Root Cause Analysis Frameworks for Information Systems

Vu Hong Hai Phan

A Thesis
in
The Department
of
Computer Science and Software Engineering

Presented in Partial Fulfillment of the Requirements
For the Degree of
Master of Computer Science at
Concordia University
Montréal, Québec, Canada

May 2022

CONCORDIA UNIVERSITY
School of Graduate Studies

This is to certify that the thesis prepared

By: **Vu Hong Hai Phan**
Entitled: **Root Cause Analysis Frameworks for Information Systems**

and submitted in partial fulfillment of the requirements for the degree of

Master of Computer Science

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

________________________ Chair
Dr. Charalambos Poullis

________________________ Examiner
Dr. Charalambos Poullis

________________________ Examiner
Dr. Sudhir Mudur

________________________ Thesis Supervisor
Dr. Brigitte Jaumard

________________________ Thesis Supervisor
Dr. Tristan Glatard

Approved by ________________________
Dr. Leila Kosseim, Graduate Program Director

__________ ________________________
Dr. Mourad Debbabi, Dean
Gina Cody School of Engineering and Computer Science

Abstract

Root Cause Analysis Frameworks for Information Systems

Vu Hong Hai Phan

Telecommunications systems have evolved to include an ever-growing number of interdependent hardware and software components with complex interactions. This exponential increase in complexity affects the reliability and stability of network systems. This thesis provides two systematic approaches to improve the speed and quality of the Root Cause Analysis task in telecommunications systems.

The first approach introduces a new fault analysis framework based on association rule mining and evaluates it for telecommunication systems. The approach describes a strategy using association rules to specifically target faults while improving runtime performance relative to the standard Apache Spark implementation. It also introduces a novel filtering strategy called Cover Set filtering that prunes and merges rule sets to produce high-quality, concise and interpretable results. The proposed framework is evaluated with real-world telecommunication datasets. Compared with other strategies, we demonstrate a better rule diversity in general and a sufficiently compact fault analysis.

The second approach tackles Root Cause Analysis from the causal perspective. It is based on Counterfactuals and Nearest Neighbour Matching concepts to identify fault types and highlight the most fault contributing variables. The proposed framework is a proof of concept for finding the root cause of problems based on the causal learning technique. It is demonstrated to be highly compatible with numerical data and highly robust with noisy data.

In conclusion, the proposed frameworks improve the quality and performance of fault troubleshooting tasks in telecommunication systems. Last but not least, the proposed frameworks can be adapted to other information systems with minor modifications.

Acknowledgments

First and foremost, I would like to express my profound gratitude to my research supervisors, Dr. Brigitte Jaumard and Dr. Tristan Glatard. Without their dedicated support, guidance and understanding, this work would have never been accomplished.

I would like to thank the experts from EXFO for their excellent collaborations. I sincerely appreciate Sylvain Nadeau and Justin Whatley for the ideas and discussions. I also would like to acknowledge Ludovik Mondou for his technical support.

Lastly, I would like to thank Mitacs and Concordia University for their financial support.

Contents

List of Figures **viii**

List of Tables **ix**

1 Introduction **1**

1.1 Motivation 1

1.2 Context 1

1.3 Plan of the thesis 2

2 Problem Statement and Background **3**

2.1 Fault Analysis in Telecommunications networks 3

2.2 Root Cause Analysis as Feature Association Problem 4

2.3 Causal Inference for Root Cause Analysis 5

2.4 Other Related Works 6

3 Root Cause Analysis with Association Rules **9**

3.1 Introduction 9

3.2 Fault Analysis and Association Rules 10

3.2.1 Frequent pattern mining 10

3.3 Fault analysis framework 12

3.3.1 Pre-processing 12

3.3.2 Data discretization 13

3.3.3 Mining of association rules 13

3.3.4 Filtering of association rules 15

3.4 Experiments 17

3.4.1 3G cellular network dataset 17

3.4.2 4G cellular network dataset 18

3.4.3 Performance improvement 21
3.4.4 Impact of discretization methods 22
3.4.5 Quality of filtered rules 23
3.4.6 Intersection and similarity of filtered rule sets 25
3.4.7 Sensitivity of the parameters in the filtering algorithm 29
3.5 Conclusions 30

4 Root Cause Analysis with Causal Inference and Embeddings 31
4.1 Introduction 31
4.2 Related materials 32
4.2.1 Dimensionality reduction 32
4.2.2 Clustering 34
4.3 Nearest Neighbour Matching in intermediate subspace 35
4.4 Experiments 37
4.4.1 Validation with synthetic datasets 38
4.4.2 Validation on real-world datasets 40
4.5 Conclusion 45

5 Conclusion and Future Work 46
5.1 Conclusions 46
5.2 Future works 47

Bibliography 49

List of Figures

1 Framework for fault analysis in big data environment 12

2 Runtime to generate association rules in seconds. 21

3 Intersections of the set of filtered rules. Two rules are considered similar if they have the same antecedent and consequent. 26

4 Similarity matrix the rules produced by CS filter in 4G dataset. The rules are reordered to form clusters of high similarity rules. The redundant areas are depicted with brighter colors. 27

5 Similarity matrix when comparing CS to other filters that work with rule-level. Rules are reordered using reverse Cuthill-Mckee algorithm [10] with 0.5 threshold binarize to move the large similarity pair closer to the main diagonal. 28

6 Similarity matrix of CS filter, ε-supp= $5e-2$, ε-conf $= 1e-1$. 29

7 Normal and Failure samples in the embedding space. The data is generated from i.i.d. random uniform distributions. 38

8 Normal and Failure samples in the embedding space of $\mathcal{S}_2$. The data is generated from i.i.d. random uniform distributions except for 3 columns, which associated with Failure samples. There are 4 clusters for Failure samples. 39

9 2-component PCA project of all data records. 41

10 2-component PCA project of non-normal data records. 41

11 The embedding projection of non-normal outcome extracted from k:1 nearest neighbour matching using PCA(40 components) + t-SNE (2 dimensions) embedding with DBSCAN cluster. 42

12 Comparison between matching method and FPM on 4G dataset. 43

List of Tables

1 An example of the records used for diagnosis 4

2 Fault distribution of datasets 17

3 Top 10 rules extracted from the 3G dataset for status Blocked, sorted by impact (support threshold $s = 1e - 4$, confidence threshold $c = 0.8$, $\varepsilon - supp = 5e - 2$, $\varepsilon - conf = 5e - 2$). To improve readability, features common to multiple rules are factorized. Cell Names are anonymized. . . . 18

4 Top 10 rules extracted from the 4G dataset for status Dropped, sorted by impact (support threshold $s = 1e - 4$, confidence threshold $c = 0.6$, $\varepsilon - supp = 5e - 2$, $\varepsilon - conf = 5e - 2$). Cell Names are anonymized. 20

5 Effect of discretization methods on runtime and size of the result, support threshold = $1e - 4$ and confidence threshold = $5e - 2$ 23

6 Comparison of filtering methods 24

7 Fault associated values extracted from Random noise with three fault-associated columns data set 40

8 The extracted top fault-contribute features from the Random noise with fault-associated data 44

Chapter 1

Introduction

This chapter provides an overview of the main topic of this dissertation.

1.1 Motivation

In many disciplines, errors, faults or problems are unavoidable. To reduce the impact of failures and potentially prevent them in the future, practitioners need to perform a series of thorough investigations to find their root causes. In engineering, the process of investigating the cause of problems is referred to as Root Cause Analysis.

In the telecommunication field, Root Cause Analysis is a vital component of the operating workflow. As networking technology evolves, the systems complexity becomes increasingly challenging for operators to monitor and resolve problems. Such complexity comes from the variety and volume of interactions between network components, which are already difficult for human operators to track manually. Unfortunately, identifying the root cause of problems is still an open research question due to many obstacles: complex component interactions, numerous points of failure, systems high evolution rate, limited time and cost. Therefore, the need for research on automatic root cause analysis methods is obvious.

1.2 Context

This work was performed in collaboration with EXFO Inc., a company focusing on network testing, monitoring and analysis. The work presented in this thesis is based on real-world problems and data of EXFO. It aims to develop methods and techniques to identify the root cause of network failure under the constraint of big data and time limits. The developed solutions are validated by telecommunication experts and proved to be promising solutions for deploying in real-world systems. The proposed solutions can also be used for root cause analysis in other information systems without significant modification.

1.3 Plan of the thesis

This dissertation is divided into five chapters. This current chapter provides an overview and context of the research. Chapter 2 will introduce the necessary technical background of the proposed approaches along with some related works. Chapter 3 and Chapter 4 are organized as separate conference papers and propose solution based on association rule and causal discovery approach respectively. Chapter 5.1 will provide the conclusions, open problems and future works.

The work from Chapter 3 has been published in IEEE Bigdata 2021 proceeding and submitted to US Patent Office:

- H-H Phan-Vu, B Jaumard, T Glatard, J Whatley, S Nadeau. "A Scalable Multi-factor Fault Analysis Framework for Information Systems". *Discriminative Pattern Mining Workshop, IEEE BigData 2021.* In this work, I contribute the methodology, experiments and the initial draft. Dr. Jaumard and Dr. Glatard provide academic guidance, feedback and manuscript modification. Whatley and Nadeau provide technical guidance, result evaluation and manuscript modification.

- H-H Phan-Vu, B Jaumard, T Glatard, J Whatley, S Nadeau. "Identification of clusters of elements causing network performance degradation or outage". *Patent (Pending) 17/649219.* In this work, I contribute the methodology. Dr. Jaumard and Dr. Glatard provide academic guidance and feedback. Whatley and Nadeau provide technical guidance and feedback.

The work of Chapter 4 will be submitted as another publication.

Chapter 2

Problem Statement and Background

This chapter starts with a detailed description of the research problem of this thesis, i.e., the fault analysis of a network system. After that, it will provide basic foundations for the work in Chapter 3 and Chapter 4. Some related works are also summarized in each respective section.

2.1 Fault Analysis in Telecommunications networks

Root Cause Analysis (RCA), also known as *Fault Localization* or *Fault Analysis* is the process of determining the sources of a problem or event. Root Cause Analysis is a crucial part of many disciplines, including industrial process control, IT operations, medical diagnosis, or epidemiology. In telecommunications, Root Cause Analysis is an essential component that helps reduce downtime and increases the information systems' stability. The work in [20] reviews standard techniques for solving RCA problems, including the five whys, fault tree analysis, Ishikawa diagrams and Pareto charts. All of them require one or many human analysts.

In telecommunications and network systems, fault analysis is commonly performed by professional experts. Usually, the network operators record the technical state of the network into a log file. The records can contain information related to the provided service, network and users. There are two primary types of records: Call Data Records (CDRs), which are created every time a subscriber establishes a call, and Service Data Records (SDRs), which are created when there is a new Internet connection in the network. Table 1 is an example of the data records from the telecommunication network; each row corresponds to a newly established call, and each column corresponds to a feature; the Status column describes if the call attempt is succeeded or not. The task of RCA is to identify the values of the elements that cause the degradation. However, even very experienced professionals cannot envision all possible explanatory factors to diagnose faults present in the data. Thus, when systems become more complex, traditional diagnostic approaches can lead to bias and might miss new or unusual patterns in the data. Further, manual explorations by experts can be time-consuming and computationally inefficient.

Table 1: An example of the records used for diagnosis

Timestamp	Cell ID	Device type	Service	Interface	Status
0	a1b1	ip5s	s1	i1	failed
1	a1b1	ss10	s2	i2	successful
2	a1b2	ip8x	s1	i3	successfull

As the complexity of information systems increases each year, the need for reliable methods to analyze large amounts of data from various input sources, representing many components of a system, has become increasingly important. Furthermore, uncovering the areas in a complex system that are possible root causes or at least warrant further investigation has become so challenging that human operators can no longer keep up with this task. Thus, automation of fault analysis is necessary to ensure service reliability and customer satisfaction experience in telecommunications systems.

2.2 Root Cause Analysis as Feature Association Problem

Many studies have created automated fault analysis or RCA for information systems. One can divide these automated techniques into two main categories: *deterministic* and *stochastic* [45].

In deterministic fault analysis models, there is no uncertainty when constructing models or getting inferences. The works in [6, 35, 42, 57] use a finite state machine as part of the reasoning process. However, such expert systems require many handcrafted features and can lack robustness. Some other techniques use supervised classification models to predict possible causes, such as Decision Trees [8] or Support Vector Machines [11, 55]. These techniques, however, can only provide predicted possible causes with minimal explanations. In addition, they can be difficult to enrich with prior knowledge, and will not easily generalize to new problems when the underlying data changes.

Stochastic methods used to find the possible causes of problems are robust and can be applied to many related tasks without significant modifications. The works in [14, 54] are among the first works using a neural network to detect and resolve faults in communication networks. While they are resilient to noise and inconsistencies in the data, stochastic methods are often hard to interpret due to the requirement to make assumptions about the data and set parameters and can therefore produce unexpected results. Another key issue with stochastic methods, especially deep learning classifiers, is that they are not designed for generating multiple dimensional results.

In [26], the authors introduced a framework using Apriori and FP-Growth [18], i.e., the two main association rule mining techniques, to find values strongly correlated with errors. When describing fault analysis as a rule finding problem, one can relate it to the Subgroup discovery [32], a special case of general rule learning. However, the goal of fault analysis is directed toward finding fault sources, whereas the target of Subgroup discovery also includes non-fault classes. The methods proposed in Chapter 3 of this thesis are most akin to the multivariable (multi-factor) fault correlation made possible by the high computational availability in recent years.

2.3 Causal Inference for Root Cause Analysis

Causal inference [33] is the process of estimating the effect of a treatment on the outcome of a data analysis, i.e., the data investigation for root cause analysis in the context of the study of this thesis.

There are two main approaches for learning causal relationships: Randomized Controlled Experiments (RCT) or Observational data. In RCT, participants are randomly assigned to either the treatment group or the control group, and the expected difference between these groups is the outcome of the variable being studied. However, such an experiment process is very expensive from a computational perspective and also requires substantial amounts of expert knowledge, e.g., to identify the values to divide the population into identical (i.e., same distribution) groups. For large systems, such a process is very complex to be put in practice concerning the partitioning process of equivalent groups.

Therefore, data-driven methods are favored to find causal relations from observational data, also referred to as *Observational Causal Discoveries* [16]. Observational data contains groups of individuals taking different treatments and outcomes from these treatments. Deducting causality from observational data is still a challenging task and there are multiple work directions to tackle it, all are based on Reichenbach's common cause principle [58]:

"if variables are dependent then they are either causal to each other (in either direction) or driven by a common driver". There are two main systematic approaches: Structure Causal Models (SCM) [33] and Potential Outcome Framework (POF) [41], which are logically equivalent. In SCM, the causal mechanisms are described using *causal graphs* and *causal equations*, which provides the ability to learn causal relations among any variable but requires knowing the complete causal graph. In contrast, the Potential Outcome Framework (POF) does not require such knowledge, which is more suitable for the goal of discovering the causal relationships in the context of this work. In the remainder of this section, we will provide the basic definitions of POF and their equivalent with the context of this work.

The atomic research object in causal inference is the *unit*, e.g., an individual or a particular time point. In this thesis, the equivalent of a unit is a data record; the term "record", "data point", "sample" will be used interchangeably. A *treatment* is an action that applies to a unit, i.e., assign a group of values to a group of a unit's features. In terms of Association Rules, a unit's counterpart is a data record and a treatment's counterpart is a basket. The *outcome* is the result of the treatment applied to a unit. In the RCA context, it is the recorded final status (fail/success) or Key Performance Indicators (KPIs) values. The *observed outcome* is the actual outcome of the treatment while *counterfactual outcome* is the outcome of the unit's treatment *if* the unit has taken a different treatment. Compared to Association Rules approach, the casual framework provides the concept of "what if" scenarios in the term of counterfactual.

There are two main problems in causal inference: learning the *causal effects* and learning *causal relations* [16]. Learning causal effects aims to understand the method to quantify the interaction between events, while learning causal relations seeks to formulate the interaction of events. According to Pearl [33], these are from an *observational* perspective and *interventional* perspective, respectively. In both perspectives, learning causal relations or *Causal discovery* for cross-sectional data (i.e., no regard for differences in time) is the most investigated causal task [31]. Generally, Root Cause Analysis (RCA) can be described as a Causal Discovery task with big data and latent variable constraints. However, RCA can also be described as a learning causal effects task with some modification. The detailed justification can be found in Chapter 4.

2.4 Other Related Works

In addition to the previously mentioned approaches, several other approaches should be mentioned. They include Classification, Feature Importance Estimation, and Abductive

Reasoning.

RCA can be viewed as a classification task where the goal is to find the groups of interested variables related to the event. In [15], the authors proposed an offline method based on classification models to identify dependencies among system variables and events automatically, followed by a summarization step. The framework operates on graphs and visualization tools. Concretely, the proposed method aimed to create a knowledge base automatically based on the "influence matrix" concept, where the interaction between variables is described. In [4], the authors aim at two objects: to detect anomaly states in virtual network infrastructure and diagnosis the root cause of these states. The former is achieved by an unsupervised learning k-Means, while the latter is performed by a Support Vector Machine classifier. In the RCA stage, a graph that describes the dependencies among events in the system was created where the nodes are misbehaving components and edges are propagation paths. SVM one-class classification is applied to determine if there is an edge between two nodes in the graph. Alaeddini and Dogan [3] proposed a Baysian network architecture for capturing the cause-effect relationship using a list of pre-determined patterns. Overall, the classification approach can take advantage of advanced machine learning models but require a limited pre-determined number of possible causal-effect candidates and may also require extensive model training.

RCA can also be considered a Feature Importance Estimation task for an individual event. Therefore, solving Feature Importance Estimation problems can also be considered as an alternative for solving RCA. From that interpretation, RCA can be solved either indirectly or directly. In the indirect approach, we first need to build a classification model with input measurements as input features and the event (normal/failure) as labels and then extract the importance of features for each sample or groups of samples as the wanted causal factors [7, 27, 39]. Opposite to the indirect approach, the direct approach does not require access to any classification model. The feature importances are directly extracted from the data itself. The approaches that can be considered include Prototypes and Criticisms [23] or Counterfactual Explanations [38].

Finally, Abductive Reasoning, Abductive Inference or Abduction [22] is a logical reasoning process that extracts the precondition from a consequence. In RCA, the abduction of an event is based on a given theory relating the events with their effects and a set of observed effects. For computational purposes, Abductive Reasoning is often implemented under the Markov Logic Networks, a formal combination of logical formulas and probabilities. In [43] and [57], the prior knowledge about the systems is previously injected into the inference network, the inference about the root causes of events are extracted such that it fits the scenario with the highest probability given the observed events. Abductive Reasoning is also considered a rule-based approach, which gives exceptional computation efficiency but requires extensive engineering efforts and is not suitable for highly evolving systems.

Chapter 3

Root Cause Analysis with Association Rules

3.1 Introduction

Despite the concept behind fault analysis are *causality* and *explanation*, a fault analysis model does not necessarily satisfy all the requirements of a causal model. There are wide ranges of aspects and requirements that can affect the nature of the fault analysis problem. The first concern is about the *intent*, whether to only obtain the group of components/interactions that caused the problem or obtain the explanation. The second aspect is *problem complexity*, which involves the time allowed to spend, the system size, the effect of error propagation, the system's evolution rate and the number of involved components (single or multiple causality). A third important aspect is the required *knowledge* about the *domain* (about the problem) and the *system* (about the diagnosing system) [45]. From these requirements, we define a framework for generic postmortem fault analysis which works with generic information systems and has the following properties: (1) be able to provide possible problem causal components/interactions; (2) be able to work with any problem complexity; (3) work as an assisted generation model: it requires some partial knowledge about possible causal components and investigating the results; (4) be easy for operators to understand the model and interpret the results.

Our proposed model for fault analysis is based on the concept of Frequent Pattern Mining (FPM), also known as association rule mining [2], which is an effective avenue for fault analysis due to its ability to identify multi-factor patterns associated with system statuses in complex systems. Although there is potential in FPM for fault analysis, strategies to reduce the high-computational cost of FPM and improve the quality of the resulting associations are critical for real-world applications.

In this work, the key contributions are:

(i) analysis and demonstrate that frequent pattern mining is an appropriate approach for fault analysis of complex systems

(ii) improvement of FP-growth and a new filtering procedure that further reduces the size of the set of filtered rules, while improving its performance (*support/confidence*) metrics, result interpretability and scalability

(iii) expert validation of the refinements and increased scalability of the improved algorithms on two real-world telecommunication datasets consisting of 3G and 4G call-data-records

(iv) generic tools (i.e., combination of similarity matrices with various metrics) for comparing different sets of filtered rules, in which rules are compared on their meaning, and not only on their analytical expressions.

3.2 Fault Analysis and Association Rules

This section reviews approaches for fault analysis and presents background information on association rules.

3.2.1 Frequent pattern mining

First introduced in [2] for market basket transactions, Frequent Pattern Mining (FPM) has become a widely used method in many big data related tasks such as recommendation systems, Web mining, bioinformatics, and medical diagnosis [25, 50]. FPM is used to find meaningful patterns, associations, or clusters from a data set consisting of transaction records, such as a transactional database. Consider a transactional database $T = \{t_1, t_2, \dots, t_N\}$ that contains N records, where $t_i \subseteq I$ represents a basket containing one or multiple items from $I = \{i_1, i_2, \dots, i_M\}$, the set of all possible items. For each itemset $X \subseteq I$, the *support* of X is defined as follows. Let $\sigma(X)$ be the number of transactions containing itemset X:

$$\sigma(X) = |\{t_i : t_i \in T, X \subseteq t_i\}. \quad (1)$$

The *support* [2] of X is the proportion of transactions containing X, calculated as follows:

$$support(X) = \frac{\sigma(X)}{N}. \tag{2}$$

An itemset is called frequent (frequent pattern) when its *support* is greater than a predefined threshold.

Association rules defined from frequent patterns show the co-occurrence between itemsets. An association rule takes the form of $X \to Y$, where X and Y are disjoint itemsets, X is called *antecedent* and Y is called *consequent*.

In addition to *support*, *confidence* is also used to measure the "relevance" of rules. The *confidence* of a rule determines the probability of the consequent given the antecedent:

$$confidence(X \to Y) = \frac{\sigma(X \cup Y)}{\sigma(X)}. \tag{3}$$

The brute-force approach to compute every possible rule is extremely expensive. For M unique items, the number of all available rules is $3^M - 2^{M+1} + 1$ [50]. Therefore, in most applications, only rules with *support* and *confidence* greater than predefined thresholds are kept. In the case where association rule mining is used for a certain type of consequents (i.e., faults), we are mostly interested in the *support* of the antecedent, defined as the rule *cover*:

$$cover(X \to Y) := support(X). \tag{4}$$

The exclusive use of *support* and *confidence* can produce misleading results. For example, consider the scenario where X and Y are frequent itemsets and *support*(X) as well as *confidence*($X \to Y$) both satisfy the pre-defined thresholds. In this case, $X \to Y$ is accepted while in reality, X and Y can be mutually independent. To address this potential for misleading results, the *lift* (also known as interest) [5] metric is defined:

$$lift(X \to Y) := \frac{\sigma(X \cup Y)}{\sigma(X) \times \sigma(Y)} = \frac{P(X, Y)}{P(X) \times P(Y)}. \tag{5}$$

The *lift* metric compares the observed frequency of a pattern against the baseline frequency computed as if the antecedent and consequent were independent. When *lift* > 1, the antecedent and the consequent are dependent to some extent, while *lift* ≤ 1 indicates independence or anti-correlation. In this work, *lift* is mainly used as a preliminary filter to construct rules where the antecedent and consequent are significantly related.

Finally, to measure the proportion of all observed faults explained by a rule, we define the *impact* measurement.

$$impact(X \to Y) := \frac{\sigma(X \cup Y)}{\sigma(Y)}. \tag{6}$$

3.3 Fault analysis framework

A high-level view of the fault analysis framework is illustrated in Figure 1. The following sub-sections describe these components in details.

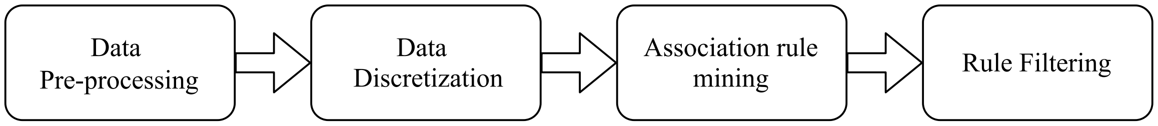


Figure 1: Framework for fault analysis in big data environment

3.3.1 Pre-processing

The data used for mining association rules typically comes from structured data: tabular data where rows correspond to event records and columns to event parameters or the system state. The size of such data grows considerably in complex systems, with hundreds of columns and millions of rows. A common option for storage is the Hadoop Distributed File System (HDFS) [44], which is ideal for a use-case where the features most relevant for a problem of interest are pre-selected in parsing.

Since our model is designed to be an assisted generation model, the first pre-processing problem is selecting fields of interest. In many cases, there are hundreds of columns in the database and only some of them will be relevant for fault localization and analysis. Including them all would not only greatly increase computational complexity, it would also add a lot of irrelevant characteristics to the generated rules. Unfortunately, since potential problematic candidate fields are task dependent, choosing fields algorithmically, as in the case of *automated generation* models, is a difficult task. To further complicate field selection, data can be organized with certain fields representing objective characteristics of the record (e.g., hardware version), measurements, or even the result of predefined rules to characterize problems. Thus, as many of these characteristics might be useful depending on the use case for fault analysis, field selection must be determined carefully.

In order to select the most relevant columns for the use-case, columns were chosen primarily based on the recommendations of the experts for their relevance to the problem at hand. Once selected, data from the remaining columns are converted to the market-basket structure commonly used in FPM: each basket corresponds to a row and items are defined as <*column_name*>= <*value*>. To reduce the number of items in the mining process, every item *not* containing the target faults are removed.

3.3.2 Data discretization

Frequent pattern mining algorithms are designed to work with discrete data, such as categories and discrete ranges. Consequently, all continuous values must be transformed through a process called *Discretization* or *Binning* prior to use in FPM. There are multiple possible approaches for discretization, such as equal frequency discretization (EFD), equal width discretization (EWD), statistic-based discretization (ChiMerge, StatDisc), etc. [28].

The selection of the right discretization method must consider the number and distribution of the generated categories. When many categories are generated, the frequency of each category is reduced, which decreases the likelihood that an individual bin is part of a frequent pattern or that many adjacent bins appear in different rules describing the same causal group. The number of generated categories also increases computational times, as more items must be evaluated. Finally, the distribution of generated categories can also directly affect the quality of the discovered rules by skewing the original data distribution.

In this study, we consider the use of three classical discretization methods [17]: Equal Width Discretization (EWD), Equal Frequency Discretization (EFD) and rounding to integers (a particular case of EWD). The EWD method divides numerical values into k bins of equal interval length, where k is predefined. The EFD method divides numerical values into k bins such that each bin contains approximately the same number of data points. The main difficulty in applying either method is in selecting the best number of bins k.

The rules created from discretized items need an interval merging step to increase the quality. Concretely, when two rules differ only in neighbouring intervals, the two rules are merged to become one rule with the union of both intervals. For example, two rules $\{i_1, i_2, \ldots, [v_t, v_{t+1}), \ldots, i_m\} \rightarrow Y$ and $\{i_1, i_2, \ldots, [v_{t+1}, v_{t+2}), \ldots, i_m\} \rightarrow Y$, can be merged to into $\{i_1, i_2, \ldots, [v_t, v_{t+2}), \ldots, i_m\} \rightarrow Y$ if their measurements are approximately equal.

3.3.3 Mining of association rules

Several factors influence algorithm selection for the mining of association rules. First, there can be large differences in the number of rules generated depending on the dataset used to produce these rules. For instance, fault analysis conducted on data from regular system logs will require a low *support* threshold because faults are not frequent, while analysing rules for logs of a system failure can use a much higher threshold. To address this issue, we propose to apply the frequent pattern mining algorithm to find clusters of items that are associated with system faults. A second factor is the scalability of the algorithm, as many fault diagnosis tasks are conducted on datasets with millions of entries. Finally, it must be decided whether the algorithm should produce deterministic or approximate results. Although efficient, approximation mining algorithms may lack important frequent patterns necessary to characterize a fault.

In this work, we used a modified version of the FP-Growth [18] algorithm which we modified to extract frequent patterns and generate association rules for the target faults. The FP-Growth algorithm consists of two steps: *(i)* Construct frequent-pattern trees (FP-trees) and *(ii)* Generate frequent itemsets from FP-trees. An FP-tree is a tree-like data structure where the root is labelled as “NULL” and other nodes contain three fields: *item-name* that represents the item, *count* that records the number of transactions that contains items from the root to the current node, and *node-link* that points to other nodes in the same tree that represent the same item. FP-trees often come with a look-up table storing the actual items and pointers (item-names) to the first node that contains the item.

Frequent itemsets are generated by traversing FP-trees in a bottom-up fashion where the support count of a combination is the *count* of the deepest node in the corresponding path. The tree traversal can subsequently be done using a divide and conquer algorithm. In our variation, the look-up table for FP-trees is optimized by removing any item that is not related to the fault, i.e. item i_k is ignored if $\sigma(i_k, fault) = 0$. The fault-related items are encoded using integers when scanning the dataset for the first time (determining items' support count phase). This helps to reduce the size of the trees and make moving data more efficient.

After obtaining frequent patterns, the association rules are usually extracted based on the *confidence* threshold. In this work, we improved this rule generation phase by targeting the target fault directly, where the consequents are faults. Since the rule extraction phase is based on the previous frequent patterns, we can reduce the number of rules generated with a frequency constraint based on a specific target consequent (i.e., the target faults).

One can argue that FP-Growth can be applied directly to the fault-related transactions to generate frequent patterns based on the impact threshold (i.e., the *support* threshold for the fault-related transactions) and compute *confidence*, *support* and *lift* in a post-processing step. This approach requires traversing the data at least one more time and does not guarantee the reductions of antecedent candidates.

3.3.4 Filtering of association rules

In this framework, the process of mining in Section 3.3.3 and selecting rules performs in two different stages. The reason is that the result of FPM contains many interactions of elements, therefore, although merging the extraction and selection process can lead to a significant reduction in compute resources, it can also lead to biased results by covering unpopular / exotic models. The design into different stages helps to increase interpretability by allowing the human operator to retrace all the rules generated if he has any doubts about the result.

When using frequent pattern mining for fault analysis, the number of found rules is bounded by an exponential function of the number of unique items 2^M, which can result in the production of too many rules to offer insight into a fault. The number of rules can be considerably reduced by applying thresholds for *support*, *confidence* and *lift*, but this can lead to ignoring interesting patterns which could have values in fault analysis. Thus, tuning these thresholds to ensure the result's compactness and interpretability is a challenge. In addition, filtered rules also contain many redundant rules that describe the same association.

Different filtering methods have been devised to address this issue. In [49], other measurements are used as filtering/ranking after obtaining the mined rules. In **FAR** filtering [13], the best rules are selected using an iterative procedure based on the highest *support* and highest *confidence*. The authors of [21] proposed a rule with a selection method based on hierarchical clustering, which we referred to as **HC** filtering. In [26] (**FB** filtering) proposed a filtering method based on *support* and *lift* for controlling the condition on the rule antecedents' lengths. In [51], mining and selecting rules happen concurrently, which involves traversing the transaction database for at least a logarithmic factor of the number of transactions..

To address the drawback of the unfocused generation of excessive and overlapping rules, we introduce a novel Cover Set (**CS**) filtering procedure, which consists of two parts: a pruning algorithm (Algorithm 1) and a merging algorithm (Algorithm 2). The **CS** filtering is motivated by real-world set of rules for dealing with the potential root causes of a failure, and their order of priority based on their explanatory factors to the failure.

Pruning in the Cover Set filtering algorithm leverages the following observations: consider a set of items that are "good enough" at explaining a fault. For such a set, adding a new item to the set should not add much value since the original set of items already explains the fault. In other words, the goal is to reduce the number of rules that describe the same group of possible root cause items so that each rule is as concise as possible. In the Cover Set filtering algorithm, the relaxation terms ε-supp, ε-conf are in a range of [0, 1). Since the antecedents' *support* vary from dataset to dataset, the ε-supp threshold is designed to control the *support* decline rate.

Algorithm 1 Cover Set: pruning

```
Input: Set of rules $\mathbf{R} = \{X_i \rightarrow Y : i = 1 \dots n\}$,
  support relaxation term $\varepsilon$-supp, confidence relaxation term $\varepsilon$-conf
Output: Rule cover $\Delta_p$
  $\Delta_p := \emptyset$
  while $|\mathbf{R}| \neq 0$ do
    $r_i \leftarrow (X_i \rightarrow Y) \in \mathbf{R}$ where $i = \arg\max_{r_j \in \mathbf{R}} \mathit{cover}(r_j)$
    $\Delta_p \leftarrow \Delta_p \cup \{r_i\}$
    $\mathbf{R} \leftarrow \mathbf{R} \setminus \{r_i\}$
    for $r_k \in \mathbf{R}, X_i \subset X_k$ do
      $s \leftarrow [\mathit{cover}(r_i) - \mathit{cover}(r_k)] / \mathit{cover}(r_i) \geq \varepsilon$-supp
      $c \leftarrow \mathit{confidence}(r_k) - \mathit{confidence}(r_i) \leq \varepsilon$-conf
      if $s \vee c$ then
        $\mathbf{R} \leftarrow \mathbf{R} \setminus \{r_k\}$
      end if
    end for
  end while
```

An additional merging step is used to combine rules, further compressing the results for fault analysis. To do this, the following reasoning is applied: if there exist two rules $X_1 \rightarrow Y$, $X_2 \rightarrow Y$ with heavily dependent antecedents $P(X,Y) \approx P(X) \approx P(Y)$ and the items from their antecedents coinciding, those antecedents can be merged to one larger antecedent. Not only a larger antecedent helps to investigate a problem, but it also reduces the number of rules that needs to be considered. The merging procedure is described in Algorithm 2. Although merging can help reduce the number of rules, it also decreases the quality (*confidence*) of merged rules, the detail effect will be discussed in Section 3.4.7. The filters have two hyperparameters: ε-supp and ε-conf, which have been defined empirically for now, future work could study a theoretical approach to automatically define these hyperparameters based on data characteristics.

Algorithm 2 Cover Set: merging

```
Input: Result of Cover Set: pruning $\Delta_p$, support relaxation term $\varepsilon$-supp, confidence relaxation $\varepsilon$-conf
Output: The final cover $\Delta$
  for $r_i, r_j \in \Delta_p$ do
    $C_1 \leftarrow |\mathit{confidence}(r_i) - \mathit{confidence}(r_j)| \leq \varepsilon$-conf
    $C_2 \leftarrow |\mathit{cover}(X_i \cup X_j) - \mathit{cover}(X_i)|/\mathit{cover}(X_i) \leq \varepsilon$-supp
    $C_3 \leftarrow |\mathit{cover}(X_i \cup X_j) - \mathit{cover}(X_j)|/\mathit{cover}(X_i) \leq \varepsilon$-supp
    if $C_1 \wedge C_2 \wedge C_3$ then
      $\Delta \leftarrow \Delta \setminus \{r_i, r_j\}$
      $\Delta \leftarrow \Delta \cup (X_i \cup X_j \rightarrow Y)$
    end if
  end for
```

3.4 Experiments

Two cellular network datasets were used to assess the proposed fault analysis framework. Along with the evaluation of the rules for these datasets, the time execution performance of the modified FPM was evaluated as well as the consequences of discretization methods, and an evaluation of the impact of different filters on the interpretability. For client's privacy reasons, we cannot provide the public version of the dataset.

Table 2: Fault distribution of datasets

Status	3G dataset		4G dataset	
	Count	*support*	Count	*support*
Normal	14,578,263	87.8%	1,458,773	96.4%
Dropped	258,829	1.6%	8,984	0.6%
Blocked	1,333,789	8.0%	4,330	0.3%
Non-Progress	440,664	2.6%	88	0.1%
Undefined	0	0.0%	39,906	2.6%

3.4.1 3G cellular network dataset

The first dataset was extracted from a 3G cellular network over 24 hours. The fault distribution in the dataset is presented in Table 2. The data consists of approximately 16 million rows with 28 columns. We selected 16 columns, all categorical types, as candidates for fault analysis as they relate to call status failures. Of the possible call failure outcomes there are three categories: Dropped, Blocked and Non-Progressed. The most common failure category for this dataset is Blocked call, which accounts for approximately 8.0% of all records and is the fault analyzed for this use case. After converting to items of the form <*column_name*>= <*value*>, the dataset contains 16,286 unique pairs.

The rules extracted from the dataset are presented in Table 3. The *support* and *confidence* thresholds were chosen manually to balance the execution time, the number of produced rules, and the quality of the rules following multiple experiments. The most common reason for blocked calls were an $Undefined$ service type (not to be confused with the Undefined call status category), with a high impact score of 77.9%. It corresponds to cell towers block calls with $Undefined$ service type, leading to an interrupted session or a call failure. Another common, albeit less prevalent, rule demonstrated that calls were more often blocked during $Handover$.

Table 3: Top 10 rules extracted from the 3G dataset for status Blocked, sorted by impact (support threshold $s = 1e-4$, confidence threshold $c = 0.8$, $\varepsilon - supp = 5e-2$, $\varepsilon - conf = 5e-2$). To improve readability, features common to multiple rules are factorized. Cell Names are anonymized.

Antecedent	*cover*	*confidence*	*lift*	*impact*
Service = Undefined	0.0625	1.0	12.5	77.9%
Establishment Cause = Handover				
Cell Name = 1	0.0093	0.824	10.3	9.52 %
Cell Name = 2	0.0069	0.848	10.6	7.28 %
Cell Name = 3	0.0031	0.839	10.5	3.24 %
Cell Name = 4	0.0010	0.864	10.8	1.04 %
Cell Name = 5	0.0008	0.896	11.2	0.914%
Cell Name = 6	0.0008	0.892	11.1	0.856%
Cell Name = 7	0.0006	0.859	10.7	0.691%
Cell Name = 8	0.0005	0.828	10.3	0.526%
Cell Name = 9	0.0005	0.866	10.8	0.504%

3.4.2 4G cellular network dataset

FPM was also applied to a 4G dataset to analyze faults related to a more modern cellular network. This dataset contains approximately 1.6 million rows from 23 selected columns, containing 15 numerical columns. As in the previous dataset, there are three categories of call status failure (Dropped, Blocked and Non-progressed) and the distribution of call statuses is detailed in Table 2. The Undefined category is excluded from the analysis as not directly associated to call status failure. 4G being a more mature technology, there is a considerably lower incidence of call failures overall. The most common type of fault in this dataset is Dropped calls.

The top 10 extracted rules are presented in Table 4. From the results, the most common reason for dropped calls is a failed attempt to reconnect during $Handover$. Tuple (Establishment cause = LTE reestablishment, Handover Attempts = Failed, Handover Successful = Failed) is present in the first rule (highest impact) and in most of the top-10, highest impact rules. It clearly indicates that unsuccessful $Handover$ is the predominant cause of dropped calls. The additional items forming each rule help characterize and reduce the possible causes of dropped calls. For example, very low uplink signal-to-interference-plus-noise (UL SINR) of $[-5.0, -4.0)$ is likely a contributing factor for dropped calls and is present in many of the rules. In a more specific example, in the second most impacting rule, item Cell Frequency UID = Band A indicates that this specific frequency band is more strongly associated with dropped call than the others. In general, faults were not as common in the 4G network dataset as in the 3G dataset. Besides, impact distributions were more uniform, with the highest impact rule at 29.82% and the lowest in the top-10 at 2.75%. The 4th rule is redundant, considering the 5th rule due to the current ε-supp condition is set to 0.05 while the difference in support is 0.12. Nevertheless, the 4th and 5th rules reassure the importance role of Handover to the failures.

In this experiment, discretization methods have been applied on numerical columns (e.g., UL SINR, End cell Quality). The impact of the discretization methods is discussed in Section 3.4.4.

Table 4: Top 10 rules extracted from the 4G dataset for status Dropped, sorted by impact (support threshold $s = 1e-4$, confidence threshold $c = 0.6$, $\varepsilon - supp = 5e-2$, $\varepsilon - conf = 5e-2$). Cell Names are anonymized.

Antecedent	Cover ($\times 10^{-4}$)	*confidence*	*lift*	*impact*
Carrier Aggregation Flag Set = False Establishment cause = LTE reestablishment Handover Attempts = Failed Handover Successful = Failed	27.59	0.64	108.06	29.82%
Carrier Aggregation Flag Set = True Cell Frequency UID = Band A Handover Attempts = Failed Handover Successful = Failed UL SINR = [-5.0, -4.0)	12.08	0.61	103.03	12.44%
Carrier Aggregation Flag Set = True Connection Type = Data Handover Attempts = Failed Handover Successful = Failed UL SINR = [-5.0, -4.0) User Equipment Category = B	5.03	0.66	110.62	5.57%
Carrier Aggregation Flag Set = False Cell Frequency UID = Band B Establishment cause = LTE reestablishment	4.03	0.67	113.25	4.57%
Carrier Aggregation Flag Set = False Cell Frequency UID = Band B Establishment cause = LTE reestablishment Handover Attempts = Failed Handover Successful = Failed	3.52	0.77	128.88	4.54%
Establishment cause = LTE-mo-data UL SINR = [-5.0, -4.0) User Equipment Category = B	3.37	0.65	108.82	3.66%
Avg. Channel Quality = [3.0, 2.0) Carrier Aggregation Flag Set = True Handover Attempts = Failed Handover Successful = Failed UL SINR = [-5.0, -4.0)	3.11	0.62	105.09	3.27%
Avg. Channel Quality = [4.0, 3.0) Carrier Aggregation Flag Set = True Handover Attempts = Failed Handover Successful = Failed UL SINR = [-5.0, -4.0)	2.89	0.64	107.10	3.09%
Handover Successful = Failed UL SINR = [-5.0, -4.0) End cell Quality = -20.0	2.67	0.66	111.27	2.97%
End cell UID = 5380 Establishment cause = LTE reestablishment	2.50	0.78	132.02	2.75%

3.4.3 Performance improvement

Experiments were conducted to compare runtimes of two Association rule generation strategies based on FP-Growth: the implementation available in Spark [56] and the novel implementation described in Section 3.3.3. The runtime was measured in a Spark cluster with 8 workers, 1 executor per worker, 4GB RAM per worker, and 1 CPU core per worker. For each minimum *support* threshold, Association rule generation was executed 7 times and the average runtime was reported. As can be seen in Figure 2, our modified association rule generation resulted in a significantly shorter average execution time for all minimum *support* thresholds.

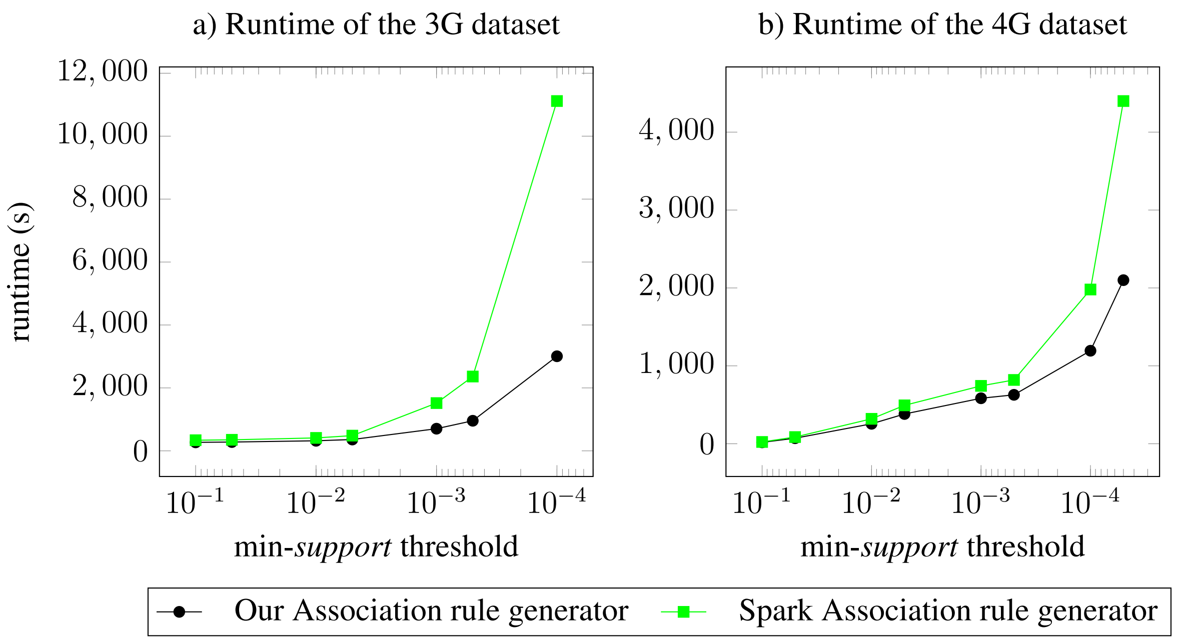


Figure 2: Runtime to generate association rules in seconds.

As anticipated, the runtime of Association rule generation increased exponentially when decreasing the minimum *support* threshold value. The improved performance observed with our modified Association rule generation comes from the reduced number of generated association rules to evaluate. For such an experiment to be conclusive, the *support* threshold must be chosen carefully to ensure a sufficient number of "interesting" rules and runtime. The *support* threshold of 10E-4 was chosen to consider a sufficiently broad rule set while ensuring achievable execution time. The key reason to restrict rules from being formed with low *support* is that these rules have a much higher chance of being spurious, only existing by coincidence. Further, restricting *support* prior to rule formation improves runtime performance, as previously discussed.

On the other hand, the *confidence* threshold only impacts the number of generated rules and is chosen to facilitate interpretability. The *confidence* threshold is decided based on dataset characteristics and the observed output, with the best *confidence* threshold of 0.8 for the 3G dataset, and 0.6 for the 4G dataset. Consequently, such an approach will likely involve some level of domain-specific expert tuning prior to integration into a system.

3.4.4 Impact of discretization methods

The 4G dataset was used to evaluate the effect of discretization of numerical columns on runtime, the number of resulting association rules and the quality of results. The 3G dataset does not have numerical columns and, thus, is excluded from this analysis.

The EFD method is considered first when it comes to converting numerical columns into categorical columns. This method is widely used in many machine learning problems since it treats the generated categories equally. The numerical values are divided into intervals so that each interval contains approximately an equal number of datapoints.

When applying the equal frequency method to FPM, some drawbacks need to be considered. First, it changes numerical values into equally distributed categories, which means that when building the FP-tree, items in each newly generated category are either present in the tree or not at all. The second drawback is that the generated rules can have lower *confidence*, since the faults usually happen within a small interval of value.

Alternatively, the equal-width binning can be a promising option, but still requires choosing the correct number of bins. As a general rule, the number of bins should be large enough to avoid considering only popular items. On the other hand, selecting a very large number of bins will reduce the *support* and impact of all generated rules and require $O(N \log(N))$ (sorting then traverse) operations to recombine the neighboring rules.

Table 5: Effect of discretization methods on runtime and size of the result, support threshold $= 1e - 4$ and confidence threshold $= 5e - 2$

Method	Bins per numerical columns	Runtime (hours)	Number of rules	Kept numerical values (%)
EFD	10	0.73	62,814	100.00
	20	0.46	45,716	”
	40	0.39	22,375	”
	80	0.36	12,676	”
	100	0.34	11,030	”
EWD	10	5.10	125,713	100.00
	20	2.26	32,823	99.99
	40	0.76	15,332	99.98
	80	0.48	9,447	99.96
	100	0.44	8,704	99.96
Categories $\rightsquigarrow$ integers	100 (average)	0.33	7,936	97.79

Table 5 compares the runtime, the number of resulting association rules and the quality of results related to each binning method. We have observed for the 4G dataset that by increasing the number of bins, the execution time decreases In addition, when the size of the intervals becomes too large, the quality of the results deteriorates considerably (due to loss of information). From these results, the Categories $\rightsquigarrow$ integer method provides the best results.

3.4.5 Quality of filtered rules

Filtering improves interpretability by reducing the number of rules required to explain a given set of faults. In this section, we investigate the quality of generated rules and compare with 3 other filter methods: FB, FAR and HC filtering (Section 3.3.4). While the number of rules before filtering was far too high to be practical or actionable, filtering significantly reduced the number of rules generated in both datasets, see Table 6. Despite the large number of deleted rules, the total impact remained close to the value obtained before filtering, which shows that the filtered rules describe very well the root causes for 3G, and a slightly less for 4G.

Other characteristics of the filtered results such as the number of rules, number of items found across all rules, the average length of the rules and diversity are shown in Table 6. For the FB filter, the support and confidence filter conditions are all set to ***1.01***. For the CS filter, the number of required rules is set to ***30***. There are no hyper-parameters for FAR filter. In particular, rule diversity indicates how well the algorithm produces a diverse set of rules to characterize the fault. In the 3G dataset, Cover Set (CS) filtering has an item diversity ratio (items over rules) higher than FB but lower than FAR. HC filtering was not possible due to the memory requirements exceeding the capacity of our computing infrastructure. While HC filtering offers the best diversity for the 4G dataset, CS filtering had the second-best diversity score. In the last row of Table 6, we report the results with the top-10 rules of CS filtering, sorted according to their *impact* values, which acts as a brief report for the human operator.

Table 6: Comparison of filtering methods

Method	3G dataset					4G dataset				
	Number of rules	Number of items	Avg. length of antecedent	Diversity (# items/ # rules)	Total impact (%)	Number of rules	Number of items	Avg. length of antecedent	Diversity (# items/ # rules)	Total impact (%)
Before filtering	107,147	218	7.31	2×10^{-3}	100.00	511	78	4.10	0.15	54.50
FB filtering	285	90	3.56	0.31	100.00	146	33	4.03	0.23	54.50
HC filtering	N/A	N/A	N/A	N/A	N/A	30	31	5.33	1.03	51.28
FAR filtering	11	15	3.55	1.36	99.98	20	14	4.45	0.70	38.84
CS filtering	59	66	3.27	1.11	100.00	42	29	3.97	0.69	50.79
CS filtering @ top10	10	25	5.30	2.50	82.97	10	15	4.10	1.50	29.82

Quantitative analysis of the quality of filtering methods is difficult. There are multiple desiderata for good compact set of rules: *(i) The size of the set:* the smaller the set, the better for human tracking; *(ii) Coverage of the set:* the filtered set should be able to provide a high fraction, ideally of all, potential root cause items; *(iii) Quality of the measurements:* the higher *support* and *confidence* of each individual rule in the set, the better it helps to perform the fault analysis; *(iv) The correlation between rules in the set:* the rules should have no high similarity between them. In other words, each rule should explain a different perspective (e.g., root cause) of the fault analysis.

The desiderata mentioned above are strongly entangled For example, high coverage often increases the size of the set, or a set of rules that have high confidence cannot cover all of the data. In this work, we give great preference to desiderata *(iv)*. In an ideal scenario, the similarity matrix of the resulting rule sets should be the identity matrix, or the rules to be proved equivalent.

3.4.6 Intersection and similarity of filtered rule sets

To further compare the filtering strategies, the intersection of the filtered rule sets is depicted in Figure 3. Although there is some overlap between the different sets of filtered rules, the results appear to produce different subsets of the rules. We then deepen this observation, with the idea of checking whether we have different rules or similar/equivalent rules in the subsets that seem not to intersect. The result of the investigation is that the apparent intersections of small sets are misleading because many of the rules are equivalent in terms of *support* and *confidence*, while not having exactly the same expression.

Even though comparing association rules is a classical topic [49], determining the best set of rules that can explain the targeted faults is still a challenge. In this work, we evaluated the similarity between rules using the Conditional Market-basket Probability Distance (CMPD) from [47]. The similarity between two rules $r_i : X_i \rightarrow Y$ and $r_j : X_j \rightarrow Y$ is defined by the probability that two rules are valid for the same basket and is calculated as follows:

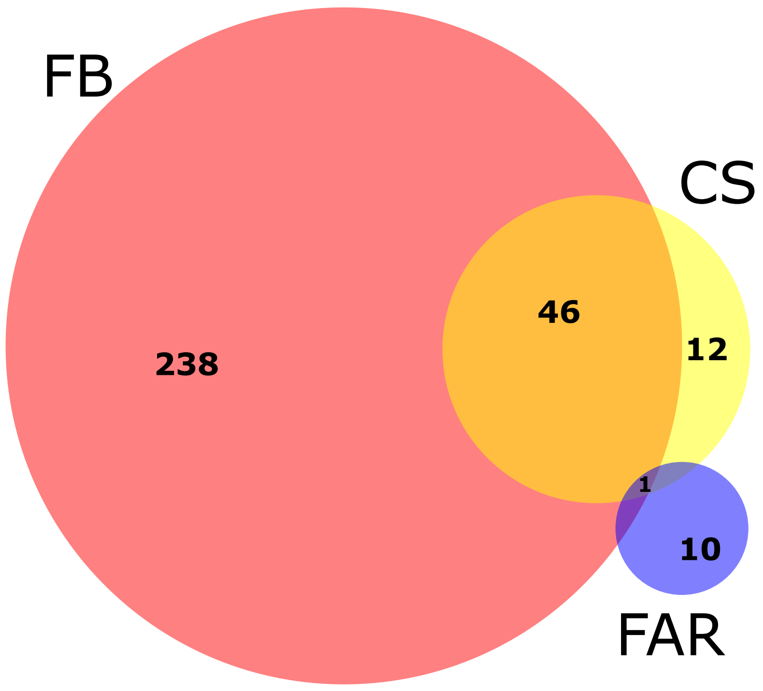
$$s(r_i, r_j) = \frac{|\sigma\left(r_{ij}\right)|}{|\sigma\left(r_i\right)| + |\sigma\left(r_j\right)| - |\sigma\left(r_{ij}\right)|}, \tag{7}$$

where $r_{ij} = (X_i \cup X_j) \rightarrow Y$. Rules without a common basket have a similarity of 0 and rules with an identical set of baskets have a similarity of 1. This similarity measure is expensive because it involves considering all the original data, with millions of records, to calculate the similarity between each pair of rules. Therefore, it is only suitable for small scale diagnosis, not to be used directly for rule filtering.

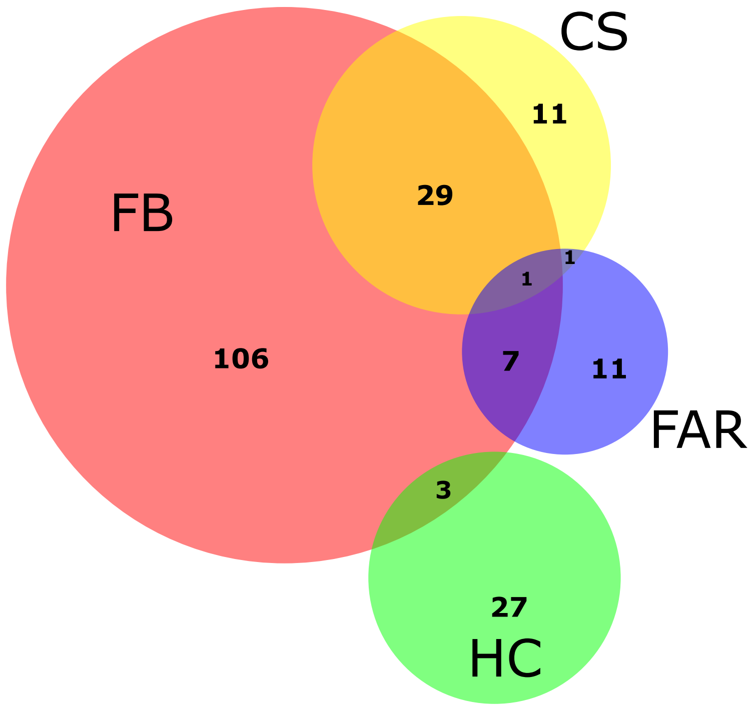
The similarity matrix of the rules produced by CS filter, as measured by (7), is shown in Figure 4. Therein, we observed multiple groups, in which the rules are highly similar. Comparing pairwise similar rules, there are two noticeable ways that can explain this:

(1) Highly probability dependent items (often appears together in the data), for example, the pairs
Handover Attempts = Failed; Handover Successful = Failed
or *end_cell=A; start_cell=A*
are the only different items among the rules, in these cases, these rules can be merged. However, there is a delicate compromise to be found between merging rules, and keeping the *support/confidence* values as high as possible.

(2) Two rules contain very different items but are applied on the same set of baskets. This is the most difficult case to solve. Because looking at the dataset is expensive, it is not easy to avoid this issue.



(a) 3G dataset



(b) 4G dataset

Figure 3: Intersections of the set of filtered rules. Two rules are considered similar if they have the same antecedent and consequent.

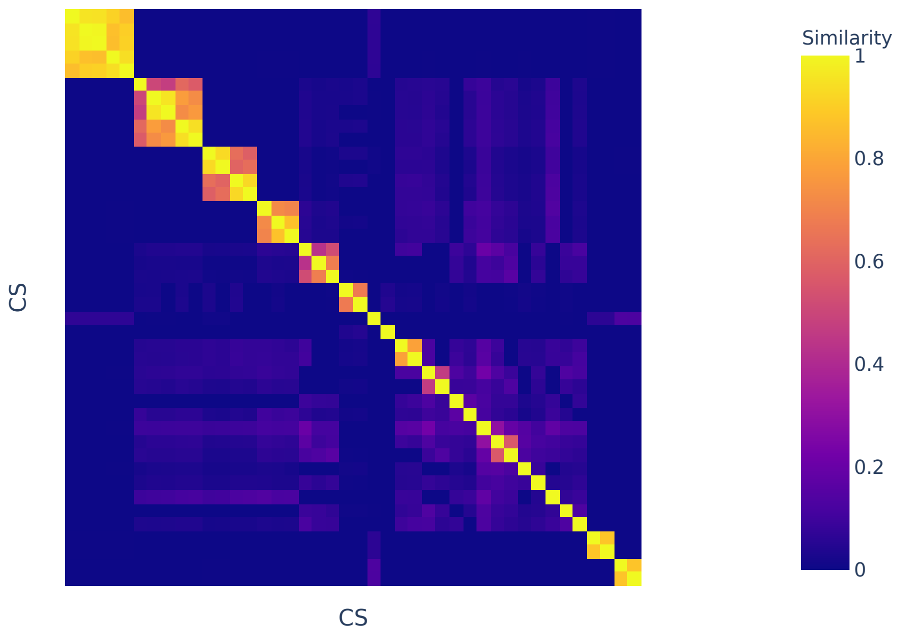
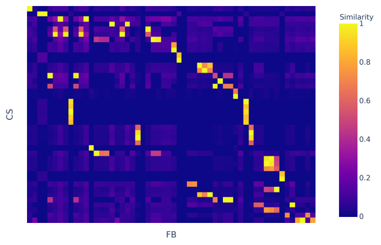


Figure 4: Similarity matrix the rules produced by CS filter in 4G dataset. The rules are reordered to form clusters of high similarity rules. The redundant areas are depicted with brighter colors.

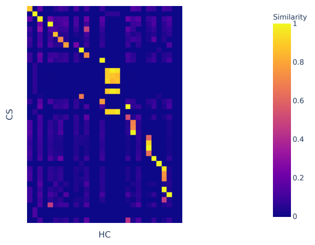
We can also validate the rule similarities between two sets of rules produced by two different filtering algorithms in 4G dataset. Figures 5(a), 5(b), and 5(c) depict the similarity of rules from the CS filter to those from the FB, HC, FAR filters, respectively. For the FB filter, we only consider the first 100 rules according to their *support*, since the others have negligible *support*.

Note that these similarity matrices allow an evaluation of the set of rules which do not seem to intersect, as well as an identification of redundant rules, but do not validate the quality of the filters.

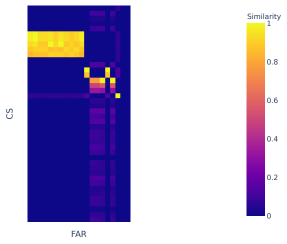
Observing Figure 5(a), we can conclude that the rules from CS and FB are quite similar since every rule in CS has at least one highly similar rule from FB and vice versa. We also observe an acceptable redundancy from the rules in CS as well as FB. If there are more than 1 high similarity element from the matrix in the same row, it means these according to rules from FB can be replaced by the according to rule from CS and vice versa.



(a) CS vs. FB filters



(b) CS vs. HC filters



(c) CS vs. FAR filters

Figure 5: Similarity matrix when comparing CS to other filters that work with rule-level. Rules are reordered using reverse Cuthill-Mckee algorithm [10] with 0.5 threshold binarize to move the large similarity pair closer to the main diagonal.

Figure 5(c) highlights a similar significant group of rules generated by CS and FAR filters. The redundant rule set in the CS group can be explained by the self-similarity observed in Figure 4 while the redundancy of some FAR rules is caused by the selection of rules with the highest *support*/*confidence*. There is also a key difference between CS and FAR, the two rules with respective antecedents $\{$*Start cell UID=9678*, *Handover Successful=False*, *Connection Type=Data*, *Establishment cause=Cat. 1*$\}$ and $\{$*Start cell UID=9678*, *Handover Attempts=False*, *Connection Type=Data*, *Handover Successful=False*$\}$ do not appear in CS filter's result. The reason is that these rules are replaced by the rule with the antecedent $\{$*User Equipment Category=B*, *Handover Successful=Failed*, *Handover Attempts=Failed*, *Connection Type=Data*, *Carrier Aggregation Flag Set=True*, *UL SINR*=$[-5.0, -4.0)\}$ in CS. This case shown the limitation of filter based on the items set relation only. Once again, this problem can be overcome by observing the dataset every time making a filter decision, which is costly. The same problem also observed when comparing CS to HC in Figure 5(b) where the rules with *Start cell UID=9678* is ignored.

Figures 5(a), 5(b) and 5(c) also show that the redundancy from FB, HC and FAR algorithms is higher comparing to CS filter.

From this analysis, we concluded that while CS filter still produces redundant rules or ignores some good rules, it is better than other filters that apply with the rule level. When comparing to filter by using dataset directly, CS also retains a good performance, while cost a negligible computation resource. In the next subsection, we will discuss the effect of pruning and merging thresholds on the quality and quantity of rule sets in CS filter.

3.4.7 Sensitivity of the parameters in the filtering algorithm

In the CS filtering, the condition for removing and merging the rules in the final rule set are governed by the *support* and *confidence* parameters. For the pruning phase, increasing ε-supp increases the removal of parent rules (in terms of set) while increasing ε-conf allows keeping the parent rules with high *confidence* and replacing the child rule in the final result. Note that since every parent rule has lower or equal *support*, compared to the child rule, therefore, the removing condition is satisfied when the *support* condition is satisfied or the *confidence* condition is not. In the merging phase, the ε-supp makes sure the two rules are highly dependent, which means the rules are applied in two highly similar baskets. The ε-conf governs the quality of merged rules: even if two rules are highly dependent, merging them can produce a bad rule when the *confidence* difference between them is noticeable. For example, by increasing the ε-conf from 0.05 to 0.1, we get a less meaningful rule set with very negligible redundancy, which is shown in Figure 6.

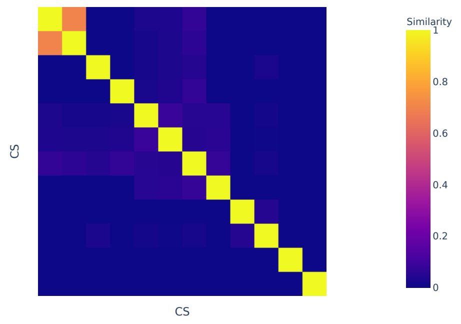


Figure 6: Similarity matrix of CS filter, ε-supp$= 5e-2$, ε-conf $= 1e-1$.

3.5 Conclusions

This work proposes a novel fault analysis strategy for information systems that uses a target-specific FPM followed by a filtering and merging procedure. The proposed framework was evaluated on two real-world telecommunication use-cases with incremental difficulty from a 3G to a 4G network. The use of target specificity significantly reduces runtime (by a factor of 2-3 with *support* threshold 10^{-4}), opening the door to a wide range of applications in big data environments. The proposed Cover Set (CS) filter then reduces the number of rules, which increases the interpretability of the results and contributes to shorten the time to resolution of network faults.

The validation by telecommunications experts has confirmed the improved interpretability and scalability of the newly proposed framework. In both datasets, the analysis of dropped calls indicated that they were mostly due to failed attempts to reconnect during $Handover$, a common issue when experiencing signal quality degradation. In addition, ranking by *impact* metric emphasizes the prevalence of certain rules across the network and allows an operator to deepen the investigation of faults. In particular, this strategy highlights the most problematic cells in order of their contribution to dropped calls for the 3G dataset, giving the optimal sequence for further investigation. When inspecting the rules for the 4G dataset, the presence of rules combining unsuccessful handover with very low call-quality-indicator and up-link interference confirms the network conditions generally leading to dropped calls.

While this strategy offers a general approach that is valuable in a data-agnostic scenarios, the addition of automatic feature selection and rules post-processing analysis could increase the interest for an integration of the proposed framework into a cellular network automatic fault analysis system. Such a system would require clear and actionable insights from the established rules.

In the comparison with other filtering algorithms, we provided generic tools to provide comparison of filtered rules, even when these rules only share the same role, but expressed differently.

Overall, the performance and interpretability improvements provided by our filtering framework allow the processing of larger datasets with less effort from human operators. Consequently, a more specific analysis of the root causes could be considered, for example at the level of cell towers or specific regions.

Chapter 4

Root Cause Analysis with Causal Inference and Embeddings

This chapter will first provide an overview of causal learning methods and their application to Root Cause Analysis. Next, Section 4.1 will provide the related materials for the proposed methodology, the detail of which can be found in Section 4. The experiments in Section 4.4 will evaluate the proposed methodology with synthetic and real-world datasets. The conclusions and future improvements are provided in Section 4.5.

4.1 Introduction

In this thesis, we are working on the RCA task for cross-sectional data, assuming that the treatment variables are independent and the outcome is the result of the treatment variables' interactions. Concretely, given the set of D treatment variables (i.e., pairs of feature and value) $x_1, \ldots, x_D$ and the outcome y, the goal is to determine whether y changes if x_i is modified for every $i \in \{1, \ldots, D\}$. Straightforwardly, this problem can be considered as a causal discovery problem. There are two main classical approaches to answer this causal discovery question: *Constraint-based* (CB) and *Score-based* (SB) algorithms.

CB algorithms reduce the number of candidates using the conditional independence assumption [46]. The most well-known constraint-based algorithm is *Peter-Clark* (PC) algorithm, proposed by Sprites et al. [46]. PC algorithm consists of two steps: 1) learning an undirected (*skeleton*) graph from data and 2) assigning the directions of the edges. Variants of the PC algorithm such as PC-stable, parallel-PC, Copula-PC [36] were introduced to improve the performance of PC with big data and mix-type data. On the other hand, SB algorithms relax the conditional independence using a score-based learner, which grades the score of the candidate causal graphs [36]. The locally optimal solutions can be found using heuristics approaches such as GES [9], or Fast GES [37]. Other approaches using causal graphs can be found in [16]. Generally, the causal discovery approach requires immense computational resources and can provide a more accurate answer.

With the assumption that all covariates are not causally related, RCA can also be considered a causal effect learning task where the covariates are all considered as treatments. In causal inference, matching methods [48] aim to ensure that the differences between treated and control groups (on all background covariates: observed and unobserved) are minimum, for instance, the differences between a group of treated and a group of untreated patients can be attributed to the effect of treatment. Research studies using matching methods have been scattered across disciplines such as, e.g., statistics [40], sociology [30], or economics [19]. Generally, matching methods aim to estimate the causal effects of treatments and can be divided into Nearest Neighbour Matching, Subclassification and Weighting. Among these methods, Nearest Neighbour Matching is the most common [48].

4.2 Related materials

In this section, we will describe the Dimensionality reduction and Clustering techniques that are used in the experiments. As discussed above, the proposed method requires that the original data is transformed into a lower-dimensional space and divided into different groups. Other techniques for data embedding and clustering can be found in [1,53].

4.2.1 Dimensionality reduction

In the context of this thesis, we decided to investigate the Principal Component Analysis (PCA) in conjunction with other techniques, as PCA is one of the most computationally economical embedding method. We next briefly recall the key features of PCA.

Principal component analysis

Principal Component Analysis (PCA) is a linear dimensionality reduction method proposed by Karl Pearson [34]. The principal components correspond to the directions of the axes where there is the most variance (most information): they are characterized by the eigenvectors of the data covariance matrix. Eigenvalues are then the coefficients attached to eigenvectors, which give the amount of variance carried in each principal component. We next define the feature vector as a matrix that has as columns the eigenvectors of the components with greater significance. The low-dimensional representation of the data is the result of a projection that retain the data's variance as much as possible. More precisely, PCA solves the following eigenvalue problem:

$$cov(\mathbf{X})\mathbf{M} = \lambda \mathbf{M} \tag{8}$$

where $\mathbf{M}$ is the linear mapping, $cov(\mathbf{X})$ is the sample covariance matrix of the data $\mathbf{X}$ and λ is the eigenvalue vector. The eigenvalue problem is solved for the selected d largest eigenvalues (components). For dimensional reduction tasks, the value of d is often much smaller than the dimension of the original data. The value of d is often chosen as a trade-off between the information losses and the compression effectiveness. The lower-dimensional data is computed by a mapping into the linear basis $\mathbf{M}$:

$$\mathbf{X}' = \mathbf{X}\mathbf{M} \tag{9}$$

There are multiple methods to solve the eigenvalue problem. The most common ones are the Singular value decomposition (SVD) when the data size is small or a Randomized method [29] for large data sizes.

t-SNE

t-Distributed Stochastic Neighbour Embedding (t-SNE) is a statistical non-linear dimensional reduction method developed by Van der Maaten and Hinton [52] and widely used for data visualization in two- or three-dimensions space. There are two stages in t-SNE: 1) construct a probability distribution over pairs of datapoints in original space and 2) reconstruct these distances using Student t-distribution in the embedded space.

For a set of N high-dimensional objects $x_1, \ldots, x_N$, the similarity of x_i and x_j is computed as the symmetrized probability $p_{ij} : p_{ij} = (p_{j|i} + p_{i|j})/2N$, where

$$p_{j|i} = \frac{exp\,(-||x_i - x_j||^2/2\sigma_i^2])}{\sum_{k \neq i} exp\,(-||x_i - x_k||^2/2\sigma_i^2)} \tag{10}$$

where $p_{i|i} = 0$, σ_i is the Gaussian kernel's bandwidth, and $\sum_j p_{j|i} = 1$ The lower-dimensional similarities, denoted by q_{ij}, are then constructed based on the Student t-distribution (one-degree of freedom):

$$q_{ij} = \frac{(1 + ||y_i - y_j||^2)^{-1}}{\sum_k \sum_{l \neq k} (1 + ||y_k - y_l||^2)^{-1}}. \tag{11}$$

The location of the corresponding projection y_i of x_i is determined by minimizing the Kullback-Leibler [24] divergences using gradient descent:

$$KL(P||Q) = \sum_{i \neq j} p_{ij} \log \frac{p_{ij}}{q_{ij}}. \tag{12}$$

4.2.2 Clustering

This subsection will provide an overview of the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) clustering method. In this work, we chose the density-based clustering method because of its ability to adapt to complex data distribution.

DBSCAN

While k-means is the most popular clustering algorithm, it also assumes spherical data clusters, which is not the case in many real-world datasets with complex structures. Furthermore, clustering of large datasets also requires the ability to remove outliers and noise, which k-means does not provide.

These issues can be solved using the density-based clustering paradigm. Density-based clustering is a non-parametric method with no assumption about the number of clusters and data distribution. In density-based clustering, clusters are viewed as regions of high data point density (i.e., all points can be reached with a small distance), separated by low data point density regions and the sparest areas treated as noise.

One of the most well-known density-based clustering methods is Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [12]. In DBSCAN, the density is estimated using the number of points in a fixed-radius neighbourhood, and two points are connected if they are in each other's neighbourhood. Concretely, given a radius threshold ε and the minimum number of neighbour $MinPts$, a point p is called core point if there are at least $MinPts$ points (including p) within the distance ε of p. A point q is considered *directly density-reachable* from the core point p if q is in that neighbourhood of p (asymmetric relationship). A point q is called *density-reachable* from p if q belong to the transitive closure of directly density-reachable points of p, i.e. there are a list of point

$p_1, \ldots, p_n$ such that p_{i+1} is directly density-reachable from p_i and $p_1 = p$, $p_n = q$. Two points are called *density-connected* if there is a third point from which both are density-reachable. A cluster is maximal set of density-connected points. The points that not belong to any cluster are called *outlier* or *noise points*.

4.3 Nearest Neighbour Matching in intermediate subspace

This section will provide the detail of the proposed approach using NNM in an intermediate lower-dimensional subspace of the covariates.

One of the most used methods for randomized experiments for estimating causal effects is neighbour matching [48]. The well-matched samples from treated and control groups are assigned to randomized experiments and help reduce selection bias of the covariates. In this case, with observational data, the matching methods are employed to investigate the contributions of one or more feature values to the outcomes.

In the Root Cause Analysis context, the causal relationship between the features' value and the outcome can come from multiple causal structures, corresponding to multiple failure events in the same dataset. Furthermore, the goal of causal inference for RCA is to provide the specific values of the variables that cause the problems. For example, in a cellular network, failed connections can be caused by a bad cell tower, hand-over problem or blocked signal. Therefore, the first step is to split data into groups, where each group only represents one type of causal structure. After that, for each group, the effectiveness of treatments on the outcome, i.e., the effect of features' value on the final status, can be estimated by comparing the normal and abnormal records by the matching method. In the view of the matching framework, the normal records and abnormal records play the roles of the factual and counterfactual outcome, respectively.

Since most of the data is high dimensional, the distance metric should be susceptible to irrelevant information due to the problem with high dimensional data. Another problem is the effect of group size: matching results can be driven mainly by a small part of the groups, leading to bias. Therefore, there will be a need for some restrictions on the number of considered neighbours. This restriction also helps to navigate when there are poor matches (i.e., the dissimilarity between two groups is significant).

The proposed framework consists of the following steps: 1) projecting original covariates to an intermediate lower-dimensionality subspace; 2) dividing the dataset into small groups of similar covariates; 3) in each divided group, find counterfactual examples of each abnormal record using NNM and gather the critical difference treatments and 4) summary to critical difference treatments of the whole dataset.

For the above mentioned reasons, we proposed a method that consists of the following steps:

1. Project the raw feature vector into a lower-dimensional space (embedding). This process applies to all data points. Due to the nature of high-dimensional and high volume data, we need to trade-off between the quality and complexity of the embedding technique. In this work, we apply PCA first to reduce the data's dimensional to an acceptable size before applying t-SNE. Projecting data into a lower-dimensional space increases the significance of the distance metric, in this case Euclidean, thus increasing the precision of neighbour clustering/selection.

2. Divide failure instances into clusters using a clustering method on the embedded space using DBSCAN.

3. For each cluster:

   (a) Find k *normal* neighbors for each failure instance in the embedded space.

   (b) Calculate the dissimilarity in the original space between the failure instance X and its non-failure neighbours N_i: $DS(X, N_i) = [|x_1 - N_{i,1}|, |x_2 - N_{i,2}|, \ldots, |x_d - N_{i,d}|]$. Each component in the dissimilarity vector corresponds to a feature x_i in the original space. By sorting $DS(X, N_i)$ in descending order, we can get a ranking list of top-ranked features that are likely associated with the cause of the failure.

   (c) The ranking can be aggregated to get the most important features for the current failure instance, i.e., the top features that contribute to the difference between a failure data point and its normal neighbours. Let us call these features the selected features cluster.

4. Report all the selected feature cluster.

The formal definition is presented in the Algorithm 3. The missing value of the original data (numerical columns) will be replaced by the average values. After that, the category columns are one-hot encoded, and the numerical columns are normalized. The data will be transformed into an intermediate subspace dimension using one or more embedding methods, and the detail can be found in the Experiments section 3.4. After the above-mentioned steps, the data will be fetched to the Algorithm 3 as the input dataset X of D

dimensional. The data will be transformed into an intermediate subspace of d dimensional using one or more embedding methods, and the detail can be found in the Experiments section 3.4. A clustering function g on the dataset E will also be required for divided the dataset into smaller groups with highly related samples. The domain of c is d, and the co-domain is called $S = \{s_1, s_2, \dots, s_t\}$.

Algorithm 3 k:1 nearest neighbour matching with embedding

```
Input: The $D-$dimensional dataset $X = \{x_1, x_2, \dots, x_m\}$ with list of normal/abnormal
  label $Y = \{y_1, y_2, \dots, y_m\}$, $y_i \in 0, 1$; a clustering function $g()$ that assigns a data point
  to its corresponding cluster in the set of $S$ clusters; a pre-defined threshold $p$, $0 < p < 1$;
Output: Clusters of variables $G$
  Step 1: $E = \{e_1, e_2, \dots, e_m\}$ is the $d-$dimensional representation of the original dataset
  $X, d < D$.                                                  ▷ Embedding step
  Step 2: Set of clusters $S = \{s_1, s_2, \dots, s_t\}$, where $S$ is the co-domain of the clustering
  function $c()$.                                              ▷ Clustering step
  $G \leftarrow \emptyset$
  for $s \in S$ do
    $g \leftarrow$ zero vector of $D$-dimensional
    for $x \in X, y_i = 1, c(x) = s$ do
      Find set of $k$ nearest normal neighbours of $x$
      for $t \in k$ do
        $d_t \leftarrow (|x_1 - t_1|, |x_2 - t_2|, \dots, |x_D - t_D|)$
        $g \leftarrow g + d_t$
      end for
    end for
    $F \leftarrow argsort(g)$    ▷ sort the variables based on their contribution to the difference
    between normal and abnormal records
    $G \leftarrow G \cup \{F_1, \dots, F_t\}, t \leftarrow \inf\{j \leq D | \frac{\sum_{i=1}^{j} N_i}{||N||_{L^1}|} \geq p\}$    ▷ select the top
    $t$ contributed variables such that the sum of the top $t$ variables' contribution over all
    variables' smaller than $p$
  end for
```

4.4 Experiments

This section demonstrates and evaluates the proposed method with multiple datasets. The first two experiments were performed using two synthetic datasets: without causal relationship (randomly generated) and with linear causal relationship (randomly generated with fault-associated features). We also experiment with the 4G Telecommunication dataset described in Chapter 3.4.

4.4.1 Validation with synthetic datasets

We evaluated our approach using two synthetic datasets generated as follows:

Purely Random data $\mathcal{S}_1$: 10 numerical features drawn uniformly from $[0, 1]$; 10 categorical features drawn uniformly from $\{0, \ldots, 10\}$. The failure ratio is set to 0.2, and failures status (abnormal) is assigned randomly to the samples. The rest are deemed normal. Since the abnormal are assigned randomly, the dataset contains no causal relationship.

Random data with weak signal $\mathcal{S}_2$: 10 numerical columns drawn uniformly in [0,1]; 10 categorical columns drawn uniformly from $\{0, \ldots, 10\}$; 3 columns drawn from $\mathrm{Lognormal}(0, 0.5)$ with the chance associated with fault 1.0, 0.8 and 0.5 respectively, namely $prob\_1$, $prob\_2$ and $prob\_3$; failure rate 0.2. In this data set, there are significant associations between the features values of the last 3 columns with the failures.

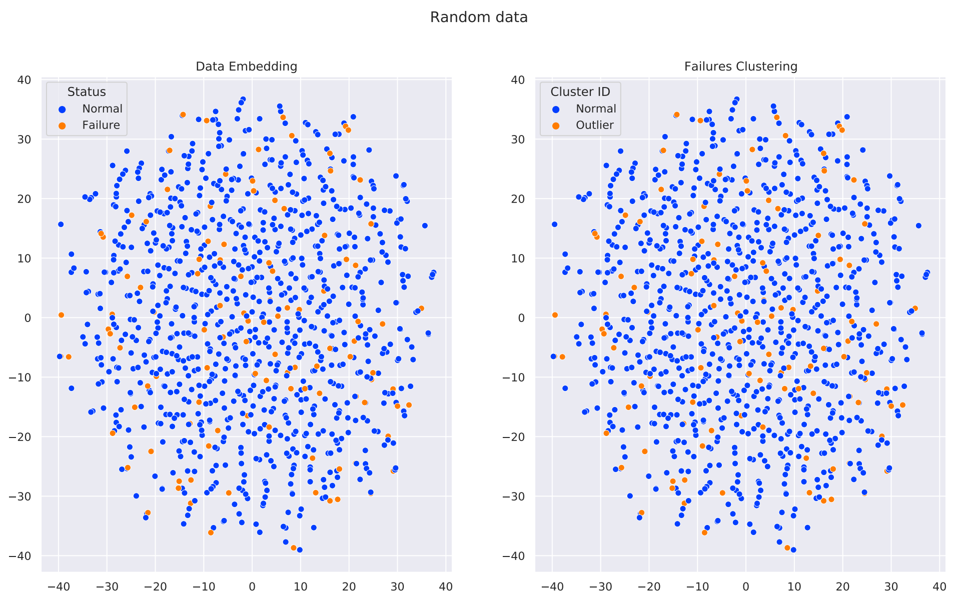


Figure 7: Normal and Failure samples in the embedding space. The data is generated from i.i.d. random uniform distributions.

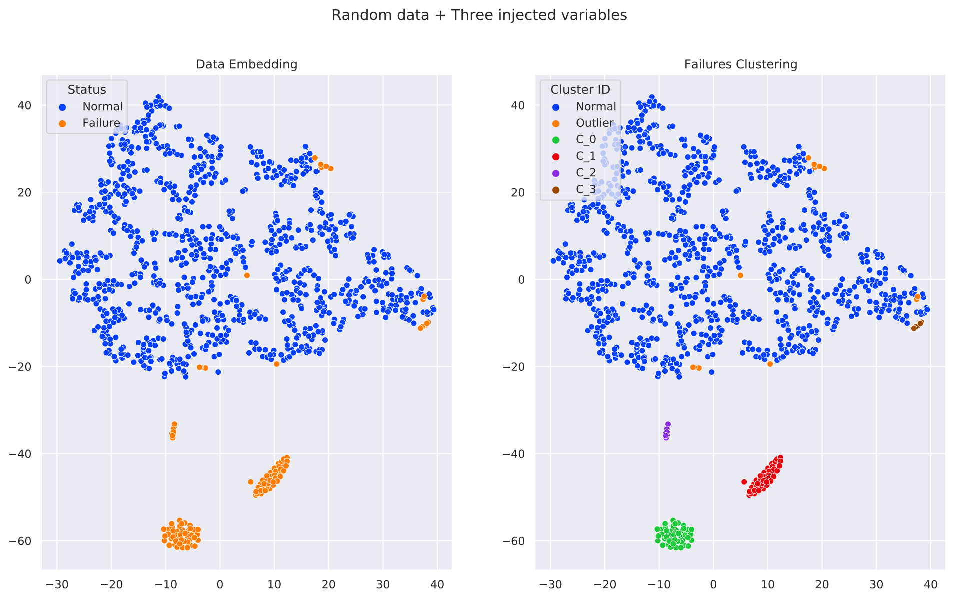


Figure 8: Normal and Failure samples in the embedding space of $\mathcal{S}_2$. The data is generated from i.i.d. random uniform distributions except for 3 columns, which associated with Failure samples. There are 4 clusters for Failure samples.

The embedded representation of dataset $\mathcal{S}_1$ (purely random) is shown in Figure 7. When performing $k : 1$ nearest neighbour matching using Algorithm 3. The data set is first embedded in a 40-dimensional representation using PCA and then a 2-dimensional representation with t-SNE. We used DBSCAN with 5 minimum neighbours and 2 minEps distance in this experiment. As observed in Figure 7, zero clustered sample can be found, and there is no output of causal factors. Hence, the proposed $k : 1$ nearest neighbour matching method provides the anticipated answer: there is no root-cause factor for this data root cause from observed variables in this dataset.

The embedded version of dataset $\mathcal{S}_2$ (random with weak signal) is shown in Figure 8. After performing nearest neighbour matching, four abnormal clusters can be observed in Figure 8. The figure clearly shows the contribution of fault-associated features that can be used to create and separate the group of Normal and Abnormal data points. Therefore, even when there are few weak fault associations with the original features, the proposed algorithm can still provide the fault associate feature groups. The found causal factors are represent in the Table 7.

Results obtained on the two synthetic datasets show that the proposed method works correctly in noisy environments where the input features have weak or zero association with the outcome. However, this result still can not be interpreted as causation.

Table 7: Fault associated values extracted from Random noise with three fault-associated columns data set

Cluster	Group	Number of failures in cluster

0	prob_3=$(-10.0, -1.02]$,
	prob_2=$(-10.0, -2.25]$,
	prob_1=$(-10.0, 0.93]$
		44

1	prob_2=$(-10.0, -2.25]$,
	prob_3=$(7.11, 10.01]$,
	prob_1=$(-10.0, 0.93]$
		39

2	prob_3=$(-10.0, -1.02]$,
	prob_2=$(9.70, 12.49]$,
	prob_1=$(-10.0, 0.93]$
		8

3	prob_2=$(-10.0, -2.25]$,
	prob_1=$(-10.0, 0.95]$,
	prob_3=$(28.5, 81.07]$
		13

4.4.2 Validation on real-world datasets

In this section, we will experiment with the proposed method on 30% of the 4G dataset in Chapter 3.4. For the sake of comparison, the data is reduced to two dimensions using the PCA algorithm. The output are presented by Figure 9 and 10. From these figures, we can conclude that the abnormal and normal outcomes are barely distinguishable; hence the quality of the matching groups will be poorly if only using PCA without t-SNE. This latter issue demonstrates the side effect of projecting high-dimensional data to a lower-dimensional space using a linear method.

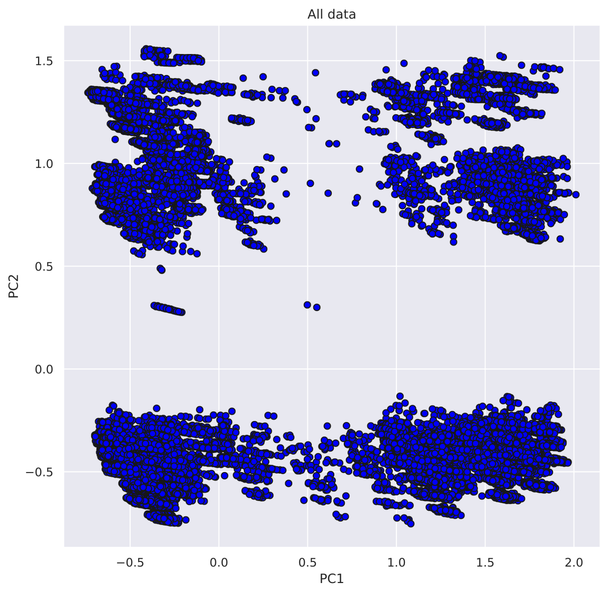


Figure 9: 2-component PCA project of all data records.

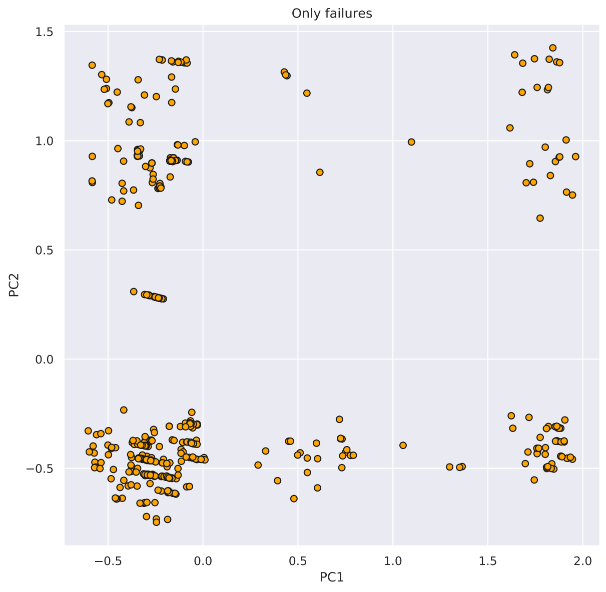


Figure 10: 2-component PCA project of non-normal data records.

Figure 11 shows the projection of the cluster from 40-component PCA and 2-component

t-SNE using DBSCAN ($eps = 2, min\_samples = 5$) clustering. We can observe clearly-separated clusters of abnormal records in the embedded space. The group of the clustered samples are distinguished using different colour coding.

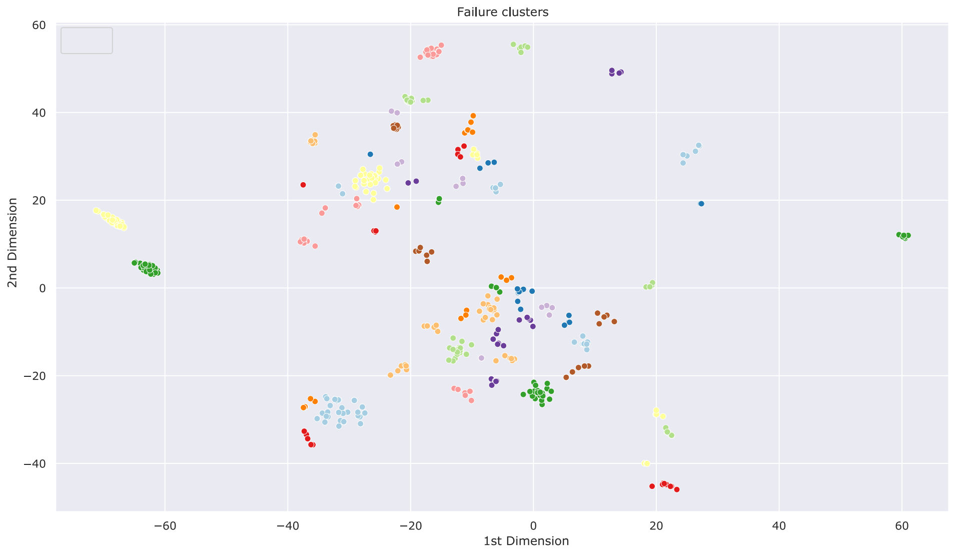


Figure 11: The embedding projection of non-normal outcome extracted from k:1 nearest neighbour matching using PCA(40 components) + t-SNE (2 dimensions) embedding with DBSCAN cluster.

The output of Algorithm 3 is shown in Table 8. The most popular causal factors group corresponds to the total 7% of the data. The outputs should be interpreted as "the observed variables together or alone will likely cause a problem". However, since there is no clear objective function to optimize, the Algorithm 3 results are challenging to track. Therefore, we re-employed the comparison in Chapter 3.4.6 to demonstrate the similarity between FPM and the Matching method. The result in Figure 12 shows that only a part of matching's result is compatible with FPM's result. We observed that the results from the two methods are mostly different. There are two main contributors to that problem. Firstly, the matching method's result is the result of voting, which implies that the variables may not all coincide. Secondly, the proposed method's indirect objective and the information losses when applying embedding to the high dimensional data have lots of one-hot encoded features that also contribute to the difference.

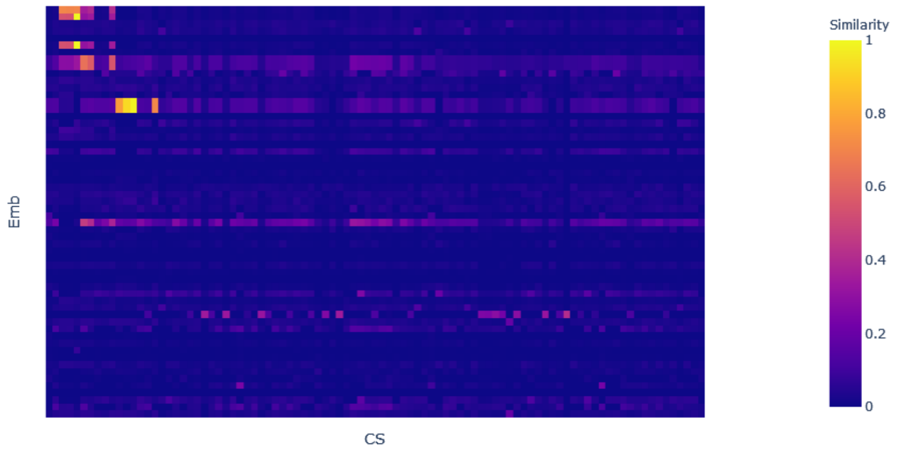


Figure 12: Comparison between matching method and FPM on 4G dataset.

Table 8: The extracted top fault-contribute features from the Random noise with fault-associated data

ID	Group	N
2	User Equipment Category=3.0, Establishment cause=LTE Mobile signalling, UL SINR=(13.97, 15.94], Avg. Channel Quality=(9.17, 10.57], Delay=(234.0, 390.0], Start cell UID=96, Carrier Aggregation Flag Set=True	198
12	First location coverage=(-107.36, -103.5]	55
0	UL SINR=(13.97, 15.94], Coverage=(-109.5, -105.5], End cell Quality=(-12.13, -10.25], Avg. Channel Quality=(9.17, 10.57], Start cell Quality=(-11.25, -10.25], First location coverage=(-112.5, -107.36], Delay=(468.0, 624.0], Last location coverage=(-112.5, -107.39], First location signal quality=(-11.84, -10.34], Last location signal quality=(-11.86, -10.44], Download(B)=(32247.0, 304845.49], Uploaded(B)=(6324.0, 22674.4], Start cell UID=70	39
26	UL SINR=(10.75, 13.97], Avg. Channel Quality=(9.17, 10.57], End cell UID=54, Start cell UID=106, End cell Quality=(-12.13, -10.25], Start cell Quality=(-11.25, -10.25], Coverage=(-105.5, -101.5], Delay=(234.0, 390.0], Last location signal quality=(-11.86, -10.44], First location signal quality=(-13.26, -11.84], First location coverage=(-107.36, -103.5], Last location coverage=(-107.39, -103.5], Uploaded(B)=(6324.0, 22674.4], Download(B)=(32247.0, 304845.49]	35
18	UL SINR=(10.75, 13.97], Avg. Channel Quality=(8.03, 9.17], End cell Quality=(-12.13, -10.25], Start cell Quality=(-11.46, -11.25], Start cell UID=80, Coverage=(-113.5, -109.6], End cell UID=80, First location signal quality=(-11.84, -10.34], Last location signal quality=(-11.86, -10.44], First location coverage=(-112.5, -107.36], Delay=(1562.0, 14920.0], Last location coverage=(-112.5, -107.39], Download(B)=(32247.0, 304845.49], Uploaded(B)=(2483.8, 6324.0]	35
4	End cell UID=400, UL SINR=(10.75, 13.97], Start cell UID=400, User Equipment Category=11, Avg. Channel Quality=(8.03, 9.17], Coverage=(-109.5, -105.5], End cell Quality=(-10.25, -8.75], Delay=(468.0, 624.0], Start cell Quality=(-10.25, -8.75], First location coverage=(-107.36, -103.5], Last location coverage=(-107.39, -103.5], Last location signal quality=(-11.86, -10.44], First location signal quality=(-11.84, -10.34], Uploaded(B)=(6324.0, 22674.4], Download(B)=(32247.0, 304845.49]	31

4.5 Conclusion

This chapter proposes a concrete causal-based framework approach for fault analysis in information systems with cross-sectional data records. The proposed framework was evaluated on two synthetic datasets and one real-world telecommunication use case of 4G network. The framework proves that it can provide better approaches with numerical data types compared to the proposed method in the previous chapter.

The result of experiments with synthetic data proves the robustness of the proposed algorithm. The result from the 4G dataset are highly interpretable and are compatible with the previous result. When analyzing the result from 4G dataset, we found a high degree of dissimilarity with the result from the previous chapter. Mainly because of different natures of forming and interpreting the result. Others come from the separating nature of components in the method. Computational-wise, the proposed method is significantly faster than the association rule methods while still providing a high degree of transparency.

Despite many advantages, the proposed method still retains three main problems. Firstly, the components in the method are formed using different criteria, which can lead to a bad quality result. Secondly, there are no clear evaluation criteria for the final result due to the nature of the data, which can be critical to evaluating future models. Lastly, the assumption of causal independence between components in the network and lack of propagating method between timestamps can also lead to a low-quality result.

Chapter 5

Conclusion and Future Work

5.1 Conclusions

This research aimed to increase the efficiency and efficacy of Root Cause Analysis for information systems, especially telecommunication networks, in the context of big data requirements.

Chapter 2 provided details about Root Cause Analysis and a concise review of possible approaches. It also provided a detailed analysis for choosing the proposed methods in the later chapters.

In Chapter 3, we proposed a novel fault analysis strategy for information systems that uses a target-specific Frequent Pattern Mining approach followed by a filtering and merging procedure. The proposed framework was evaluated on two real-world telecommunication use-cases with incremental difficulty from a 3G to a 4G cellular network. The use of target specificity significantly reduces runtime (by a factor of 2-3 with *support* threshold 10^{-4}), opening the door to a wide range of applications in big data environments. The proposed Cover Set (CS) filter then reduces the number of rules, which increases the interpretability of the results and helps reduce the time to resolution of faults. In comparison with other filtering algorithms, we provided generic tools to provide a comparison of filtered rules, even when these rules only share the same role but are expressed differently. The validation by telecommunications experts has confirmed the improved interpretability and scalability of the newly proposed framework. While this strategy offers a general approach that is valuable in a data-agnostic scenario, the addition of automatic feature selection and rules post-processing analysis could significantly improve the performance of the proposed framework in cellular network automatic fault analysis systems. Such a system would require clear and actionable insights from the established rules.

In Chapter 4, we demonstrated a different method for solving the Root Cause Analysis problem from the causality perspective. The proposed method was evaluated with synthetic datasets and demonstrated robust and accurate in noisy environments. When qualitatively evaluated with the 4G dataset, the proposed method demonstrated to be robust with numerical data, provided more detailed results and promised to be faster than Association rule approaches. However, there are still more improvements that should be considered before deploying it in real-world systems. The automatic evaluations should be investigated in more detail, which can help experts reduce the investigation time and improve result accuracy. The components of the proposed method are currently optimized independently, which also contributes to the difficulty of finding the optimal solutions. The results should also be interpreted carefully since the method would likely suffer from the Rashomon effect (i.e. multiple contradictor interpretations for an event).

Overall, the proposed methods in this research provided a solution to the research question of improving the performance of Root Cause Analysis in telecommunication networks and information systems in general by reducing the search space size. While the proposed framework based on Association rules in Chapter 3 is a good prototype, the method based in Chapter 4 still needs to be improved. The methods and their results are highly interpretable to system operators, which helps improve the accuracy and reduce the models' troubleshooting time.

5.2 Future works

Root Cause Analysis is a vast topic that involves many techniques and approaches, ranging from Finite-state machine and First-order logic to Unsupervised learning and Reasoning. After all, Root Cause Analysis's ultimate goal is to become fully automatic, which means there is no need for human labour in the deduction process. However, it will require many significant breakthroughs in Machine Learning, specifically Reasoning and Causality Discovery, before operators voluntarily withdraw the human from the analysis process. In the meantime, multiple future works can be addressed to help improve the Root Cause Analysis framework's efficiency and efficacy:

Automatic identifying and removing background variables. Background variables are the variables that are not related to the problem in the systems. In telecommunication networks, most variables in the log records are background variables and must be manually removed by the experts before fetching the data to the analysis step. The background variables can change depending on the network configuration and the analysis task. An automatic variable selector will facilitate the adaptation of the Root Cause Analysis framework on a broad spectrum of networks and analysis intents.

Fault propagation problems. In real-world systems, a component failure can create a chain of failure reactions, which increases the difficulty of the diagnosis task. In many diagnosis methods, including our proposed methods in Chapter 3 and 4, for the sake of simplicity, the fault propagation is often deemed non-exist, which is often not fully correct. To thoroughly analyze the problem requires processing the data in a time-series manner (non i.i.d. data), which promises to provide a better answer for the fault causal but will require more computing resources.

Analysising non-frequent failures. Root Cause Analysis often couples with the Anomaly Detection module, which often triggers if the error rate is beyond some pre-determined threshold. As the network resilience improves, the failure rate can be significantly low, and some infrequent failures can become chronic problems if not addressed. In this case, the proposed methods in this work can still be applied after removing some of the non-failure data in the total data records. However, this approach requires a carefully designed removal mechanism. Another method is to design the Root Cause Analysis as an infrequent pattern mining problem, which will be better suitable in terms of computing efficiency.

Bibliography

[1] C. C. Aggarwal and C. K. Reddy. Data clustering. *Algorithms and applications. Chapman&Hall/CRC Data mining and Knowledge Discovery series, Londra*, 2014.

[2] R. Agrawal, T. Imieliński, and A. Swami. Mining association rules between sets of items in large databases. In *ACM SIGMOD international conference on Management of data*, SIGMOD, pages 207–216, New York, NY, USA, June 1993. Association for Computing Machinery.

[3] A. Alaeddini and I. Dogan. Using bayesian networks for root cause analysis in statistical process control. *Expert Systems with Applications*, 38(9):11230–11243, 2011.

[4] H. Bouattour, Y. Slimen, M. Mechteri, and H. Biallach. Root cause analysis of noisy neighbors in a virtualized infrastructure. In *IEEE Wireless Communications and Networking Conference (WCNC)*, pages 1–6, May 2020.

[5] S. Brin, R. Motwani, J. Ullman, and S. Tsur. Dynamic itemset counting and implication rules for market basket data. *SIGMOD Rec.*, 26(2):255–264, June 1997.

[6] H. Büning and T. Lettmann. *Propositional logic: deduction and algorithms*, volume 48. Cambridge University Press, Cambridge, United Kingdom, 1999.

[7] M. Carletti, C. Masiero, A. Beghi, and G. A. Susto. Explainable machine learning in industry 4.0: Evaluating feature importance in anomaly detection to enable root cause analysis. In *2019 IEEE international conference on systems, man and cybernetics (SMC)*, pages 21–26. IEEE, 2019.

[8] M. Chen, A. Zheng, J. Lloyd, M. Jordan, and E. Brewer. Failure diagnosis using decision trees. In *International Conference on Autonomic Computing*, pages 36–43, New York, NY, USA, 2004. IEEE.

[9] D. M. Chickering. Optimal structure identification with greedy search. *Journal of machine learning research*, 3(Nov):507–554, 2002.

[10] E. Cuthill and J. McKee. Reducing the bandwidth of sparse symmetric matrices. In *Proceedings of the 1969 24th National Conference*, ACM, page 157–172, New York, NY, USA, 1969. Association for Computing Machinery.

[11] M. Demetgul. Fault diagnosis on production systems with support vector machine and decision trees algorithms. *The International Journal of Advanced Manufacturing Technology*, 67(9-12):2183–2194, 2013.

[12] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231, 1996.

[13] A. Faria. A methodology for filtering association rules. *Revista de Ciências da Computação*, II(2):14 – 25, Jan. 2007.

[14] R. Gardner and D. Harle. Alarm correlation and network fault resolution using the Kohonen self-organising map. In *IEEE Global Telecommunications Conference (GLOBECOM)*, pages 1398–1402, Phoenix, AZ, 1997. IEEE.

[15] J. M. N. Gonzalez, J. A. Jimenez, J. C. D. Lopez, and H. A. Parada G. Root Cause Analysis of Network Failures Using Machine Learning and Summarization Techniques. *IEEE Communications Magazine*, 55(9):126–131, Sept. 2017. Conference Name: IEEE Communications Magazine.

[16] R. Guo, L. Cheng, J. Li, P. R. Hahn, and H. Liu. A survey of learning causality with data: Problems and methods. *ACM Computing Surveys (CSUR)*, 53(4):1–37, 2020.

[17] J. Han and M. Kamber. *Data Mining: Concepts and Techniques*. Elsevier, San Francisco, second edition, 2006.

[18] J. Han, J. Pei, Y. Yin, and R. Mao. Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach. *Data Mining and Knowledge Discovery*, 8(1):53–87, Jan. 2004.

[19] G. W. Imbens. Nonparametric estimation of average treatment effects under exogeneity: A review. *Review of Economics and statistics*, 86(1):4–29, 2004.

[20] ISO/IEC 31010:2019 Risk management — Risk assessment techniques. Standard, International Organization for Standardization and The International Electrotechnical Commission, June 2019.

[21] A. Jorge. Hierarchical Clustering for thematic browsing and summarization of large sets of Association Rules. In *SIAM International Conference on Data Mining*, pages 178–187. SIAM, Lake Buena Vista, Florida, USA, April 2004.

[22] J. R. Josephson and S. G. Josephson. *Abductive inference: Computation, philosophy, technology*. Cambridge University Press, 1996.

[23] B. Kim, R. Khanna, and O. O. Koyejo. Examples are not enough, learn to criticize! criticism for interpretability. *Advances in neural information processing systems*, 29, 2016.

[24] S. Kullback and R. A. Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

[25] J. Leskovec, A. Rajaraman, and J. Ullman. *Mining of Massive Datasets*. Cambridge University Press, USA, 2nd edition, 2014.

[26] F. Lin, K. Muzumdar, N. Laptev, M.-V. Curelea, S. Lee, and S. Sankar. Fast Dimensional Analysis for Root Cause Investigation in a Large-Scale Service Environment. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 4(2):1–23, June 2020.

[27] S. M. Lundberg and S.-I. Lee. A unified approach to interpreting model predictions. *Advances in neural information processing systems*, 30, 2017.

[28] O. Maimon and L. Rokach, editors. *Data mining and knowledge discovery handbook*. Springer, Boston, MA, 2005.

[29] P.-G. Martinsson, V. Rokhlin, and M. Tygert. A randomized algorithm for the decomposition of matrices. *Applied and Computational Harmonic Analysis*, 30(1):47–68, 2011.

[30] S. L. Morgan and D. J. Harding. Matching estimators of causal effects: Prospects and pitfalls in theory and practice. *Sociological methods & research*, 35(1):3–60, 2006.

[31] A. R. Nogueira, A. Pugnana, S. Ruggieri, D. Pedreschi, and J. Gama. Methods and tools for causal discovery and causal inference. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, page e1449, 2022.

[32] P. K. Novak, N. Lavrač, and G. I. Webb. Supervised descriptive rule discovery: A unifying survey of contrast set, emerging pattern and subgroup mining. *J. Mach. Learn. Res.*, 10:377–403, June 2009.

[33] J. Pearl. Causal inference in statistics: An overview. *Statistics surveys*, 3:96–146, 2009.

[34] K. Pearson. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin philosophical magazine and journal of science*, 2(11):559–572, 1901.

[35] S. Perri, F. Scarcello, and N. Leone. Abductive logic programs with penalization: semantics, complexity and implementation. *Theory and Practice of Logic Programming*, 5(1-2):123 – 159, 2003.

[36] V. K. Raghu, A. Poon, and P. V. Benos. Evaluation of causal structure learning methods on mixed data types. In *Proceedings of 2018 ACM SIGKDD Workshop on Causal Discovery*, pages 48–65. PMLR, 2018.

[37] J. Ramsey, M. Glymour, R. Sanchez-Romero, and C. Glymour. A million variables and more: the fast greedy equivalence search algorithm for learning high-dimensional graphical causal models, with an application to functional magnetic resonance images. *International journal of data science and analytics*, 3(2):121–129, 2017.

[38] M. T. Ribeiro, S. Singh, and C. Guestrin. Anchors: High-precision model-agnostic explanations. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[39] C. M. Roelofs, M.-A. Lutz, S. Faulstich, and S. Vogt. Autoencoder-based anomaly root cause analysis for wind turbines. *Energy and AI*, 4:100065, 2021.

[40] P. R. Rosenbaum. Overt bias in observational studies. In *Observational studies*, pages 71–104. Springer, 2002.

[41] D. B. Rubin. Estimating causal effects of treatments in randomized and nonrandomized studies. *Journal of educational Psychology*, 66(5):688, 1974.

[42] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. Teneketzis. Failure diagnosis using discrete-event models. *IEEE transactions on control systems technology*, 4(2):105–124, 1996.

[43] J. Schoenfisch, C. Meilicke, J. von Stülpnagel, J. Ortmann, and H. Stuckenschmidt. Root cause analysis in it infrastructures using ontologies and abduction in markov logic networks. *Information Systems*, 74:103–116, 2018.

[44] K. Shvachko, H. Kuang, S. Radia, and R. Chansler. The hadoop distributed file system. In *Symposium on mass storage systems and technologies (MSST)*, pages 1–10, Incline Village, NV, USA, 2010. IEEE.

[45] M. Solé, V. Muntés-Mulero, A. I. Rana, and G. Estrada. Survey on Models and Techniques for Root-Cause Analysis. *arXiv:1701.08546 [cs]*, July 2017. arXiv: 1701.08546.

[46] P. Spirtes, C. N. Glymour, R. Scheines, and D. Heckerman. *Causation, prediction, and search*. MIT press, 2000.

[47] A. Strehl, G. Gupta, and J. Ghosh. Distance based clustering of association rules. In *Artificial Neural Networks in Engineering (Proceedings of ANNIE)*, volume 9, pages 759–764, 1999.

[48] E. A. Stuart. Matching methods for causal inference: A review and a look forward. *Statistical science: a review journal of the Institute of Mathematical Statistics*, 25(1):1, 2010.

[49] P.-N. Tan, V. Kumar, and J. Srivastava. Selecting the right objective measure for association analysis. *Information Systems*, 29(4):293–313, 2004.

[50] P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*. Addison-Wesley, USA, 2005.

[51] H. Toivonen, M. Klemettinen, P. Ronkainen, K. Hätönen, and H. Mannila. Pruning and grouping discovered association rules, 1995.

[52] L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[53] L. Van Der Maaten, E. Postma, J. Van den Herik, et al. Dimensionality reduction: a comparative. *J Mach Learn Res*, 10(66-71):13, 2009.

[54] H. Wietgrefe. Investigation and practical assessment of alarm correlation methods for the use in GSM access networks. In *IEEE/IFIP Network Operations and Management Symposium (NOMS)*, pages 391–403, Florence, Italy, 2002. IEEE.

[55] F. Ye, Z. Zhang, K. Chakrabarty, and X. Gu. Board-level functional fault diagnosis using multikernel support vector machines and incremental learning. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 33(2):279–290, 2014.

[56] M. Zaharia, M. Chowdhury, M. Franklin, S. Shenker, and I. Stoica. Spark: Cluster computing with working sets. In *Proceedings of the 2nd USENIX Conference on Hot Topics in Cloud Computing*, volume 10 of *HotCloud'10*, page 10, USA, 2010. USENIX Association.

[57] H. Zawawy, K. Kontogiannis, J. Mylopoulos, and S. Mankovskii. Requirements-driven root cause analysis using Markov logic networks. In J. Ralyté, X. Franch, S. Brinkkemper, and S. S. Wrycza, editors, *Advanced Information Systems Engineering*, pages 350–365, Berlin, Heidelberg, 2012. Springer.

[58] H.-D. Zeh. *The Direction of Time*. Springer, 1989.