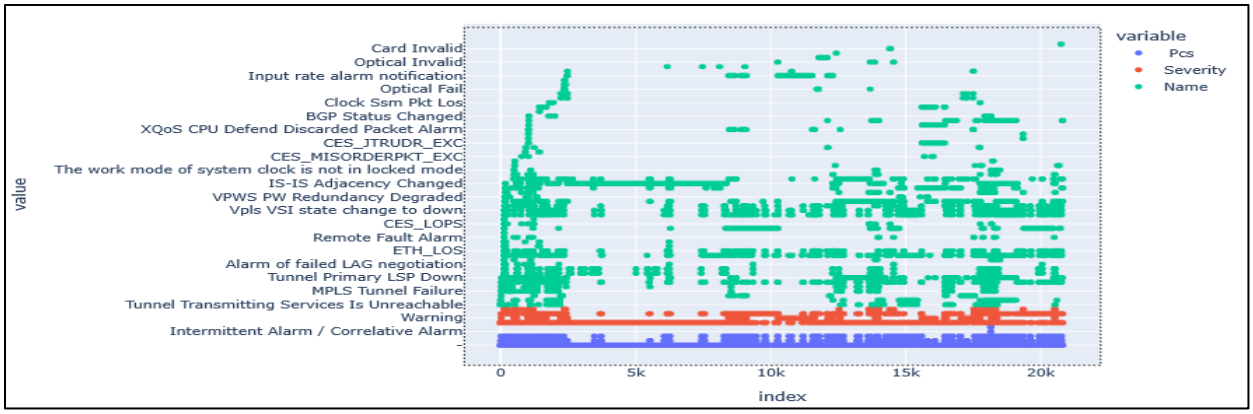


Figure 4-7 Distribution of the alarm logs.

We can observe from the alarm severity frequency distribution in Figure 4.8 below that few minor alarms were available. And from the PCS and severity variable distributions, Major and critical alarms have contained correlative and root alarms.

In addition, warning and minor alarms comprised a higher noise as compared to the others. Therefore, we have filtered and removed the noises by using the system knowledge.

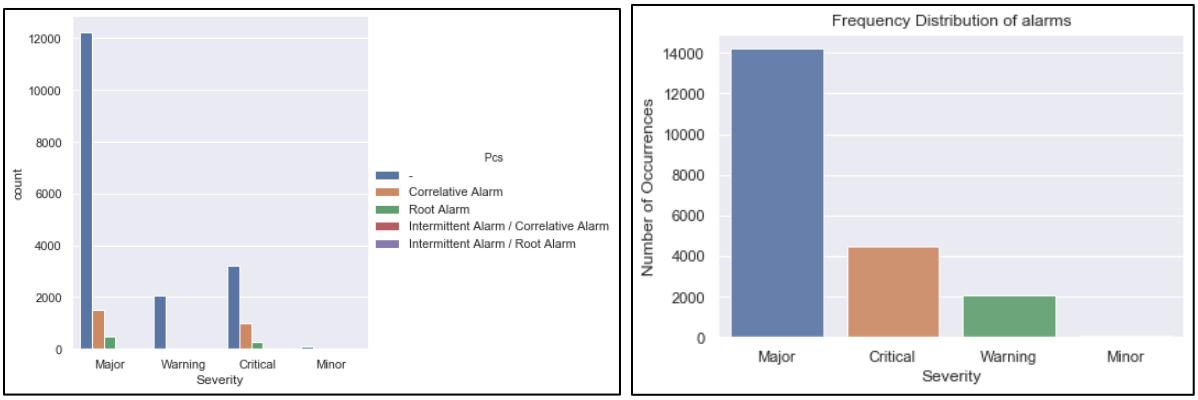


Figure 4-8 Frequency distribution of probable cause (left) and severity (right).

Moreover, we designed the timestamps in the original log files for machine learning by preparing a new time index to transform each timestamp into real-valued UTC, to keep the consistency among the datasets, to find one large chronologically ordered log file based on the data collection period. Then, we got three (3) features with 16,235 hourly interval instances from the alarm log.

Overall, after we cleaned the data sets, we were able to get 10 features from the three data sets that have a uniform time-index with the listed data properties in Table 4.4 below.

Table 4.4 Properties of the cleaned data sets.

Data set	Instance	Number of features	Type
Alarm	16,235	3	Catagorical
Link performance per link	16,125	4	Numerical
Device performance	16,125	3	Numerical

4.2.2.3 Data Aggregation

Selecting appropriate aggregation functions and fulfilling timing constraints is vital to meet the study purpose. For these reasons, this study chooses a generic data aggregation function, adapted from the heuristic data aggregation process formulated in[9], then we aggregated the cleaned data features in-network and feature level. In addition, the study has used design matrix preparation for complex network topology, as stated in[28], to summarize and answer the research question.

The idea behind the data aggregation is that we initially collected the complete raw data instances within 23 months period hourly from the same network segment. Hence, the study therefore assumes that the data contain enough temporal and contextual relationships in the network to be discriminated. Second, our goal was to determine the failure mode with a known location, devices, and specified time windows. Thus, the alarms in one hour can correspond to the link and device performances at least in 24 hours intervals.

Based on the above assumptions and using the network topology in Figure 3.5 in Section 3.3, we have framed the link performance, device performance, and the alarm data aggregation (fusion):

4.2.2.3.1 Link performance fusion

Fifteen (15) links and 12 LSR routers were available for the data aggregation process, pointing out the topology. We have fused the KPI data of all rings (R_1, R2, and R_3) based on their input source relation. For instance, joint KPI of a ring is the union of all links in one ring, such that, after combining the three rings separately, then the joint KPIs of the three IS-IS areas combined as (R_1 U R_2 U R_3). Finally, to aggregate the link performance data we computed their mean values using Equation (4.1).

$$L\bar{F} = L\frac{1}{n}\sum_{i=1}^{n} f_{Li} = \frac{1}{n}(f_{L1} + f_{L2} + \cdots + f_{Ln}) \ldots\ldots\ldots\ldots\ldots\ldots (4.1)$$

Where *f* represents all features in a link, L_i is a link, and f_{Li} contains all link performance features.

For instance, if we look at 24-hour sample jitter data in Figure 4.9 below that was stacked from fifteen links, using their time index to calculate their mean value, and we followed the same procedures for other performance features in the KPI data set.

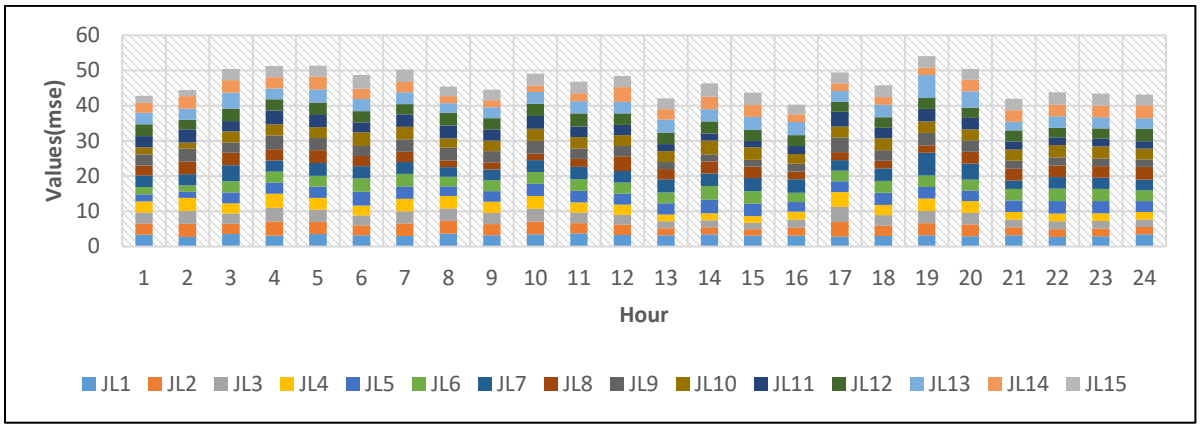


Figure 4-9 Sample aggregated jitter data.

4.2.2.3.2 Device performance data fusion

We have fused the device performance data based on device-level aggregation. In addition, each device group belongs to the same IS-IS rings. So that, we used explanatory (lossy) type (SUM) aggregation function for directly connected devices in the same ring. Then we applied average aggregation function to get the three features aggregate values in one data frame. Mathematically, expressed as:

$$D\bar{F} = \frac{1}{n}\sum_{i=1}^{n} f_{Di} = \frac{1}{n}(f_{D1} + f_{D2} + \cdots + f_{Dn}) \ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (4.2)$$

Where f represents all features in a device, D_i is the number of routers in a ring, and f_{Di} contains all device performance features.

4.2.2.3.3 Alarm data fusion

We assumed one IS-IS area failure affects the other areas at the physical topology level, or the failure impact on one ring disseminates to others. Also, the alarm data contains spontaneous events. Therefore, we merged all available alarms in each ring to get one data frame, and then we filtered alarms reported within the 24-hour window using their time index. Further, since the two (2)

aggregator routers contain more or less the same alarms, we extracted their intersection and combined them with other routers' alarms to get aggregated alarm data in a ring. We also filled gaps between the alarms dataset features. For instance, if we missed one alarm in a router at a specific time, we filled it with other routers' alarm data. Mathematically, total aggregate alarm data (A_{RT}) is expressed as below in Equation 4.3.

$$A_{RT} = \{A_{R1}\}U\{A_{R2}\}U\{A_{R2}\} \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots \dots (4.3)$$

Where A_{R1}, A_{R2}, and A_{R3} are the alarm data in the three rings in the topology.

To construct the design matrix, we aggregated the KPI (link and device performance) data according to their level of abstraction by concatenating, merging, stacking 98 columns from the KPI data sets. We developed 16125 instances of seven (7) features after taking their mean. Then, we join the alarm data with the KPI data after combing the alarm data in each ring and filtering 16125 instances only using the Pandas libraries. Finally, we merge them with the 16125 rows of the three (3) alarm features.

Then, we obtained 16125 rows with ten (10) columns after joining the three datasets using their time index, and the final design matrix values that were ready for data labeling were integer and category, as depicted in Figure 4.10 below.

	Date Time	Ping 1-15 ave	D1-15 mean	PL1-15 mean	J1-15 Ave	CPU 12 Mean	Memory 1-12 mean	AV1-12 mean	PCS	Severity	Alarm name	Class
0	12/22/2020 12:00:00 AM - 1:00:00 AM	1.752227	2.739997	0.0	3.15	29.750000	30.083333	91.666667	Correlative Alarm	Major	Tunnel Transmitting Services Is Unreachable	NaN
1	12/22/2020 1:00:00 AM - 2:00:00 AM	1.492207	2.088333	0.0	3.21	29.729167	30.083333	91.666667	Correlative Alarm	Major	Tunnel Transmitting Services Is Unreachable	NaN
2	12/22/2020 2:00:00 AM - 3:00:00 AM	2.895547	4.290560	0.0	3.15	29.833333	30.083333	91.666667	Correlative Alarm	Major	Tunnel Transmitting Services Is Unreachable	NaN
3	12/22/2020 3:00:00 AM - 4:00:00 AM	1.655553	2.118340	0.0	3.14	29.895833	30.083333	91.666667	Correlative Alarm	Major	Tunnel Transmitting Services Is Unreachable	NaN
4	12/22/2020 4:00:00 AM - 5:00:00 AM	1.883353	2.533893	0.0	3.42	29.666667	30.083333	91.666667	Correlative Alarm	Major	Tunnel Transmitting Services Is Unreachable	NaN

Figure 4-10 The final design matrix.

4.2.2.4 Data Labeling

Beyond understanding the problem, clarity of the end aim of the study and knowledge of the data is essential to engineer domain-specific features and combine external data that add value to the proposed supervised model. Therefore, the study applied weak-supervision labeling (data

programing) to extract IP and power classes from 10 features and 16125 instances in the design matrix using the domain knowledge captured by risk analysis.

4.2.2.4.1 Risk analysis on IP-backhaul system failure

To label the design matrix, capturing experts' knowledge on normal and faulty behavior of the power and IP subsystems in the network was necessary. Therefore, following semi-quantitative risk analysis using FMEA functional approach, segmenting the system into subsystems and components level by:

- Brainstorming on the potential failure modes;
- Identifying the effects of each failure mode;
- Assigning failure severity, occurrence, and detection rate;
- Calculating the risk priority number for each failure mode;

To implement the FMEA, first we brainstorm on both IP and power systems failure modes, then we used five (5) months' sample failure data as input, and 20 domain experts engaged in the RPN evaluation using an Excel sheet. Based on the identified failure modes on the network, Figure 4.11 shows the failure mode functional block diagram of IP- backhaul network.

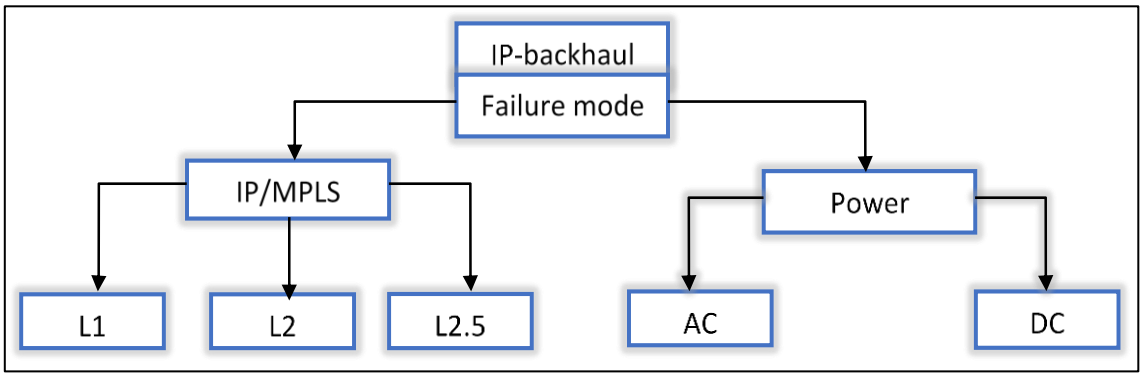


Figure 4-11 IP backhaul failure modes functional diagram

❖ *Identification of the power system failure modes*

This study did not consider the data center's power environments since it was safe and consistent, as confirmed by experts. In addition, from critical loads in the power system, we only considered the ATN (LSR), since Base Band Unit (BBU), Remote Radio Unit (RRU), Indoor Unit (IDU), and Outdoor Unit (ODU) get inputs from power and the LSRs. Besides, we excluded the accessory

loads that include lighting, socket outlets, emergency lights for the analysis. Therefore, we illustrate the identified power-system failure modes using the functional block diagram in Figure 4.12 below.

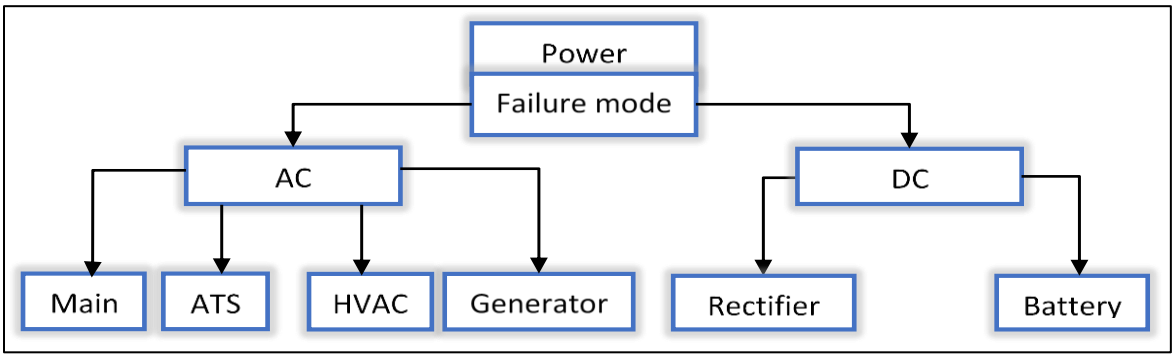


Figure 4-12 Power system failure modes functional block diagram.

After brainstorming on the power subsystem failure modes, we collected and identified the power subsystems failure modes together with power experts, and we present the detailed excels spreadsheet in Appendix A-I. Table 4.5 below shows the summary of the failure modes, their causes, and their effects.

Table 4.5 The Power system failure modes

Subsystems	Failure mode	Cause	Effect
DC	Rectifier	Overload & controller failed	Service interruption & Battery over-discharge
	Battery	Rectifier failure	Over/under voltage
AC	ATS	Overvoltage, blackout, phase loss	DG start failure, High temperature
	DG	ATS controller failure, solenoid burn, fuel system failure	Service interruption, High temperature
	HVAC	Overvoltage & fluctuation, pipe gas leakage, condenser motor blower failure, circuit board failure, and dust.	High temperature, service interruption

Later, we evaluated the failure modes' effect on global and local scales, and we were able to identify the most catastrophic effects in both scales on the system, and we summarized them in Appendix A-I in Table 2. From the evaluation result, the local effect of AC power subsystems brings more catastrophic global effects than DC on the system. However, the DC subsystems affect services. For instance, one component in the AC system might interrupt totally/partially affect

services. Therefore, to weigh their total impact finally, we did a severity analysis on components of the subsystems. Then, we figured out their RPN.

❖ *Risk Priority Number (RPN) Evaluation for power system*

After collecting and preparing RPN evaluation form for each expert, then taking their average value, we have ranked the power subsystems failure modes, considering the three risk factors:

- **Severity(S)**: the seriousness (consequence) of the failure modes
- **Occurrence (O)**: occurrence probability of the failure modes
- **Detection (D):** availability of spare parts, level of difficulties, and ability to maintain.

Based on 10-point scale risk factors, one was the lowest, and ten was the highest, except in reverse order for D. Also, if two failure modes were the same in their RPN result, then the severity determine their risk. For simplicity, throughout the analysis, we assigned failure mode 1 (FM_1) and failure mode 2 (FM_2) and referred to DC and AC subsystems, respectively.

Details of the evaluation sheet are available in Appendix-A Section I in Table 3. As a result, we identified Eight (8) potential severe power failure causes in the power system and ranked them based on their RPN values in Figure 4.13 below.

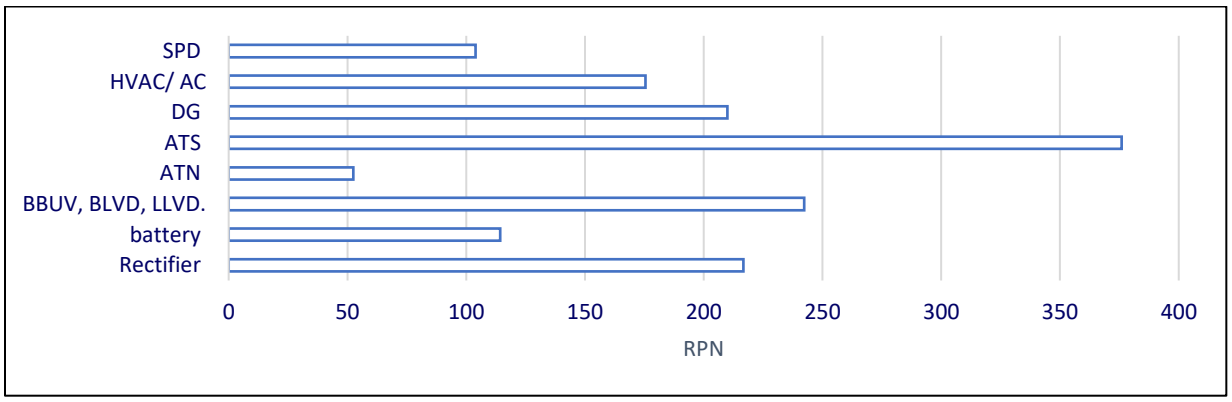


Figure 4-13 Power system RPN evaluation result.

From the result, ATS relay failure was first from the other failure causes in the AC subsystem, and the riskiest failure cause was CSU from DC failure modes.

❖ *Failure mode identification for IP-MPLS system*

From an IP-backhaul maintenance point of view, IP/MPLS contains three subsystems described in Figure 4.14 below. And, similar to the power subsystems, we followed the same procedures to

analyze the failure modes, their causes, effects, and finally, to get their RPN using ten (10) experts and considering the risk factors.

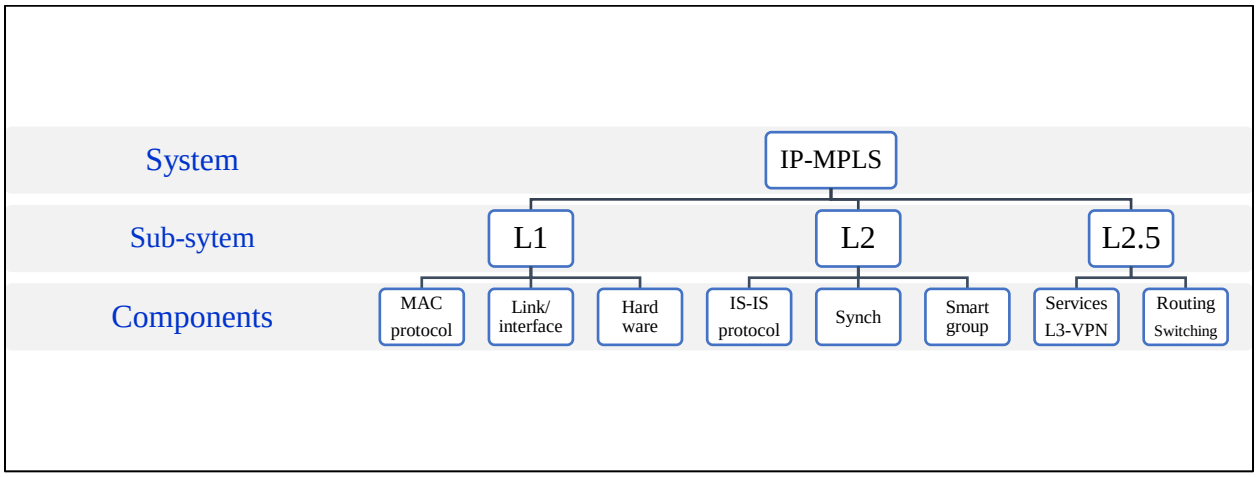


Figure 4-14 IP-MPLS system failure modes functional diagram.

Based on the failure history data, there were fifty-nine (59) alarm codes in the access and aggregation layer of the network, and from 59, thirty-nine (39) of them belong to L2.5, 10 in L2, and the rest 10 were from L1. Thus, after filtering and associating these alarm codes (names), we were able to create nine (9) main failure mode groups for RPN evaluation.

For instance, the failure mode (FM_3) has three (3) sub failure modes (causes). Also, for simplicity, FM_3, FM_4, and FM_5 are layer 2.5, layer two, and layer one subsystem failure modes in the IP-MPLS system, respectively. Table 4.6 below lists the IP service failure modes.

Table 4.6 Layer 2.5 subsystem /routing failure modes and their causes

Failure modes	Protocols	Code	Descriptions
FM_3	IS-IS	FM_{31}	Link flapping / down because of configuration mismatch
	BGP	FM_{32}	Physical connection (fiber disconnection)
	MPLS	FM_{33}	MPLS-TE / RSVP-TE configuration loss /tunnel down
	OSPF	FM_{34}	Interface state changed
	L3VPN	FM_{35}	VRF interface configuration failure

Based on the domain experts' feedback and criteria such as rare occurrence, less severity, and high detection rate, we have selected IS-IS, MPLS, and L3VPN routing protocols in the L2.5 failure modes. Then we used their RPN matrix mean values. In the same fashion, we investigated L2 and L1 subsystems' failure modes, causes, and effects, and evaluated them. See Appendix-A Section II for more details.

For instance, Table 2 in Appendix-A. II shows the average RPN evaluation in L2 from all experts. From the result, Input BW error has the highest severity and occurrence, which means an interface capacity and threshold problem were severe and happened frequently. However, they were simply detectable by domain experts. In addition, virtual configuration losses and smart group failure have often occurred because of the physical link failure, and they were less severe than the Input BW error. Based on the severity criteria, smart group failure was higher than configuration losses.

Similarly, Table 4 in Appendix-A-II shows the physical layer failure modes confirmed by the domain experts. From the identified physical layer failure modes, we have selected three sub failure modes (causes) FM_{51} (Local/remote physical interface down), FM_{52} (Optical TX & RX power mismatch), and FM_{53} (Faulty power module slot) to get their RPN evaluation result as depicted in Table 4.7 below.

Table 4.7 Physical layer failure modes mean RPN evaluation result.

Failure modes	Failure causes	Severity	Occurrence	Detection	RPN
FM_5	FM_{51}	9.333333	9.666667	7	631.5556
	FM_{52}	6.5	7	7.333333	333.6667
	FM_{53}	9.333333	2.5	5.333333	124.4444
	FM_{54}	Have lease severity and RPN values			
	FM_{55}				

From the result, physical interface and power card module-slot failures were severe than optical interface failure, and the frequency of interface failure was higher than power slot problems.

However, faulty power module slots were as severe as a physical interface failure mode. The experts could easily detect optical power mismatch and physical interface than power module-slot failure modes. These two failure modes were riskier than power module slots in the physical layer failure modes of the system. We also asked why the power failure modes were less risky than the others were. The experts confirmed that there were two power modules on the routers, which could serve as backup.

- *RPN evaluation for IP-MPLS subsystems*

After evaluating all failure modes in the IP-MPLS system, we also have calculated the overall mean RPN rank as provided in the Appendix A-II. Figure 4.15 below shows the summary of their RPN evaluation results.

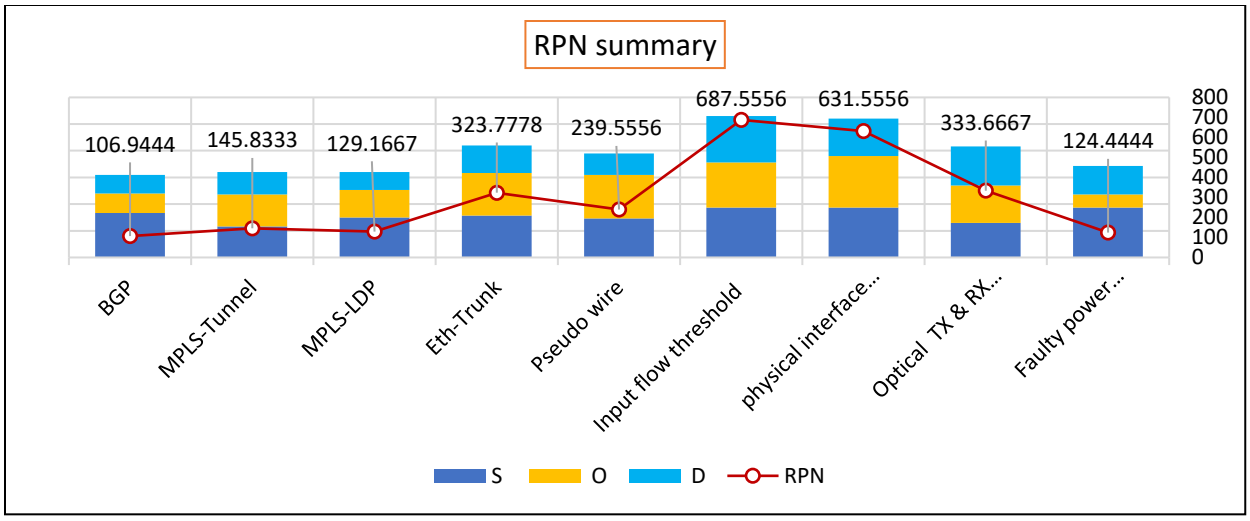


Figure 4-15 IP-MPLS system RPN evaluation summary.

We were able to identify seventeen (17) failure causes from the IP-backhaul system using FMEA. In addition, from the RPN evaluation result, we ranked the potential severe failure cause from both subsystems. When we list them in their RPN ranks FM_{21}>FM_{13}>FM_{22}>FM_{11}>FM_{23}>FM_{24}> FM_{14}, and From IP subsystem FM_4 > FM_5 > FM_3. However, in terms of their severity, FM_{43} and FM_{51} are equal, and the least RPN value is FM_{31}.

Overall, they ranked as FM_{43}>FM_{51}>FM_{52}>FM_{41}>FM_{42}>FM_{32}>FM_{33}>FM_{53}>FM_{31}. Based on these, finally, we merged, compared the results, and summarized them in Figure 4.16 below for the next-level decision-making process.

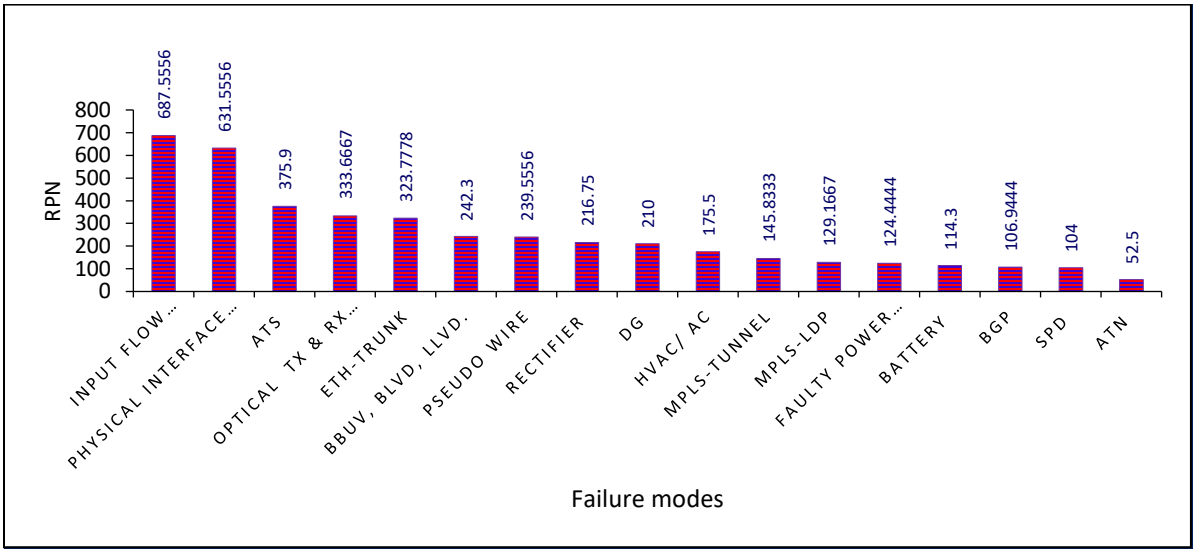


Figure 4-16 IP-backhaul system failure modes risk rank.

Even if the FMEA results were satisfactory from the failure modes identification aspect, the RPN result was uncertain and created ambiguity among the experts on the failure cause magnitudes, where some of them were highly skilled in power subsystems, and some were highly skilled in IP subsystems. To clear the ambiguity in RPN results for accurate data labeling, we needed an additional setup by studying the relationship and influence of the severe failure causes.

4.2.2.4.2 Modeling the failure modes uncertainty

To choose the ultimate failure causes and decide on the failures mode data labeling based on their interrelationship. We applied DEMATEL since it is more applicable for complex systems like IP backhaul. Following the procedure described in Section 3.2, we have used the risk analysis result captured by FMEA from the system as input to a customized DEMATEL Python script intended for similar studies on other domains.

We have used the 17 severe failure modes (FM_{xy}) as failure causes since it is better to dig deeper for decision-making while using the failure modes for classification. To prepare the failure causes influence matrix for direct relationship weighing, we set a threshold between 1 and 4 using the RPN results and criteria based on the failures' causal relationship. To mention some of these criteria used for weighting;

- BGP is an exterior gateway protocol used for route selection purposes from one autonomous system (AS) to another, which is a less influential factor in the access layer protocols IGPs, or on the MPLS, IS-IS of the network, and then we gave it the lease weight.
- From AC subsystems, ATS failure could affect the overall system. And from DC subsystems the rectifier controller circuit (CSU), was risky than the other. Then, we gave the higher weight. Also, we gave the AC (cooling system) failure causes medium weight since they are receptors of the design and load failure causes.

After collecting the weight ranks from the experts based on similar criteria as listed above, we have developed a 17 X 17 failure causes influence matrix to the DEMATEL Python script input on an Excel sheet using the mean value of the weights as depicted in the Appendix A-II.

After implementing the DEMATEL algorithm, we were able to find the most severe and influential failure causes that belong to the failure modes, as seen below in Figure 4.17.

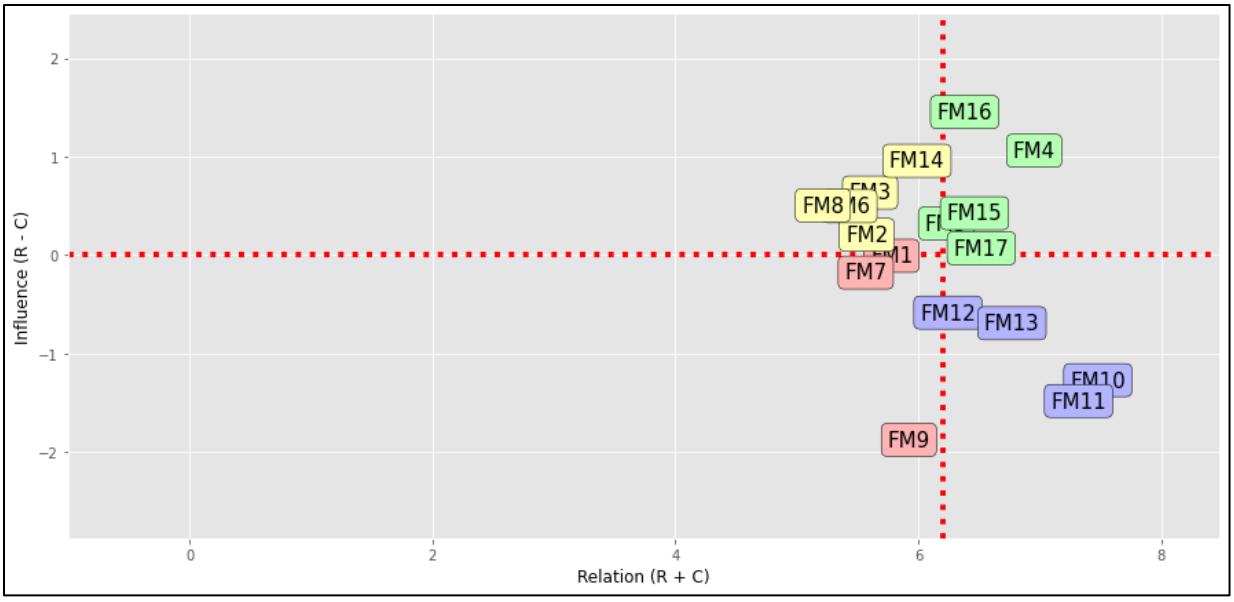


Figure 4-17 Severity influence(R-C) and relation(R+C) of the failure modes

Table 4.8 below shows the quadrants under which the failure modes lie after DEMATEL implementation and its meaning from the influence and relation aspect.

Table 4.8 IP-backhaul failure modes severity influence and relation

Quadrants	I	II	III	IV
Severity influence(R-C)	High	Low	Low	High
Relation (R+C)	High	High	Low	Low
Failure modes	FM_{52}	FM_{43}	FM_{11}	FM_{32}
	FM_{14}	FM_{13}	FM_{23}	FM_{33}
	FM_{51}	FM_{22}	FM_{31}	FM_{42}
	FM_{21}	FM_{24}		FM_{41}
	FM_{53}	FM_{12}		

The value of R-C is more effective and applicable than R+C because R-C is a good criterion or factor for failure mode prioritization. Such that, nine (9) failure causes have high severity influence on the IP backhaul network, and five (5) of them are highly related to each other.

You can find the detailed chart in Appendix A-II. Moreover, Figure 4.18 below shows the captured severity influence summary of the failure model.

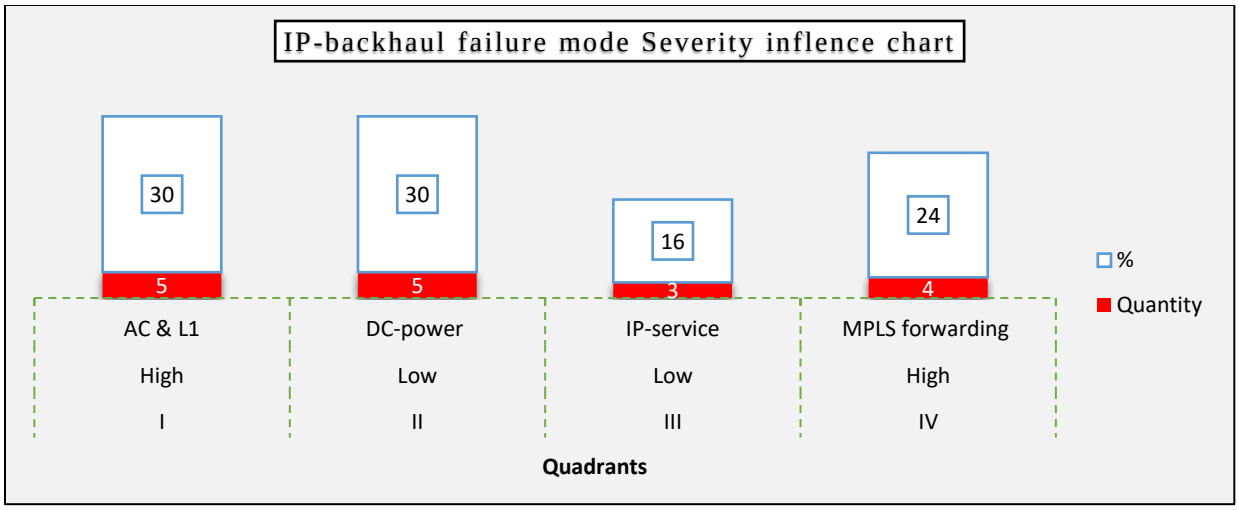


Figure 4-18 Summary of failure modes severity influence.

Based on the above result, the most severe and influential failure causes/modes lie in the physical layer failure mode and power failure modes. The second in severity influence ranking comes from the MPLS layer, followed by the Eth-trunk (virtual circuit failure). The other failure causes were highly related, but they were less influential. In other words, they were not as dominant factors as the other nine (9) failure modes that have a 54% impact on the IP-backhaul network failure aspect.

Therefore, to generate categories, map, and classify the data between power and IP failure modes, we emphasized nine (9) influential variables that belonged to quadrants I and IV.

4.2.2.4.3 Alarm category development

To generate the categories for alarm features based on the nine (9) influential variables (failure causes), we have used the 59 alarm IDs in the data to simplify the analysis. First, by assigning less influential IP failure modes to the power category. Second, filtering rare, irrelevant, and less influential 16 Alarm IDs (failures). Then we analyzed the rest of the 43 alarm IDs. Then 22 IDs represented IP/MPLS service and configuration failures, and we categorized them under class 0. In addition, we have grouped the remaining 21 physical layer and power failure IDs under class1.

4.2.2.4.4 Alarm grouping and mapping

To group and map the alarms, we have used the alarm names. Then, to assign the selected forty-nine (43) alarm categorical features into four (4) new nominal variables, we used Python group by (), map (), and dictionary () functions. We have also mapped the four severity and the two (2) PCS

alarm features into two classes using get-dummies () and label encoder functions, respectively. Details of the alarm mapping are found in Table 1 and 2 in Appendix B-III.

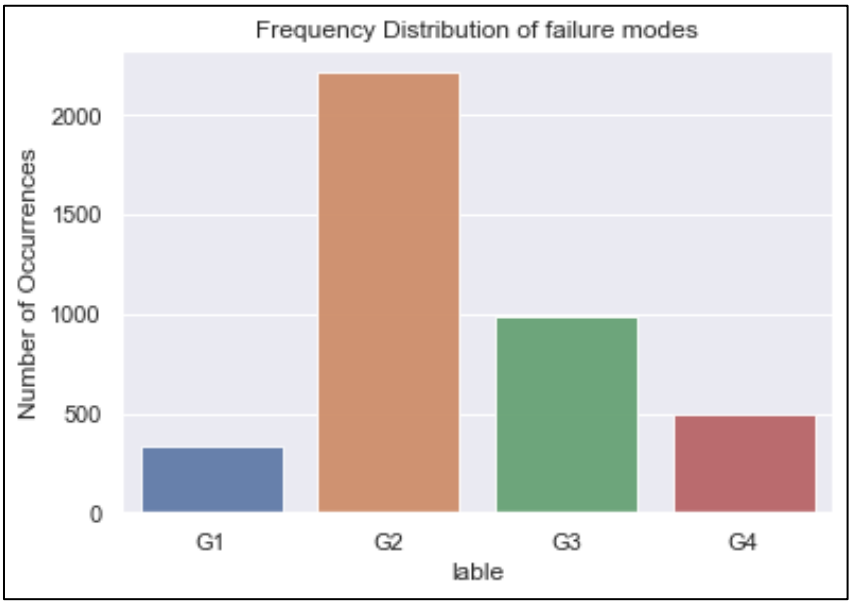


Figure 4-19 Alarm names group frequency distribution

Figure 4.19 above shows the frequency distribution of the four-alarm name groups. G1 and G2 represented power and physical layer failures, where G3 and G4 include layer two (2) and layer 2.5 failure alarms. Further, the graph shows that most alarm name features come from the physical layer, followed by IS-IS, IP services, and power alarms.

4.2.2.4.5 Criteria development and data programming

The purpose of data labeling was to differentiate the failure modes between IP (0) and power (1). Such that, we used a criteria function to decide on the binary class labels using weak supervision data labeling based on domain knowledge gained from experts, DEMATEL results, and applying user-defined functions (IF- then) conditions.

The dataset was sparse, and all performance features were at their optimal threshold values. There was also a high correlation between the KPI features. For this reason, we created seven (7) median (threshold values) from the existing features' mean value before applying the label function. And then, for comparison, we used their mode values to write the performance criteria functions (CRP). Further, we developed three (3) alarm criteria (CRA) to compare with CRP and to decide on their class label from the three (3) alarm features by comparing them with each other.

To mention some criteria used on the four (4) nominal groups and the CPU utilization features, if the alarm is in either G1 or G2, then it is in class 1(power), or if it was in either G3 or G4, then it is in class 0(IP). Similarly, if the average CPU utilization was higher than its median value, it is in class 1 otherwise 0. If the two labels were the same, then we decided on the final label. Otherwise, we decide on the instance label using majority voting and the severity of influence weights (ranks). Appendix B-V contains details of the algorithm. Table 4.9 below shows the overview of the labeling process.

Table 4.9 Design matrix labeling setup

Data	Criteria								Decision
	Performance criteria (CRP)					Alarm criteria (CRA)			Class lable
Instance	CP	CPL	CJ	CD	C..	C-AG	C-PCS	C-S	Mode (CRP) + weight (CRA)
1	1/0	1/0	1/0	1/0	1/0	1/0	1/0	1/0	1/0
2	1/0	1/0	1/0	1/0	1/0	1/0	1/0	1/0	1/0
3	1/0	1/0	1/0	1/0	1/0	1/0	1/0	1/0	1/0
.	1/0	1/0	1/0	1/0	1/0	1/0	1/0	1/0	1/0
16125	1/0	1/0	1/0	1/0	1/0	1/0	1/0	1/0	1/0

To develop the final label design matrix based on CRA and CRP decisions, we have used the label encoder function from the sklearn library, as depicted in Figure 4.20 below.

ıg 1- ı ave	Ping1-15 median	D1-15 Ave	D1-15 mean	PL1-15 median	PL1-15 mean	J1-15 Ave	J1-15 median	CPU 12 Mean	CPU 1-12 median	Memory 1-12 mean	Memory 1-12 median	AV1-12 mean	Av1-12 median	PCS	Severity Group	Alarm Group	CRA	CRP
2227	1.7833	2.50000	2.739997	0.0	0.0	3.15	2.855333	29.750000	36.0	30.083333	33.0	91.666667	100	1	G1	G3	Power	IP
2207	1.4333	1.87500	2.088333	0.0	0.0	3.21	2.965333	29.729167	36.0	30.083333	33.0	91.666667	100	1	G1	G3	IP	IP
5547	1.9333	2.62500	4.290560	0.0	0.0	3.15	3.362000	29.833333	36.0	30.083333	33.0	91.666667	100	1	G1	G3	IP	IP
5553	1.5833	1.83335	2.118340	0.0	0.0	3.14	3.417333	29.895833	36.0	30.083333	33.0	91.666667	100	1	G1	G3	IP	IP
3353	1.7667	2.50000	2.533893	0.0	0.0	3.42	3.425333	29.666667	36.0	30.083333	33.0	91.666667	100	1	G1	G3	IP	IP

Figure 4-20 Labeled design matrix.

At this level, as we can see in Figure 4.21 below, after applying One-Hot-Encoder to get their binary vector, there were 20 features, including the label column and excluding the time index in the design matrix.

	PiA	PiM	DA	DM	PLM	PLA	Jav	Jm	CPUA	CPUM	...	MEMM	Avm	AvM	PCS	SVG	AG1	AG2	AG3	AG4	label
0	1.752227	1.7833	2.50000	2.739997	0.0	0.0	3.150	2.855333	29.750000	36.0	...	33.0	91.666667	100.0	1.0	0.0	0.0	0.0	1.0	0.0	1
1	1.687222	1.6958	2.34375	2.577081	0.0	0.0	3.165	2.882833	29.744792	36.0	...	33.0	91.666667	100.0	1.0	0.0	0.0	0.0	1.0	0.0	0
2	1.622217	1.6083	2.18750	2.414165	0.0	0.0	3.180	2.910333	29.739583	36.0	...	33.0	91.666667	100.0	1.0	0.0	0.0	0.0	1.0	0.0	0
3	1.557212	1.5208	2.03125	2.251249	0.0	0.0	3.195	2.937833	29.734375	36.0	...	33.0	91.666667	100.0	1.0	0.0	0.0	0.0	1.0	0.0	0
4	1.492207	1.4333	1.87500	2.088333	0.0	0.0	3.210	2.965333	29.729167	36.0	...	33.0	91.666667	100.0	1.0	0.0	0.0	0.0	1.0	0.0	0

5 rows × 21 columns

Figure 4-21 The encoded design matrix.

4.2.2.4.6 Lable criteria evaluation

To eliminate a high probability of misclassification of the minority class over the majority class and overcome over-fitting and under-fitting problems. We have evaluated the label imbalance within the groups and between target classes using the event rate [50]. In imbalanced datasets, there is a 1:100 or 1:1000 skewness ratio between minorities and the majority class. Based on these criteria, we calculated the event rate (Er) from the distribution of minority class (M) over the majority class (m) as:

$$Er = \frac{M}{m} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots .Equation\ (4.4)$$

Thus, the event rates within the first and second-class were 0.66 and 0.33, respectively, which shows that 66% of the data labeling results come from layer1 alarms in the power failure modes. Also, 33% were from MPLS failure alarms in the IP class.

- IS-IS and layer one (1) alarms have a majority distribution in their respective classes;
- Power and IP service alarms have a minor distribution in their respective classes;
- The event rate between IP and power classes was 0.56. That is a 1:1.25 ratio.

Based on the two criteria results, the data skewed somewhat within a class. However, there was no skewness between the IP and power classes. Therefore, we have fulfilled the criterion of data processing in supervision models for binary classification problems.

4.2.2.5 Data Standardization

We standardized all features to their float values to reduce the memory and computational cost of the optimizer. Besides, we bring all attributes to the same scale [1, -1] to eliminate poor data

modeling using Z-score Normalization (standard-normal) Equation 4.5 before data segmentation. Figure 4.22 below shows the standardized design matrix after applying the sklearn standard scale.

$$f_{norm(i)} = \frac{(f(i) - \mu_f)}{\sigma_f} \dots\dots\dots\dots\dots\dots\dots\dots (4.5)$$

Where f is the features in the design matrix.

	PiA	PiM	DA	DM	PLM	PLA	Jav	Jm	CPUA	CPUM	...	MEMM	Avm	AvM	PCS	
0	-0.490042	-0.256142	-0.195501	-0.323706	-0.072276	-0.104926	-0.655837	-0.497418	-1.004816	-0.872981	...	-0.052832	-0.817335	0.0	0.493596	-0.6
1	-0.530186	-0.307809	-0.266586	-0.405182	-0.072276	-0.104926	-0.604767	-0.475944	-1.006296	-0.872981	...	-0.052832	-0.817335	0.0	0.493596	-0.6
2	-0.570331	-0.359475	-0.337672	-0.486659	-0.072276	-0.104926	-0.553696	-0.454470	-1.007777	-0.872981	...	-0.052832	-0.817335	0.0	0.493596	-0.6
3	-0.610475	-0.411141	-0.408757	-0.568135	-0.072276	-0.104926	-0.502626	-0.432996	-1.009258	-0.872981	...	-0.052832	-0.817335	0.0	0.493596	-0.6
4	-0.650620	-0.462808	-0.479843	-0.649611	-0.072276	-0.104926	-0.451555	-0.411522	-1.010739	-0.872981	...	-0.052832	-0.817335	0.0	0.493596	-0.6
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
16120	0.093228	0.176911	0.088841	0.069160	-0.072276	-0.104926	0.876278	0.048672	-1.905115	-4.017934	...	-0.052832	-4.903177	0.0	-2.025948	-0.6
16121	0.192720	0.270397	0.231013	0.171543	-0.072276	-0.104926	0.748602	0.031493	-1.650425	-3.803505	...	-0.052832	-4.569568	0.0	-2.025948	-0.6
16122	0.292212	0.363884	0.373184	0.273927	-0.072276	-0.104926	0.620925	0.014313	-1.395735	-3.589077	...	-0.052832	-4.235960	0.0	0.493596	-0.6
16123	0.391704	0.457371	0.515355	0.376311	-0.072276	-0.104926	0.493249	-0.002866	-1.141045	-3.374648	...	-0.052832	-3.902352	0.0	0.493596	-0.6
16124	0.491196	0.550857	0.657526	0.478694	-0.072276	-0.104926	0.365573	-0.020045	-0.886355	-3.160219	...	-0.052832	-3.568744	0.0	0.493596	-0.6

16125 rows × 21 columns

Figure 4-22 Normalized design matrix.

4.2.2.6 Data Segmentation

Data segmentation is the means of information mining from an array of data using window slicing and wrapping methods to train CNN image classifiers. The performance of the classifier models depend on the purpose of the study and precise windows criteria selection[18],[51],[35].

In this study, we used fixed window-slicing technique to extract fixed window slices of 2D images from multi-variate, highly correlated features, and sparse array of data to discriminate them in image form using 2D-CNN. In addition, we have considered the following key parameters while selecting the windows' size:

- *The objective:*
  - To classify the failure modes daily (maximum maintenance hour);
- *The data size and sampling interval:*

We measured the KPI data every 1sec interval (1Hz). In addition, we collected the average values of the data every one (1) hour interval, and it contains 370,875 sample points. Windows size is the number of data points in a time window with a given sampling rate. Mathematically, the sampling theorem is:

$$Frame\ size(T) = \frac{N}{Fs} = N * \Delta t \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (4.7)$$

Where *N* is the number of block size in one frame, which is equal to 24 hours in our case.

Based on the minimum sampling rate, we were able to extract 24 window frames (slices) using all 20 features in 24 hours. Thus, the time window in one day(T_{1D}) was:

- T_{1D} = 1 x 24 x 3600 = 86,400sec = 24 hr time steps.

- *Overlapping ratio (OVR):*

Most studies used ½, ¼, and ¾ overlapping ratios, depending on their data nature, to satisfy the sampling theorem. So, we used a 50% overlapping ratio to increase the size of the data and maintain information between consecutive frames, and then to keep the model's generalization ability.

To get feature space and target variable from the design matrix in NumPy array format (2D gray-scale images), we used SciPy modules from stat-library, get-frame () function from Python, and TensorFlow libraries to slicing the data in 24 hours using the constant variables (window parameters).

At this point, from 1-D 16125 instances and 20 features vectors (370,875 data points), 1342 gray-scale images with 1342 labels each having 20 features within 24 hours window segments retrieved with their most often used label in the segment. Figure 4.23 displays four (4) IP and four (4) power 24 hour labeled window samples horizontally, as well as the 20 features in each frame vertically.

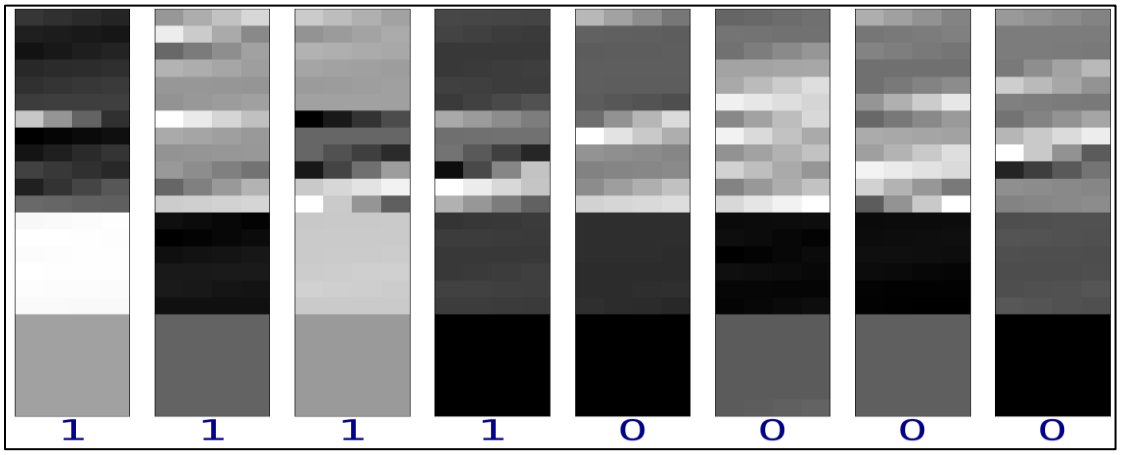


Figure 4-23 Random IP (0) and power (1) failure mode frames.

Finally, we saved the sliced train 80% and test 20% frames in separate folders in CSV format using the sklearn random-train-test-split function. As can be seen in Table 4.11, to validate and test the model with three configurations, 1073 and 269 frames were prepared respectively.

Table 4.10 Test frames slice for model evaluation

Training window sizes	Experiments	Features	Test windows slices
[1073, 24, 20]	1	Design matrix	[269, 24,20]
	2	KPI (Link & device)	[269, 24,14]
	3	Alarm	[269, 24, 6]

The selected CNN architecture should train on three-dimensional (3D) input shapes. Consequently, we reshaped both the validation and test 2D array segments to their 3D versions. For instance, by adding one feature space for all training features to find 24, 20, 1 input shapes for CNN. After we completed data segmentation, finally, the data frames have been ready for modeling.

5 Model Training

After we finished knowledge-based data preprocessing, we have implemented the 2D CNN model using tensor flow and Keras deep learning libraries. and we carried it out on Jupiter-notebook using a laptop with dual-core CPUs and 8GB RAM as follows.

5.1 CNN Setup

The 3D input shape of the IP backhaul data has three values (24, 20, 1), which represent width (w), height (h), and the number of channels of the frame(C). W represents the classification time window, h represents the number of features in the design matrix, and C=1 shows gray-scale images. Hence, we defined the input shape of the first layer 2D- CNN sequential model with 16 filters and receptive fields that were 3 pixels wide and 3 pixels height and using a default stride width of 1 to fit with the 24×20×1 (width × height × channel) of the input.

The selected CNN model architecture was 2D sequential. In addition, we changed the default parameters to fit the IP-backhaul data shape and to meet the study aim. Further, we have replaced SoftMax and sparse categorical cross-entropy functions with sigmoid and binary cross-entropy functions respectively in the output layer of the network to make it suitable for the binary classification problem before fitting the model. Figure 5.1 below shows the simplified architecture and changed parameters of the network. We defined two convolutional layers that have 48 total filters in both layers.

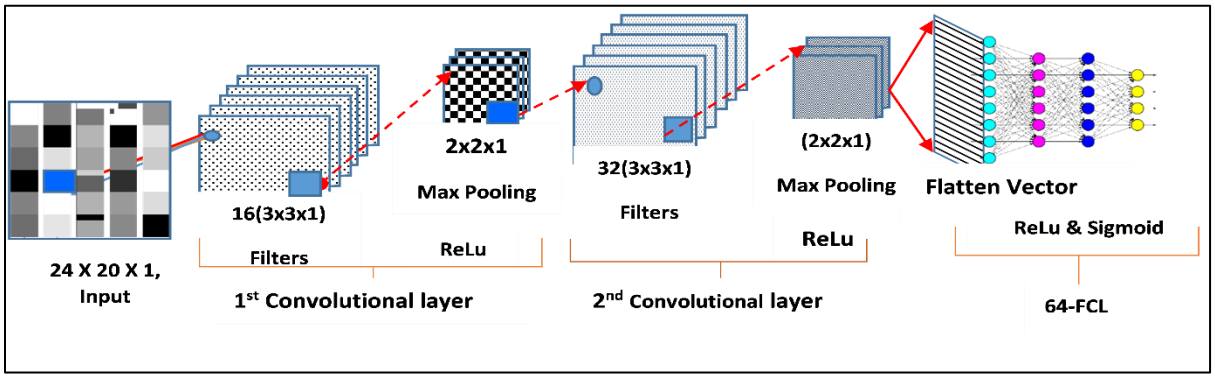


Figure 5-1 Simplified architecture of sequential 2D-CNN.

After we configured the model, the first convolutional layer contains a (3x3) size 16 filters with the same padding to learn 20 unique features from the input 20x24 matrix, and the output of this

layer is a 22x12x16 matrix of neurons. Also, the second convolutional layer contains (3x3) 32 kernels that enable the network to pick up higher-level features missed in the First CNN layer. The output of this layer is a 20x10x 32 matrix. Both convolutional layers followed by a 2x2 size max-pooling to reduce the size of the output matrix to 1/2 of the input matrix with ReLu activation functions to reduce the complexity of the output and prevent over-fitting the data.

Moreover, the network contains one flatten layer that cascades the filtered 3D feature vectors into 1600 1D vectors to feed into a fully connected neural network (MLP). Also, there are 64 dense layers (FCLs) and one (1) final dense layer to classify the 1D-feature vectors with ReLu and Sigmoid activation functions, respectively. The final dense layer determines the class probabilities prediction using binary cross-entropy loss function. Overall, the network has 107,329 total trainable parameters.

5.2 Parameter Selection and Tuning

We integrated Grid search (GS) from the sci-kit-learn library with Keras API on tensor flow automatically configured with K-fold cross-validation for 2D-CNN classifier.

We implemented cross-validation (G-CV) to split the dataset in 5-partitions, fit the model with 4/5 of the dataset, and validate the model with 1/5 of the data K times (where k = 5 & 10 iterations). We also applied grid-search to fit the model with an overall grid of hyper-parameters for each iteration in a brute force manner, trying every combination that maximizes the accuracy and minimizes the variance of the model. Then it fine-tunes the model to illuminate the model's bias-variance trade-off.

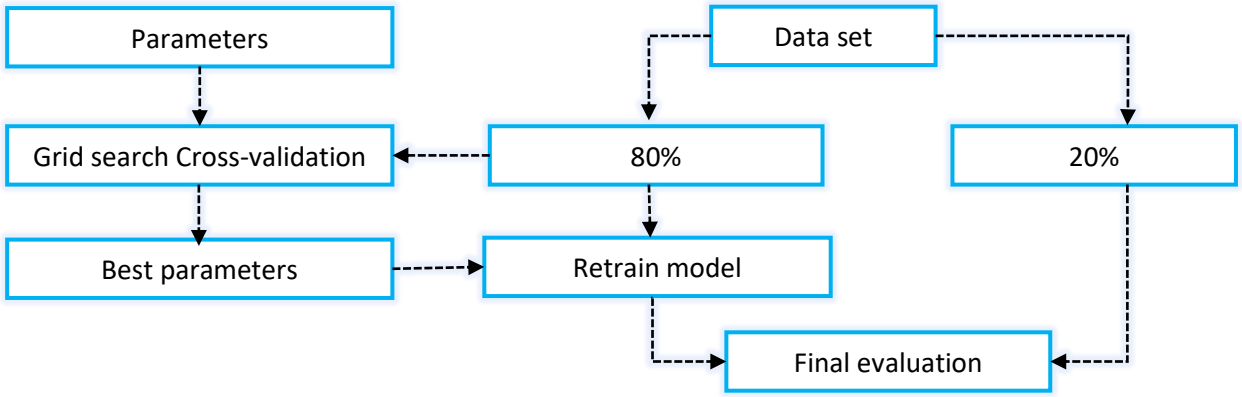


Figure 5-2 Grid search Cross-validation pipeline[52].

In this study, we used both K=5 and k=10. We used K=5 to find most parameters except for the number of epochs, batch size, and best optimizer in the first and second run time. Besides, as the number of candidate parameters and run-time increase, the grid search algorithm interrupts even using five (5) folds. However, we found successful outputs by setting 64 and below batch size with 64 epochs. Moreover, the number of iteration, batch size selection, and CPU power determine the training effectiveness. The grid search algorithm tests the dictionary of parameters specified in Table 5.1 below.

Table 5.1The model candidate parameters and values.

Steps	Hyper Parameters	Candidate values				
1	Batch size	16	32	64	128	
2	Epoch	10	14	20	25	
3	Optimizer	Adam	RMSprop	SGD	Adadelta	Adagrad
4	Learning rate	0.0001	0.001	0.01	0.1	
5	Momentum	0.2	0.3	0.6	0.8	0.9
6	Activation function	ReLu	SoftMax	Soft plus	Sigmoid	tanh
7	Dropout rate	0.1	0.2	0.3	0.4	0.5
8	Neurons in the hidden layer	8	16	32	64	128

For example, the first and second iterations of the grid search results are displayed in Figure 5.3 (left) and (right), respectively. The first graph shows the model accuracy scores for various batch sizes and epoch counts, while the second graph depicts the model standard deviation for five optimizers.

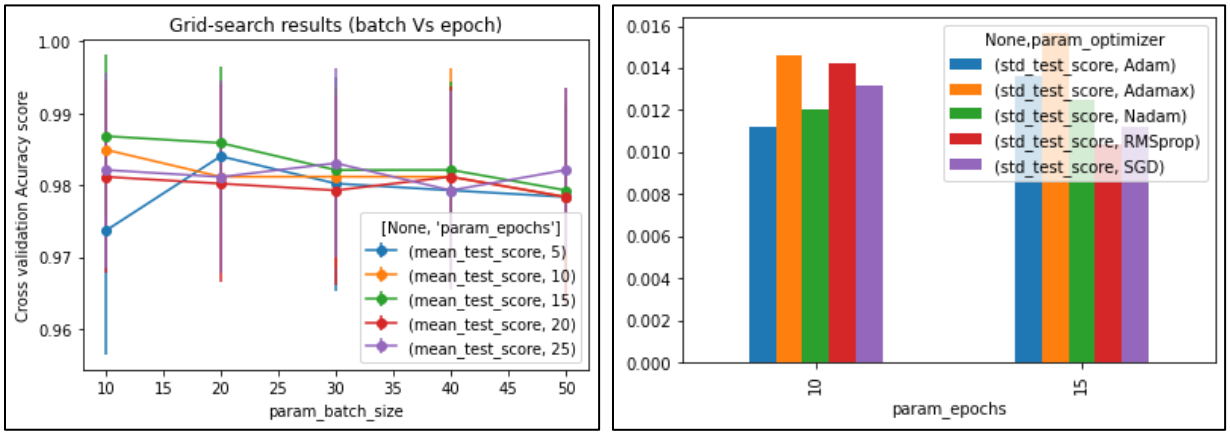


Figure 5-3 Grid search results for batch size vs. epochs (left) and optimizer (right).

From the candidate parameters, 15 epochs and 10-batch size with a 0.001 learning rate, the model's mean accuracy reached 0.986. Thus, these were the selected hyper-parameters at first and were ready to be tuned in the next runtime to identify the best optimizer among the five candidates, as depicted in the above Figure (right). In addition, the training data was so small to train the model longer with a batch size of 10. However, its standard deviation shows that the model was stable to converge using Adam with 15 epochs and a 0.3 dropout rate. As a result, we selected the Adam optimizer that scored 0.97 mean accuracy and 0.014 standard deviation from the second iteration.

Furthermore, the model's accuracy has decreased with the learning rate, and when we reduce the batch size and epoch, it has shown improvement. These show that the model has learned the data quickly and converges to global minima. Further, the model was over-fitted. Hence, to reduce over-fitting, fine-tune the model, and find optimal parameters, we ran the G-CV five times using k=5 and 10. In Appendix B-I, details of G-CV result plots are available. Eventually, we summarized the best scores in Table 5.2 below.

Table 5.2 The selected model parameters

M	K	P	F	M	σ	E	η	B	O	D	T
1	10	25	250	0.986	0.023	15	10^{-3}	10	SGD	0.5	5h
2	10	6	60	0.972	0.014	15	10^{-3}	64	Adam	0.3	1h
3	5	250	1280	0.987	0.009	25	10^{-1}	64	Adam	0.3	16h
4	5	36	180	0.988	0.007	25	10^{-1}	64	Adam	0.25	3hr
5	5	20	100	0.989	0.006	25	10^{-2}	32	Adam	0.15	2hr

M = model, K= the number of iteration, P= parameters, F is Fits, μ is the mean values, σ is standard deviation, E is the number of epochs, alpha(η) is the learning rate, B= batch size, o = optimizer, D stands for dropout rate, T stands for run time.

From the above five models 320 candidate parameters, after fitting the model 1870 times using 10 and 5 folds, for 27 hours successful runtime, we selected the last five (5) hyper-parameters that score 98.9% mean accuracy and a 0.006 sigma values.

In this study, we used the Grid search only to select and tune the optimal parameters. Therefore, we saved the optimal parameters to retrain the model once.

5.3 Model Validation

To validate whether the model was underfitting, overfitting, or a good fit and improved its loss and/or accuracy. We compiled the model using the Adam optimizer with a 1e-2 learning rate,

binary cross-entropy loss function, accuracy metric, and other optimal parameters as soon as we finished model tuning. In addition, we defined early stopping regularization from the Keras call - back library to interrupt the training and save the physical model with five (5) epochs of patience to the model's loss.

After fitting the model on 1073 training samples for 10 extra minutes using 5 fold, 32 batch size, and 0.15 dropout rate, we assessed the model's performance using the training and validation learning curves as depicted in Figure 5.4 below, which shows the learning and generalizing ability of the model.

- ***Observation***

We can see the model bias and variance trade-off from the training loss and performance learning curves with epochs in Figure 5.4 below. The training loss curve did not remain flat and decreased continuously until the end of the training. Consecutively, the training and validation losses have continued to reduce to a stable point without oscillation, and the model had converged after 14 epochs. In addition, the model did not over-fit because of; a 0.017% error variation observed between the validation and training. Thus, it was a good fit.

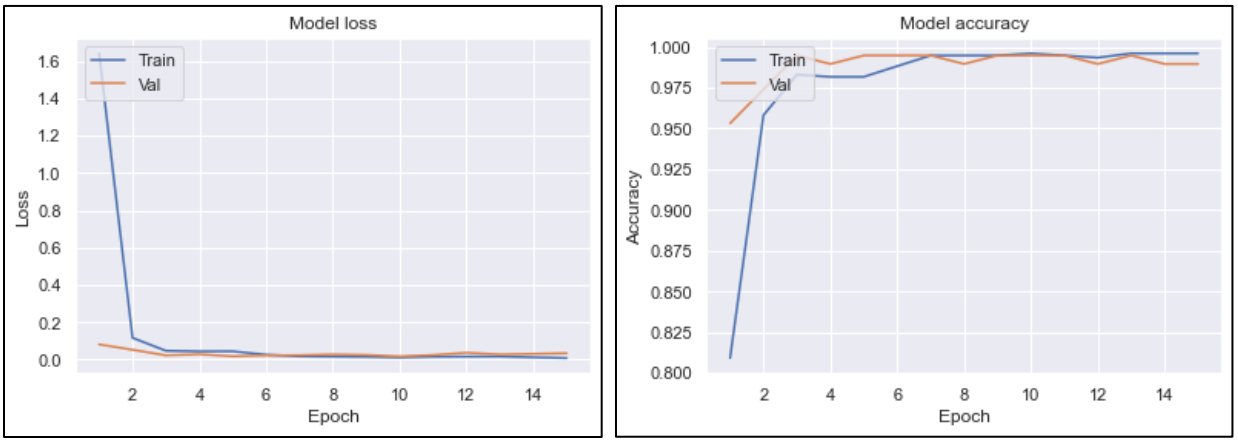


Figure 5-4 Learning curve of the model left (loss) and right (performance).

Moreover, from the training and validation accuracy learning curve, performance of the model has reached 0.98 validation accuracy. As a result, the model has learned well the complexity of the dataset (it was not under fitted), or it has low bias. Thus, we could prove that the learning ability of the model on the data was higher, and it was a good fit.

Overall, the model was able to learn the problem well and faster from the training set with 10^{-2} learning rate; this suggests that a moderate learning rate would be 0.01 to test the generalization ability of the model. Such that, we saved the model configuration and weights to evaluate the model.

5.4 Model Evaluation

After validating the model, it was necessary to measure the generalization ability of the trained model using the saved model parameters with all test frames kept separately from the training phase. We also separately examine the impact of alarm and KPI data set performance to capture which data sets can more classify failure modes and second, to verify which data set is suitable and applicable for optimal decision making in the same 2DCNN architecture.

We evaluated the models using the F1 score to find the optimal outcome, as it is a measure of the balance between recall and precision for an unequal distribution. In addition, the F1score takes into account FP and FN, which also have an equal impact on the fairness of model. In addition, to compare the models, we used the weighted average of F1score because; it gives different weights to unbalanced classes to fairly calculate their loss.

However, the macro average treats the two classes equally regardless of the imbalance between the classes. Therefore, it only describes the overall performance of the model. In addition, to find a more optimal result, we used ROC-AUC because it is scale and threshold invariant, and it eliminates the compromise between FN and FP.

6 Result and Discussion

6.1 Model Result on all Features

In this study, the negative and positive classes for the suggested analysis were IP and Power, respectively. In addition, we used FP (Type1) and FN (Type2) errors to check whether the classification was good or bad. When a power failure mode was classified as IP instance, it was FP, and when IP failure mode was classified as power cases, it was FN.

During all the feature tests, it took 1 second to generate a classification report from the confusion matrix in 269 test frames. As can be seen from the confusion matrix results in Figure 6.1 below, there are 106 IPs and 163 power failure modes in the 269 test samples.

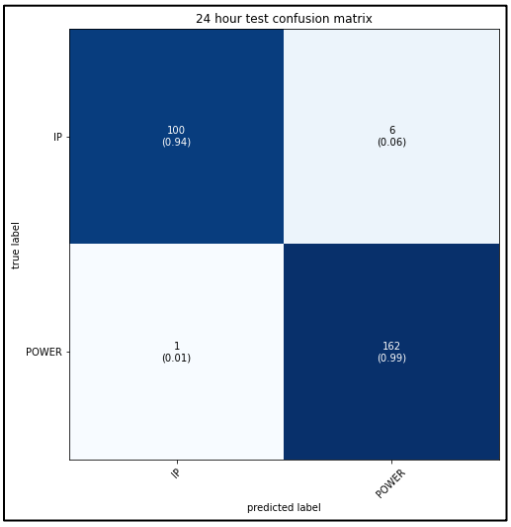


Figure 6-1 Confusion matrix of all features.

In addition, we extracted the indicators of the model's accuracy rate (error rate), TPR (sensitivity / recovery rate) and FPR (1- specificity). Also, the correct classification rate is the sum of the numbers on the diagonal divided by the sample size in the test frame. That is, the error rate of (100 + 162) / 269 = 97%) is 0.1023.

- *Interpretation 1/confusion matrix /*

From 269 total test frames, the model misclassified 7 frames of the two failure modes with a 7% error rate, which means that the model could correctly classify 93% failure modes. The report below in Table 6.1 also shows the main classification metrics calculated using TP, TN, FP, and FN.

Table 6.1 Classification report for all features

	Recall	Precision	F1-score	Support
IP	0.99	0.94	0.97	106
Power	0.96	0.99	0.98	163
Accuracy			0.97	269
Macro average	0.98	0.97	0.97	269
Weighted average	0.97	0.97	0.97	269

- *Interpretation 2 /F1-score/*

When we evaluate the model using recall and precision, we have faced a trade-off of minimizing and maximizing these metrics. However, based on the F1-score (harmonic average), the model could classify more power classes than IP with a difference of 1%, interestingly the model has achieved equal 97% macro and weighted average scores.

Apart from F1score, the model's AUC provides an aggregate measure of performance to all classification thresholds and scales because it shows how the model can separate the two classes correctly and differentiate categories in one class.

The vertical and horizontal axis of the ROC-AUC graph below in Figure 6.2 represents the TPR and FPR score thresholds, respectively. The higher the x-axis shows higher FP than TP, and the higher y-axis shows higher TP than FN. In addition, an AUC of 0.0 is 100% wrong, and an AUC of 1.0 is 100% correct. Also, the diagonal line y = x represents the class's random guess, and the vaster area under the curve, the better the separation between different groups.

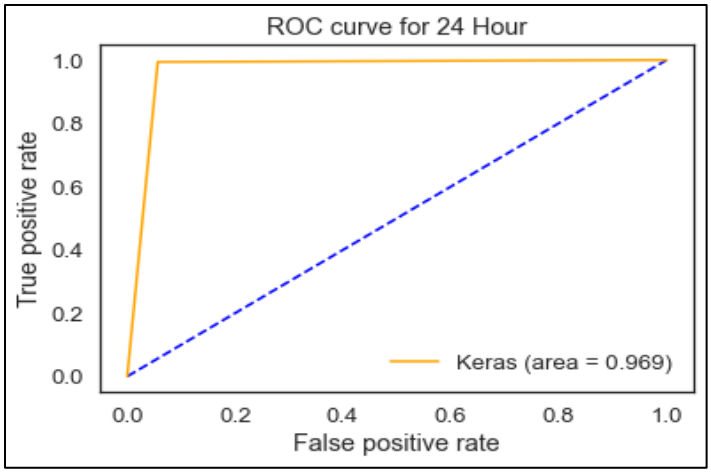


Figure 6-2 ROC & AUC result from all data set.

❖ *Interpretation: 3/ROC-AUC/*

The model shows 0.96 TPR and 0.1 FPR. As a result, the model was close to perfection because; the point (0, 1) on ROC represents the perfect model. Also, the model correctly captured 96% of the IP failure modes from TPR (True IP cases), with 1% FPR (True power cases). In addition, A 0.969 AUC shows the model's ability to capture and separate the groups correctly in the labeled frames.

6.2 Model Result on the Selected Features

During the selected features test, the pre-trained 2D-CNN algorithm has run 35ms to get the confusion matrix and therefore the classification report, employing a similar procedure on all features tests. First, we evaluated the 14 features within the KPI data before the six (6) alarm features; however, we put their confusion matrices side by side for comparison in Figure 6.3 below.

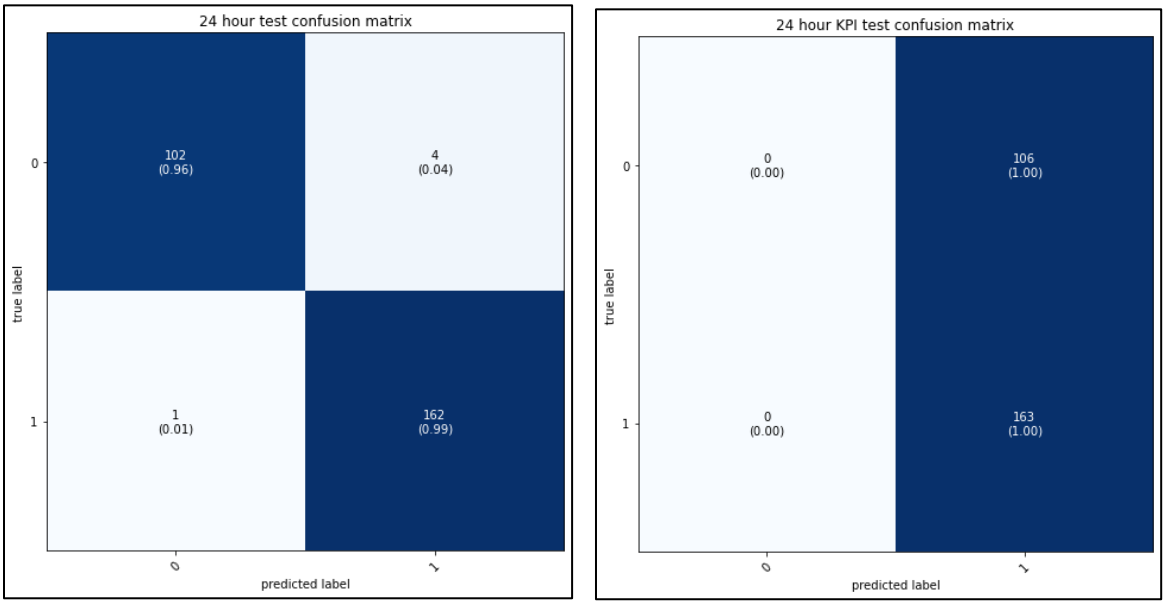


Figure 6-3 Confusion matrix result of alarm (left) and KPI (right) model.

❖ *Interpretation 4: confusion matrices*

When we look at the alarm model's performance, we can observe that it could classify both failure modes from 269 test frames with 0.98 overall accuracies and a 0.0242 error rate. The KPI model gets 0.61 overall accuracies and a 0.394 error rate. However, out of 102 actual IP failure events, the alarm model misclassified 4 IP frames as a "power failure mode" with a 4 percent error rate.

On the other hand, the KPI model only identified all 163 power failure scenarios with 100% accuracy. To be sure, look at the F1score results in Table 6.2 from the below classification report.

Table 6.2 The two models F1 score report

	f1-score		Support
	Alarm	KPI	24 hours
IP	0.98	0.00	106
Power	0.98	0.75	163
Accuracy	0.98	0.61	269
macro average	0.98	0.38	269
weighted average	0.98	0.46	269

❖ *Interpretation 5: F1-scores*

We can see from the classification reports in Table 6.2 that the model could classify failure modes with 61% overall accuracy using only KPI data, but only with a 75 percent F1-score for the power class. When compared to alarm features, the model's overall performance drops by 52%. With this in mind, we attempted to capture their performance using ROC-AUC for ultimate fairness comparison reasons. Figure 6.4 shows the ROC-AUC results from sklearn for the two classifiers.

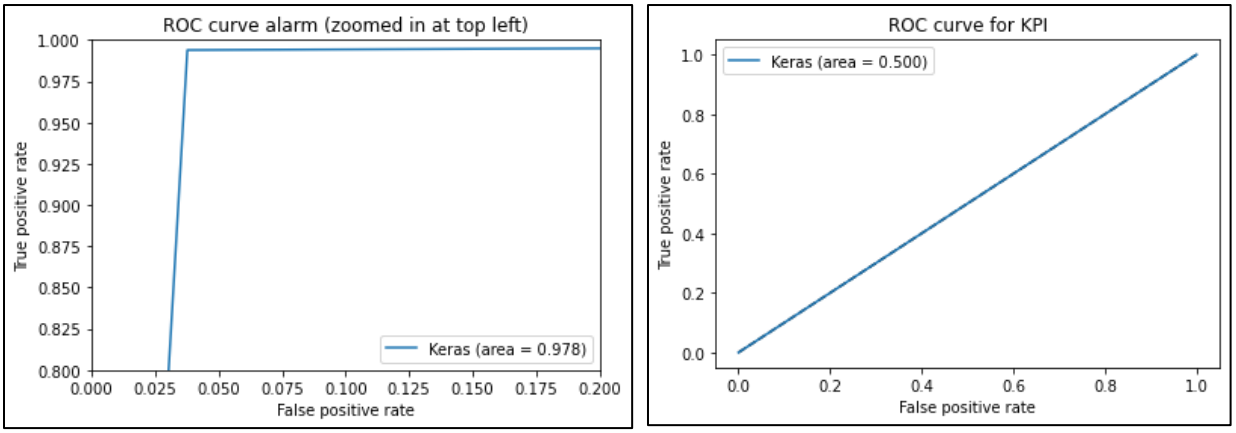


Figure 6-4 ROC-AUC result of the alarm (left) and KPI (right) models.

❖ *Interpretation 6: ROC-AUC*

Based on the ROC KPI in Figure 6.4 above, the point (0.5, 0.5) on the ROC represents a random classification. Therefore, the KPI model estimated TPR and FPR with 0.5 AUC. The model correctly captured 50% of the two failure modes of TPR (Real IP Case) and FPR (Real Power Case). Also, the AUC of the model is narrower at 0.5. The KPI model has a high bias, which reduces its generalization capability. Therefore, we retrain the model using the parameters

specified in Appendix II to improve its optimality, and then increase the validation accuracy of the model from 0.61 to 0.8, with a loss of 0.23. However, the model was inconsistent, required more training time, or the algorithm could not converge to the local optimum.

Meanwhile, the alarm model has scored a 0.978 TPR and a 0.03 FPR, which is close to perfection (the point at (0, 1)). The model has correctly captured 97% and 3% of the IP and power failure modes from FPR (truly power cases) and TPR (truly IP cases), respectively. Also, the AUC of the model is wider at 0.969.

6.3 Discussion

This research was designed to determine the influence of domain knowledge and data aggregation on optimal decision-making problems of maintenance resource allocation. An optimal knowledge base binary classifier model should obtain higher scores on the F1 (fairness) and AUC-ROC (cost vs. benefit) evaluation metrics, as well as its ability to capture domain insights through data programming. In addition, the performance of each data set should be tested separately for further study to decrease the model uncertainty.

Based on the results of the reliability maintenance analysis of the power supply and IP systems in the network, the FMEA and DEMATEL tools can complement each other, such that, the study identified and captured 17 severe failures and 9 influential failure modes, which allowed us to generate 2 categories and define 8 criteria from 59 alarm IDs using weak supervision method to label the design matrix. However, using this method required a concentrated effort among maintenance sections. Besides, system knowledge was crucial to integrate data with knowledge. In the meantime, the identified risks and their influence can be used to decide on maintenance resources allocation in the company along with the existing environmental, economic, and health maintenance metrics.

In addition, we trained the 2D-CNN algorithm successfully using Grid search CV to classify failure modes between IP and source based on 24 hours of sliced and labeled window frames using domain knowledge. As a result, we achieved a training and test accuracy of 98.0% and 97.0%, respectively, although the class imbalance ratio of 0.56 between the IP and the power frame affects the model optimality to reduce power instances by 1%. Moreover, the study result indicates that many

monitoring tools and data aggregation can fill in missing values to get a complete design matrix for optimal model construction.

The most important conclusion of the F1 score is that the model can classify two failure modes with an error of only 3% in both the same and different weights, and it can distinguish the classes with a ROC-AUC of 0.967 that shows the model TPR (benefit) is higher than its FPR (cost). Together, these results effectively meet the study's first question that the model is 97% optimal.

However, from the results of the selected data model, as shown in Figure 6.5 below (right), we observe a significant difference that the KPI model contradicts the results of the alarm model. The KPI model is only 46% fair, which means its performance decreased by 51% and 52%, respectively, compared to using all the design matrix features and only alarm features. However, we should bear in mind that the KPI data helps the model in data segmentation and could help the decision makers to be fair at least 46% of the time. Again, the graph shows that the alarm data increases the model's confidence by 1% and reaches 98%. As a result, the model is more optimal.

Moreover, as illustrated in Figure 6.5 below (left), their ROC-AUC results show that the KPI model is far from perfect at 0.5, revealing that the ability of the model to capture and separate the marked frames between groups has reduced. As a result, the model is more conjecture with equal cost and benefit. However, the alarm model performed better than the two models in all metrics. As a result, this will increase the model's ability to capture the domain knowledge and separate the label data by 1%, reaching 97.8% that reduced the uncertainty towards optimal solution.

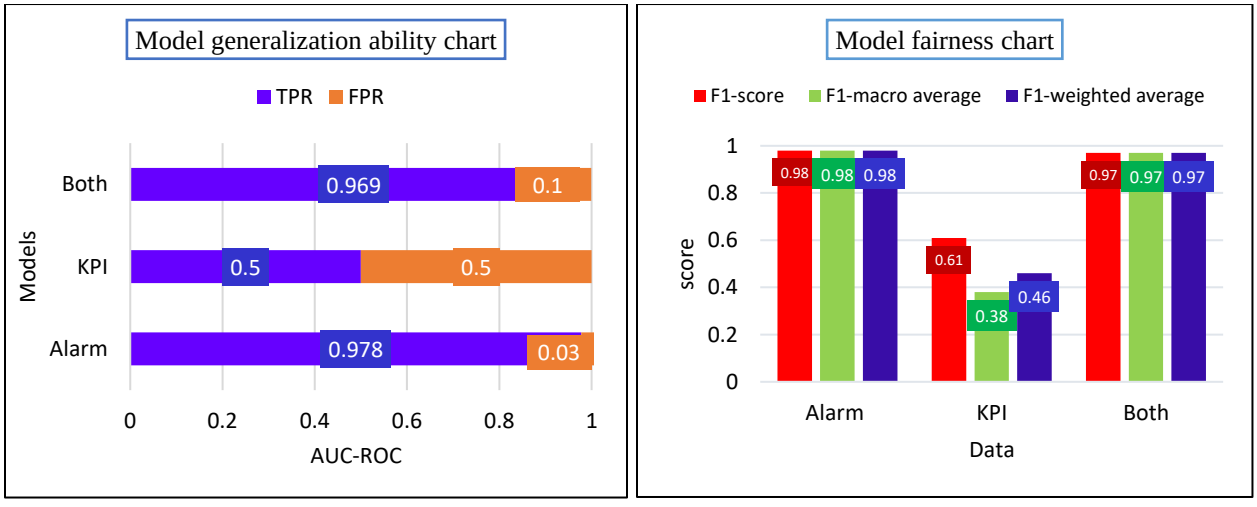


Figure 6-5 Model fairness comparison (right) and classification capacity (left) chart.

One explanation for the weak link between IP data and power is the lack of device health features, such as voltage, power, and current that were not included in this study because of the vendor's licensing issues. Furthermore, the difference in the KPI data results is due to the lack of an inherent pattern in the data to teach the model, a strong correlation between features, and slight differences between their instances. Therefore, another study that focuses more on improving the performance of the KPI model uses a dynamic threshold with further statistical analysis, or the use of the minimum and maximum feature value in feature engineering is thus recommended.

Apart from these, there was no similar model either from data or methodology aspects. On one hand, this hinders us from comparing our model with others to justify our findings more. On the other hand, to the best of our knowledge, this makes our work unique and can be used as a benchmark for IP-backhaul domain.

7 Conclusion and Future Work

7.1 Conclusion

In summary, this thesis has introduced a new 97% optimal approach to study the failure modes classification model based on domain knowledge and system data fusion, to help decision- making, related to the allocation of maintenance resources.

This study developed a design matrix from failure and system performance data in an Ethiotelecom IP backhaul network to classify IP and power failure modes using a CNN binary classification algorithm. The study has also applied reliability-based maintenance as a tool for risk assessment, feature mapping, and data programming. We also looked at the magnitudes of separate features on the model performance using sliding windows.

As a result, the risk analysis-based model is more efficient than a random allocation of maintenance resources over 80% of the time, implying that the model can reduce the unnecessary economic loss incurred by the company's inefficient decision-making framework. Also, the model can suggest ways for decision-makers to shape their verdicts and develop a rational and optimal policy.

Overall, the model result implies that the alarm data is preferable to KPI data for optimal solutions in IP-backhaul maintenance based on risk analysis. Although we have obtained substantial benchmark results and answered the research question, we therefore recommend further research, focusing more on improving the performance of the KPI data model by using the network reliability data with dynamic threshold in frequency domain.

7.2 Future Work

In the future: (i) our approach can be extended to solve multi-class problems using other CNN architectures and additional explanatory data. (ii) The results can be more reliable by using heuristic data labeling techniques with advance system knowledge.

References

[1] W. N. Cahyo, K. El-Akruti, R. Dwight, and T. Zhang, “Managing maintenance resources for efficient asset utilization,” *undefined*, Feb. 2014, [Online]. Available: https://www.semanticscholar.org/paper/Managing-maintenance-resources-for-efficient-asset-Cahyo-El-Akruti/917e1d1227af0f0798c971ee81338bffaab50650

[2] ethio-telecom, “Ethiotelecom Mobile Network Traffic & KPI Analysis from Business Perspective,” presented at the monthly CEO meeting, Addis Ababa, Mar. 04, 2020.

[3] B. Steenwinckel *et al.*, “FLAGS: A methodology for adaptive anomaly detection and root cause analysis on sensor data streams by fusing expert knowledge with machine learning,” *Future Generation Computer Systems*, vol. 116, pp. 30–48, Mar. 2021, doi: 10.1016/j.future.2020.10.015.

[4] S. Pranoto and R. Nurcahyo, “Implementation of Integrated System Six Sigma and Importance Performance Analysis for Quality Improvement of HSDPA Telecommunication Network and Customer Satisfaction,” p. 10, Jan. 2015.

[5] S. M. Seyed-Hosseini, N. Safaei, and M. J. Asgharpour, “Reprioritization of failures in a system failure mode and effects analysis by decision making trial and evaluation laboratory technique,” *Reliability Engineering and System Safety*, vol. 91, no. 8, pp. 872–881, 2006, [Online]. Available: https://ideas.repec.org/a/eee/reensy/v91y2006i8p872-881.html

[6] S.-L. Si, X.-Y. You, H.-C. Liu, and P. Zhang, “DEMATEL Technique: A Systematic Review of the State-of-the-Art Literature on Methodologies and Applications,” *Mathematical Problems in Engineering*, vol. 2018, p. e3696457, Jan. 2018, doi: 10.1155/2018/3696457.

[7] C.-C. Hung and C.-C. Hsieh, “Chapter 5 - Big Data Management on Wireless Sensor Networks,” in *Big Data Analytics for Sensor-Network Collected Intelligence*, H.-H. Hsu, C.-Y. Chang, and C.-H. Hsu, Eds. Academic Press, 2017, pp. 99–116. doi: 10.1016/B978-0-12-809393-1.00005-2.

[8] M. Fouad, N. Oweis, T. Gaber, M. Ahmed, and V. Snasel, “Data Mining and Fusion Techniques for WSNs as a Source of the Big Data,” Apr. 2015, vol. 65. doi: 10.1016/j.procs.2015.09.023.

[9] S. Cai, B. Gallina, D. Nyström, and C. Seceleanu, “Data aggregation processes: a survey, a taxonomy, and design guidelines,” *Computing*, vol. 101, no. 10, pp. 1397–1429, Oct. 2019, doi: 10.1007/s00607-018-0679-5.

[10] A. J. Ratner, C. M. D. Sa, S. Wu, D. Selsam, and C. Ré, “Data Programming: Creating Large Training Sets, Quickly,” p. 9, Jan. 2016, [Online]. Available: https://arxiv.org/abs/1605.07723

[11] A. Ratner, S. H. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré, “Snorkel: rapid training data creation with weak supervision,” *The VLDB Journal*, vol. 29, no. 2, pp. 709–730, May 2020, doi: 10.1007/s00778-019-00552-1.

[12] E. Bringer, A. Israeli, Y. Shoham, A. Ratner, and C. Ré, "Osprey: Weak Supervision of Imbalanced Extraction Problems without Code," in *Proceedings of the 3rd International Workshop on Data Management for End-to-End Machine Learning*, New York, NY, USA, Jun. 2019, pp. 1–11. doi: 10.1145/3329486.3329492.

[13] G. B. Goh, C. Siegel, A. Vishnu, and N. Hodas, "Using Rule-Based Labels for Weak Supervised Learning: A ChemNet for Transferable Chemical Property Prediction," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, New York, NY, USA, Jul. 2018, pp. 302–310. doi: 10.1145/3219819.3219838.

[14] W. Ji, S. Duan, R. Chen, S. Wang, and Q. Ling, "A CNN-based network failure prediction method with logs," in *2018 Chinese Control And Decision Conference (CCDC)*, Shenyang, Jun. 2018, pp. 4087–4090. doi: 10.1109/CCDC.2018.8407833.

[15] F. Aziz, A. Ul Haq, S. Ahmad, Y. Mahmoud, M. Jalal, and U. Ali, "A Novel Convolutional Neural Network-Based Approach for Fault Classification in Photovoltaic Arrays," *IEEE Access*, vol. 8, pp. 41889–41904, 2020, doi: 10.1109/ACCESS.2020.2977116.

[16] H. Ren, Z. J. Hou, B. Vyakaranam, H. Wang, and P. Etingov, "Power System Event Classification and Localization Using a Convolutional Neural Network," *Frontiers in Energy Research*, vol. 8, p. 327, 2020, doi: 10.3389/fenrg.2020.607826.

[17] M. S. Singh, V. Pondenkandath, B. Zhou, P. Lukowicz, and M. Liwickit, "Transforming sensor data to the image domain for deep learning — An application to footstep detection," in *2017 International Joint Conference on Neural Networks (IJCNN)*, May 2017, pp. 2665–2672. doi: 10.1109/IJCNN.2017.7966182.

[18] A. Dehghani, O. Sarbishei, T. Glatard, and E. Shihab, "A Quantitative Comparison of Overlapping and Non-Overlapping Sliding Windows for Human Activity Recognition Using Inertial Sensors," *Sensors*, vol. 19, p. 5026, Nov. 2019, doi: 10.3390/s19225026.

[19] A. Mukhopadhyay *et al.*, "A Review of Incident Prediction, Resource Allocation, and Dispatch Models for Emergency Management," *arXiv:2006.04200 [cs]*, Feb. 2021, Accessed: Feb. 25, 2021. [Online]. Available: http://arxiv.org/abs/2006.04200

[20] L. Lei, Y. Yuan, T. X. Vu, S. Chatzinotas, and B. Ottersten, "Learning-Based Resource Allocation: Efficient Content Delivery Enabled by Convolutional Neural Network," in *2019 IEEE 20th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, Jul. 2019, pp. 1–5. doi: 10.1109/SPAWC.2019.8815447.

[21] F. Hussain, S. A. Hassan, R. Hussain, and E. Hossain, "Machine Learning for Resource Management in Cellular and IoT Networks: Potentials, Current Solutions, and Open Challenges," *arXiv:1907.08965 [cs, eess]*, Jul. 2019, Accessed: May 04, 2021. [Online]. Available: http://arxiv.org/abs/1907.08965

[22] J. Zhang, W. Li, P. Ogunbona, and D. Xu, "Recent Advances in Transfer Learning for Cross-Dataset Visual Recognition: A Problem-Oriented Perspective," *arXiv:1705.04396 [cs]*, May 2019, Accessed: Nov. 26, 2021. [Online]. Available: http://arxiv.org/abs/1705.04396

[23] N. Hatami, Y. Gavet, and J. Debayle, "Classification of Time-Series Images Using Deep Convolutional Neural Networks," *arXiv:1710.00886 [cs]*, Oct. 2017, Accessed: Jan. 18, 2021. [Online]. Available: http://arxiv.org/abs/1710.00886

[24] Y. Ran, X. Zhou, P. Lin, Y. Wen, and R. Deng, "A Survey of Predictive Maintenance: Systems, Purposes and Approaches," *arXiv:1912.07383 [cs, eess]*, Dec. 2019, Accessed: Dec. 31, 2020. [Online]. Available: http://arxiv.org/abs/1912.07383

[25] S. Rastayesh, S. Bahrebar, F. Blaabjerg, D. Zhou, H. Wang, and J. Dalsgaard Sørensen, "A System Engineering Approach Using FMEA and Bayesian Network for Risk Analysis—A Case Study," *Sustainability*, vol. 12, no. 1, 2020, doi: 10.3390/su12010077.

[26] JOYCE E. MORROW and PETER J. SCHOOMAKER, "TM 5-698-4 Failure Modes, Effects and Criticality Analysis (FMECA) For Command, Control, Communications, Computer, Intelligence, Surveillance, and Reconnaissance (C4ISR) Facilities | WBDG - Whole Building Design Guide." https://www.wbdg.org/ffc/army-coe/technical-manuals-tm/tm-5-698-4 (accessed Dec. 17, 2020).

[27] H.-C. Liu, *FMEA Using Uncertainty Theories and MCDM Methods*. Springer Singapore, 2016. doi: 10.1007/978-981-10-1466-6.

[28] M. X. Cheng and W. B. Wu, "Data Analytics for Fault Localization in Complex Networks," *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 701–708, Oct. 2016, doi: 10.1109/JIOT.2015.2503270.

[29] M. G. Ölçer, "Developing a spreadsheet based decision support system using DEMATEL and ANP approaches," p. 167, Dec. 2013, [Online]. Available: http://acikerisim.deu.edu.tr:8080/xmlui/bitstream/handle/20.500.12397/7682/346156.pdf?sequence=1&isAllowed=y

[30] N. Haghighat, "Evaluating Airline Service Quality Using Fuzzy DEMATEL and ANP," *Strategic Public Management Journal*, vol. 3, no. 6, pp. 57–77, Dec. 2017, doi: 10.25069/spmj.351296.

[31] J. Heaton, "Ian Goodfellow, Yoshua Bengio, and Aaron Courville: Deep learning," *Genetic Programming and Evolvable Machines*, vol. 19, no. 1, pp. 305–307, Jun. 2018, doi: 10.1007/s10710-017-9314-z.

[32] S. Says, "Convolutional Neural Network (CNN) Tutorial In Python Using TensorFlow," *Edureka*, Nov. 27, 2018. https://www.edureka.co/blog/convolutional-neural-network/ (accessed Oct. 22, 2020).

[33] P. Galeone, *Hands-On Neural Networks with TensorFlow 2.0*, 1st ed., vol. one. Livery Place 35 Livery Street Birmingham: Packt Publishing Ltd., 2019. Accessed: Apr. 25, 2020. [Online]. Available: www.packt.com

[34] J. Dawani, *Hands-On Mathematics for Deep Learning: Build a Solid Mathematical Foundation for Training Efficient Deep Neural Networks*. Packt Publishing, 2020.

[35] J. Brownlee, *Deep Learning With Python: Develop Deep Learning Models on Theano and TensorFlow Using Keras*. Machine Learning Mastery, 2016.

[36] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv:1609.04747 [cs]*, Jun. 2017, Accessed: May 26, 2021. [Online]. Available: http://arxiv.org/abs/1609.04747

[37] "Cross Validation and Grid Search." https://amueller.github.io/ml-training-intro/slides/03-cross-validation-grid-search.html#1 (accessed Jun. 04, 2021).

[38] S. Cai, Y. Shu, G. Chen, B. C. Ooi, W. Wang, and M. Zhang, "Effective and Efficient Dropout for Deep Convolutional Neural Networks," *arXiv:1904.03392 [cs]*, Jul. 2020, Accessed: Aug. 17, 2021. [Online]. Available: http://arxiv.org/abs/1904.03392

[39] M. Bramer, "Measuring the Performance of a Classifier," in *Principles of Data Mining*, M. Bramer, Ed. London: Springer, 2013, pp. 175–187. doi: 10.1007/978-1-4471-4884-5_12.

[40] T. Fawcett, "ROC Graphs: Notes and Practical Considerations for Researchers," *Machine Learning*, vol. 31, pp. 1–38, Jan. 2004.

[41] HUAWEI Technologies co., ltd, "Ethio Telecom LTE Network Planning," presented at the LTE Project, Addis Ababa, May 20, 2020.

[42] HUAWEI TECHNOLOGIES, "Ethiotelecom LTE Expansion Project." Jan. 03, 2020.

[43] T. Kibatu, "Recurrent Neural Network-based Base Transceiver Station Power System Failure Prediction," Addis Ababa Institute of Technology School of Electrical and Computer Engineering Telecommunication Engineering Graduate Program, Addis Ababa, 2019.

[44] Eaton, "Automatic transfer switches (ATS) fundamentals | Eaton," *Fundamentals of automatic transfer switches (ATS)*, Jul. 29, 2020. https://www.eaton.com/us/en-us/products/low-voltage-power-distribution-control-systems/automatic-transfer-switches/automatic-transfer-switch-fundamentals.html (accessed Aug. 17, 2021).

[45] ZTE Technology, "ZXDU58 B121 Product Description." ZTE, Aug. 17, 2017. [Online]. Available: http://support.zte.com.cn

[46] R. Boutaba *et al.*, "A comprehensive survey on machine learning for networking: evolution, applications and research opportunities," *Journal of Internet Services and Applications*, vol. 9, no. 1, p. 16, Jun. 2018, doi: 10.1186/s13174-018-0087-2.

[47] E. Rozaki, "Network Fault Diagnosis Using Data Mining Classifiers," Apr. 2015, vol. 5, pp. 29–40. doi: 10.5121/csit.2015.50703.

[48] M. Daka, “Traffic Analysis of IP Core Networks: the Case of Ethio Telecom,” Thesis, Addis Ababa University, 2017. Accessed: Nov. 16, 2021. [Online]. Available: http://etd.aau.edu.et/handle/123456789/3493

[49] K. Habtamu, “Analysing Impact of Seamless MPLS on QoS,” Thesis, AAU, 2018. Accessed: Nov. 16, 2021. [Online]. Available: http://etd.aau.edu.et/handle/123456789/15237

[50] Upasana, “Imbalanced Data,” *How to handle Imbalanced Classification Problems*, Aug. 12, 2017. https://www.analyticsvidhya.com/blog/2017/03/imbalanced-data-classification/

[51] S. Nisar, O. U. Khan, and M. Tariq, “An Efficient Adaptive Window Size Selection Method for Improving Spectrogram Visualization,” *Computational Intelligence and Neuroscience*, Aug. 24, 2016. https://www.hindawi.com/journals/cin/2016/6172453/ (accessed Dec. 11, 2020).

[52] Tom Dupre la Tour, “Grid-search and cross-validation pactools 0.1 documentation,” *Grid-search*, Jan. 01, 2017. https://pactools.github.io/auto_examples/plot_grid_search.html (accessed Apr. 29, 2021).

Appendix-A

A–Risk Analysis Excel Sheets

I- FMEA on IP-backhaul Power System, Subsystem, Components

Table 1-power subsystems

Power system	Power subsystem					
AC	Main	DG	ATS	SPD	CB	ACDU
Cooling system	Gas pipe	Condenser	Compressor	Controller circuit		
DC	Rectifier	Battery	Controller board	Protection circuits	Circuit breakers	DCDU

Table 2 -Summary of the identified failure effects in the power system

Subsystem	Component failure modes/ causes	Local effects	Final effects
AC	ATS	Power fluctuation b/n main and standby	Total service interruption
	DG	Battery load	Higher interruption time
	SPD	Catastrophic damages to AC circuits	Total service interruption
	CB	Damages to devices	Medium interruption
	ACDB	Over or under voltage	Total service interruption
DC	Rectifier	Unbalanced load on Rectifier bank (AC-DC)	Service affected
	Battery	Be in floating charge mode & load handling capacity decreases	Service affected
	Controller board	Load sharing	Service interruption
	DC CB	Unbalanced load	Service affected
	DC DB	Unbalanced load	Service affected
Cooling	Gas pipe	Leakage	affect the services
	Condenser	Blower failed	Affect Service
	Compressor	Motor failure	Affect Service
	Controller board	Over-temperature	Service interruption

Table 3- Power system RPN evaluation and rank

Sub system	Code	Failure Modes	Component	Failure causes	Mean S	Mean O	Mean D	Mean RPN	Rank
DC FM_1	FM_{11}	DCDU	Rectifier	LLVD /BLVD	8.5	6	4.25	216.75	4
	FM_{12}		Battery	Discharge	8.75	4.75	2.75	114.3	7
	FM_{13}		BBUV, BLVD, LLVD	CSU	9.5	8.5	3	242.3	2
	FM_{14}		ATN/LSR	Software, power module	7	2.5	3	52.5	8
AC FM_2	FM_{21}	ACDU	ATS	Relay failure	8.25	6.75	6.75	375.9	1
	FM_{22}		DG	Starter battery battery charge	7	6	5	210	3
	FM_{23}		HVAC/ AC	Compressor failure	6.5	6	4.5	175.5	5
	FM_{24}		SPD	Connection, lighting	6.5	4	4	104	6

RPN ranks: FM_{21}>FM_{13}>FM_{22}>FM_{11}>FM_{23}>FM_{24}> FM_{14}

II- FMEA on IP-MPLS System, Subsystem, and Components

Table 1. Layer 2.5 subsystems mean RPN evaluation sheet

Failure modes	Failure causes	Severity	Occurrence	Detection	RPN
FM3	FM_{31}	8.333333	3.666667	3.5	106.9444
	FM_{32}	5.833333	6	4.166667	145.8333
	FM_{33}	7.5	5.166667	3.333333	129.1667
	FM_{34}	lease severity and RPN values and excluded for analysis			
	FM_{35}				

Table 2. Layer 2 subsystem failure modes and their causes

Failure modes	Forwarding protocols	Code	Descriptions
FM_4	ETH-Trunk	FM_{41}	Member down when physical connection of the link fails
	ETH_LOSS	FM_{42}	VSIs configuration failure on the local or the peer device
	Input BW error	FM_{43}	when the interface capacity less than threshold & Input flow
	Pseudo wire	FM_{44}	VPLS VC Down when physical link failed & configuration loss
	Clock	FM_{45}	When the work mode of system clock is not in locked mode
	Flow down	FM_{46}	Input threshold configuration problem
	XQoS	FM_{47}	CPU defends when packet discard passes the threshold
	CRC error	FM_{48}	Optical module disconnected or TX/RX power mismatches, patch cords (UTP cable) problem

Table 3. Layer 2 subsystem failure modes mean RPN evaluation result

Failure modes	Failure causes	Severity	Occurrence	Detection	RPN
FM_4	FM_{41}	7.833333	8	5.166667	323.7778
	FM_{42}	7.333333	8.166667	4	239.5556
	FM_{43}	9.333333	8.5	8.666667	687.5556
	FM_{44}	Least severe merged with FM_{43} (the same family)			
	FM_{45}				

Table 4. Physical layer failure modes and their causes

Physical layer	Mode	Causes	Descriptions	
Hardware	FM_5	FM_{51}	Card invalid	Physical damage and insertion problem
			Board invalid	Firmware problem
			Chassis failed	Dusty air filter
			Fan removed	Software, module
			MAC transition	MAC hooping b/n source & transit nodes
Physical port		FM_{52}	Optical fail	Disconnected
			Optical invalid	TX & RX power mismatch
			Optical remove	Module removed /failed
			Port down	Local/remote physical interface down
Power supply		FM_{53}	Power fail(entity)	Input voltage < min value
			Power invalid	faulty power module or slot

Table 5. IP MPLS RPN evaluation summary

Modes	Name	Causes	S	O	D	RPN	Rank
FM_3 Routing	BGP	FM_{31}	8.333	3.667	3.5	106.9444	9
	MPLS-Tunnel	FM_{32}	5.833	6	4.167	145.8333	6
	MPLS-LDP	FM_{33}	7.5	5.167	3.33	129.1667	7
FM_4 Forwarding	Eth-Trunk	FM_{41}	7.833	8	5.167	323.7778	4
	Pseudo wire	FM_{42}	7.333	8.167	4	239.5556	5
	Input flow threshold	FM_{43}	9.333	8.5	8.667	687.5556	1
FM_5 Physical	Physical interface down	FM_{51}	9.333	9.667	7	631.5556	2
	Optical power mismatch	FM_{52}	6.5	7	7.334	333.6667	3
	Faulty power module slot	FM_{53}	9.333	2.5	5.333	124.4444	8

III- DEMATEL on IP-backhaul

Table 1- Expert's knowledge and causal relation computation

Weights			Average weight matrix
RPN	Base line	Maximum(M)	Based on direct relation
<150	0.25	4	If FM_w is 0.5, then it is 0 or1
150- 200	0.5	4	If FM_x is 1.5, then it is 1or 2
201- MAX	0.75	4	If FM_y is 2.7, then it is 2 or 3 If FM_z is 3.5, then it is 3 or 4

The weights were assigned for the seventeen failure causes (FMs), using Table 2 below where x and y represent any two failure causes in the identified seventeen cases.

Table 2- The minimum and maximum criteria weights

Failure causes	Failure causes	Minimum weight	Maximum weight
FM $_x$	FM $_y$	1	4

Table 3- IP backhaul failure causes direct influence weight matrix

M	FM 11	FM 12	FM 13	FM 14	FM 21	FM 22	FM 23	FM 24	FM 31	FM 32	FM 33	FM 41	FM 42	FM 43	FM 51	FM 52	FM 53
FM 11	0	4	2	2	2	1	1	1	1	1	1	1	1	1	1	1	2
FM 12	4	0	2	2	1	1	1	1	1	1	1	1	1	1	2	1	2
FM 13	4	3	0	2	2	1	1	1	1	1	1	1	1	1	1	1	3
FM 14	1	1	1	0	1	1	1	1	3	4	3	4	3	1	3	1	4
FM 21	2	2	2	1	0	4	4	3	1	1	1	1	1	1	1	1	1
FM 22	1	1	1	1	4	0	4	2	1	1	1	1	1	1	1	1	2
FM 23	1	1	1	3	2	1	0	1	1	1	1	1	1	1	1	1	3
FM 24	1	1	1	1	4	2	3	0	1	1	1	1	1	1	1	1	2
FM 31	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1
FM 32	1	1	1	1	1	1	1	1	4	0	4	2	4	1	1	1	1
FM 33	1	1	1	1	1	1	1	1	3	4	0	1	4	1	1	1	1
FM 41	1	1	1	1	1	1	1	1	2	3	4	0	1	2	1	1	1
FM 42	1	1	1	1	1	1	1	1	3	4	4	1	0	2	1	1	1
FM 43	1	1	1	1	1	1	1	1	2	4	3	2	3	0	2	3	1
FM 51	1	1	1	1	1	1	1	1	2	3	4	4	2	1	0	3	1
FM52	1	1	1	1	1	1	1	1	2	4	4	4	3	2	4	0	1
FM53	2	1	2	4	1	1	1	1	2	1	1	2	1	2	3	1	0

The input FM_{11} is the same as output FM_1 and the other 16 variables were related in the same fashion as depicted in Table 4 below.

Table 4, Failure modes input (direct relation) to Dematel output (indirect relation)

IN	FM_{11}	FM_{12}	FM_{13}	FM_{14}	FM_{21}	FM_{22}	FM_{23}	FM_{24}	FM_{31}	FM_{32}	FM_{33}	FM_{41}	FM_{42}	FM_{43}	FM_{51}	FM_{52}	FM_{53}
OUT	FM_1	FM_2	FM_3	FM_4	FM_5	FM_6	FM_7	FM_8	FM_9	FM_{10}	FM_{11}	FM_{12}	FM_{13}	FM_{14}	FM_{15}	FM_{16}	FM_{17}

FM_1 is the same as FM_{11} and the same is true for other 16 variables.

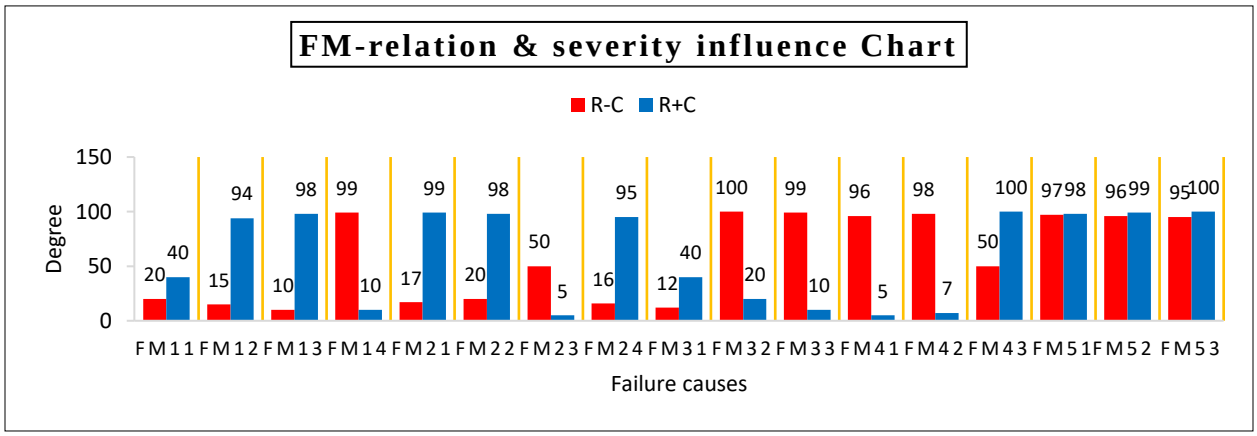


Figure 3- Failure modes R+C and R-C result chart.

IV- Category Generation Setup

Table 1- Category generation on alarm features

Alarm IDs	Categories			
Power & Physical layer alarms	Power failure modes (1)			
MPLS & IP service alarms	IP failure modes (2)			
Probable cause (PCS)	Root alarm	Correlative alarm		
Category	1	2		
Severity	Critical	Major	Minor	Warning
Category	1		2	

Table 2 Overview of the 43 IDs categorization process into the two classes.

Power(G1)	Physical layer(G2)	IP-MPLS (G3)	IP –SERVICE (G4)
3	7	2605064	1100882
2505228	2505222	2600454	1101056
2505230	2603044	1101064	2600456
1103100	2601241	2600151	2601500
Class 1		Class 2	

Table 3. Summary of design matrix formation.

Data	Performance		Alarm
Quantity	15 links	12 devices	12 devices
Raw (Rows X Columns)	16170x9x15	16180 x 4 x12	17,000 x 18
After Feature selection	16170x4x15	16180 x3 x 12	17,000 x4
After Cleaning	16125x 4x 15	16,125 x 3x12	16,607 x3
Before Aggregation	16125x 4x 15	16,125 x3 x12	16,235 x3
After aggregation	16125x 4	16,125x3	16125x3
Final Design matrix	16125 X 10		

Note: We have aggregated the alarm data while cleaning and KPI data aggregated after cleaning.

V- Design Matrix Labling Rules / Functions

Criteria 1:

Alarm names splits into four (4) groups (i.e., G1, G2, G3, and G4) based on their severity influence ranks obtained from the DEMATEL result, then using following conditions the first class labling criteria was generated as:

- *If the alarm is in either G1 or G2, then it is class 1(power failure mode).*
- *If the alarm is in either G3 or G4, then it is class 0(IP failure mode).*

Criteria 2:

The severity & PCS features from the system preserved to include the existing critical filtering policy and support the decision-making on the class lable. Severities were categorized in 2 (i.e., G1, G2) such that, based on the following three conditions, the second labling criteria was developed

- *If the alarm is in G1 or G2, & the Severity is in G1, then it is in class 1, and if it is in Severity G2, then it is in class 0.*
- *If the alarm is in G3 or G4 and Severity is in G1 and then it is in class 0.*
- *If the alarm is in G3 or G4 and Severity is in G2 and then it is in class 1.*

Criteria 3:

PCS has two values (i.e., root alarm (R) and correlative alarm (C)), and it has three (3) conditions.

- *If alarm is in G1 or G2 and PCS value is R or C, then it is in class 1.*
- *If the alarm is in G3 or G4 and the PCS is C, then it is in class 0.*
- *If the alarm is in G3 or G4 and the PCS is R, then it is in class 1.*

Criteria 4:

The mean aggregated KPI values compared with their corresponding median values, to capture their deviation and decide on their lable class based on the following conditions.

Ping and Packet loss directly related to connectivity and device performance problems respectively that means, they belong to G2 and G1 failure modes i.e., power, and physical connectivity (layer one) classified as 0.

- *If the mean > median, then it is in class 1, otherwise it is in class 0.*

Average Delay (Dav): related to latency (service failure) i.e., G4 and it is in class 1 and jitter (delay variation): related to connectivity interruption and congestion i.e., in G2 and class 1.

- *If the mean > median, then it is in class 1, otherwise it is in class 0.*

Criteria 5:

The mean aggregated device performance compared with their corresponding median values to capture their deviation and to decide on their lable class based on the following conditions.

- *If average CPU utilization > its median value, then it is in class 1 other wise 0.*
- *If average memory utilization > its median value, then it is in class 1other wise 0.*
- *If the mean availability of the device < its median value, then it is in class 1 other wise 0.*

The final lable column created by comparing the KPI features mode values (i.e., CRP) with alarm features (severity influence) criteria values (i.e., CRA).

- *If the lable from CRP is the same as CRA, then the final lable is decided*
- *If CRP is different from CRA, then the final label decided by majority voting (weight of the criteria).*

VI- Lable Distribution Graphs

In Figure 1 below, the PCS feature and the group label distributions were plotted as a function of the CRP and CRA, respectively.

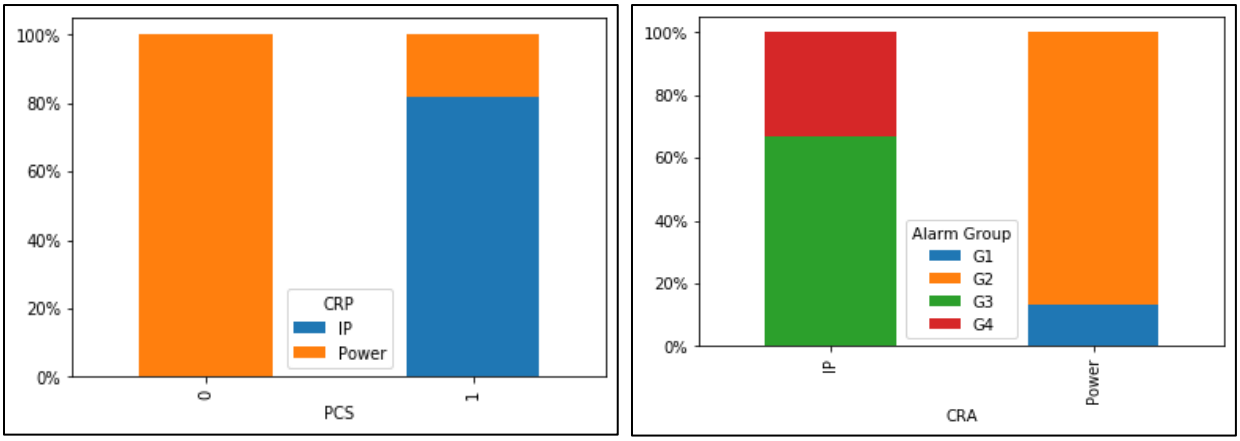


Figure 1. Label distribution of CRP on PCS (left) and CRA on Alarm group (right)

Appendix-B

I. Model Parameter Selection

1. *GCV – For 3rd Iteration*

To fine tune the model, using Adam optimizer, after 16-hour tuning, from 256 candidate parameters with 1280 fits, 0.987 mean accuracy and 0.009 sigma was achieved with 64 batch, 0.3 dropout rate, 25 epochs, with the same learning rate (0.1).

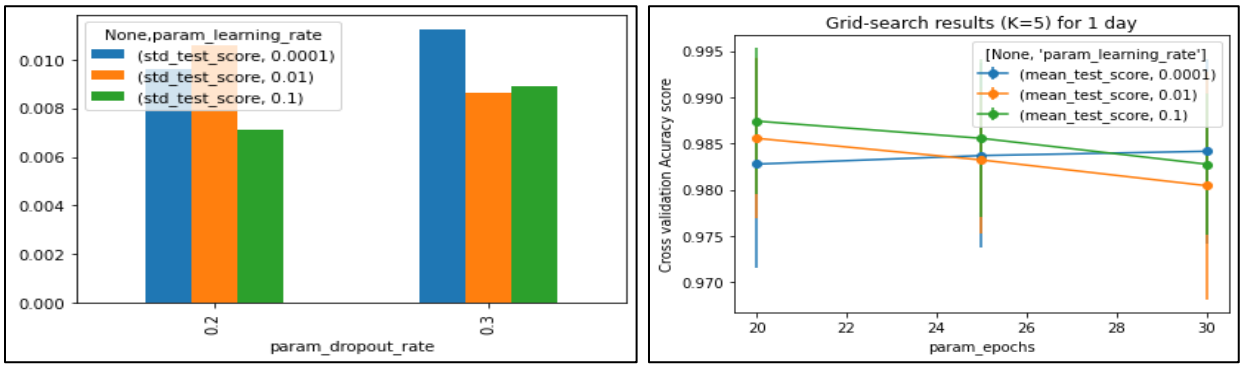


Figure 1. G-CV optimal learning rate, epochs, batch size, and dropout rate results

2. *G-CV for 4th iteration*

To fine-tune the model and find the optimal number of neurons, the dropout set to 0.25 manually by using the other parameters obtained from the above experiments. The final learning curve from the grid search plot as seen in Figure 3 below.

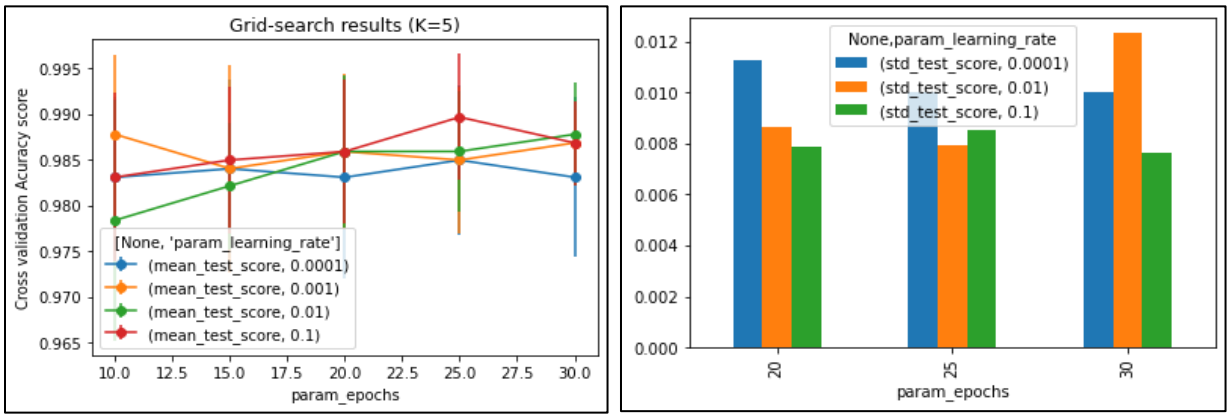


Figure 2. G-CV optimal number of neuron results

3. G-CV For 5th Iteration

The model was validated for total 1280 times to find the best parameters from 250 candidates by fitting 5 times, for 5 hours, then the accuracy and loss were improved from fold 1 to fold 5.

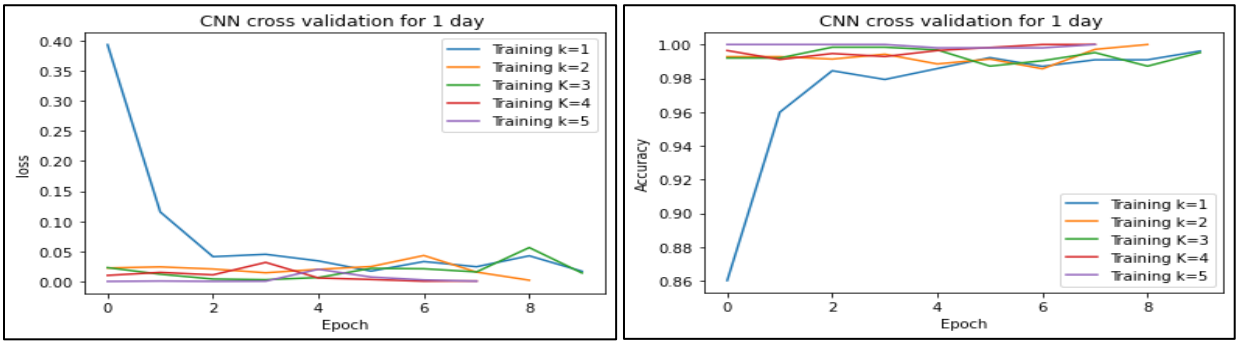


Figure 3. The model validation learning curve for k=5

4. KPI Model Grid search -CV

Figure 5 below shows the optimal parameters for the KPI data.

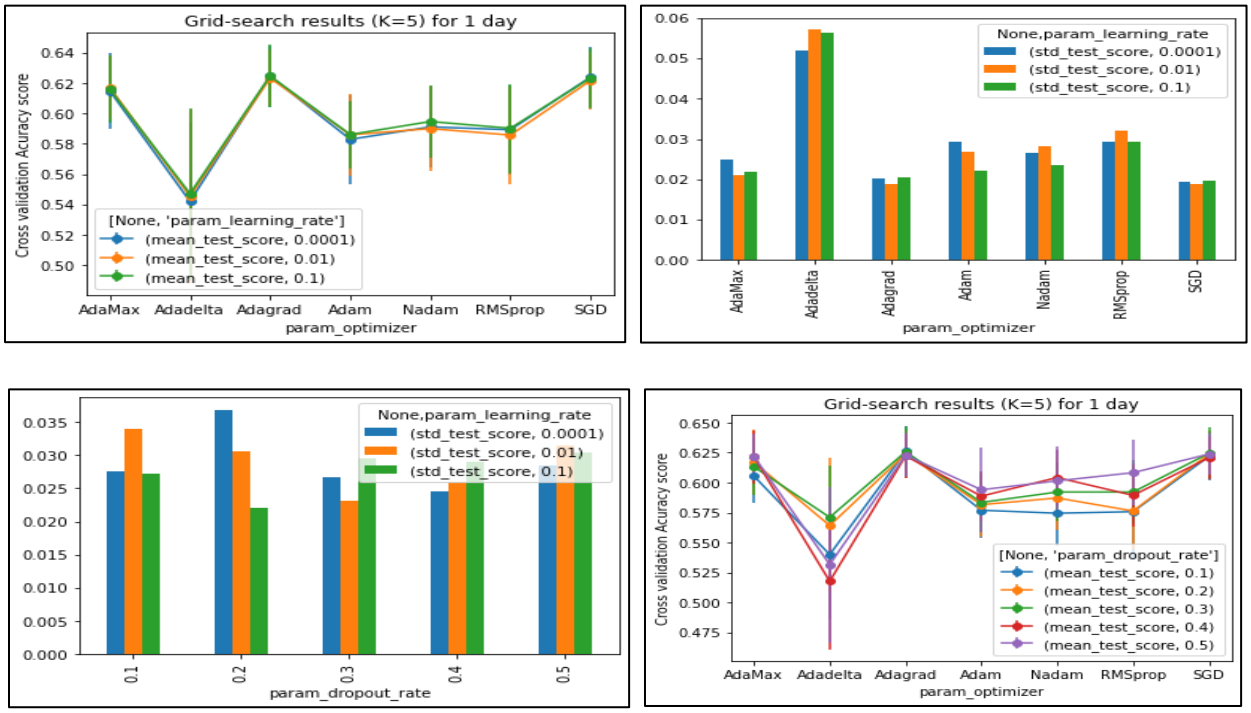


Figure 5. Optimal model parameter results.

5. *KPI Model Manual Tuning Results*

By using Adadelta and SGD optimizer with 10-2 learning rate, 1e-6 decay, and 0.9 momentum.

Table 1. Best CV scores for the KPI features only.

KPI data Val scores			
Model	**Accuracy**	**Loss**	**Parameters**
1	0.705	0.61	η =0.01, batch 32, epoch 25 dropout =0.2
2	0.625	0.62	η = 0.1, batch =32, epoch 20, dropout =0.2
3	0.695	0.44	η = 0.1, batch=64, epoch =20, dropout 0.3
4	0.565	0.73	η =0.1, batch =32, epoch 20, dropout = 0.2 for 1st CL, 0.3 for DL
5	0.546	0.78	η =0.1, batch =32, epoch 20, dropout = 0.3 for 2nd CL, 0.3 for DL
6	0.572	0.8	η =0.1, batch =32, epoch 20, dropout = 0.4 for 1st CL, 0.4 for DL
7	0.568	1.0	η =0.1, batch =32, epoch 20, dropout = 0.35 for 1st CL, 0.35 for DL
8	0.683	0.5	η =0.000, batch =32, epoch 20, dropout = 0.35 for 1st CL,0.35 for DL
9	0.654	1.0	η =0.001, batch =32, epoch 20, dropout = 0.35 for 1st CL,0.35 for DL
10	0.639	1.0	η =0.01, batch =32, epoch 20, dropout = 0.35 for 1st CL, 0.35 for DL

- DL is Dense layer and CL is CNN layer
- For testing, model 5 was selected and we obtained, loss**: 0.6959 - accuracy: 0.605**

Appendix-C

In the next pages of this section, you can find the paper of this study that was prepared for journal publication.

Resource Allocation Model for IP-Backhaul Maintenance Using Risk Analysis and Convolutional Neural Network

Tamrat Demena Guda
Addis Ababa Institute of Technology
Addis Ababa University
Addis Ababa, Ethiopia
nanuna2009@gmail.com

Rosa Tsegaye Aga(PhD)
Addis Ababa Institute of Technology
Addis Ababa University
Addis Ababa, Ethiopia
rosatsegaye@gmail.com

***Abstract*—For telecommunication business sustainability and productivity, efficient allocation of maintenance resources is vital. Maintenance resources such as cars, tools, technicians, and others are the major causes of longer repair times and dissatisfied consumers. This paper aims to assist the maintenance framework for daily resource sharing between two maintenance teams in the IP-backhaul network at Ethiotelecom. We have developed a knowledge-based optimal failure mode classifier model by fusing alarm and key performance indicator (KPI) data using 2D convolutional neural networks via weak supervision data programming by using risk analysis and multi-criteria decision-making with data segmentation in the time domain Windows. Overall, our model has correctly identified failure modes 96.6 percent of the time and has been 97 percent fair in maintenance resource allocation.**

Keywords— IP-backhaul, Maintenance, Resource allocation, 2D CNN, KPI, Alarm, Domain Knowledge, FMEA, DEMATEL.

I-INTRODUCTON

Intelligent and optimal maintenance resource management is the key enabler in telecommunication and other domains. An optimal allocation of maintenance resources includes modeling techniques, maintenance environments, data sources, system configurations, optimization criteria, and planning periods. To this end, there have been extensive studies on reliability-centered maintenance, multi-criteria decision making, and mathematical and deep neural network models that have determined that optimal allocation of maintenance resources is the greatest contributor to decision-making and business growth in telecommunications and other fields.

Maintenance activities involving time and resources (labor, tools, materials, and site visit funds) and often encounter allocation problems while setting priority, handling customer requests and complaints. For example, Ethiotelecom has set up the HUAWEI IP-backhaul network in Addis Ababa, which includes power and IP subsystems, mainly to provide advanced cellular services over Intermediate System to Intermediate System (IS-IS) protocol and Multi-Protocol Lable Switching (MPLS) provided routers as a backbone. However, according to the Network Management System (NMS) experts, 9,500 alarms per day occur in connection with the services. As a result, due to a lack of failure risk analysis in the maintenance of the IP backhaul network and unavailability of the services, Ethiotelecom loses 2.073 billion ETB every month [2]. Based on the alarm prioritization and classification in the network, the problem belongs to the classification in the context of a neural network. The key research problem in this paper is: how the optimal failure mode classification model can be implemented using knowledge of telecommunication network alarm and performance data.

The main aim of this paper was to identify and classify the failure modes in the network in order to assign the priority for resources requirement between Power and IP maintenance teams in an optimal way, using the network alarm and KPI data, the reliability-centered maintenance, and convolutional neural network model to help the decision makers in the company.

The rest of the paper is organized as follows. Section II provides related works and the contribution of this work related to the previous studies. Section III introduces materials and methods that have been used to develop the classification model. In Section IV, the results of our experiment will be demonstrated, and a discussion of our findings will follow. Summarization is in Section V, and it also raises some possible improvements for future work in this Section.

II-RELATED WORKS

A. Resource allocation models

Most resource allocation studies deal with mathematical models to optimize different objectives[2]. However, neural network models have emerged in telecommunication and other industries. For example, [3] tried to assist the resource allocation decision for emergency fleet management system to minimize the time gap between incidence and response, using risk analysis and convolution neural network (CNN) and Long Short Term Memory (LSTM) on spatial-temporal incident data . In addition, deep learning models have been applied to minimize the total transmission time and energy in real-time for a congested content delivery network. By preparing channel matrix from mobile terminal's data as input to CNN and deep neural network algorithms, as a result, they have been found to be optimal to time and energy-efficient allocation to NP-hard (nondeterministic polynomial time) problem with more than 95% accuracy [4]. Moreover, a recent work has investigated feature extraction and classification capability of deep learning (DL) algorithms. The author has recommended machine learning to resource allocation problems when either a shortage of model or algorithm occurs in the study domain [5].

B. Convolutional Neural Network

Ethiotelecom

CNN is one of the deep learning universal approximator algorithms that imitate the real brain and should solve the problem of simple machine learning algorithms such as, learning and mapping complex functions in high-dimensional spaces to solve intelligent tasks[31].

There are more than 10 CNN architecture with sparse interactions, parameter-sharing capability and transitional invariance as feature extractor for multilayer neural networks. and because of the hardware acceleration, in addition to its statistical efficiency in training big data, it is more efficient in terms of storage and runs time requirements. Furthermore, 2D-CNN in particular has proven to be more efficient than 1D-CNN in order to classify images from a feature vector of fixed size (image). In addition, its classification performance in computer vision [32].

C. Data preparation for decision-making

To assist decision support system optimally in complex network data aggregation, data labeling and data segmentation play a prominent role in localization, classification, and detection of network faults. For example, [8] has proposed user-defined functions, such as; complementary, redundant, and cooperative data integration based on input source relation. In addition, [9] proposed features level abstraction on different sensors' data. Further, another pioneer work [10] has provided real-time data aggregation taxonomy by setting out design constraints in hard, firm, soft-real time scenarios using both SQL standards and user-defined aggregator functions. Prior to data aggregation, the study has implemented periodic features selection, worst-case scenarios selection, replacement of sporadic with aperiodic features.

An early and relevant study [11], has presented fault localization approach aiming at algorithmic aspect in a complex networks topology. In addition, using a logistic regression algorithm to study the correlation among network events based on end-to-end measurements of throughput or delay datasets. The authors proposed a design matrix to the available historical data from network events and diagnosis reports. Estimation error and the MacDiarmid inequality or concentration property used for model evaluation. Based on the study result, the model was free from NP-hard problems, scalable with complex network, and suitable for data compression and integration. However, the author stated that accurate dependencies and prior probabilities are difficult to obtain with the absence of causality graph.

D. Knowledge-base data labeling techniques

The major challenge of addressing optimization problems using neural network models is to label data from different sources, to this end, studies [12] and[13] that had applied data programming techniques have developed weak-supervision labeling based on domain knowledge to extract text relation patterns using the LSTM algorithm on three domain datasets. Based on the classification report, data programming was 10 times better than the baseline handheld labeling techniques. In addition, [14] has proposed rule-based and weak- supervision data programming for chemical domain data, using CNN and RNN , and has achieved an equal 93% predictive accuracy. In contrast, weak supervision models without the programming skills of experts proposed for text mining. The study has found that this approach has been better than the other methods for reducing time and cost [15].

E. Reliability — centered maintenance

To implement weak-supervision labeling for supervised classifier models, domain knowledge is essential. Such that, Failure Mode and Effect Analysis (FMEA) is a reliability-centered maintenance tool standardized by IEC 60812 to prepare telecommunication and other domain knowledge to identify risk in system failure modes for optimal decisions [16]. In addition, FMEA is a systematic approach to analyze a system, design, process, and service to identify and reduce the potential failure modes' cause and effects. Besides, it is a structured, logical, and engineering technique to define, rank, and prevent the risk of failure modes in a complex system. In addition, there is functional and hardware approaches to implement FMEA based on the objective and the result of the study.

In FMEA, Risk Priority Number (RPN) is cumulative result of severity, occurrence, and detection of failure modes in a system. Risk analysis can be implemented as quantitative, qualitative, and semi quantitative depending on the failure rate data availability[17]. Moreover, [18] has investigated FMEA application from relative importance consideration aspect and the study suggested that identification of failure modes needs domain experts and brainstorming on a system. However, mostly RPN evaluation is both subjective and objective that creates vague results. Therefore, to mitigate the uncertainty of RPN evaluation, the study proposed uncertainty theories, multi-criteria decision-making (MCDM), or hybrid methods.

From MCDM techniques, Decision Making Trial and Evaluation Laboratory (DEMATEL) is an effective technique for complex systems to analyze the severity impact of failure modes by pairwise comparison and assigning unique criteria taking the FMEA's uncertain result for better decision-making [19]. Mathematically, we can express DEMATEL easily as linear algebra, digraph, and matrix form [11]. For instance, [19] examined the DEMATEL by using a weighted digraph to represent two complex systems and experts' knowledge, and then to prioritize failures based on direct and indirect influence relationships. The research result have shown that DEMATEL can assign unique levels and group failure modes with equivalent effects and relationships.

F. Data transformation techniques

To transform fused data from different sources and train CNN, Gated Recurrent Unit (GRU), and LSTM models, signal processing in the time domain[20] and frequency domain[21] have been studied to solve network and power failure detection problems using simulated logs in Python and MATLAB, correspondingly. They have achieved 74.6% and 73.53% detection accuracies, respectively. In addition, another study[22] has proposed an ensemble simulation method to train the CNN model for failure classification in a power system. The study has utilized lable for four (4) types of failure data via category generation using MATLAB. They have compared sliding windows, Discrete Wavelet Transform (DWT), and Gramian Angular Field (GAF) techniques. The analysis result has showed that CNN's accuracy has reached 76%, 87%, and 76% for all three methods.

Moreover, [23] has solved the footstep detection problem by encoding temporal 2-D pressure sensor data to the visual domain (pixels), using histogram and windowing techniques for binary frame preparation on a pre-trained CNN model.

Based on the result, average frames has outperformed the SVM classifier with DWT by 10%. Furthermore, [24] has investigated the impact of overlapping and non-overlapping windows approaches for human activity detection. The study has used a 10 Cross validation on 'HAR' dataset on stacked and not stacked windows to train three-machine learning and one neural network (NN) model. From the results, the overlapping window improved the model F1-score by 10% for NN models with subject-independent cases. However, within machine learning classifiers, the F1 score even declined by 4%.

We respect the past studies effort to classify failure modes and optimize maintenance resource allocation in complex network maintenance using mathematical and neural network models. However, combining reliability study and data analysis with neural network models can close the gaps in mathematical models. This paper provides mainly with insight into optimal resource sharing based on reliability study and system understanding by learning relevant variables from network metrics and device logs to classify failure modes. In addition, the paper made a significant contribution to:

- Automate maintenance frameworks to manage tasks;
- Improve the NMS critical alert filtering policy;
- Aggregate knowledge and data in complex network;
- Enhance the neural network with ground truth labels.
- Enable domain knowledge to be leveraged for deeper risk assessments to improve operational performance and revenue while reducing maintenance costs and time.

III-MATERIAL AND METHODS

Knowledge-driven studies require knowledge acquisition and transformation. In this paper. To preprocess the data we have used Excel and Pandas, to implement the 2D CNN model we used tensor flow and Keras deep learning libraries. And we have carried out the experiment on Jupiter-notebook using a laptop with dual-core CPUs and 8GB RAM. The methodology applied in this paper is illustrated in Fig.1below

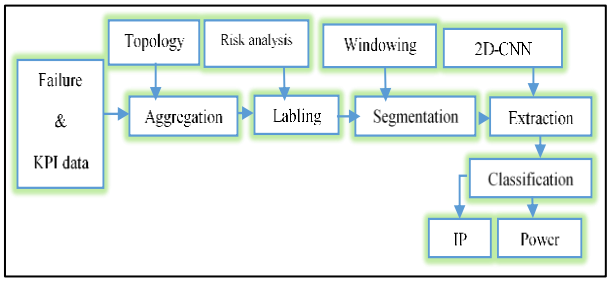


Fig. 1 The proposed methodology pipeline

A. Data preparation

Data collection from a complex network topology is possible to solve NP-hard problems using causality graph, domain knowledge, and statistical data analysis on KPI and alarm data. Therefore, in this study to select topology for data collection and to select important features, discussing the study aim with the network experts has been the first step.

We have collected the data sets using PRTG and U2000 NMS between August 2019 and June 2021 for 23 months from access and aggregator layer routers that are more susceptible to failure and suitable for data collection and aggregation in the IS-IS ring topology that contains 15 links and 12 devices as depicted in Fig .2 below.

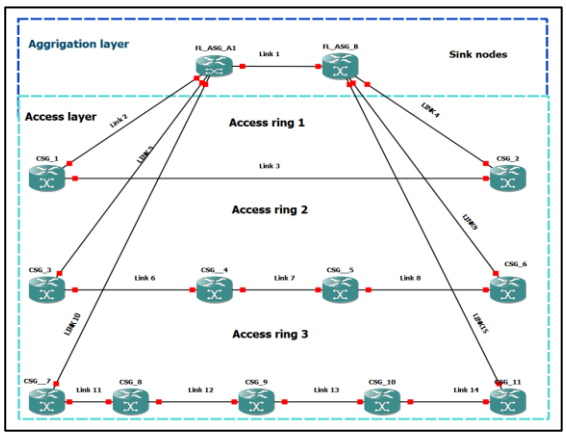


Fig. 2 IP-backhaul IS-IS topology

The collected KPI data have contained 15 links and 12 devices' performance with 1hour granularity, using peer-to-peer interfaces with a 1sec measurement interval on all links at the same time . And the alarm data have contained random faults from the 12 devices with a 1-hour interval. Overall, we gathered 16,170 hourly instances with 9 features from each link, 16,180 device performance instances with 4 features and 17,000 alarm instances with 19 attributes.

Following supervised data preprocessing pipelines to get structured data, to fit and evaluate the CNN algorithm. In this paper, data preprocessing has included feature selection, data cleaning, data aggregation, and data labeling, feature scaling, and data segmentation using domain and system knowledge.

We have selected ten (10) features from three data sets using domain expert's knowledge and the routers vendors' recommendation. And the selected features for data cleaning are:

- Link jitter, delay, ping, packet loss ratio;
- Device CPU, Memory and availability;
- Alarm name, Severity, Probable cause;

B. Data cleaning

We have cleaned the dataset by modifying long words, deleting blank space, removing units and redundant columns in the data before aggregation. After data cleaning, we have got 17,400 rows, and over four (4) numerical columns in the link performance data, besides delay, Ping, Jitter, and packet loss ratio feature. Besides, from the device performance data we have extracted 17,000 instances with three (3) numerical columns. Further, we have extracted three (3) catagorical features with 16,235 hourly interval instances from the alarm log after grouping and filtering incidents within the same time interval.

Finally, we designed the timestamps in the original log files for machine learning by preparing a new time index, to keep the consistency among the datasets, to find one large chronologically ordered log file based on the data collection

period. Together, we have obtained ten (10) cleaned features from the three data sets.

C. Data aggregation

data aggregation process contains raw data, aggregation function, and the aggregated output. Aggregation function applies a certain characteristics on the input data features.

We have chosen a generic data aggregation function to summarize and answer the research question, adapted from the heuristic data aggregation process formulated in [10], and then we have aggregated the raw data features in-network and feature level. In addition, we used design matrix preparation for complex network topology, as stated in [11]. To aggregate the data sets we have deemed the following assumptions:

- The data has contained enough temporal and contextual relationships in the network to be discriminated.
- Our goal has been to determine the failure mode with a known location, devices, and specified time.
- The alarms in one hour can correspond to the link and device performances at least in 24 hours.

Based on the above assumptions and using the network topology in Figure 3.2, we have framed the link performance, device performance, and the routers alarm data fusion:

1) Link performance fusion:

Joint KPI of a ring is the union of all links in one ring, such that, after combining the three rings separately, then the joint KPIs of the three IS-IS areas combined as (R1 U R2 U R3). Finally, we have computed their average to aggregate the link performance data. Mathematically, expressed as:

$$L\bar{F} = L\frac{1}{n}\sum_{1}^{n} f_{Li} = \frac{1}{n}(f_{L1} + f_{L2} + \cdots + f_{Ln}) \qquad Eq\ (1)$$

Where f_{Li} contains all link performance features, *Li* is a link.

For instance, if we look at 24-hour sample jitter data in Fig. 3 below that has been stacked from fifteen links, using their time index to calculate their mean value, and we followed the same procedures for other performance features in the KPI data set.

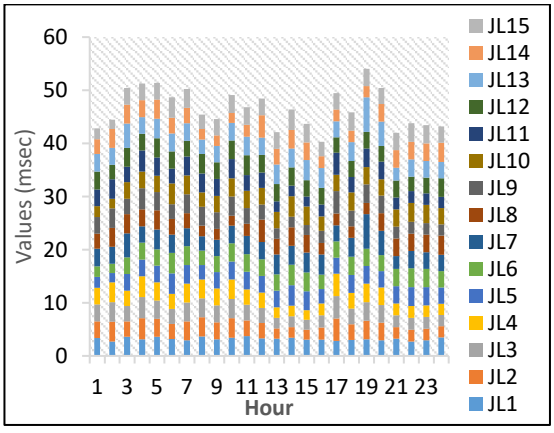


Fig. 3 Sample jitter aggregated data

2) Routers' performance data fusion

We have fused the device performance data based on device-level aggregation. Since, each device group belongs to the same IS-IS rings. So that, we used explanatory (lossy) type SUM aggregation function for directly connected devices in the same ring. Then we have applied average aggregation function to get the three features aggregate values in one data frame. Mathematically, expressed as:

$$D\bar{F} = \frac{1}{n}\sum_{1}^{n} f_{Di} = \frac{1}{n}(f_{D1} + f_{D2} + \cdots + f_{Dn}) \quad Eq\ (2)$$

Where f_{Di} contains all device performance features, *Di* is the number of routers in a ring.

3) Alarm data fusion

We have assumed one IS-IS area failure affects the other areas at the physical topology level, or the failure impact on one ring disseminates to others. Also, the alarm data has contained spontaneous events. Therefore, we have merged all available alarms in each ring to get one data frame, and then we filtered alarms reported within the 24-hour window using their time index. Further, since the two (2) aggregator routers contain more or less the same alarms, we extracted their intersection and combined them with other routers' alarms to get aggregated alarm data in a ring. We have also filled gaps between the alarms dataset features. For instance, if we missed one alarm in a router at a specific time, we filled it with other routers' alarm data. Mathematically, total aggregate alarm data (A_{RT}) is expressed as below in Equations (3).

$$A_{RT} = \{A_{R1}\}\, U\, \{A_{R2}\}\, U\, \{A_{R3}\} \qquad Eq\ (3)$$

Where A_{R1}, A_{R2}, and A_{R3} are the three rings alarm data,

To construct the design matrix, we have aggregated the KPI (link and device performance) data according to their level of abstraction by concatenating, merging, and stacking 98 columns from the KPI data sets. We developed 16125 instances of seven (7) features after taking their mean. Then, we join the alarm data with the KPI data after combining the alarm data in each ring and filtering 16125 instances only using the Pandas libraries. Finally, we merge them with the 16125 rows of the three (3) alarm features. Then, we obtained 16125 rows with ten (10) columns after joining the three datasets using their time index, and the final design matrix values that were ready for data labeling were integer and category.

D. Data labeling

After we completed data aggregation, we have applied weak-supervision data labeling (programing) to extract IP and power classes from 10 features and 16125 instances in the design matrix using the domain knowledge captured by reliability-centered maintenance (risk analysis).

1) Risk analysis on IP-backhaul failure mode

In this paper, the role of risk analysis has been to capture experts' knowledge on the normal and faulty power and IP subsystems behavior of the IP backhaul network. Therefore, following semi-quantitative FMEA with functional approach, using 20 domain experts from both sections, segmenting the system into subsystems and components level by:

- Brainstorming on the potential failure modes;

- Identifying the effects of each failure mode(FM);
- Assigning failure severity(S), occurrence(O), and detection (D) rate;
- Calculating the RPN for each failure mode;

Fig. 3 below shows the functional block diagram of the IP-backhaul system failure modes.

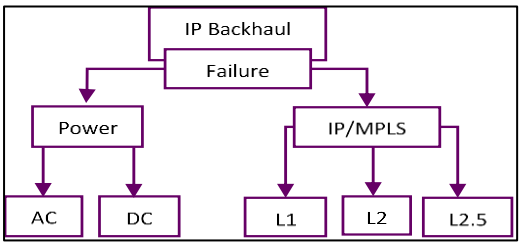


Fig. 4 IP-backhaul functional block diagram

In this paper, we have considered the IP/MPLS and power systems failures as failure modes (FMs). In addition, we have considered their subsystems to capture failure causes.

After collecting twenty-one (21) failure modes from the IP-backhaul network experts, we have identified eight (8) power failure modes in AC and DC sub systems, for IP failure modes, we used the failure history data, and there have been fifty-nine (59) alarm codes in the selected network. From 59 codes, thirty-nine (39) have belonged to L2.5, 10 in L2, and the rest 10 IDs have been from L1. Thus, by filtering and associating these alarm codes (names), we have obtained nine (9) main IP/MPLS failure modes for RPN evaluation based on the three (3) risk factors in as stated in Equation (4).

$$RPN = R * D = O * S * D \qquad Eq(4)$$

Severity(S): the seriousness (consequence) of the failure modes, Occurrence (O): occurrence probability of the failure modes, Detection (D): includes spare parts, methods, level of difficulties, and ability. Based on 10-point scale factors, one has been the lowest, and ten has been the highest, except in reverse order for D. Also, if two failure modes were the same in their RPN result, then the severity has determined their risk.

By taking the experts' average RPN, considering the risk factors; we have been able to identify seventeen (17) severe failure causes from the IP-backhaul system. Then, we have ranked the potential severe failure cause from both subsystems as can be seen in Fig 5 below.

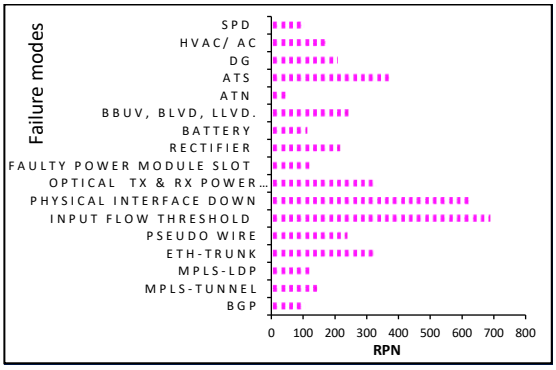


Fig. 5 IP-backhaul failure mode risk ranks

2) Modeling the failure modes uncertainty

Even if the FMEA result had been satisfactory from the failure modes identification aspect, the RPN result has been uncertain and created ambiguity among the experts on the failure cause magnitudes, where some of them had been highly skilled in power subsystems, and some had been highly skilled in IP subsystems. To clear the ambiguity in RPN results for accurate data labeling, we have needed to study the relationship and influence of the severe failure causes.

We have followed the procedures in [25], to apply the DEMATEL function since it has been more applicable for complex systems like IP backhaul. The function performs matrix normalization, extracts the indirect relationship of the failure modes by pair wise comparison. We have used a 17 X 17-failure causes' matrix result from the risk analysis as input for a customized DEMATEL Python script that had been intended for similar studies in other domains.

To prepare the failure causes influence matrix for direct relationship weighing, we have set a threshold between 1 and 4 using the RPN results and criteria based on the failures' causal relationship obtained from power and IP maintenance team. The graph in Fig. 6 below shows the DEMATEL result

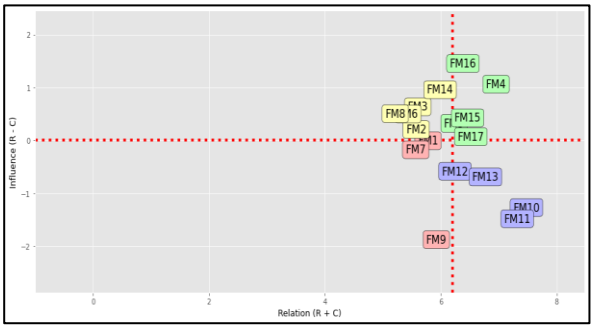


Fig. 6 IP backhaul failure modes influence and relationship

On the above graph, the horizontal (R+C) and the vertical (R-C) represent the relationship and influence of the failure modes, respectively. In addition, the quadrants under which the failure modes lie determine their degree of influence and relation on the network. For instance, the first and the fourth quadrants represent high degree of severity influence and relationship of the failure modes. Moreover, the value of R-C is more effective and applicable than R+C because R-C is a good criterion or factor for failure mode prioritization.

Based on the above result, the most severe and influential failure causes/modes lie in the physical layer failure mode and power failure modes. The second in severity influence ranking comes from the MPLS (Layer 2.5), followed by the Eth-trunk (virtual circuit failure). The other failure causes had high relationships, but they have been less influential. In other words, they have not been as dominant factors as the other nine (9) failure modes that have a 54% impact on the IP-backhaul network failure aspect. Therefore, to generate categories, map, and classify the data between power and IP failure modes, we have emphasized nine (9) influential variables that have belonged to quadrants I and IV.

3) Alarm category development

To generate the categories for alarm features based on the nine (9) influential failure causes, we have used the 59 alarm IDs in the data to simplify the analysis. First, by assigning less influential IP failure modes to the power category. Second, filtering rare, irrelevant, and less influential 16 Alarm IDs (failures). Then we have analyzed the rest of the 43 alarm IDs. Then 22 IDs have been represented IP/MPLS service and configuration failures, and we have categorized them under class 0 (IP). In addition, we have grouped the remaining 21 physical layer and power failure IDs under class1 (Power).

Fig. 7 below shows the frequency distribution of the four-alarm name groups. G1 and G2 have represented power and physical layer failures, whereas G3 and G4 have included layer two (2) and layer 2.5 failure alarms. Furthermore, the graph shows that most alarm names come from the physical layer, followed by IS-IS, IP services, and power alarms. Likewise, we have also categorized the other alarm features

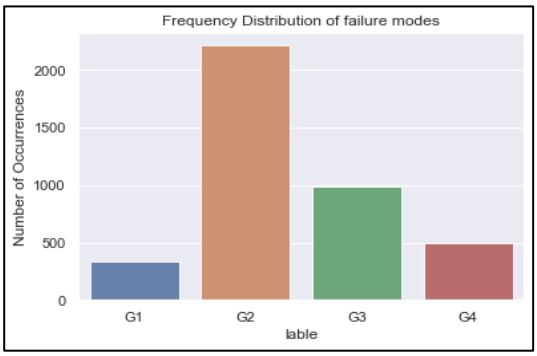


Fig. 7. Distribution of the data over the groups

4) Criteria development

In this study, data labeling has been to differentiate the failure modes between IP (0) and power (1). Such that, we have used a criteria function to decide on the binary class labels using weak supervision data labeling based on domain knowledge gained from experts, DEMATEL results, and applying user-defined functions (IF then) conditions.

The dataset has been sparse, and all performance features had been at their optimal threshold values. There has been also a high correlation between the KPI features. For this reason, we have created median (threshold values) from the existing features' mean value before applying the label function. And then, for comparison, we have used their mode to write the performance criteria functions (CRP). Further, we have developed alarm criteria (CRA) to compare with CRP and to decide on their label from the alarm features by comparing them with each other.

To mention some criteria we have used for the four (4) nominal groups and the CPU utilization features, if the alarm is in either G1 or G2, then it is in class 1 (power), or if it is in either G3 or G4, then it is in class 0 (IP). Similarly, if the average CPU utilization was higher than its median value, it is in class 1, otherwise 0. If the two labels had been the same, then we would have decided on the final label. Otherwise, we have used majority voting and the severity of influence weights (ranks).

To eliminate a high probability of misclassification of the minority class over the majority class and overcome over-fitting and under-fitting problems. We have evaluated the label imbalance within the groups and between target classes using the event rate (minority over Majority), then we have obtained a 0.56. That is a 1:1.25 event ratio between IP and power classes. Therefore, the labling process has fulfilled the criterion of data preparation processing in supervision models for binary classification problems.

E. Data segmentation

In this study, to extract fixed window slices of 2D images from multi-variate, highly correlated features, and sparse arrays of data to discriminate them in image form using 2D-CNN we have used a fixed window-slicing technique. In addition, we have considered the following led parameters while selecting the windows' size:

1) The objective: has been to classify the failure modes daily based on the maximum maintenance hour.

2) Sampling interval: We have measured the values of the KPI data every 1s interval (1Hz). Also, we have collected the average the data every one (1) hour interval, and it has contained 370,875 sample points. Windows size is the number of data points in a time window with given sample interval. Mathematically, the sampling theorem is:

$$Frame\ size(T) = \frac{N}{Fs} = N * \Delta t \qquad Eq(3)$$

Where N is the number of block size in one frame,

Based on the minimum sampling rate, we have been able to extract 24 window frames (slices) using all 20 features in 24 hours.

3) Overlapping ratio (OVR):

Most studies have used ½, ¼, and ¾ overlapping ratios, depending on their data nature, to satisfy the sampling theorem. So, we have used a 50% overlapping ratio to increase the size of the data sets and maintain information between consecutive frames, then to keep the model's generalization ability.

To get feature space and target variable from the design matrix in NumPy array format (2D gray-scale images), we have sliced the data in 24 hours using the above constant variables (window parameters). At this point, from 1-D 16125 instances of 20 features vectors (370,875 data points), 1342 gray-scale images with 1342 labels each having 20 features within 24 hours window segments have been retrieved with their most often used label in the segment. Fig. 8 displays four (4) IP and four (4) power 24 hour labeled window samples horizontally, and the 20 features in each frame vertically. To validate and test the model with three configurations we have prepared 1073 and 269 frames, respectively.

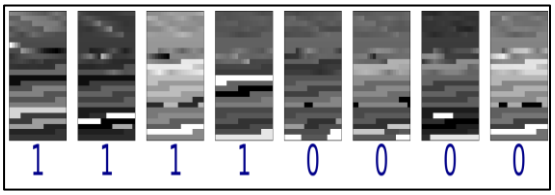


Fig. 8. Sample grayscale sliced failure modes

CNN architecture receives three-dimensional (3D) input shapes. Consequently, we have reshaped both the validation and test 2D array segments to their 3D versions. After we had completed data segmentation, finally, the data frames have been ready for modeling.

F. Expermental setup

The selected CNN architecture was 2D sequential and we have changed the default parameters to fit the IP-backhaul data shape and meet the study aim. Further, we have replaced SoftMax and sparse categorical cross-entropy functions with sigmoid and binary cross-entropy functions respectively in the output layer of the network to make it suitable for the binary classification problem before fitting the model. Fig. 9 below shows the simplified architecture of network with the changed parameters. We have defined two (2) convolutional layers that have 48 filters in both layers. Overall, the network has 107,329 total trainable parameters.

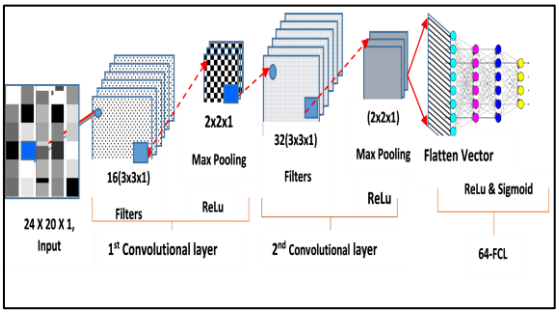


Fig.9 The simplified architecture of 2D-CNN

1) Parameter selection and tuning

We have implemented cross-validation (G-CV) to split the dataset into 5-partitions, fit the model with 4/5 of the dataset, and validate the model with 1/5 of the data K times (where k = 5 & 10 iterations). We have also applied the grid-search algorithm to fit the model with a grid of hyper-parameters for each iteration in a brute force manner. We have tried every combination to maximize the accuracy and minimize the variance of the model. Then we have fine-tuned the model to illuminate the model's bias-variance trade-off.

From five configurations of 320 candidate parameters, after fitting the model 1870 times using 5 folds, for 27 hours successful runtime, we have selected the last five (5) hyper-parameters listed in Table II that have scored 98.9% mean accuracy and a 0.006 standard deviation. In this study, we have used the Grid search only to select and tune optimal parameters.

TABLE I. OPTIMAL PARAMETERS

The selected optimal parameters						
μ	***σ***	***e***	***η***	***b***	***o***	***d***
0.98	0.006	25	10^{-2}	32	Adam	0.2

Where, μ is the mean values, σ is standard deviation, e is the number of epochs, alpha(η) is the learning rate, b is batch size, o is optimizer, d stands for dropout rate.

2) Model validation

After fitting the model on 1073 training samples using 5 folds and the selected optimal parameters, we have assessed the model performance using the training and validation learning curves as depicted in Fig.10 below that indicates learning and generalizing ability of the model.

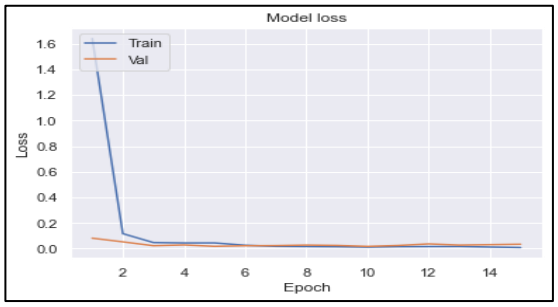


Fig. 10 . The model's learning curve

The training loss curve has not remained flat and decreased continuously until the end of the training. Consecutively, the training and validation losses continued to reduce to a stable point without oscillation, and the model had converged after 14 epochs. In addition, the model has not over-fitted because a 0.017% error variation has been observed between the validation and training. Thus, it has been a good fit. In addition, the model has achieved a 0.98 validation accuracy.

Overall, the model has detected the problem well and faster from the training set with a 1e-2 learning rate. And this suggests a 0.01 learning rate would be moderate to test the generalization ability of the model. Thus, we have saved the model configuration and weights to evaluate the model.

3) Model evaluation

After validating the model, it has been necessary to measure the generalization ability of the trained model using the saved model parameters with all test frames kept separately from the training phase, as listed in Table II below. We have also separately examined the impact of the alarm and KPI data performance to capture which data sets can more classify failure modes. And second, to verify which data set is suitable for optimal decision making in the same 2DCNN architecture.

TABLE II. SLICED TEST FRAMES

	Experiments		
	Data	***Features***	***Number of test frame size***
1	All/Both	20	269
2	KPI	14	269
3	Alarm	6	269

We have evaluated the models using the F1 score to find the optimal outcome, as it is a measure of the balance between recall and precision for an unequal distribution. In addition, the F1score takes into account FP and FN, which also have an equal impact on the fairness of the model. In addition, to compare the models, we have used the weighted average of F1score because it gives different weights to unbalanced

classes to fairly calculate their loss. However, the macro average treats the two classes equally regardless of the imbalance between the classes. Therefore, it could only describe the overall performance of the model. In addition, to find a more optimal result, we used ROC-AUC because it is scale and threshold invariant, and it eliminates the compromise between FN and FP.

IV-RESULTS AND DISCUSION

A. *Results*

In this paper, the negative and positive classes for the suggested analysis had been IP and Power, respectively. In addition, we have used FP (Type1) and FN (Type2) errors to check whether the classification was good or bad. When a power failure mode classified as IP instance, it was FP, and when IP failure mode classified as power cases, it was FN.

From the model classification report as depicted in Fig. 11 below, using all features in the design matrix the model has achieved 0.97 score in average, macro and weighted average F1 scores and the alarm model has achieved 0.98 in all three scores to classify both failure modes. However, the KPI model has achieves 0.61, 0.38, and 0.46, respectively.

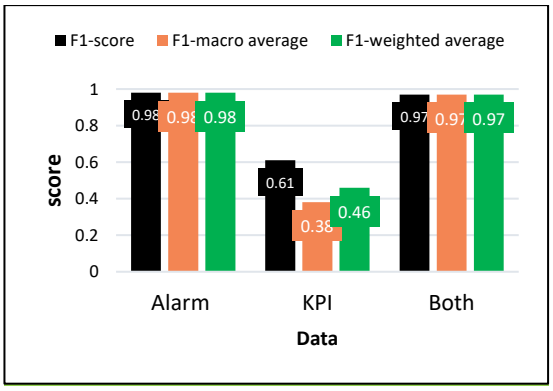


Fig.11 The model fairness (optimality) graph

From the model ROC and AUC report depicted in Fig. 12 below using all features in the design matrix and the selected alarm features, the model has scored 0.96 and 0.978 TPR and 0.1 and 0.3 FPR, respectively. Also, the model has correctly captured 96% and 97% of the IP failure modes from TPR (True IP cases), with 1% and a 0.03 FPR (True power cases), respectively. However, the model has achieved a 0.5 ROC and AUC using the KPI data.

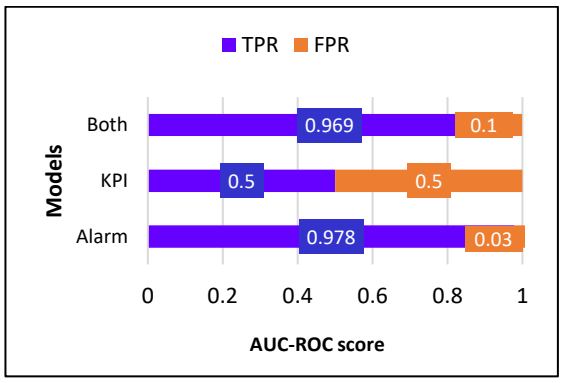


Fig.12. The model generalization ability

B. *Discussion on the results*

An optimal knowledge base binary classifier model should obtain higher scores on the F1 and AUC-ROC evaluation metrics, as well as its ability to capture the domain insights through data programming. In addition, the performance of each data set should be tested separately for further study.

From the model result, we have achieved a training and test accuracy of 98.0% and 97.0%, respectively, although the class imbalance ratio of 0.56 between the IP and the power frame affects the model optimality to reduce IP instances by 1%.

When we compare the model fairness (optimality), the most important conclusion of the F1 score is that the model can classify two failure modes with an error of only 3% in both the same and different weights using the alarm and all features. Together, the results effectively meet the study's first question that the model has been 97% optimal. However, the model fairness has reduced using KPI data by 52% and 1% as compared to the other data, respectively. However, we should bear in mind that the KPI data helps the model in data segmentation and could help the decision makers at least 46% of the time. In addition,

Moreover, the point (0.5, 0.5) on the ROC represents a random classifier. Therefore, the KPI model has estimated TPR and FPR with 0.5 AUC. In other words, the model has correctly captured 50% of the two failure modes of TPR (Real IP Case) and FPR (Real Power Case). Also, the AUC of the model is narrower at 0.5, indicating that the ability of the model to capture and separate the labeled frames between groups has reduced. As a result, the model is not optimal rather it guesses. On the contrary, the model can distinguish the classes using alarm and all (alarm and KPI) features in the design matrix with a 0.967 ROC. As a result, the model has been an optimal classifier and close to perfection because; the point (0, 1) on ROC represents the perfect model. In addition, a 0.969 AUC in both cases show the model's ability to capture and separate the groups correctly in the labeled frames.

However, from the results of the selected data model, a significant difference has been observed that the KPI model contradicts the results of the alarm model, as it is 100% correct when classifying the power failure modes only. However, the alarm model has performed better than the two models in all metrics; and this will increase the model's fairness by 1% and reach 98%.

One explanation for the weak link between IP data and power is the lack of device health features, such as voltage, power, and current, that were not included in this study. In addition, the difference in the KPI data results is due to the lack of an inherent pattern in the data to teach the model, a strong correlation between features, and slight differences between their instances.

Apart from these, as far as our knowledge concerns there has not been similar model from either data or method aspects. On one hand, this has hindered us from comparing our model with others to justify our findings more. On the other hand, this makes our work unique and can be used as a benchmark for the problem domain.

V-CONCLUSION AND FUTURE WORK

In summary, this paper has introduced a new 97% optimal approach to study the failure modes classification model based on domain knowledge and system data fusion, to help decision- making, related to the allocation of maintenance resources.

This study has developed a design matrix from failure and system performance data in an Ethiotelecom IP backhaul network to classify IP and source failure modes using the 2D CNN binary classification algorithm. The study has also applied reliability-based maintenance and MCDM for risk assessment, feature mapping, and data programming. We also looked at the magnitudes of separate features on the model performance.

Overall, the model result implies that the alarm data is preferable to KPI data for optimal solutions in IP-backhaul maintenance based on risk analysis. Although we have obtained substantial benchmark results and answered the research question, we therefore recommend further research, focusing more on improving the performance of the KPI data model by using the network reliability data with dynamic threshold in frequency domain with further statistical analysis in feature engineering.

In the future: (i) our approach can be extended to solve multi-class problems using other CNN architectures and additional data. (ii) The results can be more reliable by using heuristic data labeling with advance system knowledge.

ACKNOWLEDGMENT

The authors thank ethio-telecom for financial support and technical assistance.

REFERENCES

[1] ethio-telecom, "Ethiotelecom Mobile Network Traffic & KPI Analysis from Business Perspective," presented at the monthly CEO meeting, Addis Ababa, Mar. 04, 2020.

[2] W. N. Cahyo, K. El-Akruti, R. Dwight, and T. Zhang, "Managing maintenance resources for efficient asset utilization," 2014.

[3] A. Mukhopadhyay et al., "A Review of Incident Prediction, Resource Allocation, and Dispatch Models for Emergency Management," arXiv:2006.04200 [cs], Feb. 2021, Accessed: Feb. 25, 2021. [Online]. Available: http://arxiv.org/abs/2006.04200

[4] L. Lei, Y. Yuan, T. X. Vu, S. Chatzinotas, and B. Ottersten, "Learning-Based Resource Allocation: Efficient Content Delivery Enabled by Convolutional Neural Network," in 2019 IEEE 20th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC), Jul. 2019, pp. 1–5. doi: 10.1109/SPAWC.2019.8815447.

[5] F. Hussain, S. A. Hassan, R. Hussain, and E. Hossain, "Machine Learning for Resource Management in Cellular and IoT Networks: Potentials, Current Solutions, and Open Challenges," arXiv:1907.08965 [cs, eess], Jul. 2019, Accessed: May 04, 2021. [Online]. Available: http://arxiv.org/abs/1907.08965

[6] J. Heaton, "Ian Goodfellow, Yoshua Bengio, and Aaron Courville: Deep learning," Genetic Programming and Evolvable Machines, vol. 19, no. 1, pp. 305–307, Jun. 2018, doi: 10.1007/s10710-017-9314-z.

[7] S. Says, "Convolutional Neural Network (CNN) Tutorial In Python Using TensorFlow," Edureka, Nov. 27, 2018. https://www.edureka.co/blog/convolutional-neural-network/ (accessed Oct. 22, 2020).

[8] Chih-Chieh Hung and Chu-Cheng Hsieh, "Aggregation Network - an overview | ScienceDirect Topics," in Big Data Analytics for Sensor-NetworkCollected Intelligence, Jan. 05, 2017. https://www.sciencedirect.com/topics/computer-science/aggregation-network

[9] M. Fouad, N. Oweis, T. Gaber, M. Ahmed, and V. Snasel, "Data Mining and Fusion Techniques for WSNs as a Source of the Big Data," Apr. 2015, vol. 65. doi: 10.1016/j.procs.2015.09.023.

[10] S. Cai, B. Gallina, D. Nyström, and C. Seceleanu, "Data aggregation processes: a survey, a taxonomy, and design guidelines," Computing, vol. 101, no. 10, pp. 1397–1429, Oct. 2019, doi: 10.1007/s00607-018-0679-5.

[11] M. X. Cheng and W. B. Wu, "Data Analytics for Fault Localization in Complex Networks," IEEE Internet of Things Journal, vol. 3, no. 5, pp. 701–708, Oct. 2016, doi: 10.1109/JIOT.2015.2503270.

[12] A. J. Ratner, C. M. D. Sa, S. Wu, D. Selsam, and C. Ré, "Data Programming: Creating Large Training Sets, Quickly," p. 9, Jan. 2016, [Online]. Available: https://arxiv.org/abs/1605.07723

[13] A. Ratner, S. H. Bach, H. Ehrenberg, J. Fries, S. Wu, and C. Ré, "Snorkel: rapid training data creation with weak supervision," The VLDB Journal, vol. 29, no. 2, pp. 709–730, May 2020, doi: 10.1007/s00778-019-00552-1.

[14] G. B. Goh, C. Siegel, A. Vishnu, and N. Hodas, "Using Rule-Based Labels for Weak Supervised Learning: A ChemNet for Transferable Chemical Property Prediction," in Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, New York, NY, USA, Jul. 2018, pp. 302–310. doi: 10.1145/3219819.3219838.

[15] E. Bringer, A. Israeli, Y. Shoham, A. Ratner, and C. Ré, "Osprey: Weak Supervision of Imbalanced Extraction Problems without Code," in Proceedings of the 3rd International Workshop on Data Management for End-to-End Machine Learning, New York, NY, USA, Jun. 2019, pp. 1–11. doi: 10.1145/3329486.3329492.

[16] Y. Ran, X. Zhou, P. Lin, Y. Wen, and R. Deng, "A Survey of Predictive Maintenance: Systems, Purposes and Approaches," arXiv:1912.07383 [cs, eess], Dec. 2019, Accessed: Dec. 31, 2020. [Online]. Available: http://arxiv.org/abs/1912.07383

[17] S. Rastayesh, S. Bahrebar, F. Blaabjerg, D. Zhou, H. Wang, and J. Dalsgaard Sørensen, "A System Engineering Approach Using FMEA and Bayesian Network for Risk Analysis—A Case Study," Sustainability, vol. 12, no. 1, 2020, doi: 10.3390/su12010077.

[18] H.-C. Liu, FMEA Using Uncertainty Theories and MCDM Methods. Springer Singapore, 2016. doi: 10.1007/978-981-10-1466-6.

[19] S. M. Seyed-Hosseini, N. Safaei, and M. J. Asgharpour, "Reprioritization of failures in a system failure mode and effects analysis by decision making trial and evaluation laboratory technique," Reliability Engineering and System Safety, vol. 91, no. 8, pp. 872–881, 2006, [Online]. Available: https://ideas.repec.org/a/eee/reensy/v91y2006i8p872-881.html

[20] W. Ji, S. Duan, R. Chen, S. Wang, and Q. Ling, "A CNN-based network failure prediction method with logs," in 2018 Chinese Control And Decision Conference (CCDC), Shenyang, Jun. 2018, pp. 4087–4090. doi: 10.1109/CCDC.2018.8407833.

[21] F. Aziz, A. Ul Haq, S. Ahmad, Y. Mahmoud, M. Jalal, and U. Ali, "A Novel Convolutional Neural Network-Based Approach for Fault Classification in Photovoltaic Arrays," IEEE Access, vol. 8, pp. 41889–41904, 2020, doi: 10.1109/ACCESS.2020.2977116.

[22] H. Ren, Z. J. Hou, B. Vyakaranam, H. Wang, and P. Etingov, "Power System Event Classification and Localization Using a Convolutional Neural Network," Frontiers in Energy Research, vol. 8, p. 327, 2020, doi: 10.3389/fenrg.2020.607826.

[23] M. S. Singh, V. Pondenkandath, B. Zhou, P. Lukowicz, and M. Liwickit, "Transforming sensor data to the image domain for deep learning — An application to footstep detection," in 2017 International Joint Conference on Neural Networks (IJCNN), May 2017, pp. 2665–2672. doi: 10.1109/IJCNN.2017.7966182.

[24] A. Dehghani, O. Sarbishei, T. Glatard, and E. Shihab, "A Quantitative Comparison of Overlapping and Non-Overlapping Sliding Windows for Human Activity Recognition Using Inertial Sensors," Sensors, vol. 19, p. 5026, Nov. 2019, doi: 10.3390/s19225026.

[25] N. Haghighat, "Evaluating Airline Service Quality Using Fuzzy DEMATEL and ANP," Strategic Public Management Journal, vol. 3, no. 6, pp. 57–77, Dec. 2017, doi: 10.25069/spmj.351296.