UNIVERSITE BRETAGNE LOIRE / MATHSTIC

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

THESE DE DOCTORAT DE

L'ÉCOLE NATIONALE SUPERIEURE MINES-TELECOM ATLANTIQUE
BRETAGNE PAYS DE LA LOIRE - IMT ATLANTIQUE
COMUE UNIVERSITE BRETAGNE LOIRE

ECOLE DOCTORALE N° 601
Mathématiques et Sciences et Technologies de l'Information et de la Communication
Spécialité : *Informatique*

Par

Maha Mdini

ANOMALY DETECTION AND ROOT CAUSE DIAGNOSIS IN CELLULAR NETWORKS

Thèse présentée et soutenue à Rennes, le 20/09/2019
Unité de recherche : IRISA
Thèse N° : 2019IMTA0144

Rapporteurs avant soutenance :

M. Damien Magoni	Professeur, Université de Bordeaux
M. Pierre Parrend	Professeur, ECAM Strasbourg

Composition du Jury :

Président :	M. Eric Fabre	Directeur de recherche, INRIA Rennes
Examinateurs :	M. Damien Magoni	Professeur, Université de Bordeaux
	M. Pierre Parrend	Professeur, ECAM Strasbourg
	Mme. Lila Boukhatem	Maitre de conférences, Université de Paris XI
	M. Alberto Blanc	Maitre de conférences, IMT Atlantique
Directeur de thèse :	M. Gwendal Simon	Professeur, IMT Atlantique

Maha Mdini: *Anomaly Detection and Root Cause Diagnosis in Cellular Networks,* PhD Thesis, © October 2, 2019

To my beloved family,
Mariem and Hedi,
Houda and Mohamed,
Jawher,
and Ahlem.

ABSTRACT

With the evolution of automation and artificial intelligence tools, mobile networks have become more and more machine reliant. Today, a large part of their management tasks runs in an autonomous way, without human intervention. The latest standards of the Third Generation Partnership Project (3GPP) aim at creating Self-Organizing Network (SON) where the processes of configuration, optimization and healing are fully automated. This work is about the healing process. This question have been studied by many researchers. They designed expert systems and applied Machine Learning (ML) algorithms in order to automate the healing process. However this question is still not fully addressed. A large part of the network troubleshooting still rely on human experts. For this reason, we have focused in this thesis on taking advantage of data analysis tools such as pattern recognition and statistical approaches to automate the troubleshooting task and carry it to a deeper level. The troubleshooting task is made up of three processes: detecting anomalies, analyzing their root causes and triggering adequate recovery actions. In this document, we focus on the two first objectives: anomaly detection and root cause diagnosis.

The first objective is about detecting issues in the network automatically without including expert knowledge. To meet this objective, we have created an Anomaly Detection System (ADS) that learns autonomously from the network traffic and detects anomalies in real time in the flow of data. The algorithm we propose, Watchmen Anomaly Detection (WAD), is based on pattern recognition. It learns patterns from periodic time series and detect distortions in the flow of new data. We have assessed the performance of our solution on different data sets and WAD has been proven to provide accurate results. In addition, WAD has a low computational complexity and very limited memory needs. WAD has been industrialized and integrated in two commercialized products to fill the functions of supervising monitoring systems and managing service quality.

The second objective is automatic diagnosis of network issues. This project aims at identifying the root cause of issues without any prior knowledge about the network topology and services. To address this question, we have designed an algorithm, Automatic Root Cause Diagnosis (ARCD) that identifies the roots of network issues. ARCD is composed of two independent threads: Major Contributor identification and Incompatibility detection. Major contributor identification is the process of determining the elements (devices, services and users) that are causing the overall efficiency of the network to drop significantly. These elements have inefficiency issues and contribute to a non-negligible amount of the traffic. By troubleshooting these issues, the network performance increases significantly. The second process, incompatibility detection, deals with more fine and subtle type of issues. An incompatibility is an inefficient combination of efficient elements. Incompatibilities cannot be diagnosed by human experts in a reasonable amount of time. We have tested the performance of ARCD and the obtained results are satisfactory. An integration of ARCD in a commercialized troubleshooting product is ongoing and the first results are promising.

WAD and ARCD have been proven to be effective. However, many improvements of these algorithms are possible. This thesis does not address fully the question of self-healing networks. Nevertheless, it contributes to the understanding and the implementation of this concept in production cellular networks.

RÉSUMÉ

Grâce à l'évolution des outils d'automatisation et d'intelligence artificielle, les réseaux mobiles sont devenus de plus en plus dépendants de la machine. De nos jours, une grande partie des tâches de gestion de réseaux est exécutée d'une façon autonome, sans intervention humaine. Les dernières normes de la *Third Generation Partnership Project (3GPP)* visent à créer des réseaux dont l'organisation est autonome (*Self-Organizing Network (SON)*). Dans ces réseaux, les procédures de configuration, d'optimisation et de restauration (*healing*) sont entièrement automatisées. Ce document a pour objet l'étude du processus de restauration. Dans cette thèse, nous avons focalisé sur l'utilisation des techniques d'analyse de données dans le but d'automatiser et de consolider le processus de résolution de défaillances dans les réseaux. Pour ce faire, nous avons défini deux objectifs principaux: la détection d'anomalies et le diagnostic des causes racines de ces anomalies (*root cause diagnosis*).

Le premier objectif consiste à détecter automatiquement les anomalies dans les réseaux sans faire appel aux connaissances des experts. Pour atteindre cet objectif, nous avons créé un système autonome de détection d'anomalies qui extrait de l'information du trafic réseau et détecte les anomalies en temps réel dans le flux de données. L'algorithme qu'on propose, *Watchmen Anomaly Detection (WAD)*, est basé sur le concept de la reconnaissance de formes (*pattern recognition*). Cet algorithme apprend le modèle du trafic réseau à partir de séries temporelles périodiques et détecte des distorsions par rapport à ce modèle dans le flux de nouvelles données. Nous avons évalué les performances de notre solution sur des données venant de différents réseaux. Les résultats fournis par WAD font preuve de précision. Outre cela, WAD est efficace en termes de temps d'exécution et d'utilisation d'espace mémoire. Nous avons intégré WAD dans deux produits commercialisés pour contrôler les systèmes de supervision (*monitoring systems*) ainsi que pour gérer la qualité de service.

Le second objectif de la thèse est le diagnostic des causes racines. Ce projet a pour but la détermination des causes racines des problèmes réseau sans aucune connaissance préalable sur l'architecture du réseau et des différents services. Pour ceci, nous avons conçu un algorithme, *Automatic Root Cause Diagnosis (ARCD)*, qui permet de localiser les sources d'inefficacité dans le réseau. ARCD est composé de deux processus indépendants: l'identification des contributeurs majeurs à l'inefficacité globale du réseau et la détection des incompatibilités. L'identification des contributeurs majeurs consiste à déterminer les éléments (équipements, services et utilisateurs) qui sont à l'origine d'une chute importante de l'efficacité globale du réseau. Ces éléments ont une faible performance et contribuent à une quantité de trafic non négligeable. En résolvant ces défaillances, les performances du réseau augmentent considérablement. La détection des incompatibilités traite des problèmes plus fins. Une incompatibilité est un ensemble d'éléments fonctionnels dont la combinaison est non fonctionnelle. Les incompatibilités ne peuvent pas être identifiées par un expert en un délai raisonnable. Nous avons testé les performances d'ARCD et les résultats qu'on a obtenus sont satisfaisants. L'intégration d'ARCD dans un

produit commercialisé de diagnostic de réseau est en cours et les premiers résultats de tests sont prometteurs.

WAD et ARCD ont fait preuve d'efficacité. Cependant, il est possible d'améliorer ces algorithmes sur plusieurs aspects. Cette thèse ne donne pas une réponse complète à la question de l'auto-restauration (*self-healing*) dans les réseaux. Néanmoins, elle contribue à la compréhension et l'implémentation de ce concept dans les réseaux mobiles opérationnels. Dans ce qui suit, nous décrivons le fonctionnement de WAD et d'ARCD.

LA DÉTECTION D'ANOMALIES

Nous avons créé une solution de détection d'anomalies qui permet d'identifier les problèmes survenant dans un système de supervision en temps réel et d'une façon dynamique. Le but du WAD est de détecter des changements brusques tels que les crêtes et les creux dans des séries temporelles périodiques. Ces anomalies peuvent provenir de problèmes de configuration, de pannes d'équipements conduisant à une perte de trafic réseau, d'évènements de masse (tels que les évènements sportifs) à l'origine d'une saturation d'équipements, etc.

Notre solution doit répondre à un ensemble d'exigences spécifiées par les utilisateurs finaux de la solution qui sont les administrateurs systèmes et les techniciens réseaux. En premier lieu, la solution doit être facile à mettre en place, à configurer et à piloter. Deuxièmement, contrairement aux méthodes basées sur des seuils fixes, cette solution doit s'appliquer à des données périodiques. De plus, le modèle généré par l'algorithme doit être ajusté dynamiquement pour refléter l'évolution naturelle du trafic. La solution doit être aussi proactive pour détecter les anomalies en temps réel. En outre, la solution doit être non supervisée: Elle ne doit nécessiter aucun effort humain une fois déployée. La configuration doit être facile. En d'autres termes, la solution doit inclure un nombre réduit de paramètres pouvant facilement être compris et modifiés par les utilisateurs finaux. Outre cela, le but principal de la solution est d'avoir un taux de détection proche de 100% avec un faible taux de faux positifs. Enfin, et ce n'est pas le point le moins important, il est primordial que l'algorithme ait une complexité faible pour que le temps de calcul ainsi que les ressources requises soient raisonnables.

WAD répond aux exigences citées précédemment en analysant des métriques collectées par le système de supervision. Ces métriques peuvent être le débit du trafic en entrée des sondes, le nombre de comptes rendus de communication (*Call Data Record (CDR) et Session Data Record (SDR)*) en entrée/sortie des équipements de supervisions, le taux de déchiffrement, etc. Ces métriques forment des séries temporelles unidimensionnelles et sont fortement corrélées avec le comportement des abonnées. Ainsi, elles présentent une périodicité journalière avec des pics autour de 12h et 18h.

Pour chaque métrique, WAD génère un modèle de référence (*pattern*) décrivant l'évolution normale du trafic. Ensuite, il mesure l'écart entre le modèle de référence et les données temps réel. Si l'écart excède le seuil calculé, une alerte est déclenchée automatiquement pour prévenir les administrateurs réseaux de l'apparition d'une anomalie. WAD comprend deux phases: une phase d'apprentissage et une phase de détection. La phase d'apprentissage est exécutée une seule fois par jour dans le cas d'une périodicité journalière. Dans cette phase, WAD crée un modèle de référence pour chaque métrique et le stocke dans une base de données. Ce modèle sera utilisé durant la phase

de détection, qui s'exécute en continu. Avant de lancer l'algorithme, une étape de prétraitement (*preprocessing*) est exécutée dans le but de compléter les valeurs manquantes et de normaliser l'intervalle de temps entre les échantillons consécutifs. Pour ce faire, nous appliquons simplement une interpolation linéaire.

Vu qu'on ne s'intéresse qu'aux données périodiques, la phase d'apprentissage commence par un test de la périodicité des donnés. Pour ce faire, nous appliquons la Transformée de Fourier dans le but d'identifier une fréquence dominante. Si une telle fréquence a été détectée, on peut affirmer que la métrique est périodique et que sa période est égale à l'inverse de la fréquence dominante. Dans ce cas, nous passons à l'étape suivante du WAD qui est le calcul du modèle de référence. Ce calcul se fait en deux temps. Dans un premier temps, on segmente l'historique de la métrique en périodes et on calcule la moyenne point à point de toutes les périodes. Cette moyenne présente un modèle de référence provisoire. On écarte ensuite toutes les valeurs extrêmes par rapport à ce modèle et on recalcule la moyenne point à point. Cette moyenne représente le modèle de référence final de la métrique. Cette méthode nous permet d'obtenir un modèle de référence non biaisé par les valeurs extrêmes. Nous appliquons, par la suite, une transformation que l'on appelle *Difference over Minimum (DoM)* caractérisant les variations de la métrique durant une période. Cette transformée permet d'amplifier les variations brusques et de réduire les variations de faible amplitude. Cette transformée est appliquée également à l'historique des données. En étudiant la distribution de la transformée DoM des données autour de la transformée du modèle de référence, on calcule un seuil de normalité au delà duquel un échantillon est considéré comme très différent du modèle de référence et par la suite anormal. Le modèle de référence et le seuil calculé sont stockés dans la base de données.

Durant la phase de détection, on compare les échantillons arrivant dans le flux de données au modèle de référence construit durant la phase d'apprentissage. Si l'écart dépasse le seuil calculé durant la phase d'apprentissage, les nouvelles données présentent des anomalies. Pour ce faire, on applique la transformée DoM aux échantillons arrivant dans le flux de données. Après, on calcule la distance euclidienne entre la transformée de l'échantillon et la transformée du modèle de référence qu'on extrait de la base de données. Si cette distance est supérieure à la valeur du seuil calculé durant la phase d'apprentissage, une alerte est déclenchée. Le modèle de référence et le seuil sont mis à jour en début de chaque période lors de l'exécution de la phase d'apprentissage.

Nous avons évalué les performances de WAD en utilisant des données provenant de différents réseaux d'opérateurs. Nous avons comparé les performances de WAD à deux algorithmes de référence qui sont *Symbolic Aggregate Approximation (SAX)* et *Principal Component Analysis (PCA)*. Pour appliqer la PCA qui exige des données multidimensionnelles, nous avons transformé les séries temporelles unidimensionnelles à des séries multidimensionnelles en introduisant des décalages temporels. Dans notre contexte où on traite des séries temporelles collectées par un système de supervision, WAD est le seul parmi les algorithmes testés à offrir un bon compromis entre le taux de détection et la précision.

WAD a été industrialisé et déployé dans des réseaux opérationnels en tant que plugin de détection d'anomalies dans deux produits commercialisés d'EXFO. Les techniciens réseaux ont confirmé que WAD leur a permis de gagner en temps et en productivité et leur a facilité la tâche de dépannage du réseau. En effet, WAD automatise l'analyse répéti-

tive des métriques de supervision, effectuée manuellement par les experts. La précision et la réactivité de cet outil confirme l'avantage de l'utilisation du Machine Learning dans les réseaux mobiles.

LE DIAGNOSTIC DES CAUSES RACINES

La structure des réseaux mobiles est très complexe. Les opérateurs interfacent des technologies différentes (*3G, 4G et 5G*). Les équipements utilisés viennent de différents constructeurs. Les services proposés par les fournisseurs d'accès sont très variés. Les abonnés utilisent des téléphones différents avec des systèmes d'exploitation variés. Ces faits complexifie le diagnostic des réseaux et la détermination des causes racines des problèmes. Dans leur analyse, les opérateurs se basent sur les comptes rendus de communication pour expliquer les problèmes réseaux. Un compte rendu trace les équipements réseaux impliqués dans la communication, la nature et les caractéristiques du service requêté ainsi que des données liées à l'abonné tel que le type du téléphone utilisé dans la requête. Grâce à ces informations, les experts sont capables de déterminer l'origine des problèmes survenus. Cependant, cette analyse telle que faite aujourd'hui est très fastidieuse. Elle demande une grande expertise, du temps et un effort important de la part des experts. Pour cette raison, des recherches ont été menées pour automatiser cette tâche. Cependant, ces propositions ne répondent pas complètement à cette question.

La solution qu'on propose doit répondre à plusieurs contraintes. En premier lieu, elle doit être applicable à différents types de communications tels que les appels voix et les connections *Transmission Control Protocol (TCP)*. Deuxièmement, la solution doit s'appliquer à des données ayant un nombre de dimensions très élevé. Actuellement, le nombre dimensions d'un CDR peut être supérieur à 100. Ces dimensions sont hétérogènes. On peut distinguer des dimensions de type réseau (les équipements réseaux), des dimensions de types service tels que le type du service ou l'identifiant du fournisseur et des dimensions de types abonné tels que le type du téléphone et la version du système d'exploitation. La solution doit prendre en considération les dépendances entre différentes dimensions. Ces dépendances sont expliquées par l'architecture du réseaux et des différents services. Outre cela, un problème réseaux n'est pas forcément expliqué par un seul élément mais pourrait résulter d'une combinaison d'éléments comme dans le cas des incompatibilités. Enfin, notre solution doit hiérarchiser les différents problèmes selon leurs impacts et ne retourner que les plus pertinents.

On peut résumer les objectifs de notre solution, ARCD en trois points. Le premier point concerne l'identification des contributeurs majeurs. On appelle contributeur majeur tout élément à l'origine d'une baisse importante de l'efficacité globale du réseau. Par exemple, un *Mobility Management Entity (MME)* non fonctionnel peut bloquer la transmission d'un grand nombre d'appels. Une fois identifiés, les contributeurs majeurs peuvent être dépannés par des experts. Le deuxième point concerne la détection des incompatibilités. Une incompatibilité est un ensemble d'éléments fonctionnels lorsqu'ils sont pris séparément et non fonctionnels quand ils sont combinés. Citons à titre d'exemple une version d'un système d'exploitation incompatible avec un service à cause d'un bug dans l'implémentation de ce service. Le dernier objectif de notre solution est la création de classes d'équivalence. Une classe d'équivalence est un ensemble d'éléments liés au même problème. La corrélation des éléments d'une même classe d'équivalence peut provenir

de l'architecture du réseau. Après avoir identifié les contributeurs majeurs et les incompatibilités, une analyse statistique des dépendances nous permet de créer différentes classes d'équivalence. L'étude de ces classes nous permet de réduire chaque problème à sa cause racine.

La détermination des contributeurs majeurs se fait en cinq étapes. On commence par la labellisation des comptes rendus de communication s'ils ne sont pas labellisés. Pour ceci, on se base sur des critères liés à la qualité de service. Par exemple, si le temps de réponse est supérieur à un seuil minimal, on considère que la connexion a échoué. Après, on identifie les éléments qui sont suffisamment présents dans les comptes rendus des communications échoués et le sont beaucoup moins dans les communications correctement abouties. Pour ce faire, nous avons créé un système de classement (*scoring system*) qui permet d'identifier les éléments les plus importants. Par la suite, ces éléments sont groupés dans des classes d'équivalence. Une classe d'équivalence est un ensemble d'éléments impliqués dans les mêmes communications. On explore, ensuite, les dépendances hiérarchiques entre différentes classes d'équivalence et on produit un graphe de dépendances. Pour créer le graphe, on connecte les classes selon leurs dépendances hiérarchiques. Pour rendre le graphe lisible et exploitable, on supprime les liens superflus entre les nœuds (les liens qui portent une information redondante). Pour ceci, on utilise l'algorithme de parcours en profondeur pour déterminer tous les chemins possibles entre nœuds et on ne garde que le chemin le plus long. En utilisant cette méthode, on supprime les redondances tout en gardant la totalité de l'information. On termine par l'élagage du graphe de façon à ne garder que les causes racines des problèmes réseaux.

Pour identifier les incompatibilités, on procède en six étapes. On commence par la labellisation des comptes rendus de communication s'ils ne sont pas labellisés. Après, on parcourt tous les comptes rendus de communication pour identifier les combinaisons de deux éléments incompatibles. En d'autres termes, on cherche les éléments qui ont chacun un taux d'échec global faible, mais dont la combinaison a un taux d'échec important. L'étape suivante consiste à écarter les fausses incompatibilités. Une fausse incompatibilité est une combinaison ayant un taux d'échec important. Cependant, ce taux d'échec est expliqué par la présence d'un troisième élément non fonctionnel dans les comptes rendus contenant la combinaison. Par la suite, pour chaque élément présent les combinaisons restantes, on identifie l'ensemble d'éléments qui sont incompatibles avec lui. Puis, on groupe ces éléments dans des classes d'équivalence. Comme pour les contributeurs majeurs, on crée un graphe de dépendance et on procède à l'élagage pour ne garder que les racines des incompatibilités.

Nous avons évalué les performances d'ARCD dans la détermination des contributeurs majeurs et des incompatibilités. Les résultats sont prometteurs. Nous avons testé notre solution sur des données réelles venant de trois opérateurs différents. Avec les deux processus de détection de contributeurs majeurs et d'incompatibilités, nous avons pu expliquer respectivement 95%, 96% et 72% des échecs d'appels et de connexions TCP rencontrés par les abonnés des trois opérateurs. La précision dans le diagnostic des contributeurs majeurs est supérieure à 0.9 dans les trois cas et celle des incompatibilités est supérieure à 0.86. Nous avons comparé ARCD avec l'algorithme *Learning from Examples Module, version 2 (LEM2)* et nous avons conclu qu'ARCD est plus performant en termes de précision et de couverture de problèmes. ARCD est en cours d'industrialisation pour être intégré dans une solution d'analyse de performance de réseaux. ARCD sera aussi

connecté à WAD pour que le processus de détection d'anomalies déclenche automatiquement le diagnostic des causes racines.

PUBLICATIONS

M.Mdini, G.Simon, A.Blanc, and J.Lecoeuvre. Automatic Root Cause Diagnosis in Telecommunication Networks. Patent Pending, 2018.

M.Mdini, G.Simon, A.Blanc, and J.Lecoeuvre, ARCD: a Solution for Root Cause Diagnosis in Mobile Networks, in CNSM, 2018.

M.Mdini, A.Blanc, G.Simon, J.Barotin, and J.Lecoeuvre, Monitoring the Network Monitoring System: Anomaly Detection using Pattern Recognition, in IM, 2017.

ACKNOWLEDGEMENTS

I would like to thank, first and foremost, Professor Gwendal Simon and Professor Alberto Blanc without whom this thesis could not be done. During three years, they were guiding me through each step of the PhD and supporting me with every single task I had to accomplish. I am grateful to them for their help with the design and the validation of technical solutions. I would also thank them for their patience while reviewing and correcting my documents. I have learned a lot from their technical knowledge and personal qualities and I feel honored for the great opportunity I had to work with them. I would like also to thank IMT Atlantique for opening its doors to me for both my engineering degree and my PhD.

I had also the privilege to conduct my PhD in EXFO, the company which offered the perfect environment for this work. I would like to thank the members of my team, especially, Mr. Sylvain Nadeau, Mr. Loïc Le Gal and Mr. Jérôme Thierry for their support and help during these three years. I could not forget to thank Mr. Julien Lecoeuvre and Mr. Jérôme Barôtin with whom I worked during the first part of my PhD. I am also grateful to the Data Science team, namely, Mr. Nicolas Cornet, Mr. Fabrice Pelloin and Mr. Thierry Boussac as well as the Support team for their help on validating the thesis results. I would like to thank as well Ms. Isabelle Chabot for her help and assistance on the documentation aspect. Finally, I would like to thank all EXFO staff for their collaboration and for the great moments we have shared.

I would conclude with thanking my dear family members for their constant love and support. They are the best gift I have ever had.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

3GPP Third Generation Partnership Project

5G Fifth Generation of Cellular Network Technology

ACC Accuracy

ACF Autocorrelation Function

ADS Anomaly Detection System

AI Artificial Intelligence

AIC Akaike Information Criterion

ANOVA Analysis of Variance

AR Autoregressive

ARCD Automatic Root Cause Diagnosis

ARIMA Autoregressive Integrated Moving Average

ARMA Autoregressive Moving Average

ARP Address Resolution Protocol

AUC Area Under the Curve

BDR Bearer Data Record

BI Business Intelligence

BSC Base Station Controller

BSD Berkeley Software Distribution

BTS Base Transceiver Station

CDR Call Data Record

CP Control Plane

CPU Central Processing Unit

CS Circuit Switched

CSFB Circuit Switched Fallback

CSV Comma Separated Values

DBSCAN Density Based Spatial Clustering of Applications with Noise

DDoS Distributed Denial of Service

DET Detection Error Tradeoff

DoM Difference over Minimum

DPI Deep Packet Inspection

DSL Digital Subscriber Line

DWT Discrete Wavelet Transform

E2E End to End

EC Equivalence Class

EF Extra Features

EM Expectation Maximization

eNodeB Evolved Node B

EPC Evolved Packet Core

EUTRAN Evolved Universal Terrestrial Radio Access Network

EV Extra Values

FFT Fast Fourier Transform

FN False Negative

FNR False Negative Rate

FP False Positive

FPR False Positive Rate

FTP File Transfer Protocol

GERAN GSM EDGE Radio Access Network

G Gain

GGSN Gateway GPRS Support Node

GMM Gaussian Mixture Model

GUI Graphical User Interface

GUI Graphical User Interface

HDFS Hadoop Distributed File System

HLR Home Location Register

HMM Hidden Markov Model

HPC High Performance Computing

HSS Home Subscriber Server

HTTP Hypertext Transfer Protocol

IDE Integrated Development Environment

I Inefficiency

IMEI International Mobile Equipment Identity

IM Instant Messaging

IMSI International Mobile Subscriber Identity

IMS IP Multimedia Subsystem

IMT Institut Mines Telecom

IoT Internet of Things

IP Internet Protocol

IPTV Internet Protocol Television

IPython Interactive Python

ISP Internet Service Provider

IT Information Technology

KNN K Nearest Neighbors

KPI Key Performance Indicator

LDA Linear Discriminant Analysis

LEM2 Learning from Examples Module, version 2

LSTM Long Short-Term Memory

LTE Long Term Evolution

M2M Machine to Machine

MA Moving Average

MCC Mobile Country Code

MIT Massachusetts Institute of Technology

ML Machine Learning

MME Mobility Management Entity

MMS Multimedia Messaging Service

MNC Mobile Network Code

MRLS Multiscale Robust Local Subspace

MSC Mobile Services Switching Centre

MSISDN Mobile Station ISDN Number

OS Operating System

P2P Peer to Peer

PAA Piecewise Aggregate Approximation

PACF Partial Autocorrelation Function

PCA Principal Component Analysis

PGW Packet data network Gateway

PIP PIP Installs Packages

PLMN Public Land Mobile Network

PPV Positive Predictive Value

PR Precision-Recall

PS Packet Switched

QCI QoS Class Identifier

QoE Quality of Experience

QoS Quality of Service

RAN Radio Access Network

RAT Radio Access Technology

RDR Residual Data Rate

REST Representational State Transfer

RGR Record Generation Report

RI Residual Inefficiency

RL Reinforcement Learning

RNC Radio Network Controller

RNN Recurrent Neural Network

ROC Receiver Operating Characteristic

RPCA Robust Principal Component Analysis

SAX Symbolic Aggregate Approximation

SDR Session Data Record

SGSN Serving GPRS Support Node

SGW Serving Gateway

SIP Session Initiation Protocol

SMS Short Messaging Service

SMTP Simple Mail Transfer Protocol

SNMP Simple Network Management Protocol

SOM Self-Organizing Maps

SON Self-Organizing Network

SQM Service Quality Management

SSL Secure Sockets Layer

SVM Support Vector Machine

SVR Support Vector Regression

SV Software Version

TAC Type Allocation Code

TCPDR TCP Data Record

TCP Transmission Control Protocol

TLS Tranport Layer Security

TNR True Negative Rate

TN True Negative

TPR True Positive Rate

TP True Positive

UE User Equipment

UMTS Universal Mobile Telecommunications System

UP User Plane

USSD Unstructured Supplementary Service Data

UTRAN Universal Terrestrial Radio Access Network

VDR Video Data Record

VLR Visitor Location Register

VoIP Voice over IP

VoLTE Voice over LTE

VoWIFI Voice over WIFI

WAD Watchmen Anomaly Detection

WiFi Wireless Fidelity

Part I

BACKGROUND

In this part, we give an overview of the main topic of thesis. We present its context, objectives along with the related challenges. Then, we go briefly through some basic notions in telecommunication and data analysis, which are essential to the understanding of the technical content of the thesis.

1 INTRODUCTION

1.1 MOTIVATION

The Long Term Evolution (LTE) mobile networks provide very diverse services including classical voice calls, video conferences, video gaming, social networking, route planning, email, and file transfer, to name a few. With the Fifth Generation of Cellular Network Technology (5G), mobile operators aim to extend their activities to provide more services related to health care, public and private transport, and energy distribution. The performance of mobile networks has also significantly increased and mobile operators still work towards attaining higher bandwidth, lower latency, denser connections, and higher speed.

To provide various services with high Quality of Service (QoS), researchers in the area of cellular networks introduced new paradigms to ease the management tasks and the implementation of new functionalities. One of the key concepts introduced in LTE and 5G is the concept of Self-Organizing Networks (SONs) [38], [64]. The idea of self-organization refers to the automation of the processes that are usually conducted by experts and technicians. We identify three main functionalities related to SONs: self-configuration, self-optimization, and self-healing. Figure 1 illustrates this analysis. Self-configuration aims at the automation of the installation procedures. For instance, a new deployed Evolved Node B (eNodeB) can be automatically configured based on its embedded software and the data downloaded from the network without any human intervention. Self-optimization is the process of the automatic tuning of the network based on performance measurements. Self-healing refers to detecting network issues and solving them in an automatic and dynamic manner. In this thesis, we focus on this latter aspect. We conduct a study of self-healing techniques and propose a potential solution [98], [99].

The healing process is made up of four main tasks [75]. It starts with the detection of an issue in the network such as the degradation of a Key Performance Indicator (KPI). Once a problem is detected, one should carry out an in-depth analysis to diagnose the problem and find its root causes. Depending on the nature of the problem, two cases are possible. If the problem can be solved within a short time (few minutes), the recovery process is triggered and there is no need for compensation. On the contrary, if the recovery process is slow, compensation measures are taken to guarantee the availability of the network during the recovery process. The compensation and the recovery processes depend on the accuracy of the detection and the diagnosis. The main topic of this thesis is related to the two first phases of the healing process: the detection and the diagnosis. We aim to create an autonomous system that detects anomalies in real time and identifies their root causes in order to help experts to decide about the adequate recovery actions.

The detection of network issues is a part of the daily tasks of network technicians and administrators. To identify issues they check network metrics and KPIs to make sure their values are within the normal range. Detecting anomalies on metrics with constant values

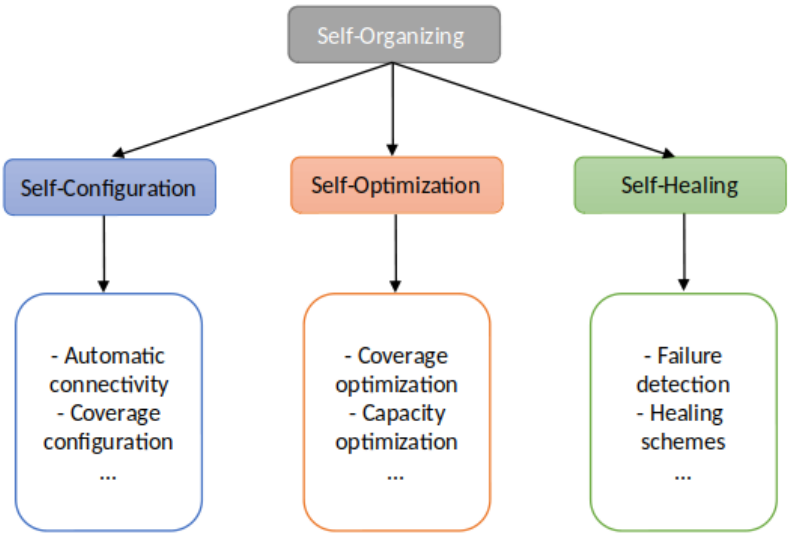


Figure 1: SON framework: Main functions.

can be easily automated. An expert may set a threshold and an alarm is raised every time the threshold is exceeded. However, in the case of non-constant metrics, network technicians analyze manually the different curves and distinguish normal patterns from anomalous ones. The automation of this task is not straightforward. For this reason, we have focused in this thesis on detecting anomalies using pattern recognition.

The diagnosis of network issues is a complex task. It requires a good knowledge of the network architecture and services. This task is carried out by telecommunication experts who analyze network logs to localize issues and identify their origins. Identifying the root of an anomaly is essential for an efficient troubleshooting. However, this process is not simple. The same problem may cause multiple dysfunctions in different points of the network. Thus, experts have to identify these failure points and point among them the one originating the issue. With the growing complexity of today's networks, this process can no longer be handled by human experts, nor can it be performed by a simple expert system implementing a set of hard coded rules. For this reason, we focused on creating a diagnosis system based on data mining.

1.2 CONTEXT

The present work has been carried out in the context of a partnership between Institut Mines Telecom (IMT) Atlantique and EXFO Incorporated. The thesis has been developed in EXFO under the academic supervision of the university. This thesis is based on real world data from different mobile networks. The prototypes developed are validated by end users (support technicians and telecommunication experts) and tested in live networks. The aim of this thesis is to create industrial products to automate network troubleshooting. It aims also to contribute to the study and development of the new generation of SONs by the integration of statistics and Machine Learning (ML) tools. This thesis marries the two fields of Cellular Networks and Data Analysis.

1.3 OBJECTIVES

Our goal is to study and propose techniques to design self-healing networks. The proposed techniques are based on data analysis concepts. In other terms, we work on algorithms allowing the extraction of information relevant to the troubleshooting process from the large amount of data produced by the cellular network. As stated earlier, we focus here on the detection and diagnosis tasks of the self-healing process. So this thesis has two major objectives:

- Anomaly detection: Our goal is to detect anomalies in the network in real time. The detection should be automatic and unsupervised. The proposed solution should learn from previous data without including any expert rules. It should also be dynamic and capable to adapt to the network evolution. Furthermore, the proposed solution should be efficient (having low error rate with few computational resources). In practical terms, the solution should be implemented as a module of a monitoring system and has to process the metrics it creates. To detect network issues, the solution should detect the anomalous patterns in the curves representing the network metrics. Once an anomaly is detected, an alarm is raised to notify the network administrator about the details of the issue.

- Root cause diagnosis: Our objective is to create a solution that analyses the End to End (E2E) cellular network and find the roots of the issues within the network. This solution should be unsupervised and automatic. It has to identify major high-impact causes at the first place. Localizing less important causes with lower impact is also part of the study. The solution should function without any prior knowledge of the network topology and its computational complexity should be reasonable. The solution for root cause diagnosis should be part of the monitoring system. It should be connected to the anomaly detection module. Once an anomaly is detected, the diagnosis could be automatically triggered. The result of the analysis should contain all the information needed for the troubleshooting process. The information should be presented in a way that is easily understandable by the experts.

For each of the two objectives, we aim to create a prototype and validate it in both a lab environment and a real network. Once the solution is validated, we aim to transform it into an industrial product that will go to the market as part of EXFO solutions.

1.4 CHALLENGES

Creating an automatic solution for network anomaly detection is challenging in two major aspects: data handling and integration within the monitoring system. The first aspect is related to data preparation and processing. The metrics of the monitoring system can be either constant, periodic, or chaotic. As our objective here is to address the issue of anomaly detection in periodic data, the solution has to automatically identify the types of the metrics and process only periodic ones. Periodicity detection is not easy, especially in irregular metrics where the interval between different samples is not constant. This is always the case in real monitoring systems where the data measurements are controlled

by queuing systems. Moreover, the solution has to learn a short history of data as monitoring systems are designed to erase data regularly. The solution has also to adapt to the natural growth of the traffic. This growth should be integrated into the model of data created by the solution in order to prevent it from being detected as an anomaly. Another challenging fact is the various patterns of anomalies. The solution has to detect new anomaly patterns that were not predefined during the implementation. The second aspect concerns the integration of the solution in the monitoring system. The solution should be transparent and not interfere with any other monitoring process. Thus, it has to have few computational and memory needs. The solution should not require any post implementation effort from experts. Consequently, all the detection steps should be fully automatic. Finally, the solution has to provide accurate results which is not easy with few calculation resources.

The diagnosis task demands a deep analysis of the data logs to identify the root cause of issues. Automating this process is challenging as it requires both domain knowledge and analysis capabilities from experts. The network architecture is needed to diagnose network issues. However defining it manually requires a large effort prior to the installation of the solution. Added to that, this information has to be updated each time the architecture is modified. To overcome this difficulty, the architecture can be discovered automatically based on the data. However, the inference of such complicated structure may require a lot of calculations. The network architecture is not the only information needed for the analysis. The structure of services provided by Internet Service Providers (ISPs) and web content providers is also required. Besides, the analysis of communication logs to extract information about issues is challenging in multiple aspects. The number of communication logs is huge, each having a large number of features. The features may be related. There is no dependency chart defining explicitly the relations between different features. Another challenging point is the large variety of network issues. Some issues may be interrelated and thus more complicated to be diagnosed automatically. The network issues do not have the same importance and thus a prioritization task must be included within the diagnosis process.

1.5 CONTRIBUTION

This thesis contributes to the research and industry in the field of network monitoring. As stated, we address here two major questions related to the concept of self-healing networks. The first subject is the detection of anomalies in network metrics. In this thesis, we propose a fully unsupervised solution, Watchmen Anomaly Detection (WAD) to this question. This solution has been industrialized and added to two monitoring tools to alleviate the load of network administrators. The second question is diagnosis of the root causes of network issues. Our solution, Automatic Root Cause Diagnosis (ARCD) is unsupervised and implements a large part of telecommunication experts tasks. ARCD analyses communication logs and creates an overview of issues at multiple levels. This fact reduces the expert role to validating the output of ARCD before triggering the adequate healing scheme. The industrialization of ARCD is ongoing and the first results are promising.

1.6 DOCUMENT STRUCTURE

This document is composed of three parts. The first part is an introductory part. It contains the introduction, a chapter about the telecommunication background and another one on the data analysis background. In the telecommunication background, we give an overview of cellular networks and monitoring systems. In the data analysis chapter, we introduce time series and multidimensional analysis as one needs some basic notions in these topics to understand the remainder of the thesis. The second part is about the first objective of the thesis: anomaly detection. In this part, we start by a study of the state of the art. Then, we present the theoretical and practical results of the solution we propose. We finish by a brief description of the industrialization process of our solution. The third part is about the second objective of the thesis: root cause diagnosis. As for the previous objective, we review the state of the art, introduce our solution and describe the industrialization process. In the conclusion, we recall the main results and give some potential future works.

2 TELECOMMUNICATION BACKGROUND

In this chapter, we introduce some telecommunication concepts relevant to the remainder of the thesis. As stated in the objective of the thesis, our aim is to create an anomaly detection and root cause diagnosis solution in cellular networks. We start by giving an overview of today's cellular networks architecture and services. Then, we highlight their complexity and list their requirements in terms of performance and reliability. Thereafter, we give a short overview of the state of the art of monitoring systems and we detail some of their functions. We finish by highlighting the gap between the required capabilities of monitoring systems and their current performance. Our work is an attempt to close this gap by proposing algorithms filling two important monitoring functions: anomaly detection and root cause diagnosis.

2.1 CELLULAR NETWORKS

Mobile operators offer different types of services ranging from basic services such as conversational voice, Short Messaging Service (SMS), Multimedia Messaging Service (MMS), and Unstructured Supplementary Service Data (USSD) to more complex ones appearing with the emergence of smart phones such as real time gaming, video chatting, video streaming and other web based services like browsing, email, social networking, and Peer to Peer (P2P) file transfer. With the arrival of 5G, more services are expected to appear based on Internet of Things (IoT) and Machine to Machine (M2M) related to smart cities and smart homes [9], [35]. The multitude and heterogeneity of services make the network infrastructure more and more complex. In this section, we give a brief overview of cellular networks architecture. Then, we explain briefly the complexity of network management and list cellular network requirements with regard to Third Generation Partnership Project (3GPP) standards.

2.1.1 *Architecture*

To reduce costs and ease the transition from one generation to another, 2G, 3G and 4G coexist in today's cellular network infrastructure. We will give now a brief view of the different technologies present in the Radio Access Network (RAN) and in the core network.

2.1.1.1 *RAN*

The RAN is the wireless part of the network that connects subscriber devices to the core network. It implements one or multiple Radio Access Technologies (RATs). As illustrated in Figure 2, in today's RAN we find the following RATs:

GSM EDGE Radio Access Network (GERAN) [49] consists in Base Transceiver Stations (BTSs) connected to a Base Station Controller (BSC). It has a bandwidth of 200 kHz, a data rate up to 1.89 MBps, and a latency of at least 180 ms.

Universal Terrestrial Radio Access Network (UTRAN) [138] is composed of Node-Bs connected to an Radio Network Controller (RNC). Its bandwidth is equal to 5 MHz. Its data rate goes to 42 MBps and its latency is at least equal to 110 ms.

Evolved Universal Terrestrial Radio Access Network (EUTRAN) [37] is a composed of a net of Evolved Node Bs (eNodeBs) that can communicate with each other. It uses many frequencies between 1 MHz and 20 MHz. Its data rate can exceed 300 MBps and its latency is less than 5 ms.

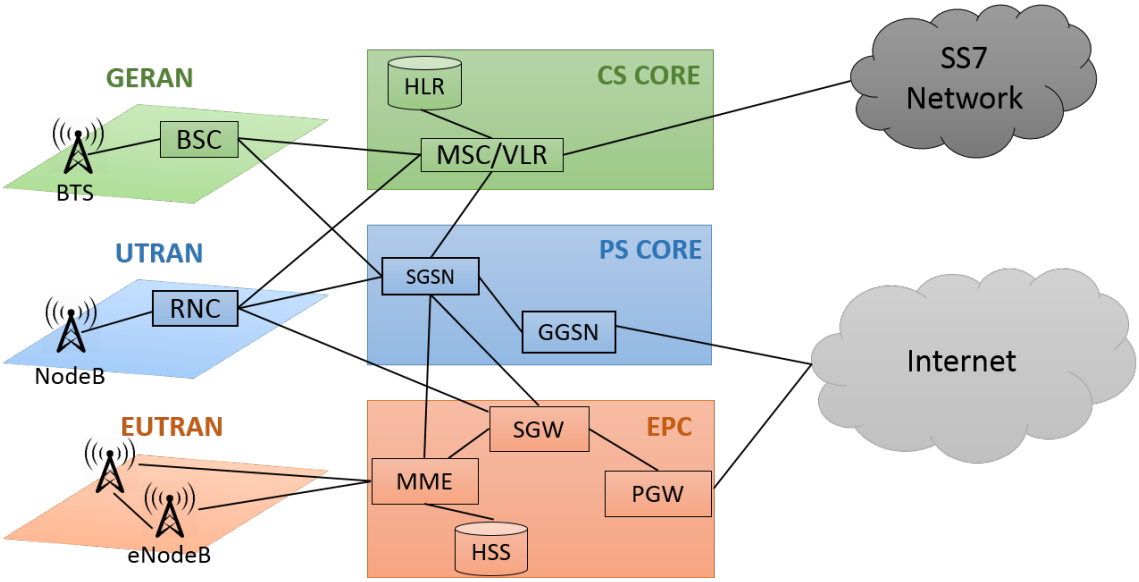


Figure 2: Cellular network infrastructure showing RAN and core elements.

2.1.1.2 *Core Network*

The core network is the network part that implements the network services. With the evolution of cellular networks, different types of core networks have been designed. Figure 2 shows the different types of core networks used by network operators [31]:

Circuit Switched (CS) Core is mainly used for voice calls (in 2G and 3G). Each communication has a dedicated channel. Figure 2 shows three main components of the CS core network: Mobile Services Switching Centre (MSC), Visitor Location Register (VLR), and Home Location Register (HLR). An MSC is a switching node that offers key functionalities such as authentication, registration, call location, and call routing. A VLR is an integral part of the MSC that provides subscriber information allowing the access to specific services. Finally, a HLR is a database that contains subscriber general information and last location. These data are relevant to service access and call routing.

Packet Switched (PS) Core is used in 3G to handle data. A message is broken into packets and its packet can go through a different circuit. In Figure 2, we see two main devices: a Serving GPRS Support Node (SGSN) and a Gateway GPRS Support Node (GGSN). The combination of a SGSN and a GGSN allows the control of subscriber attachment, service access, packet routing, and charging.

Evolved Packet Core (EPC) unifies voice and data over Internet Protocol (IP). Voice is considered as an IP application. The entry point of the core network is the Mobility Management Entity (MME). A MME handles the signalling related to mobility management such as paging. The main component of an EPC core network is the combination of a Serving Gateway (SGW) and Packet data network Gateway (PGW). This combination transports IP packets between User Equipment (UE) and external networks. The database of EPC networks is called Home Subscriber Server (HSS). It contains user information and implements functions to support MMEs in mobility management and network access.

2.1.2 *Procedures*

Cellular networks implement a set of procedures that allow subscribers to access different types of services. When a mobile phone is switched on, the *registration* process is executed. The International Mobile Subscriber Identity (IMSI) is authenticated and located. The subscriber profile is updated. When the subscriber moves from a location area, which is a group of cells, to another, the *location update* procedure is triggered. During this process, the subscriber location is updated and a new cell is selected to handle the calls and data sessions initiated by the subscriber. Another key procedure in cellular networks is the *call setup procedure*. When a subscriber attempts to call another subscriber, the called number is processed to identify the path the request will follow to reach its destination. Once the request is approved, a multimedia session is established. If, during the ongoing call, the subscriber moves from one cell to another, the call is transferred to the new channel without disconnecting the session. This process is known as the *handover procedure*. When a subscriber is outside the coverage area of its home network, he still can access services via intermediate networks. This process, called *roaming*, is guaranteed by means of agreements between different telecommunication operators.

2.1.3 *External networks*

Every cellular network is connected to other cellular networks. This fact allows cellular networks to carry outgoing calls and guarantee connectivity when subscribers are out of their coverage area. Cellular networks are also connected to service providers via internet. Service providers give subscribers access to a wide variety of services such as video streaming, social networking, and online gaming. Cellular networks link subscribers to service providers and carry the exchanged data between them. Different services do not have the same requirements in terms of bandwidth, fault tolerance, and latency. Mobile operators have to adjust their configuration to fit different requirements.

2.1.4 *Cellular network requirements*

The telecommunication market standards define challenging criteria that should be filled [3], [45]. First, *high performance* is the most important targeted criterion. Operators aim to deliver an elevated Quality of Service (QoS) and Quality of Experience (QoE). They have to assure high responsiveness by reducing the latency and increasing the throughput. They also have to ensure high connectivity by densifying their network and widening their coverage areas (indoor/outdoor). Second, operators work towards having a *resilient and flexible network* with an adaptive capacity to support disruptive as well as natural traffic growth. Third, operators aim to have a *fault tolerant network* that guarantees a constant availability of the services by implementing compensation and fast recovery mechanisms. Lastly, standards value *security and privacy*. Operators should secure their networks from attacks against both integrity and privacy.

2.2 MONITORING SYSTEMS

As shown in the previous section, cellular networks are very complex to monitor. At the same time, the telecommunication market and the standardization entities push operators to deliver high quality services. These facts make the monitoring systems a critical component of today's cellular networks. Many operators outsource a part of their monitoring tasks to specialized companies such as EXFO.

The goal of monitoring the cellular network is to have full visibility and control over the network infrastructure and processes. This task is carried out by experts in different domains: network administrators, software technicians, Information Technology (IT) specialists, and telecommunication experts. The goal of integrating monitoring systems is to increase the speed and efficiency of these experts by automating tasks and providing useful information. In the following, we detail the main functions of a monitoring system.

2.2.1 *Main functions*

The monitoring functions can be grouped into three categories: troubleshooting, performance monitoring, and planning. Some operators use a centralized monitoring system that incorporates the three categories. Others use multiple systems, each implementing a subset of the monitoring functions. We detail now each category of functions.

Troubleshooting is the most important monitoring function as operators want their services to be continuously available. The troubleshooting process can be broken down into four main functions [124]: data collection, issue detection, issue identification, and recovery. Data collection consists in collecting data that is relevant to the troubleshooting process by creating and/or collecting logs, recording events, mirroring the traffic, generating reports, and calculating metrics. The second step, issue detection, consists in analyzing logs and evaluating metrics to detect anomalies. If an abnormal event is found in the logs or a metric has an abnormal value, the operator is notified. The notification can take different forms such as raising an alarm or creating a ticket. The third function, issue identification consists in

a deep analysis of the data to diagnose the network and identify the issue. Finally, the recovery step consists in fixing the problem by triggering the adequate compensation and recovery mechanisms. The mentioned functions can be manual, partially automated, or fully automated.

Performance monitoring consists in measuring Key Performance Indicators (KPIs) to gain insights on network performance, benchmark network elements (e.g. cells) and services, and identify the low performing ones. Through performance monitoring, operators may improve the QoS and therefore the QoE [88].

Prediction and planning are based on process/resource monitoring. By analyzing KPI trends, operators can improve their services in different ways. As an example, they can anticipate events and prevent downtime [54]. Furthermore, they can predict churn and propose more personalized subscriptions [125]. Moreover, they can allocate resources more efficiently [60].

2.2.2 *Structure*

Monitoring systems have different structures depending on many factors such as the size of the network, the load of the traffic, the main afforded services, and the preferences of the operator. The monitoring system can be centralized or distributed. In the following, we focus on the four main components that are present in almost every monitoring system.

Probes [83], [136] are mirroring components that can be built in the network monitoring tool or installed separately in the monitored devices. Probes pull data from devices passively in real time using protocols like Simple Network Management Protocol (SNMP), Transmission Control Protocol (TCP), and Hypertext Transfer Protocol (HTTP).

A data processing entity is considered as the brain of the monitoring system. It implements basic functionalities such as data filtering and aggregation. It also includes advanced functionalities such as trend analysis, anomaly detection, network diagnosis, capacity planning, and QoE estimation based on Business Intelligence (BI) and Machine Learning (ML) tools. This component creates KPIs, generates status information, and summary reports.

An alerting system triggers real time alarms based on the information offered by the data processing entity. For example, it notifies the administrators if a KPI is below a predefined threshold.

A visualization system is a Graphical User Interface (GUI) that displays the summary information in a human readable manner. It enables the administrator to navigate the different components of the network and to visualize status and performance statistics.

2.2.3 *Data types*

The monitoring systems produce two types of logs: communication logs and system logs [137]. Communication logs are the logs a monitoring system keeps from subscribers activity such as call traces. Call traces are unstructured data including all the messages exchanged by network devices during a mobile communication initialized by a subscriber. These traces are aggregated into Call Data Records (CDRs) and Session Data Records (SDRs) which are structured logs containing the list of the network devices and the service details involved in the mobile communication. CDRs are the records of voice calls and SDRs are the records of TCP connections. CDRs and SDRs are aggregated into multidimensional counters reporting the performance of the network (e.g. the mean response time by the 3-tuple (RAT, service, handset type)). System logs are the logs created to keep track of the events, processes and messages of the system such as Address Resolution Protocol (ARP) requests. Alerts are created based on these logs (whenever an abnormal event occurs). System logs and communication logs are aggregated into KPIs which are time series presenting the temporal evolution of performance indicators such as the Central Processing Unit (CPU) usage and the call drop rate.

2.2.4 *Requirements*

To guarantee network high performance, the monitoring system has to be effective. The effectiveness translates in terms of different requirements. First, the monitoring system should *scale* to handle the growth of the traffic and the expansion of cellular networks [85]. Second, the monitoring system should be *flexible* to supervise multi-vendor and heterogeneous network equipments [69]. Third, the monitoring system should be sufficiently *reactive* to detect issues and trigger mitigation operations in real time to limit downtime. In addition, it should perform an in-depth analysis of the network and go to fine granularity levels to find hidden issues. Fourth, the monitoring system should be *autonomous* to a certain degree to be reactive and reduce demanded human efforts. Repetitive tasks should be automated. More advanced tasks can be partially automated by the use of ML [88], [124]. The settings of the monitoring system should be straightforward [88]. Moreover, it has to be *compliant with telecommunication standards and market*. The monitoring system should respect the standards by insuring the demanded QoS. It has also to respond to the marked needs [2], [4], [87] (e.g. integrating new functions to monitor new services). Furthermore, it should be *fault tolerant* and *easy to troubleshoot*. Lastly, the monitoring system has to be *cost-effective*. As the network traffic is huge, the analysis implemented in the monitoring system should be optimized to reduce computational needs. The monitoring system should also store only needed data and for a limited period of time.

2.2.5 *Limitations*

While the monitoring systems have evolved in terms of autonomy and efficiency, they are still below the requirements mentioned in the previous paragraph in many aspects: First, the monitoring process still relies on the human presence. Many monitoring tasks

today are carried out manually. While monitoring systems generate KPIs and alarms, these latter are analyzed by experts when troubleshooting the network. This fact makes the troubleshooting task very costly to operators. Second, the cellular networks still suffer from low efficiency occasionally (downtime) or continually in some specific cases such as roaming and mobility. The monitoring systems are not sufficiently reactive and efficient to address unavailable services in real time nor to handle roaming and handovers in an optimal way. Last, the 5G standards have set high expectations in terms of QoS. The current monitoring systems are not capable of guaranteeing such performance [127].

3 DATA ANALYSIS BACKGROUND

In this chapter, we go through the data analysis background needed for the thesis. We start by giving an overview of time series which is the main mathematical concept we need to go through Part ii. Then, we will briefly introduce multidimensional analysis which is the mathematical basis of Part iii.

3.1 TIME SERIES

3.1.1 *Definitions*

An **univariate time series** $x(t)$ is a sequence of measurements of a single variable that is time dependent [19]. It is often denoted $X(t) = \{x(t)|t \in T\}$ where T is the time index. Figure 3 is an example of time series. It shows the evolution of the number of call logs in an interface between two monitoring devices: a Neptune™probe and a hub over time.

A **multivariate time series**, as its name suggests, is a sequence of measurements of multiple variables. The variables are co-dependent and depend also on time.

In the following, we denote the **average** and the **standard deviation** of a time series $x(t)$ as:

$$\mu_x(t) = E[x(t)] \tag{1}$$

$$\sigma_x(t) = \sqrt{E[(x(t) - \mu_x(t))^2]} \tag{2}$$

where the operator $E[x]$ denotes the expectation of x.

$x(t)$ is said to be (weakly) **stationary** if μ_x and σ_x are not time dependent. This assumption is crucial in many time series models.

3.1.2 *Seasonal time series*

A **seasonal** or **periodic** time series is a series that verifies $x(t + m) = x(t)$ where m is called the period of the series. Seasonal time series exhibits a repetitive pattern at regular time intervals. This seasonality comes from many factors depending on the domain. For example, in cellular networks, the seasonality is explained by the repetitive behaviour and the lifestyle of subscribers: few calls during the night, a peak in calls at 12 am and 6 pm as shown in Figure 3.

Seasonal times can be decomposed into three components:

- The trend which is the long term variations of the series. It can be deduced using filtering or polynomial fitting techniques.
- The seasonal component which contains the repetitive pattern. It can be deduced from a harmonic analysis.
- The residual component which contains the random variation in the time series.

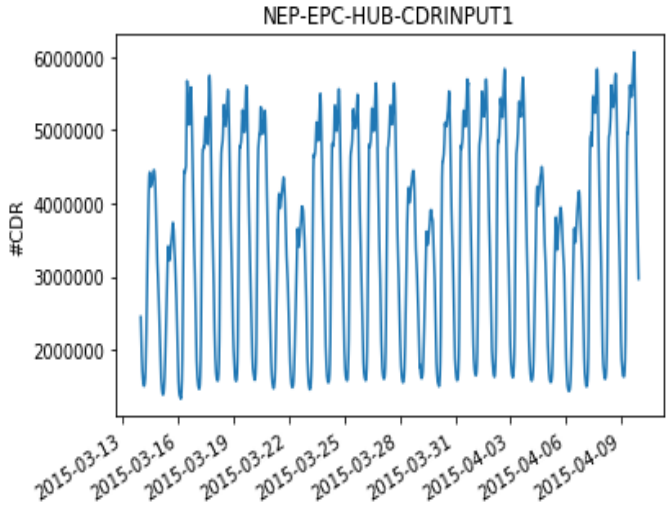


Figure 3: An example of time series:
The number of CDRs in an interface of a monitoring probe over time.

Figure 4 shows a time series of the number of CDR in a probe interface. The time series is decomposed into its trend, seasonal and residual components. The trend shows that the traffic is increasing over time. The seasonal component shows a perfect daily periodicity in the number of calls. The residual component contains noise effects and abrupt changes in the traffic.

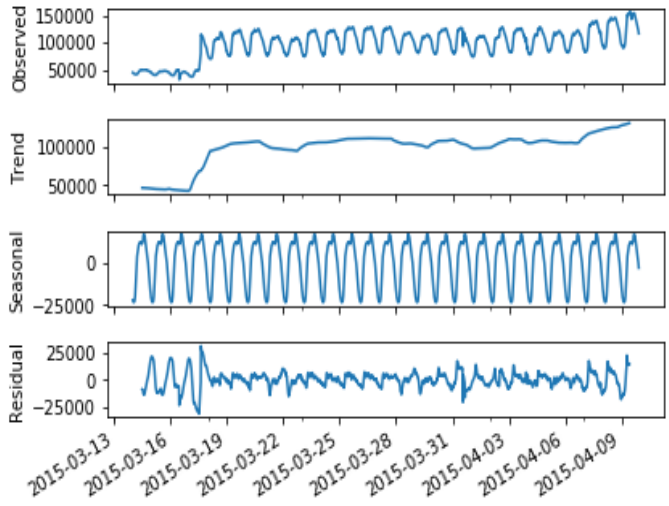


Figure 4: The decomposition of a time series into its trend, seasonal and residual components.

3.1.3 *Time analysis*

This type of analysis consists on analyzing the series in the time domain. One of the most known techniques is the Autocorrelation Function (ACF). It helps in identifying patterns and finding periodicity. The ACF analysis is a base of the auto-regressive models used in

forecasting that will be detailed further. The ACF is a measure of the correlation between two instants of the same time series. The ACF of a time series $x(t)$ is defined as:

$$\rho_x(t_1, t_2) = E[x(t_1).x(t_2)] \tag{3}$$

The dot, in this chapter, denotes the product operator. In the case of stationary time series, the ACF is only function of the lag between the two instants:

$$\rho_x(\tau) = E[x(t).x(t+\tau)] \tag{4}$$

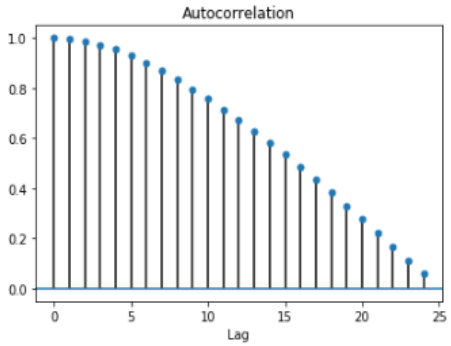


Figure 5: Time series ACF:
The lag is the number of in-between samples.

Figure 5 shows the ACF of the time series plotted in Figure 3 which is stationary. The time interval between two successive samples is 15 minutes. The plot shows a high correlation between successive samples: $ACF(10) \simeq 0.8$. This high correlation demonstrates that we can apply auto-regression and predict values from previous ones.

A more sophisticated technique to select the samples to use in an auto-regressive model is the Partial Autocorrelation Function (PACF). The PACF is the mere correlation between two samples after subtracting the influence of the samples separating them. It is defined as:

$$\pi_x(\tau) = E[(x(t) - P_{t,\tau}(x(t))).(x(t+\tau) - P_{t,\tau}(x(t+\tau)))] \tag{5}$$

where $P_{t,\tau}(x(t))$ denote the projection of $x(t)$ in the space spanned by $x(t+1), ..., x(t+\tau-1)$.

Figure 6 is the PACF of the time series in Figure 3. We see here that with a lag $\tau = 4$ samples, we have the highest value of PACF. This means that this component will have the highest coefficient in the auto-regressive model.

3.1.4 *Frequency analysis*

This analysis consists in transposing the time series to the frequency domain and analyzing the frequencies to find the characteristics of the time series such as periodicity and noise. The most known technique to do so is the Fourier Transform. In the case of time series, we can use the Fast Fourier Transform (FFT) which is an algorithm that calculates the Fourier Transform of discrete data in an optimal way [13].

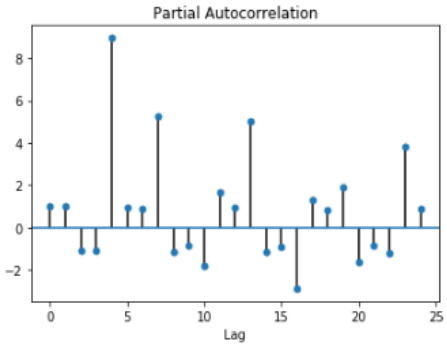


Figure 6: Time series PACF:
The lag is the number of in-between samples.

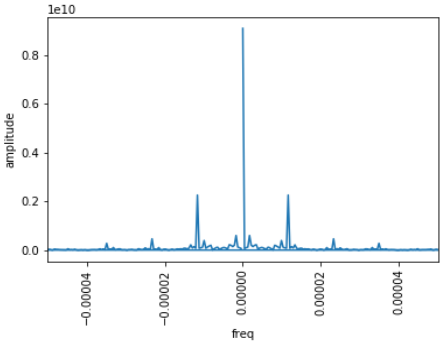


Figure 7: Time series FFT.
The two important symmetrical components correspond to daily periodicity.

Figure 7 is the FFT of the time series in Figure 3. The central component at freq=0 corresponds to the mean of the time series. The two high symmetrical components correspond to the daily seasonality of the time series. The plot shows also some other periodical variations. These is no high frequencies with important amplitudes, which means that the noise in the time series is not significant. Other techniques to study time series in the frequency domain exists such as the Wavelet Transform. The details are in [5].

3.1.5 *Forecasting in time Series*

Time series are used in many areas to do forecasting. Forecasting is important not only for planning and optimization but also in predicting issues in order to prevent them or limit their extent. The most widely used model for time series forecasting is Autoregressive Moving Average (ARMA), [19]. ARMA is the combination of two models applicable to stationary time series: an Autoregressive (AR) model and a Moving Average (MA) model. In the case of non stationarity, the time series is differentiated multiple times until stationarity. Then the model is applied (Autoregressive Integrated Moving Average (ARIMA)).

An AR model of order p, $AR(p)$ is denoted as:

$$x(t) = c + \sum_{i=1}^{p} \psi_i . x(t-i) + \epsilon(t) \tag{6}$$

where c is a constant, $\psi_1, ..., \psi_p$ are regression parameters and $\epsilon(t)$ is a variable that models the noise of the time series. Equation (6) shows that AR models can handle seasonality if samples from previous periods are included in the model $x(t-T)$.

A MA model of order q, $MA(q)$ is denoted as:

$$x(t) = \mu + \epsilon(t) + \sum_{i=1}^{q} \theta_i . \epsilon(t-i) \tag{7}$$

where μ is the expectation of $x(t)$, $\theta_1, ..., \theta_q$ are the parameters of the model and $\epsilon(t)$ is a variable that models the noise. As in Equation (7), MA models are flexible as they estimate the current value of a time series based on previous values of the noise variable.

An ARMA model of orders p and q, $ARMA(p, q)$ is denoted as:

$$x(t) = c + \sum_{i=1}^{p} \psi_i . x(t-i) + \epsilon(t) + \sum_{i=1}^{q} \theta_i . \epsilon(t-i) \tag{8}$$

The ARMA models are widely used thanks to their flexibility and precision. The coefficients of the model are fitted using the least squares regression. The selection of p and q is generally based on the PACF and the ACF. The selection can be also made on the basis of Akaike Information Criterion (AIC) [15].

3.1.6 *Patterns in time series*

Time series can display specific patterns that would be relevant to the use case of the analysis. The idea of pattern recognition in time series is not new and has been approached in different ways. The time series is split into segments [71] and the segments could be classified into classes of patterns [61]. The classification can be based on any Machine Learning (ML) algorithm such as K-means, neural networks, Support Vector Machine (SVM), decision trees, etc. Seasonal time series, because of their periodicity, are good candidates to be analyzed via pattern recognition tools.

3.2 MULTIDIMENSIONAL ANALYSIS

Multidimensional analysis presents many challenges. In the following, we give a brief tour of its main concepts.

3.2.1 *Numerical vs categorial data*

In a data set, the features may have different types: numerical, categorial [7] or mixed. In the case of numerical data, a large panel of analysis tools is available. The data can be processed in different ways by applying mathematical functions. The precision of the analysis can be measured and the error can be quantified. The visualization of data is simple and the distance between the samples is calculated in an intuitive way. However, for categorial data, the features are rather qualitative and we have less analysis tools. Values are not always comparable. The concept of distance depends on the use case and cannot be generalized, although some distance measures exist such as the Hamming distance. The error and precision are context dependent.

In both cases, we have specific tools to deal with each type of data. The challenge is when we have mixed data (numerical and categorial features). In this case, very few algorithms are applicable. To overcome this barrier, two solutions are possible: either we transpose numerical data into categorial data by discretization or we transform categorical data into numerical by encoding it. Discretization consists in mapping numerical values into qualitative levels by setting thresholds. For example, in the case of response time in Transmission Control Protocol (TCP) connections, we may consider a response time below 20 ms as good, between 20 ms and 100 ms as acceptable and a response time higher than 100 ms as bad. Discretization is practical, however it induces an obvious loss of information. Encoding is the process of converting categorial values to numerical values. Many encoders have been proposed [117]. We cite here, the ordinal and the OneHot encoders. The ordinal encoder converts a categorial feature with k different values to integers from 1 to k. By the means of this encoder, we are capable of applying regression techniques to categorial data. However, the notions of distance and proportionality are irrelevant and the results are very likely to be biased. In the case of OneHot encoder, for each value of a categorial feature, we create a binary column. So a categorial feature with k different values is replaced with k numerical (binary) features. This encoder may drastically increase the dimensionality of data and therefore the execution time of the analysis.

3.2.2 *Sampling*

In some cases of high volumes of data (Big Data), it is not advised to process the whole data for both cost and practicality reasons. In these cases, sampling is a necessity rather than an option. Sampling [27] is defined as the selection of a subset of samples from a data set on which the analysis will be performed. By using sampling, the processing time is reduced and the application of complex models is possible. There are many possible approaches to perform sampling such as random selection, cluster sampling, and stratified sampling. The selection of the sample size is not straightforward. It depends on many factors such as the analysis techniques that will be applied, the available computational resources, and the required accuracy of the results. As the selected subset is not perfectly representative of the original set, many types of errors are induced. A first type of errors is sample over-fitting. In this case, the inferred model fits perfectly the sample. However, it does not fit the original data set. In our context of designing monitoring systems for mobile operators, one may think of a solution perfectly fitting a specific network but giving random results for other networks. Another type of errors is the random sampling error, where the results display random variations originated from the random selection in the sampling process. This can be the case of a tendency artificially created by a random selection of points.

3.2.3 *Dimensionality reduction*

During the data collection process, administrators in the telecommunication domain tend to log every bit of information that could be useful for analysis. Today, we are able to have an End to End (E2E) view of communications including details about the involved devices, service information, and protocol messages. While being comprehensive, the data analysis task becomes cumbersome. Analyzing data with a high number of features requires both large computation resources and huge human efforts. That is why it is necessary to reduce data dimensionality.

Dimensionality reduction is beneficial with regard to many aspects. First, it is cost effective: It reduces the computation time and resources. It also reduces the storage resources by compressing high dimensional data into low dimensional data and removing redundant features. Second, it allows us to have simple data models that are easily understood and interpreted. It also eases data visualization. Last, it allows us to apply complex algorithms. Some algorithms perform very poorly when applied to high dimensional data compared to low dimensional data. This fact is known as the curse of dimensionality. By reducing the number of features, it is possible to remove noise features and have more accurate results. To perform dimensionality reduction, two approaches are possible [72]:

FEATURE SELECTION consists in the selection of a subset of features that are relevant to the analysis. Multiple techniques exist depending in the model input (numerical or categorial data) and the output of the analysis (classification or regression). Table 1 gives some examples of the used techniques.

FEATURE EXTRACTION consists in combining multiple features in order to transform a high dimensional analysis space to a low dimensional analysis space. The most

Input \ Output	Numerical (Regression)	Categorial (Classification)
Numerical	Pearson correlation: measures the linear correlation between the input and the output variable of the regression.	LDA: identifies a linear combination of variables that separates the different classes of a categorial variable.
Categorial	T-test/ANOVA: test if the mean of the output is the same in each class of the input.	χ^2-test: quantifies the correlation between two categorial variables based on their distributions.

Table 1: Feature selection techniques by input/output type.

known technique is Principal Component Analysis (PCA). PCA creates a linear combination of numerical inputs that maximizes the variance. The concept behind PCA is that variables with low variance (almost constant) are not useful in the data model. Feature extraction can also be performed using ML algorithms such as neural networks and decision trees.

3.2.4 *Statistical Inference*

Statistical inference aims at creating a model that highlights specific characteristics of data [22]. The model could be for example the distribution of a random variable. The model selection can be subjective based on intuition or expertise in the domain or objective based on statistical tests such normality tests. Models can be parametric or non parametric. Most of the used models are parametric and the estimation of parameters is the core of data analysis process. To this aim, many estimators have been proposed such as Bayesian estimator and Maximum likelihood. Some estimators return an interval rather than a value of the parameter with a confidence coefficient of the real parameter to be in that interval.

Hypothesis testing has been introduced to validate models or assumptions in general. Hypothesis testing is a formal procedure to accept or reject an assumption (e.g. a model parameter value). It consists on formulating a null hypothesis (the proposed model) and an alternative hypothesis. The second step is then to identify a test statistical to evaluate the truth of the null hypothesis. Depending on the type of the hypothesis many tests have been defined such as χ^2-tests, t-tests, z-test, etc. These tests allow to accept or reject the null hypothesis with a confidence parameter (p-value). If the null hypothesis is rejected, the alternative hypothesis is considered to be true.

3.2.5 *Machine Learning*

Machine Learning (ML) [142] is an area of data analysis, where the algorithm learns continuously from data and improves its precision. ML aims at imitating human learning, reasoning and decision making. Unlike expert systems which are automated systems

based on human expertise, ML-based systems rely completely on the system to learn from data. The intelligence of ML algorithms comes from the iterative processing of data until convergence or attaining a high precision. ML allows to build autonomous systems with an embedded artificial intelligence. There are two phases in an ML algorithm: a training phase where the system learns from data and a test phase when the algorithm applies the acquired knowledge to new samples. The system keeps learning from new data to increase its performance.

There are two types of ML algorithms: supervised and unsupervised. Supervised ML is when the expected outcome of the algorithm is known for the data used in the training phase. In this case, the algorithm can be finely tuned to have a high precision. In the case of unsupervised learning, the expected outcome of the training data is unknown. In most of the real world cases, we have to work with unsupervised ML as manually specifying the expected outcome for training data can be either difficult or very time consuming. However, we can specify the expected outcome for only a portion of the training data. This is an uncommon type of ML known as semi-supervised ML. It can be attained by integrating user feedback into the system in the case of online learning [51].

ML algorithms can be classified into two main categories: regression and clustering algorithms. Regression is the process of estimating a variable based on other variables. There is a wide variety of regression tools such as Linear/Logistic regression, Decision tree regression, Support Vector Regression (SVR), and neural networks. Clustering, as its name suggests, is the process of grouping samples into classes. The main algorithms of clustering are K-means, K-nearest neighbors, decision trees, random forests, SVM, and neural networks. Large neural networks, are commonly used for both regression and classification. This process has taken the name of Deep Learning where the number of hidden layers in the neural network is considerably high.

3.2.6 *Validation aspects*

To validate the relevance of a data analysis procedure, many aspects have to be considered, a critical one being the statistical significance. The analysis should be applied to fairly large data sets in order to draw general conclusions. The data set should be representative of real world data (in terms of probability distributions and variability) to prevent over-fitting issues. Moreover, the results should be consistent and precise. Decisions cannot be made based on contradictory and inaccurate results. Furthermore, the analysis should be easily reproducible. The use of heuristics may lead to different results at each execution which makes the solution difficult to validate. Another important aspect to consider is the complexity. The use of complex solutions makes the results difficult to interpret and may lead to divergence bugs such as infinite loops and extreme parameter values. Finally, the solution should not be very demanding in terms of computation and storage resources. It has also to be scalable and applicable to larger data. The used algorithms have to be easily distributed.

Part II

ANOMALY DETECTION

This part is about the first objective of the thesis: Anomaly detection. In this part, we start by a review of the state of the art. Then, we introduce a theoretical explanation and some practical results of the solution we propose. We finish with a brief description of the integration process of our solution in industrial products.

4

STATE OF THE ART

In this chapter, we present the state of the art on anomaly detection. We start by introducing the problem itself. Then, we explore the techniques used to detect anomalies. In the third section, we give a brief overview of the evaluation methods used to validate the introduced techniques.

4.1 PROBLEM STATEMENT

The question of anomaly detection has interested researchers in different domains for more than a century. The question has taken many names such as "anomaly detection", "outlier detection", "novelty detection" and "peak detection." There is no universally accepted definition of anomaly or outlier. Nevertheless, many definitions depending on the context have been proposed. In general, an anomaly is a data sample that is considerably different from the whole data set. In the cellular networking industry, the definition of anomalies is tightly related to business considerations. An anomaly is a data sample pointing to an event or a to a dysfunctional (software/hardware) component resulting, either directly or indirectly, in a financial loss.

Although anomaly identification is subjective, many Anomaly Detection Systems (ADSs) have been developed. Some ADSs are very domain specific and can not be applied to other domains due to some constraints (input/output schemes, available resources, etc.). Other ADSs are very flexible and can be applied in many different domains. The newly proposed solutions based on Artificial Intelligence (AI), are sufficiently generic to be integrated in different environments.

In general, an ADS collects a large amount of data and identifies abnormal points based on prior knowledge (expert systems) or on knowledge deduced from the data (such as Machine Learning (ML) solutions). Depending on the application field and the reliability of the ADS, the detected anomalies can either be reported to an expert to analyze them or fed to another system that runs mitigation actions automatically.

Depending on the analysis context, an ADS can have different input types [53]. The type of inputs dictates some of the constraints to consider in the selection of the anomaly detection technique [17]. We will give here some examples in our context of cellular networks. ADS inputs can be system logs produced by different devices of the network. It can be also communication logs such as Call Data Records (CDRs). The logs are presented generally as multidimensional data frames. An ADS may also take measurement time series which are stamped counters, e.g. the number of subscribers accessing a service as a function of time. An ADS can take network graphs which can be be static or dynamic. As an example, one may think of the cell topology in a cellular network to detect interference issues. Last but not least, spatial data (such as user coordinates) can reveal anomalies. For example, an abnormal distribution of users around an access point/antenna may result from a problem within the access point.

Depending on the types of the inputs, the anomalies may be of different types [146]. For example, if the input data are graphs, the anomalies could be nodes or edges. If the inputs take the form of sequences like in the case of some network logs, an anomaly may be a sequence of events. The anomalies can also be classified with regard to the their scope into anomalous points, collective anomalies and contextual anomalies, as shown in [25]. *Anomalous points* are individual samples of data having an abnormal value. For instance, a server, overloaded for a short time span, may have an abnormally long average response time. The second type of anomalies, *collective anomalies*, are collections of data samples that are abnormal as a whole. However, the individual samples are not necessarily anomalous. This case may occur in sequential data and more precisely in time series. This could be the case of a wrongly configured probe having a persistent traffic loss. The third type, *contextual anomaly*, is a sample of data that is anomalous in some specific context. However, it is considered to be normal otherwise. This is the case of periodic data where a peak can be normal at some point of the period and anomalous in another. For instance, the drop of the cell traffic is normal during scheduled maintenance operations and anomalous otherwise.

The output of an ADS could be a score or a label [25]. Flexible ADSs return an anomaly score. The user has then two options: Either, he selects the top n samples with the highest score, or he sets a cut-off threshold and considers the samples with a score higher than the threshold as anomalous. ADSs can also return a binary label (normal/anomalous) or multi-class label (normal/anomaly-type A/anomaly-type B/etc).

Detecting anomalies in general presents many challenges [8], [24], [25] in many aspects: First, the notion of normal data is very domain specific. There is no universal procedure to model normal data. The concept of normality is still subjective and difficult to validate. In addition, data classes (normal/anomalous) are in general imbalanced. And since statistical tools are not applicable to few samples, modeling anomalies is not easy because of their rarity. Second, the boundary between normal and anomalous data is fuzzy. There is no rule to decide for the points in this gray zone except subjective judgement. Added to that, the data may contain noise which makes the distinction between normal and anomalous data more difficult. Third, the concept of normality evolves with time. What is normal in a time span can turn anomalous in the future. Novel anomalies, that are not contained in the model, can also appear. Last, it is not common to have labeled data to train the model in many domains.

4.2 TECHNIQUES

Given the extent of the literature on anomaly detection, we have grouped works in a few common threads: knowledge based, regression, classification, clustering, sequential analysis, spectral analysis, information theory and rule induction. These approaches differ in many aspects such as the applicability scope, the execution mode (offline/online) and the requirement of domain knowledge. In the following, we give a brief explanation of the different techniques and enumerate their pros and cons.

4.2.1 *Knowledge based*

This approach is based on the human knowledge of the domain. It assumes that the patterns of anomalies are known and that it is possible to encode these patterns in a way that can be understood by the machine. Based on these assumptions, the creation and functioning of a knowledge based system (also called an expert system) involves three steps [44]. First, an expert has to analyze a large amount of data manually and to identify anomaly patterns. Then, the anomaly patterns are implemented in an automated system. Lastly, the system runs automatically and detects anomalies in new data. The expert knowledge can be programmed in different ways [10]:

RULE BASED SYSTEMS : Contains a set of if-then rules related to different events. It supposes that the data are deterministic (the same conditions always lead to the same consequence). It is simple to implement; however, the list of rules should be exhaustive.

STATISTICS BASED SYSTEMS : A slightly advanced version of the rule based systems. It creates decision rules based on simple statistical calculations. As an example in mobile networks, if a service has a higher delay than the average delay of all services, it is considered to be anomalous. This type of systems can also include hard coded thresholds.

FINITE STATE MACHINE : A more formal way to encode expert knowledge. A state machine encodes the succession of events in an efficient way. This method can compact a large set of rules in a simple graph which eases expert tasks.

Regarding performance aspects [36], [122], knowledge based systems are known for their robustness to false positives as they detect only the patterns specified by the user. However, to have a fair detection rate, the programmed list of anomalies has to be exhaustive which is very time consuming. Another difficulty is to translate human knowledge in a machine language which is not always easily feasible. The main flaw of knowledge based systems is their incapacity to detect novel types of anomalies. For these considerations, researchers and industrial actors moved towards more autonomous and dynamic systems.

4.2.2 *Regression*

This approach is based on the principle of forecasting. An anomaly can be defined as a gap between the reality and what was expected. An ADS based on regression operates in the following mode: First, it makes an estimation of the coming data. Then, it quantifies the gap between the real value and the predicted one. If the gap is sufficiently large (higher than a predefined threshold), the new data point is considered as an anomaly. The predicted value is generally calculated on the basis of previous values. That is why this approach applies well to time series. To forecast new values, we need an autoregressive model such as Autoregressive Integrated Moving Average (ARIMA). Many researchers have applied ARIMA to detect anomalies in different contexts [101], [115], [150]. ARIMA models are known to be highly precise in forecasting values and extendable

to seasonal time series. However, this precision depends significantly on the selection of the order of the model (auto-regressive, differentiation and moving average orders). The major weakness of ARIMA is that it lacks an efficient model updating procedure and the model parameters are recomputed using a non-linear method (Box–Jenkins). This fact makes ARIMA computationally expensive.

Neural networks can be used to perform regression by removing the activation function from the output layer and using an objective function adapted to regression such as least squares. To predict new values from previous ones, one may use Recurrent Neural Network (RNN). In order to address the issue of RNN vanishing gradient, on may use Long Short-Term Memory (LSTM) to detect anomalies [96]. By using neural networks to perform regression, we escape the need of labeled data (normal/anomalous) to train the model. While RNNs and neural networks in general seem to give promising results, they still very unstable and poorly controllable (When neural networks perform poorly, it is difficult to point the origin of the issue.). Another major disadvantage is the high time complexity of the training process. Since the complexity is linearly proportional to the number of nodes and layers in the neural network as shown in Table 3, having a deep neural network may result on a high time complexity.

4.2.3 *Classification*

This approach assumes that data points can be grouped into classes. There are two possible ways to view anomalies [25]: either as scattered points far from a dense central group of data points (two class classification) or as dense groups of data distinct from the groups of normal data (multi-class classification). The classification techniques suppose that we have a training data set with labeled data. In this approach, the anomaly detection process can be broken into the following steps [114]: First, we classify the training data and identify classification attributes. Next, we learn a model using the training data. Then, we can classify new data using the learning model. To detect anomalies, many classification algorithms have been used [25]. We focus here on the most relevant ones.

Some researchers have used decision trees to detect anomalies [43], [105], [131]. In general, systems based on decisions trees are easily comprehensible as they show clearly the succession of tests leading to classifying a point as normal/anomalous. Their complexity depends on the algorithm used to build the tree (CART, ID3, C4.5, C5.0, etc). The main issue of decision trees is that by reducing a complex classification problem with highly multidimensional data, the risk of over-fitting is very high. To overcome this flaw, one may use random forests [148]. A random forest consists on creating multiple decision trees, each on a randomly selected subset from the input data set. The output of a random forest is the mean/mode of all the decision trees. This way, the issue of over-fitting is fixed. However, the enhancement of the classification performance comes at the expense of the computational complexity: The larger the number of the decision trees in the random forest, the more accurate the classification results and the heavier the computation needs.

Likewise, Bayesian networks have been used to classify data into normal/anomalous instances [59], [97]. Bayesian networks are known to be fast and easy to implement. However, they rely on the independence assumption between data features which is

not always the case (especially in the case of high dimensional data where it is very likely to have correlations). When the independence assumption is not verified, Bayesian networks have very low performance.

Support Vector Machine (SVM) is another classification algorithm widely used in anomaly detection systems, in the case of numerical data [62], [92], [104], [130]. SVM creates a boundary between normal and anomalous data based on a kernel function. SVM allows high classification accuracy when non linear kernels (e.g., polynomial) are used. However, with non linear kernels, SVM is highly susceptible to over-fitting issues.

Neural networks classifiers have been used in the context of intrusion detection [65], [66], [134]. As stated for neural network regressors, the lack of interpretability and the time complexity still hinders their widespread adoption.

To conclude, a wide variety of classification techniques have been used for the purpose of anomaly detection. This approach has two main issues: On one hand, it requires labeled training data which is challenging especially in cellular networks where we deal with huge volumes of data. On the other hand, it detects only known issues, so an extra effort is needed to keep the model up-to-date.

4.2.4 *Clustering*

Clustering is an unsupervised technique to group similar data. This approach assumes that anomalies are either points not belonging to any cluster or belonging to small clusters compared to large and dense clusters of normal data [25]. Clustering algorithms have two phases: a training phase and a test phase. During the training phase, the data points are grouped into clusters over a large number of iterations until convergence or attaining predefined criteria (maximum number of iterations, etc). In the test phase, a new data point is assigned to a the closest clusters based on the distance/similarity measure used in the training. There exists a large variety of distance types that depend on the data types and the use case. For example, the K-means algorithm generally uses the euclidean distance for numerical data and the Hamming distance in the case of categorial data (K-modes).

Many clustering algorithms have been used to detect anomalies. One of the most used is the K-means [43], [105], [106]. K-means starts by selecting centroids and creating clusters around them by assigning each data point to its closest centroid. Then, the centroids and the clusters are updated repeatedly until convergence. While k-means is simple to implement, easy to interpret and time efficient, it has few flaws: First, one needs to know the number of clusters before running the algorithm which is not always possible especially in the case of anomaly detection, when anomalies may form multiple small clusters. Second, it is very sensitive to the scale of features. For instance, if one normalizes data, the results can change completely. Last, it is very sensitive to noise and in the case of high dimensional data, it may have very poor performance.

Another clustering algorithm used in anomaly detection is hierarchical clustering [73]. Hierarchical clustering creates a dendrogram by merging two clusters at each iteration in a greedy manner. This algorithm does not require knowing the number of clusters in advance and returns a multilevel clustering. This is very practical as it allows the user to select the adequate sensitivity based on the clustering results. It is also very sensitive to outliers and very efficient in detecting them. However, it has a quadratic time complexity.

Added to that, as it operates in a greedy manner when merging clusters, it has no global objective function that is optimized during the process. As a consequence, controlling the algorithm to improve the results is not possible.

Another option to cluster anomalies is the Expectation Maximization (EM) algorithm [129]. This algorithm alternates the steps of labeling data based on a Gaussian Mixture Model (GMM) and estimating the GMM parameters based on data labels. The main issue of this algorithm is that it assumes that the number of the components of the GMM is known. Another flaw is its sensitivity to the starting point (the initialization of the GMM parameters).

Density Based Spatial Clustering of Applications with Noise (DBSCAN) is a clustering algorithm allowing, by design, to detect outliers [23], [135]. DBSCAN does not require a prior knowledge of the number of clusters, contrary to k-means. It also allows to have non-clustered points (outliers). DBSCAN is flexible as it creates clusters of different shapes (unlike k-means where the clusters are spherical) and does not prevent clusters from overlapping. However, in the case of data sets with varying densities, DBSCAN may label many normal points as outliers (if they are situated in a low density region). This may lead to a high false positives rate.

In summary, the clustering approach is very interesting in anomaly detection as it is unsupervised. It does not require labeled data and allows to discover novel anomalies. The testing phase for new data is generally fast as it is a simple comparison with clusters predefined in the training phase. However, depending on the algorithm, the training step maybe very long as some algorithms have slow convergence issues. In addition, the performance of clustering algorithms is completely based on the used distance (or similarity measure). In the case of high dimensional data, the distance significance may be altered by noise which may lead to poor performance.

4.2.5 *Sequential analysis*

With clustering, classification and regression techniques, we can detect point anomalies. However, these techniques do not address the case of collective anomalies. As explained earlier, a collective anomaly is a sequence of samples that is anomalous as a whole. To deal with these types of anomalies, there are some specific techniques based on sequential analysis [25].

Markov Models are one of the most widely used sequential models. In their work, Ren *et al.* [118] segment a time series into sequences using a sliding window. Then, for each sequence, they discretize the time series values into N states (by splitting the interval between the min and the max of each sequence into N equal intervals). Based on these states, they create a finite state machine (a Markov model of order N) summarizing the probabilities of the possible transitions. This way they train the model for normal data. A new sequence of probability 0 is considered as an anomaly. To have a robust and dynamic model, they define a substitution mechanism to substitute new anomalies before integrating them in the model. This method allows to detect novel sequences in a dynamic way. However, substituting anomalies may lead to a large number of false positives in the long run. For example, after a reconfiguration, the shape of the traffic may change permanently. The substitution process will prevent the model from learning the new shape of the traffic and all the new sequences will be tagged as anomalous.

Li *et al.* [84] proceed in a different way dealing with multivariate time series. They start by transforming data into uni-variate time series by testing multiple techniques. Then, they create a Hidden Markov Model (HMM) model with two hidden states: normal and anomalous. For new data samples, they use the Viterbi algorithm to estimate the state (detect anomalies). The main issue with this method is that it is supervised: It requires labeled data to calculate the HMM parameters.

Another way to deal with sequential time series is to transform them into sequences of symbols before detecting anomalous patterns [50]. As an example, Symbolic Aggregate Approximation (SAX) creates symbolic representations of time series in an efficient manner [42], [70]. Keogh *et al.* [70] propose to map time series into words using SAX. Then, they create a tree of symbols representing all the possible words with their number of occurrences. A word that occurs rarely is considered as an anomaly. The tree structure of the model allows to label real time data in a very efficient way. Nevertheless, this solution has some limitations when applied to some specific time series, and especially to periodic data where certain patterns considered as normal in a given part of the period can be considered as anomalies in another one. For example, if in a weekly periodic data, there is a peak each Tuesday at 7pm but only one on a Friday at 3pm, we need to detect the peak on Friday as an anomaly. SAX, which "does not keep track of time", does not detect this anomaly.

4.2.6 *Spectral Analysis*

In this approach, we assume that normal and anomalous points are easily separable in a lower dimensional space. Thus, we project the data in this space. Then, we apply a distribution-based or an expertise-based threshold to identify anomalies. In the following, we give few examples of spectral analysis techniques: Principal Component Analysis (PCA), Fast Fourier Transform (FFT), The Wavelet transform and the Hough transform.

PCA is a linear decomposition of multidimensional data into orthogonal components based on variance maximization. Many researchers have used PCA to detect anomalies [58], [81], [119]. PCA helps detecting peaks and dips in noisy data and separates repetitive patterns from novel ones efficiently. However, it remains a linear decomposition and requires the noise to be linearly separable from significant data. In most cases, this condition is not satisfied [151].

The FFT is a widely used technique in spectral analysis. It allows transforming data from the time domain to the frequency domain. FFT is efficient in detecting anomalies as they can by defined as abrupt changes in time series. Such changes are reflected as high frequencies in the spectrum while periodicity is transformed into low frequency components. Han *et al.* [55] showed how to detect anomalies using clustering techniques like k-means on the FFT of the time series. The main difficulty of FFT-based anomaly detection is when one has to find the time at which each frequency occurs.

The Discrete Wavelet Transform (DWT) is a spectral technique that operates in both time an frequency domains. It combines and represents time and frequency information at each point. The wavelet transform has been used in anomaly detection as it addresses the flaws of FFT [11], [34], [89]. However, for fine analysis, this transformation requires complex computations and so is not applicable in real time use cases.

Fontugne *et al.* [39] have explored the Hough transform to detect anomalies. The Hough transform allows to identify lines in pictures. In [39], the network traffic is mapped into pictures. In the pictures, the data points are randomly distributed except for anomalies where they shape lines. This approach can detect effectively some types of anomalies such as Distributed Denial of Service (DDoS). However, some anomalies may have very irregular patterns and thus cannot be detected.

To conclude, spectral analysis is an interesting approach in detecting anomalies as it is does not require labeled data. It is also efficient as it reduces the dimensions of data. However, it is based on the assumption that anomalies and normal points are separable in a lower space which is not always the case. Furthermore, some of the spectral analysis tools are very computationally demanding.

4.2.7 *Information Theory*

This approach is based on the assumption that anomalies in a data set alter its information content. This approach consists mainly on measuring information indicators of the flow of data and detecting the moments of abrupt changes. The most used information indicator in information theory is the entropy that quantifies the uncertainty or randomness of the data. The entropy has been used by many researchers to detect anomalies [12], [109], [141] for two main reasons. On the one hand this approach is unsupervised and so we do not need any labeled data. On the other hand, there is no need to make any assumptions about data distribution and thus this approach can be applicable in different contexts. The issue with this technique is its high sensitivity to the presence of noise.

4.2.8 *Rule Induction*

This approach assumes that there exists a set of rules that allows to identify anomalies in a deterministic way. This approach requires labeled data. The training step consists in learning the rules based on the data labels. In the test step, we apply the rules to new samples to label them as normal/anomalous. Many techniques allow rule induction such as decision trees (see Section 4.2.3) and association rules [18]. Association rules allow to create rules based on two factors: the support and the confidence. This solution is very simplistic. However, it is very computationally demanding. To have flexible and accurate rules, Luo *et al.* [91] integrated fuzzy logic with association rules. In general, in Big Data and high dimensional data sets, association rules are not applicable for both computation and accuracy issues.

4.2.9 *Hybrid Solutions*

We call hybrid solutions those using two or more techniques of anomaly detection. There are two possible ways for combining different solutions: serial and parallel topologies [143]:

SERIAL TOPOLOGY: As shown in Figure 8, the output of a technique is the input of the following one. This is basically used in the case of multilevel classification

where we start by a rough classification and then go progressively to finer classification [6], [78]. It is also used in the case where we want to use a supervised technique but we do not have labeled data [43], [105], [126]. So, we start by an unsupervised technique to label the training set. Then we apply the supervised technique based on these labels to learn the model.

PARALLEL TOPOLOGY [46], [116]: As shown in Figure 9, we run multiple techniques independently to labels data as normal/anomalous. For each sample, we return an anomaly score proportional to the number of techniques that labeled the sample as an anomaly. The aim of this approach is to enhance the accuracy of the solution as different techniques have different flaws and can compensate each other. This way, the false positives detected by only one system will have a low anomaly score. Likewise, the false negatives missed by only one system are detected by the other techniques.

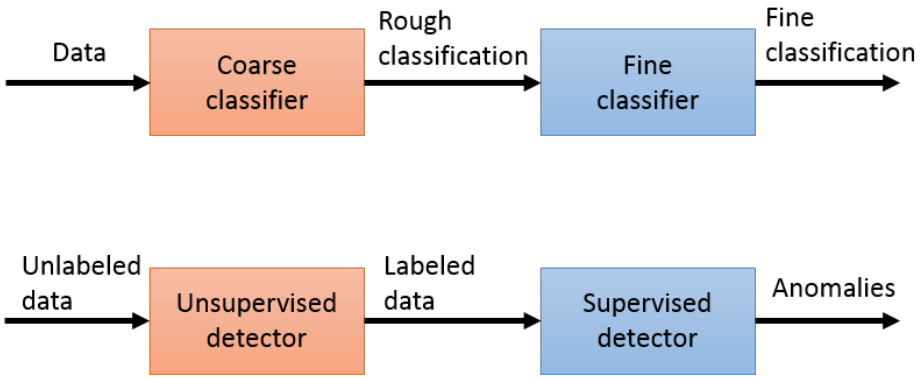


Figure 8: Examples of hybrid ADSs with a serial topology.

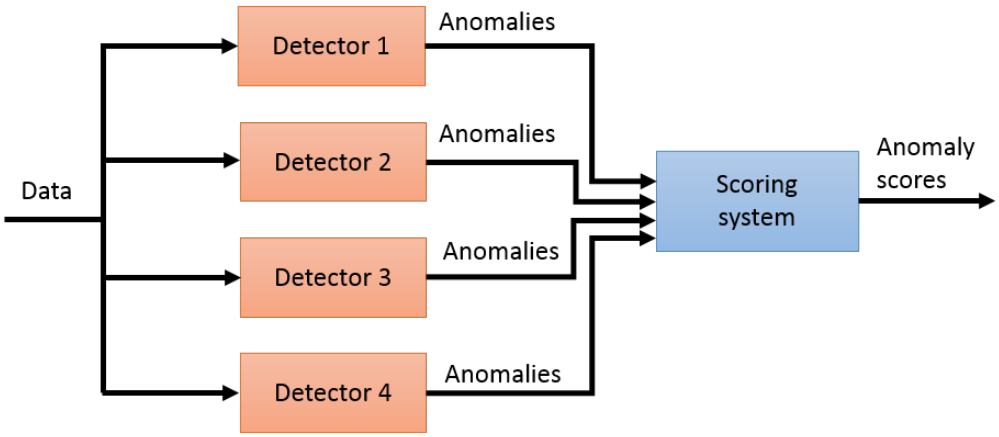


Figure 9: Hybrid ADS: Parallel topology.

4.3 EVALUATION METHODS

In this section, we give an overview of the evaluation methods that can be used to assess the performance of an ADS.

4.3.1 *Confusion Matrix*

A confusion matrix is a table that puts together the results of the application of an ML algorithm. In the case of a anomaly detection (and any binary classification in general), we have four quantities in the confusion matrix:

- True Positives (TPs): A TP is an anomaly labeled correctly by the ADS.
- False Positives (FPs): A FP (called also Type I error) is a normal value labeled erroneously by the ADS as an anomaly.
- True Negatives (TNs): A TN is a normal value labeled correctly by the ADS.
- False Negatives (FNs): A FN (called also Type II error) is an anomaly labeled erroneously by the ADS as a normal value.

		Predicted	
		Positive	Negative
Real	Positive	TP	FN
	Negative	FP	TN

Table 2: Confusion matrix.

As Table 2 shows, the confusion matrix gives an overview of the performance of an ADS. Many performance measures have been derived from the confusion matrix to evaluate the accuracy of the results.

4.3.2 *Evaluation Metrics*

The following measures allow to evaluate the accuracy of an ADS [128]:

SPECIFICITY : Called also True Negative Rate (TNR), it measures the proportion of normal points that are correctly labeled.

$$\text{TNR} = \frac{\text{TN}}{\text{TN} + \text{FP}} \tag{9}$$

PRECISION : Called also Positive Predictive Value (PPV), it measures the proportion of anomalies correctly labeled among all the points predicted as anomalies.

$$\text{PPV} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{10}$$

This measure quantifies the relevance of the points detected as anomalies.

RECALL : Called also sensitivity or True Positive Rate (TPR), it measure the proportion of anomalies that are correctly labeled.

$$TPR = \frac{TP}{TP + FN} \tag{11}$$

This measure quantifies the capability of an ADS to detect existing anomalies.

FALLOUT : Called also False Positive Rate (FPR), it measures the proportion of false positives among all the normal points.

$$FPR = \frac{FP}{FP + TN} \tag{12}$$

False Negative Rate (FNR): is the proportion of not detected anomalies.

$$FNR = \frac{FN}{FN + TP} \tag{13}$$

ACCURACY (ACC): Is the proportion of the points correctly labeled by the ADS.

$$Accuracy(ACC) = \frac{TN + TP}{TN + TP + FN + FP} \tag{14}$$

The *specificity* and the *fallout* are complementary. They measure the capacity of the system to recognize a normal point. Likewise, the *recall* and the *FNR* are complementary. They measure the capacity of the system to detect an anomalous point.

$$TNR + FPR = 1 \tag{15}$$

$$TPR + FNR = 1 \tag{16}$$

4.3.3 *Evaluation Curves*

4.3.3.1 *The Receiver Operating Characteristic curve*

Receiver Operating Characteristic (ROC) curve is a plot highlighting the discrimination capacity of an ADS to distinguish between normal and anomalous points [29]. As shown in Figure 10, the ROC curve plots the TPR as a function of the FPR. The closer the curve to the horizontal asymptote, the better the ADS performance. To compare two ADSs, we may use the Area Under the Curve (AUC) [16]. A perfect ADS has an $AUC = 1$. An ADS with an $AUC = 0.5$ (orange line in Figure 10) is a random system with no discrimination capacity.

4.3.3.2 *The Precision-Recall curve*

The Precision-Recall (PR) curve is another way to assess the quality of an ADS [29]. The PR curve displays the TPR as a function of the PPV. Like the ROC curve, the PR curve aims to show the trade-off between detecting all the anomalies and having few false positives. However, as the number of anomalies is in general very low compared to the number of normal instances, the FPR can be very close to 0 and its variations can be imperceptible. That is why the use of PPV instead is more judicious.

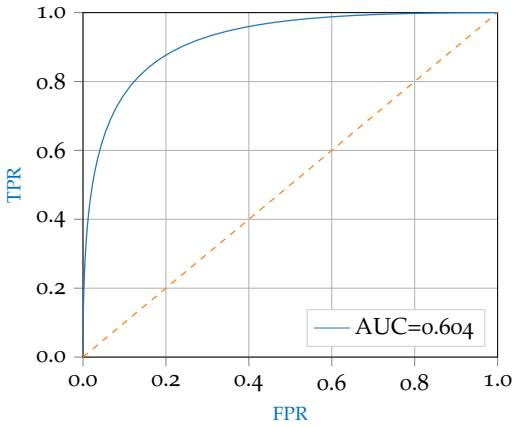


Figure 10: An example of a ROC curve.

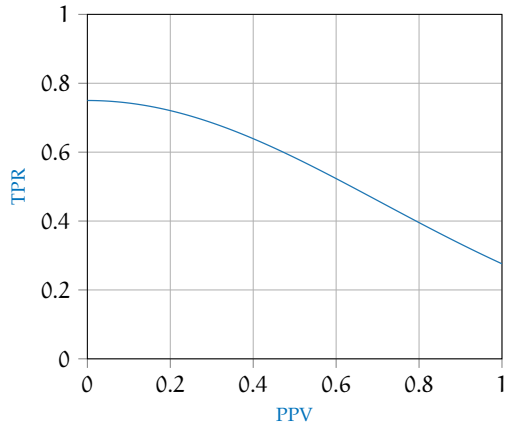


Figure 11: An example of a PR curve.

4.3.4 *Validation Techniques*

Several techniques are available to validate the results of an ADS and compute the different performance measures. We present a few examples in the following.

HOLDOUT [80]: is the simplest approach to evaluate the performance of an ADS. We split the data set into two non overlapping subsets: a training set and a test set. We learn the model based on the training set. Then, we apply the model to the test set and compute the performance measures. This way, we make sure that the model does not over-fit the training set. This approach however, can be biased if the training and test set have different distributions and thus are not representative of the overall data. For this reason, one should apply stratification, which is splitting data sets into subsets with the same probability distribution.

CROSS VALIDATION [80]: is a well known technique to validate ML models. It consists in splitting the data set into k folds. Then, each time, we use $k-1$ folds for the training and the remaining fold for the test. For each iteration, we compute the performance measures. The overall performance measures are the averages of the performance measures of all the iterations. Cross validation reduces the bias of performance measures estimation. To have a more accurate evaluation, one may use stratified cross validation which allows to have k subsets with the same probability distribution.

BOOTSTRAPPING [80]: consists in randomly selecting training instances with replacement. The remaining instances that were not included in the training set form the test set. In this technique, the size of the test set changes depending on the number of duplicates, triplicates, etc in the training set. This process is repeated and each time, one computes the performance metrics. As for cross validation, the overall performance metrics are the averages of the performance metrics of all iterations.

4.4 SUMMARY

In this chapter, we have discussed some of the different approaches the researchers have applied to address the question of anomaly detection. As shown in Table 3, many solutions with different characteristics have been proposed. Depending on the context, anomaly detection is applied to different data structures and returns different outputs. Learning may be supervised or unsupervised depending on the use case and the availability of labeled training data. Some of the algorithms process data in a simple and intuitive way which makes the results easy to control and interpret. Others are more complex and less manageable. The time complexities of the learning phase and the detection phases vary drastically from one algorithm to another. In some fields, the training phase is executed very often. Therefore, one should select an algorithm with low training computation needs. In other cases, especially in the mobile networking field, the algorithm should detect anomalies in real time. So, the detection time should be bounded. The algorithms complexity depends not only on the data size and dimensionality but also on the algorithm parameters. Some parameters, like the number of epochs (iterations) in the case of neural networks and EM algorithm may increase drastically

the complexity of the process. The accuracy aspect is not presented in Table 3 as the accuracy of an algorithm depends tightly on the statistical characteristics of data. An algorithm may have a high accuracy in the case of a specific data distribution and perform poorly in another. The thresholding and the configuration aspect in general are to take into account when selecting an anomaly detection solution. Generally speaking, there is not an optimal configuration that fits all data types types and use cases.

Approach	Algorithm	Input	Output	Supervised learning	Interpretable results	Complexity		
						Learning	Detection	Ref
Regression	ARIMA	Time series	Binary label	No	Yes	$O(m.n^2)$	$O(m.p)$	[21]
	RNN LSTM				No	$O(n_e.m.n.(n_{l_1}+n_{l_2}+...))$	$O(m.n_{l_1}+n_{l_1}.n_{l_2}+...)$	[14]
Classification	Decision Trees	Numerical/ categorial	Binary/ multi-class label	Yes	Yes	$O(n^2.m)$	$O(m)$	[133]
	Random Forest				No	$O(n^2.m.n_t)$	$O(m.n_t)$	
	Bayesian Networks				Yes	$O(n.m)$	$O(m)$	[57]
	Neural Networks				No	$O(n_e.m.n.(n_{l_1}+n_{l_2}+...))$	$O(m.n_{l_1}+n_{l_1}.n_{l_2}+...)$	[14]
	SVM	Numerical			Yes	$O(n^2.m+n^3)$	$O(m.n_{sv})$	[26]
Clustering	K-means/modes	Numerical/ categorial	Unlabeled classification	No	Yes	$O(m.n^2)$	$O(m.n_k)$	[111]
	Hierarchical Clustering					$O(m.n^3)$	$O(m.log(n))$	[30]
	EM					$O(m.n.n_k.n_e)$	$O(m.n_k)$	[139]
	DBSCAN					$O(m.n^2)$	$O(m.n)$	[140]
Sequential Analysis	SAX	Time series	Binary label	No	Yes	$O(m.n^2.n_s)$	$O(m.n_s)$	[70]
	Markov Models					$O(n.m)$	$O(n_s.m)$	[118]
	HMM			Yes		$O(n^2.n_s.m)$	$O(n_s.m)$	[20]
Spectral Analysis	FFT	Time series	Binary label	No	No	$O(m.n.log(n))$	$O(m.n)$	[33]
	DWT					$O(n.m)$	$O(n.m)$	[52]
	PCA	Numerical				$O(m^2.n+m^3)$	$O(m)$	[28]
	Hough Transform					$O(n.m)$	$O(m)$	[39]
Information Theory	Entropy	Numerical/ categorial	Binary label	No	No	$O(n.m)$	$O(m)$	[141]
Rule Induction	Association rules	Categorial	Score	Yes	Yes	$O(2^m.n)$	$O(2^m)$	[145]

Table 3: Comparison of anomaly detection approaches: n: number of training samples, m: number of features, p: order of AR, n_t: number of trees, n_{sv}: number of support vectors, n_k: number of clusters/components, n_l: number of layers ,n_{l_i}: number of nodes in layer i, n_e: number of epochs, n_s: length of the sequences. For algorithms applicable on uni-variate data (individual features), the complexity is multiplied by m.

5

PROPOSED ALGORITHM: WAD

In this chapter, we present a solution to detect anomalies in monitoring systems: Watchmen Anomaly Detection (WAD). We start with stating the problem of detecting anomalies in periodic time series created by a monitoring system. Then we clarify the objective of this work which is having an autonomous real time solution with low error rate. In Section 5.2, we explain the mathematical basis and the architecture of our solution. Then, we demonstrate the validity and robustness of our solution and compare it with some reference techniques. We conclude with our main results in the lab as well as in live networks. These results confirm that our algorithm fills the accuracy requirement. We close this chapter with a brief discussion of future work.

5.1 INTRODUCTION

5.1.1 *Motivation*

Mobile operators exploit Anomaly Detection Systems (ADSs) in their cellular network to detect and report any network anomaly as fast as possible. Since mobile networks are getting larger and more complex, the ADSs have also become more prone to faults, which eventually lead to bad network management and to a potential Quality of Experience (QoE) degradation for the customers. Monitoring the network monitoring system is thus a critical task, in order to ease network troubleshooting tasks.

Our goal is to assist network administrators in the task of investigating failures. Since both the network and the monitoring system are complex, determining whether the root cause of a reported anomaly is localized in the monitoring system or in the cellular network is not straightforward. The investigation task is hard and time consuming. Furthermore, supervising the inputs/outputs of monitoring devices (*e.g.*, probes) helps not only to address monitoring system issues but also to identify the faulty network equipment. For example, detecting an anomaly in the input of a probe that is connected to an Mobility Management Entity (MME) can suggest that the MME is faulty.

5.1.2 *Objective*

In this project, we aim to create an ad-hoc anomaly detection solution that detects real time issues in mobile network monitoring systems in a dynamic manner. The goal of WAD is to detect anomalies in periodic time series such as peaks, dips and sudden rises and falls. Those anomalies emerge from erroneously configured devices, broken equipment (resulting in traffic loss), and atypical events (e.g., traffic increase due to a major sporting event), just to name a few. Figure 12 shows the types of anomalies that we want to detect.

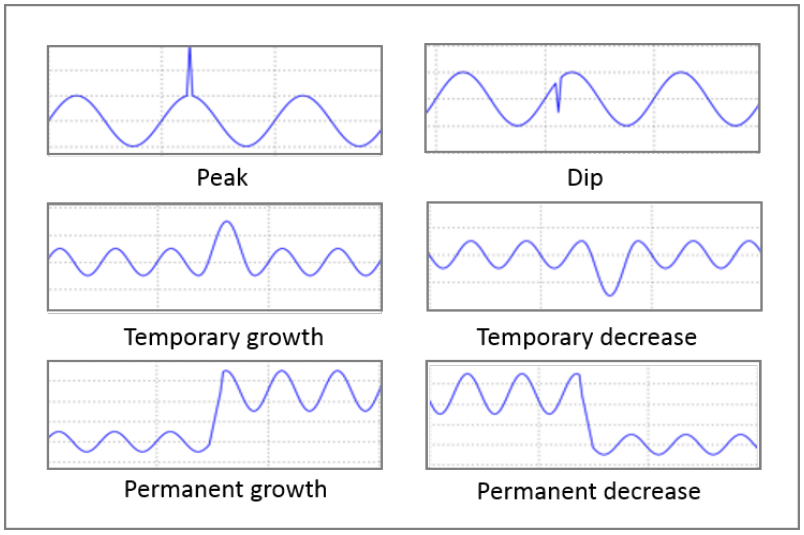


Figure 12: The different anomaly types

5.1.3 *Requirements*

The proposed solution has to meet a number of requirements. First, it has to be practical. The solution should detect anomalies in the mobile network metrics. The overall approach should be easy and intuitive to be easily managed and used by end users. Second, unlike threshold based approaches, the solution should handle data periodicity and detect anomalies based on repetitive patterns. Moreover, the model should be dynamically adjusted to embody the natural traffic evolution. The solution should also be pro-active and detect anomalies in real time. Furthermore, it should be unsupervised (work on unlabeled data) and should not require post-deployment effort. Its configuration should be easy. In other terms, it should have few parameters that can be easily understood and modified by the end user. Most importantly, the solution should have low error rate. It has to detect major anomalies detection have few false positives. Last but not least, the solution should have few computation needs. The learning phase should have low complexity. The detection task should have a constant time complexity to guarantee real time response.

5.1.4 *Data types*

The WAD algorithm detects anomalies in a cellular network monitoring system, by analyzing the traffic generated by the monitoring system itself. We illustrate the overall system in Figure 13. Each monitored entity generates Call Data Records (CDRs), which are sent to the *monitoring system* module of the ADS, and Record Generation Reports (RGRs) to the WAD. RGRs are WAD reports generated from monitor system metrics. RGRs include the number of CDRs, TCP Data Records (TCPDRs), *tuples* (User Plane traffic counters), the input/output rate of probes, and the sessions/bearer deciphering rate. The stream of data logs is a univariate discrete time series with time index t. Since the data come from the network probes, they are highly correlated with the behavior of the population of

subscribers (*e.g.*, they show a daily pattern). In the remainder of the paper we use a period of one day but WAD works with any period length.

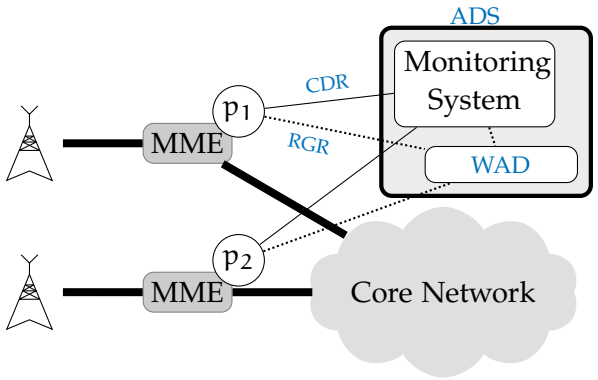


Figure 13: Overall Architecture: two probes p_1 and p_2 monitor MMEs and send reports to the ADS

5.2 THE WATCHMEN ANOMALY DETECTION ALGORITHM

Watchmen Anomaly Detection (WAD) is an ad-hoc solution to the network anomaly detection problem. It is dynamic and adapts to the natural traffic evolution. It runs in real time on data coming from monitoring devices, such as network probes measuring traffic throughput in network interfaces. Based on these measurements (time series), the ADS creates reference patterns describing data in normal state. Then it measures the gap between the reference pattern and real time data. If the gap is large, the network administrator is notified about an anomaly occurring in the related equipment.

WAD consists of two phases: a learning phase and a detection phase. The learning part of the algorithm is executed once a day (i.e., the period of the data). In this phase, we create a reference model, which is stored in a database. This reference model is then used in the detection phase, which runs in real time. Before running the algorithm, there is a pre-processing step to fill missing values and have the same time interval between consecutive samples. To do so, we apply a simple linear interpolation.

In the following, we will use the metric in Figure 14 to explain the different steps of WAD. As in the real world process, we will use the 29 first days to build the model and will apply the detection procedure to the 30^{th} day.

5.2.1 *Learning Phase*

Figure 15 shows the four main steps of the learning phase, whose input is one month worth of data for each metric (time series). Note that WAD handles each metric independently of the others: if we use 40 metrics, we run the learning phase 40 times to produce 40 reference models.

5.2.1.1 *Periodicity Check*

We compute the Fourier transform of the time series to check whether the data is periodic, in which case there is dominant frequency (equal to the inverse of the period)

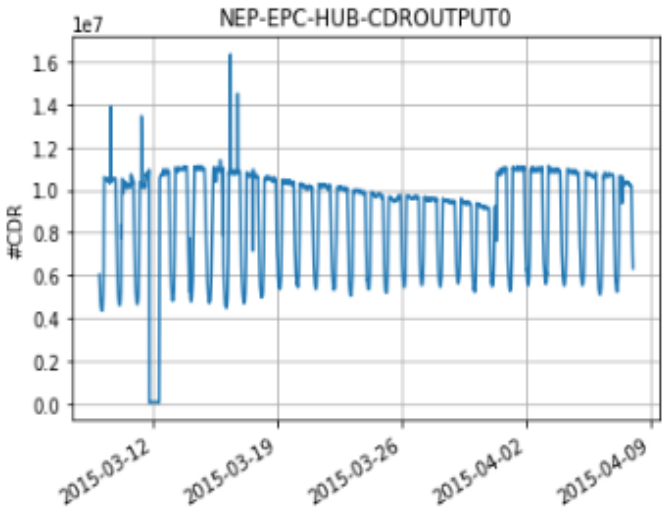


Figure 14: The CDR number at an output interface of a hub.

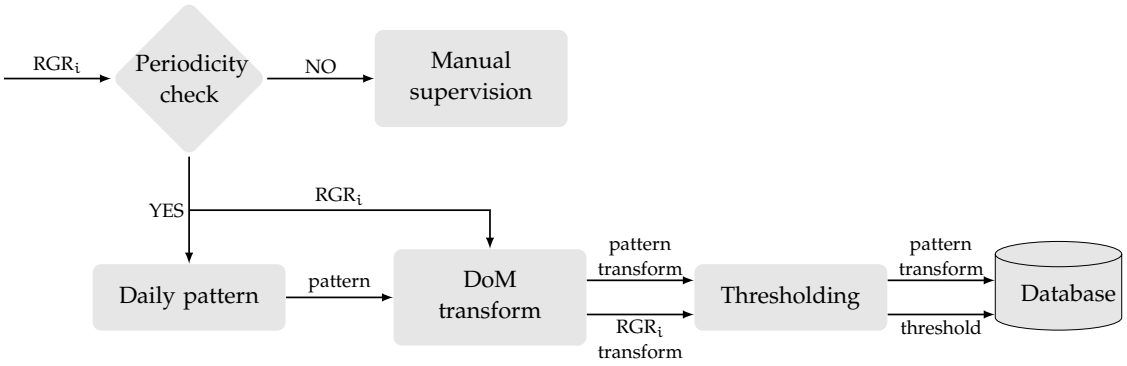


Figure 15: The learning phase for a metric RGR$_i$

in the spectrum. To verify this, we calculate the ratio between the sum of the components close to the dominant frequency and the sum of the rest of the components (high frequencies/noise). The ratio should be greater than one otherwise the noise masks the periodicity. In practice, after empirical testing, we took five as a lower bound of the ratio. Figure 16 shows that the metric in Figure 14 has a highly daily periodicity ($1\mathtt{day} \leftrightarrow 1.157 \times 10^{-5}$ Hz).

If a given metric is not periodic, we ignore it for a day. This metric will be checked manually by the network administrators. This can happen on freshly installed systems that are converging to a steady state.

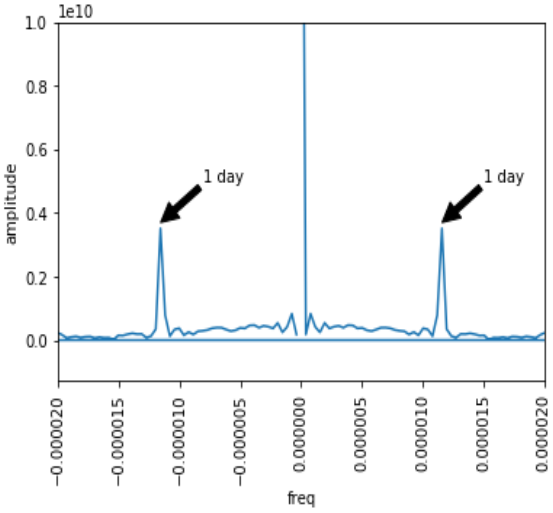


Figure 16: The FFT of the metric in Figure 14.

5.2.1.2 *Daily Pattern*

In order to obtain the anomaly free behavior, we compute a daily pattern for each metric. To do so, we split our training set into periods and calculate the average value on each point of the period. In practice, we have one sample every 15 minutes. We compute the average for all the days of the month at that time. This way, we get a rough pattern describing the average evolution of the metric.

We then calculate the Euclidean distances between the average vector and the periods of the training set. We sort these distances and discard all the periods whose distances to the average vector are greater than the 95th percentile. This way, we rebuilt a new training set with no extreme values. Afterwards, we calculate the average vector of all the periods of the anomaly-free training set. This vector is the metric daily pattern. Figure 17 shows the rough pattern in orange and the anomaly free pattern after removing extreme values. The anomaly free pattern has smoother variations than the rough pattern.

5.2.1.3 *Difference over Minimum (DoM) Transform*

The data we use for the model generation have to be smooth, since discontinuities are considered to be anomalies. For this reason, we need a transform that amplifies jumps

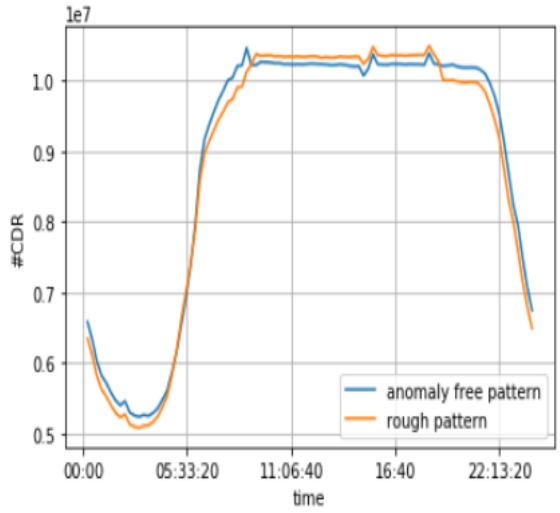


Figure 17: Daily pattern:
Rough pattern and anomly free pattern after removing extreme values.

and minimizes smooth variations. To this end, we define the DoM transform (T) of a function f as:

$$T(f, L) = d\left(f\left(t\right), f\left(t - L\right)\right) \tag{17}$$

where d is defined as:

$$d(x, y) = \frac{x - y}{\min(x, y)} \tag{18}$$

and $L > 0$ is the order of the transform (the lag between the two samples of f). This transform quantifies the jumps in f after L time steps, starting from a given time t. In the remainder of this paper, we focus on the transform of first order since it acts like a derivative and it features that it goes to infinity when there is a peak (or a dip) and it sticks to small values otherwise. Figure 18 shows the DoM transform applied to the metric in Figure 14.

In this phase, we calculate the DoM transform of the metric and the DoM transform of its daily pattern.

5.2.1.4 *Thresholding*

At this point of the algorithm, we measure the distance between the transformed value of each sample of the metric and the corresponding (with regard to the time of the day) transformed value of the pattern. The histogram of distances in Figure 19 shows that most of the distances between the transformed training data and the transformed pattern are close to 0 as expected. We sort these distances and we set the threshold to the 99th percentile.

We store the threshold and the DoM transform of the daily pattern in the database for real time detection phase.

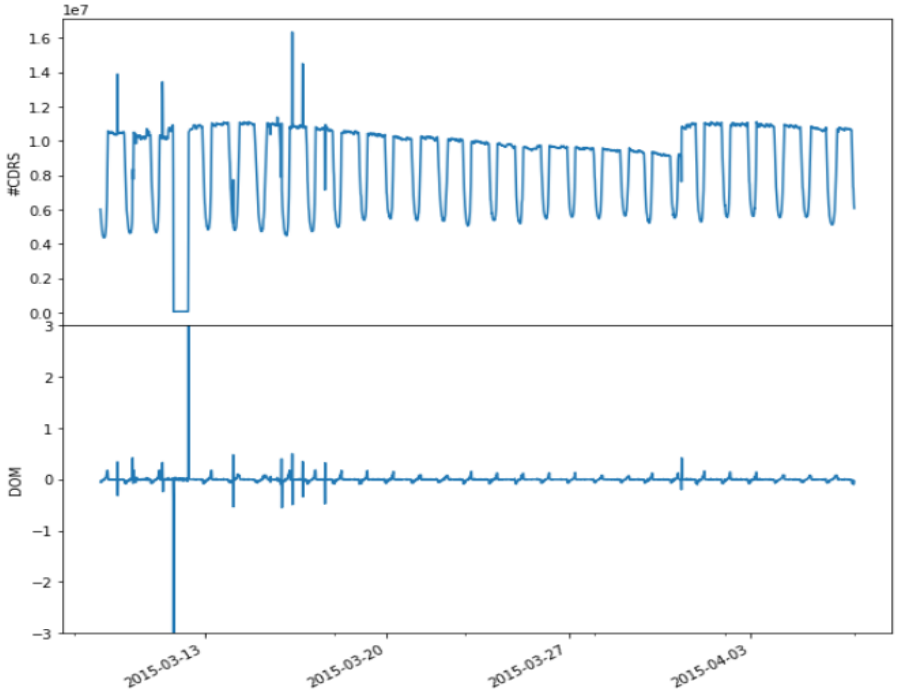


Figure 18: DoM transform applied to the metric in Figure 14.

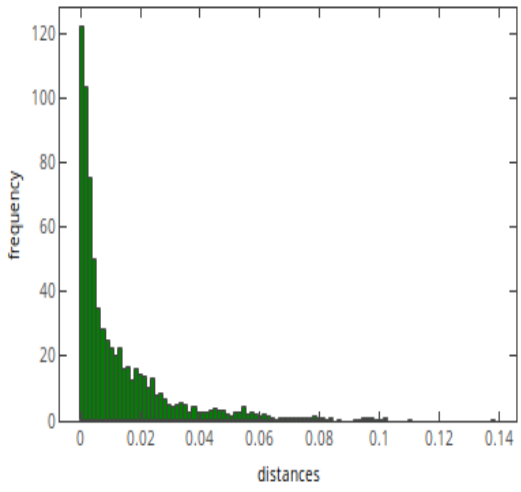


Figure 19: The histogram of distances between the transformed training data and the transformed pattern.

5.2.2 *Detection Phase*

Figure 20 shows the three main steps of the detection phase, which runs in real time and compares the incoming sample to the reference model constructed in the previous phase. As in the case of the learning phase, this phase runs independently for each metric.

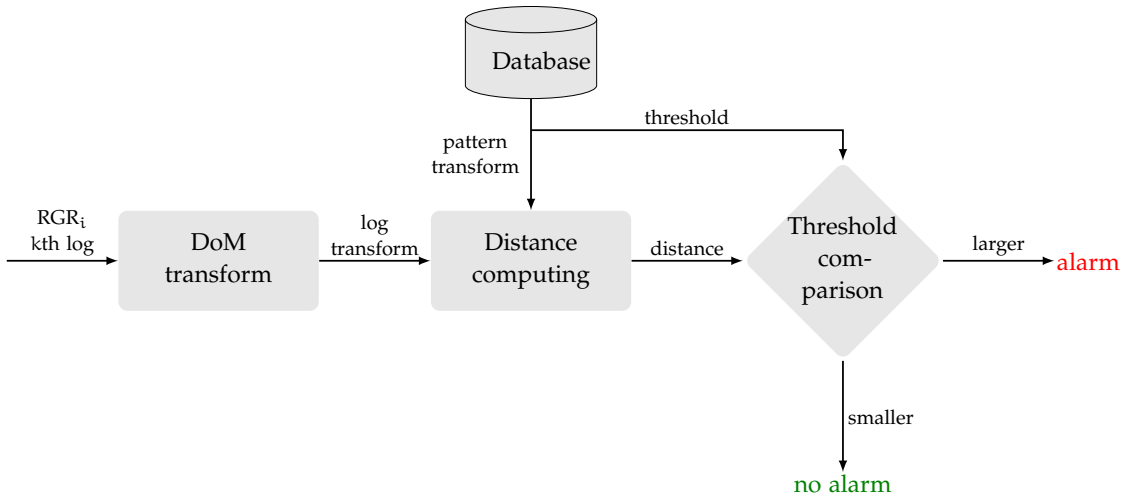


Figure 20: Detection phase for the kth sample of the RGR$_i$.
Detection phase for the kth sample of the RGR$_i$.

After computing the DoM transform of a new sample, we calculate the Euclidean distance to the reference pattern for the same time of the same day, which is stored in the database since the learning phase.

If the distance is larger than the threshold, plus a tolerance band, WAD concludes that there is an anomaly and it raises an alarm. The tolerance band reduces the number of false positives, as there can be occasional fluctuations that cross the threshold without being anomalies. For the time being, we set the tolerance band based on empirical observations but we are working on computing its value during the learning phase. Figure 21 shows how based on the 29 days of training data we can detect a dip on the 30^{th} day.

5.3 VALIDATION TESTS

The goal of WAD is to be a lightweight solution to detect anomalies in the monitoring system of a cellular network. We report in the following some of our main findings after a series of evaluation, first in the lab based on real data traces, and then after we installed WAD in the production network of two operators.

5.3.1 *Parameters Setting*

As mentioned in Section 5.2, we set the lag parameter to one. We use the 95th percentile for the pattern generation and the 99th percentile for the threshold calculation. The only parameter we need to tune is the tolerance term. We ran empirical tests in the lab on

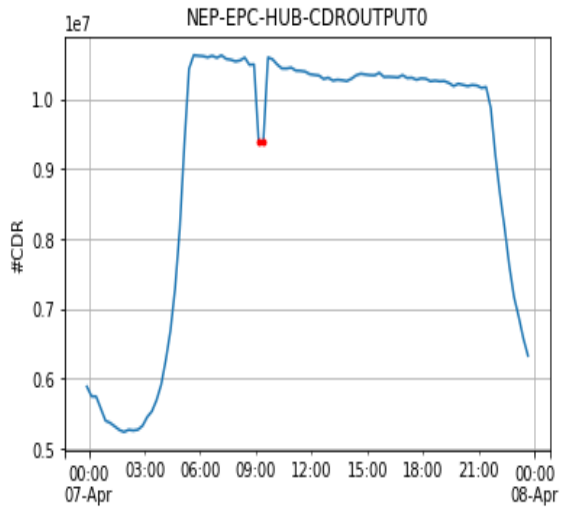


Figure 21: Detection of two anomalies in the 30^{th} day of the metric in Figure 14.

data provided by a mobile operator and set the error term to 0.1, which means that we tolerate a raise (or fall) of 10% around the 99th percentile. For both operators, the same value of the tolerance has given satisfactory results. These latter will be detailed in 5.3.2.2.

5.3.2 *Accuracy*

5.3.2.1 *Lab results*

We tested WAD in the lab based on historical data from an European operator. We worked with the values of 58 metrics over a month. The total number of samples is equal to 167 040 with 90 anomalies. We compared our algorithm with two other solutions suitable for our use case: Symbolic Aggregate Approximation (SAX) and Principal Component Analysis (PCA). For SAX, we used an alphabet size of 15, a word size of 4 and a maximum number of abnormal words occurrence to 2 [70]. Figure 23 shows how the values of the metric in Figure 14 are approximated by a Gaussian distribution and then converted into equiprobable symbols. The histogram demonstrates that the gaussian is not a good approximation of this metric.

We used the PCA decomposition with original space dimension equal to 4 and projection space dimension equal to 4. We detected anomalies on the 4^{th} component and we used a threshold equal to the 99.5^{th} percentile of the data [81]. Figure 22 shows the results of the PCA decomposition of the metric in Figure 14. The four components represent the projection of this metric on the basis defined by the PCA. The components are ordered by decreasing variance. As one may notice, the anomalies are located in the 3^{rd} and 4^{th} (the noise components).

We have used the recommended parameters for each algorithm and we have tried a few other values without obtaining any significant improvement in the results.

We evaluated the results in terms of True Positive (TP), False Positive (FP) and False Negative (FN). The TP are real anomalies detected by the system. The FP are normal

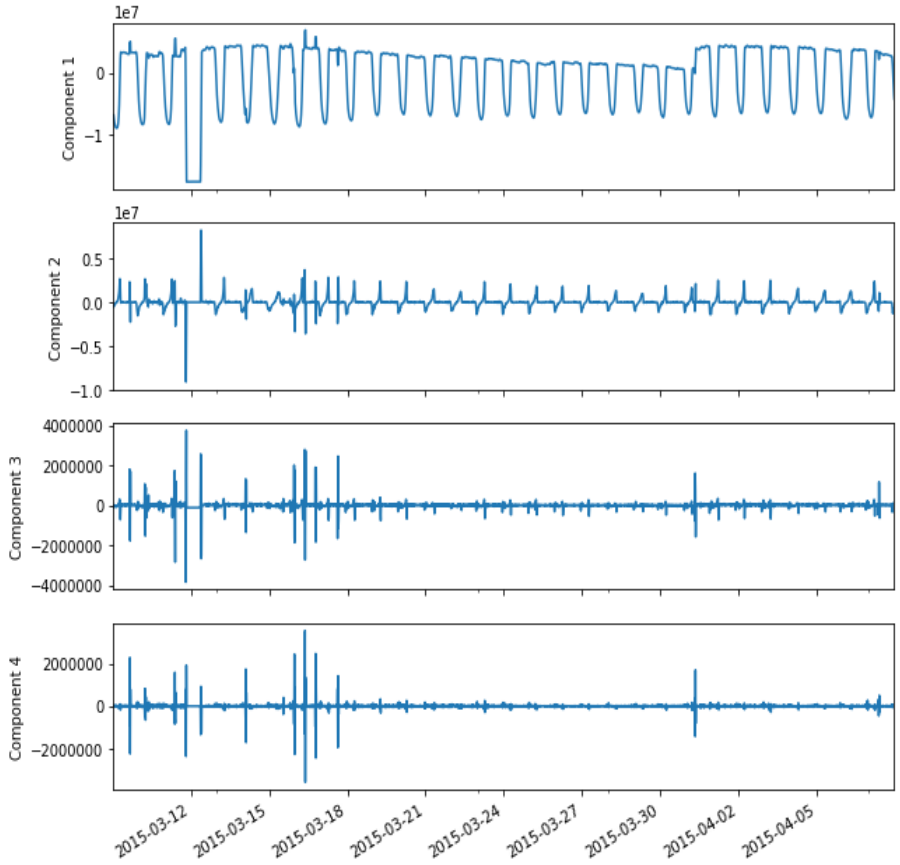


Figure 22: PCA decomposition applied to the metric in Figure 14.

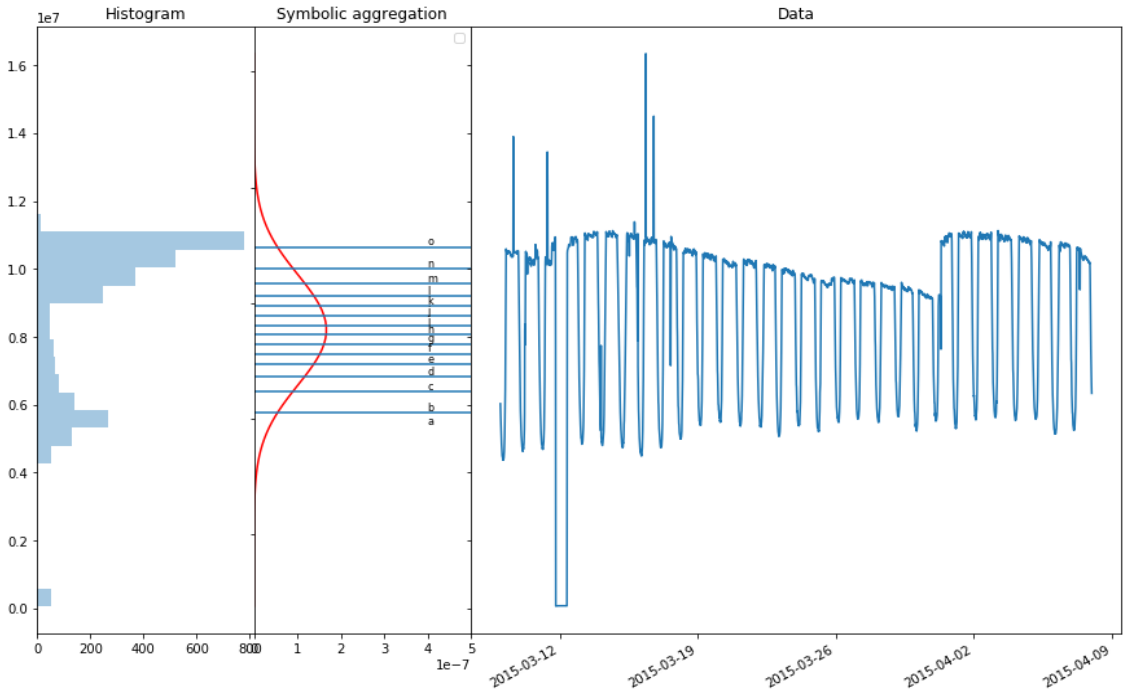


Figure 23: SAX applied to the metric in Figure 14:
The histogram, the guassian approximation and the time series.

points erroneously labeled as anomalies. The FN are anomalies not detected by the system. We also measured the sensitivity, which is the ratio of correctly detected anomalies to the total number of anomalies, and the specificity, which is the ratio of points correctly identified as normal to the total number of normal points. Table 4 shows the results of these tests.

Algorithm	TP	FP	FN	Sensitivity	Specificity
SAX	59	6	31	0.65	0.99
PCA	85	85	5	0.94	0.99
WAD	81	9	9	0.90	0.99

Table 4: Lab Evaluation Results

The results show that the PCA decomposition is a good method to detect abrupt changes. However, it does not fit all types of data. For example, PCA supposes that the noise and anomalies are linearly separable from significant data, although it is not always the case. Another problem with PCA-based anomaly detection is that including anomalies in the noise component implies a high FPs rate. Another source of uncertainty lies in the fact that the number of noise components depends highly on the metric. When we run our tests, we have noticed that for different metrics in the same data set, anomalies are not in the same component. As previously said, we used a 4-dimensional projection space. For some metrics, we detected anomalies only in the third component, for some other metrics in the fourth component and for other different metrics in both the third and the fourth components. This implies that, in order to have accurate results, one needs the intervention of an expert, who can analyze the data, for each metric, to determine which component contains the anomalies. Such a solution is not feasible in a production environment.

The SAX algorithm obtains a high FN number since SAX does not keep track of time. Thus, it considers some behaviors as being normal although they abnormally took place at an unusual time (e.g., a peak at 5am while usually there is not peak at that time). Our algorithm offers a good trade-off between FPs and FNs rates.

Figures 24,25 and 26 show the Precision-Recall (PR) curves of SAX, PCA and WAD created using different parameters applied to five metrics selected by experts: a metric with no anomalies, two metrics with few anomalies and two metrics with many anomalies. These curves confirm that SAX has a highest precision , PCA has a highest recall and WAD give a fair trade off between the two.

5.3.2.2 *Live network results*

We install WAD during three months to supervise the network of two mobile operators (in Europe and Africa). The first operator gave a general positive feedback, without giving specific numbers. The second one gave us access to a detailed feedback. In the remainder of this section, we will refer to these results. Figure 27 shows an example of results given by WAD on the number of *tuples* at the output of a probe. The dots here are the anomalies detected by WAD.

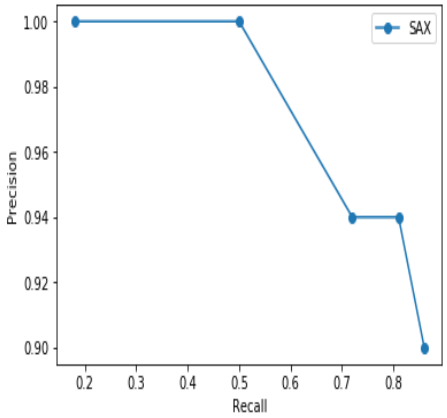


Figure 24: The PR curve of SAX results.

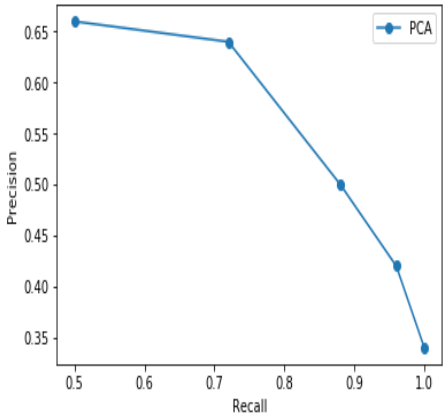


Figure 25: The PR curve of PCA results.

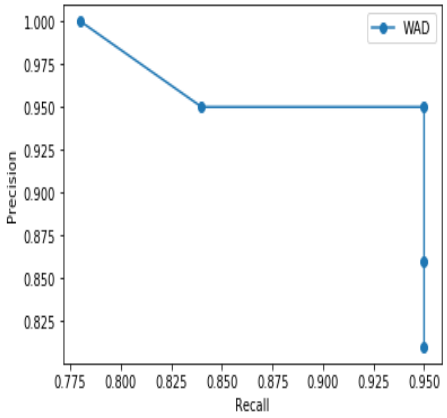


Figure 26: The PR curve of WAD results.

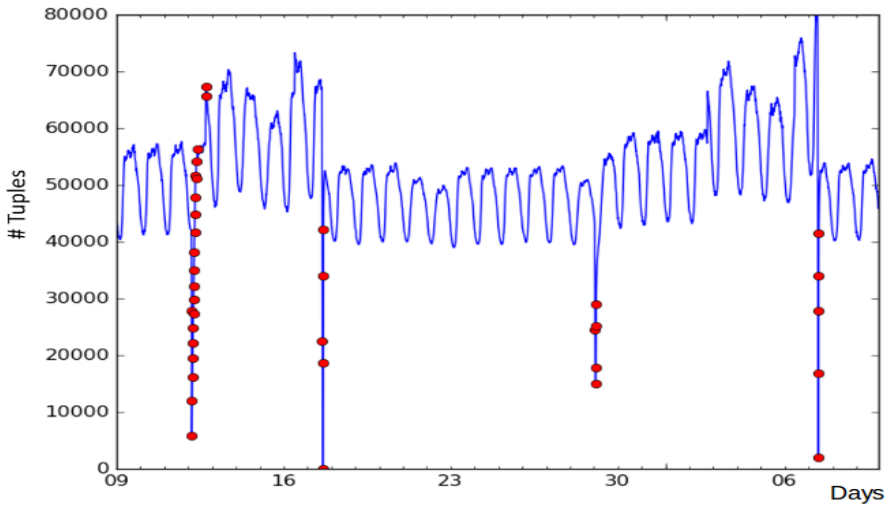


Figure 27: Anomalies in the number of Tuples – probe output

The network administrator selected 18 metrics and provided feedback about the performance of our algorithm. During the three months, 83 anomalies occured (among 155 520 samples). We detected 75 of them, which means we had eight false negatives. We also detected four false anomalies. The sensitivity of the system is equal to 0.90 and the specificity is equal to 0.99. The accuracy of WAD meets the requirements of both operators.

5.3.3 *Memory and Computation Needs*

The step of periodicity check has a complexity of $O(n\log(n))$ since it is based on FFT [33]. Each of the remaining steps of the training phase (pattern creation, DoM transform and thresholding) has a $O(n)$ complexity. Thus, WAD has a $O(n\log(n))$ complexity, as the FFT is the term with the dominant time complexity. We have evaluated WAD hardware performance on a machine with the following features: 4 Cores, 4GB RAM, 30GB hard drive. We have noticed a peak once a day in the CPU utilization related to the learning phase. The training phase takes about 30 minutes to process 850MB of data (we monitor 55 metrics, we have one month of data per metric with one sample every 15 minutes). Tables 5 and 6 show the results of these tests.

	Median	**95th percentile**	**Max**
Learning Phase	12%	14%	25%
Detection Phase	0.1%	1%	2%

Table 5: WAD CPU Usage

Based on these experiments and on the successful deployment in production networks, we conclude that WAD has modest requirements in terms of memory and computing power, even during the learning phase.

	Median	95th percentile	Max
Learning Phase	1.2%	1.5%	1.6%
Detection Phase	0.5%	0.8%	0.9%

Table 6: WAD Memory Usage

5.3.4 *Discussion*

WAD gives promising results in terms of gain of productivity, time saving, and ease of outage troubleshooting. After using WAD for a few months, the network administrators of two cellular operators have highlighted the following aspects. They have noticed that they do not have to check all the metrics manually, since they are alerted in real time. They also save a lot of time and effort in troubleshooting and understanding issues: our algorithm helps them localize the problem and therefore fix broken devices and/or configurations efficiently. The configuration of our solution is straightforward since it produces dynamic models (updated automatically). It does not classify days into working days and holidays, therefore there is no need for calendar integration. Finally, as an added benefit, it does not require a large training set.

To sum up, automatic supervision of network monitoring systems and real time feedbacks processing is crucial to automatize mobile network troubleshooting. Thus, integrating an ADS based on Machine Learning techniques like WAD within a network monitor system seems to be an essential task for the creation of self healing networks. Our research may contribute to the definition of basic functions for self-organizing and more resilient networks.

6 INDUSTRIALIZATION

Watchmen Anomaly Detection (WAD) can be applied to different types of data in different contexts. In this Chapter, we detail how we industrialized WAD to achieve two different objectives. The first objective is to automatically monitor the monitoring system in order to ease the network administrators tasks. The second objective is to help telecommunication experts to carry out service quality management.

6.1 MONITORING THE MONITORING SYSTEM: NOVA COCKPIT™

In the following, we go through the scoping phase and the deployment phase of integrating WAD in Nova Cockpit™.

6.1.1 *The scoping phase*

Our goal is to implement the WAD algorithm to monitor the monitoring system by integrating it as a plugin of the monitoring tool: Nova Cockpit™. We present here the scope of the project starting by the development environment which is the Cockpit™ framework. Then we detail the accessible data for WAD application. We finish by stating the requirements and constraints of this project.

6.1.1.1 *Environment*

Cockpit™ is a software designed to monitor EXFO monitoring solution. Figure 28 shows how it collects monitoring metrics that are aggregated measurements of the input/output data of probes and other monitoring devices. Network administrators analyze these metrics during their daily checks to find issues within the monitoring solution.

Cockpit™ offers straightforward monitoring interfaces containing human readable information status. It processes data in real time, centralizes information and creates interactive graphics. It provides alerting services for probes, applications, switches, etc. It also offers network service monitoring (e.g., for Simple Mail Transfer Protocol (SMTP), File Transfer Protocol (FTP)) and host monitoring (e.g., Central Processing Unit (CPU) load, disk usage).

In this project, we aim to add an anomaly detection plugin based on WAD as a feature of Cockpit™ to monitor the EXFO monitoring system. The plugin has to analyse Cockpit™ data to detect issues in real time. The plugin will inter-operate with the different components of Cockpit™.

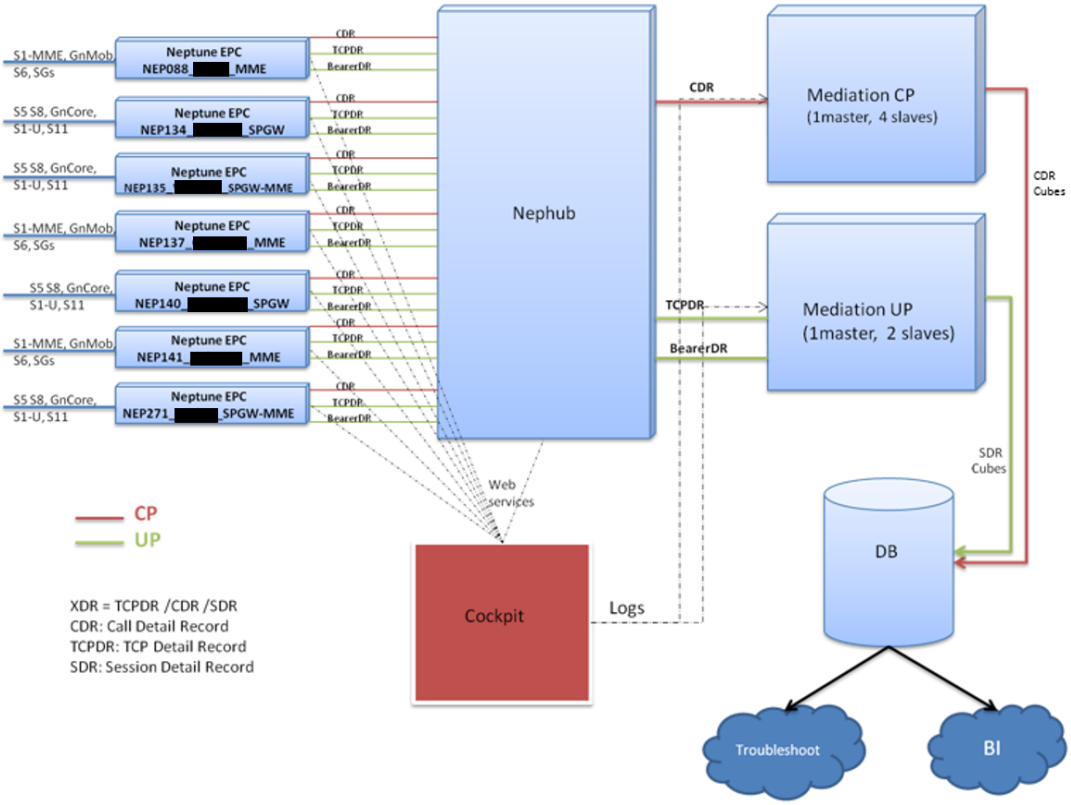


Figure 28: EXFO monitoring system.
Cockpit™ collects data from Neptune™ probes, Nephub™, and Mediation™ interfaces.

6.1.1.2 *Accessible Data*

The anomaly detection plugin has to analyze the data collected by Cockpit™ from EXFO monitoring solution. As Figure 28 shows, the monitoring solution has three main components:

NEPTUNE™ is a passive high capacity probe that captures the traffic for multiple applications such as network optimization, customer care and network management and supervision. A Neptune™ probe captures and processes huge volumes of Control Plane (CP) and User Plane (UP) traffic in real-time. It generates Call Data Records (CDRs) and TCP Data Records (TCPDRs) for key interfaces of core network sites such as Mobility Management Entitys (MMEs), Serving GPRS Support Nodes (SGSNs) and Gateway GPRS Support Nodes (GGSNs).

NEPHUB™ is a hub that centralizes configurations for up to 16 Neptune™ probes. It also aggregates CDRs and TCPDRs.

MEDIATION™ is a software component that processes, enriches and aggregates data into cubes (aggregated counters). It also provides multidimensional Key Performance Indicators (KPIs). Nova Mediation™ is designed for Business Intelligence and Big Data processing aims.

The Nagios scheduler runs services to pull data regularly from the different components. It queries the components to get the metrics needed to supervise the monitoring system. Table 7 details the list of metrics at each interface of the monitoring system. Our goal, in this project, is to apply anomaly detection to these metrics to find issues in the monitoring system.

6.1.1.3 *Objectives*

The main goal of this project is to detect anomalies in the devices monitored by EXFO solution. The deployed plugin should process data and alert users in real time. It has to have a low error rate. The plugin should not interfere with Cockpit™ functions and should not impact Cockpit™ computational performance. The configuration tasks such as adding new metrics should be straightforward. The plugin should also respect Cockpit™ plugin architecture in order to ease maintenance operations.

6.1.2 *The deployment phase*

The anomaly detection plugin has been deployed in two Cockpit™ instances installed in two different networks for two years. The plugin has given promising results in terms of gain of productivity, time saving, and ease of outage troubleshooting. More than 100 metrics were supervised on each network. The overall results are satisfactory. The detection of false positives and false negatives is relatively rare and no precision issue was signaled by network administrators. The computational complexity and the CPU and memory needs are low. The anomaly detection plugin did not have any visible impact on Cockpit™ computational performance. The initial parameters were well tuned. They have been changed only once in one of the networks and not modified in the other.

Monitoring component	Metrics
Neptune™	Input Rate: Average input rate (Mbps) Output Rate (CDR+ TCPDR+ BDR): Average output rate (Mbps)
Nephub™	CDR Input: Number of CDR received per Neptune™ (not cumulative) CDR Output: Number of CDR sent to each Mediation™ (not cumulative) TCPDR Input: Number of TCPDR received per Neptune™ (not cumulative) TCPDR Output: Number of TCPDR sent to each Mediation™ (not cumulative)
Mediation™ UP	CDR Input: Number of CDR received per interface CDR Output: Number of CDR inserted in the counter database (per interface)
Mediation™ CP	TCPDR Input: Number of TCPDR received per interface SDR Output: Number of SDR inserted in the counter database (per interface)

Table 7: Cockpit™ Metrics by monitoring component.

Figures 29 and 30 are examples of graphs generated by Cockpit™ anomaly detection plugin. The blue envelope is the pattern plus/minus the threshold (the maximum acceptable distance to the pattern). In the two graphs, it is obvious that the envelope is adapted dynamically to the data which confirms that the algorithm is learning from new data.

Figure 29 is a plot of the input rate of a Neptune™ probe captured in September 2016. It shows detected dips related to maintenance and configuration issues. The traffic growth is detected at end of the graph is due to a new network configuration. Figure 29 is a plot of the number of CDRs at the interface between the Nephub™ and its first connected Neptune™ probe captured in November 2017. The graph shows a detected peak caused by the change to the winter time in the corresponding country and a dip related to a maintenance operation.

6.1.3 *Sample Use Case*

We now describe a specific incident to highlight the efficiency and usefulness of the Cockpit™ anomaly detection plugin. The network administrators supervise only a few metrics as the number of metrics is large. They also focus on constant metrics as detecting problems on them is easier than in periodic metrics. In September 2016, during a daily check, a network administrator detected an important decrease in the deciphering rate which has to be constant around 100%. The graph of the deciphering rate is shown in Figure 31. The anomaly detection plugin had detected an anomaly in the input of

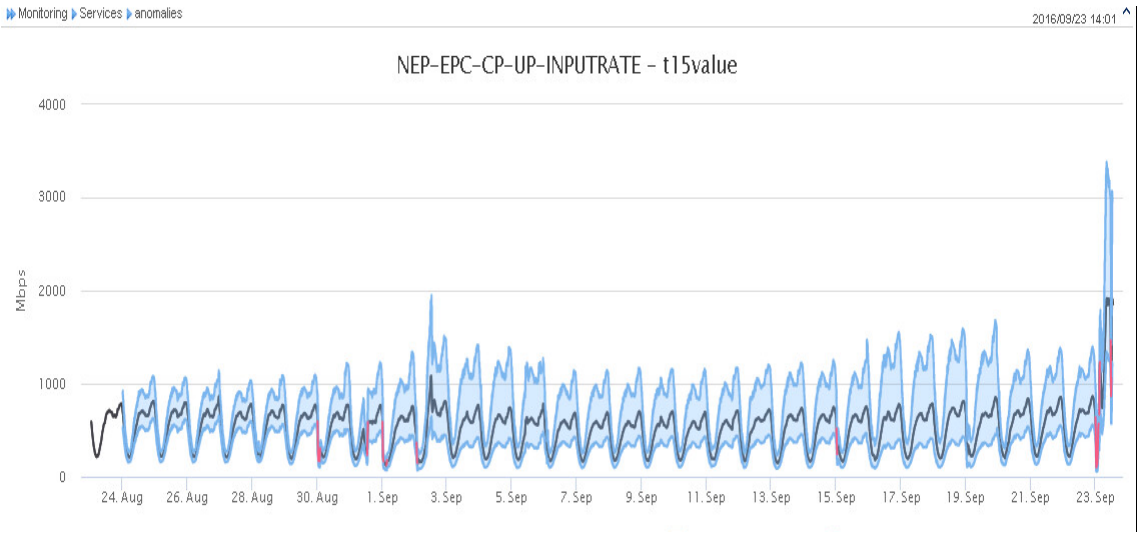


Figure 29: Cockpit™ capture of a Neptune™ probe Input Rate.
The envelope of acceptable values is in blue and the detected anomalies in red.

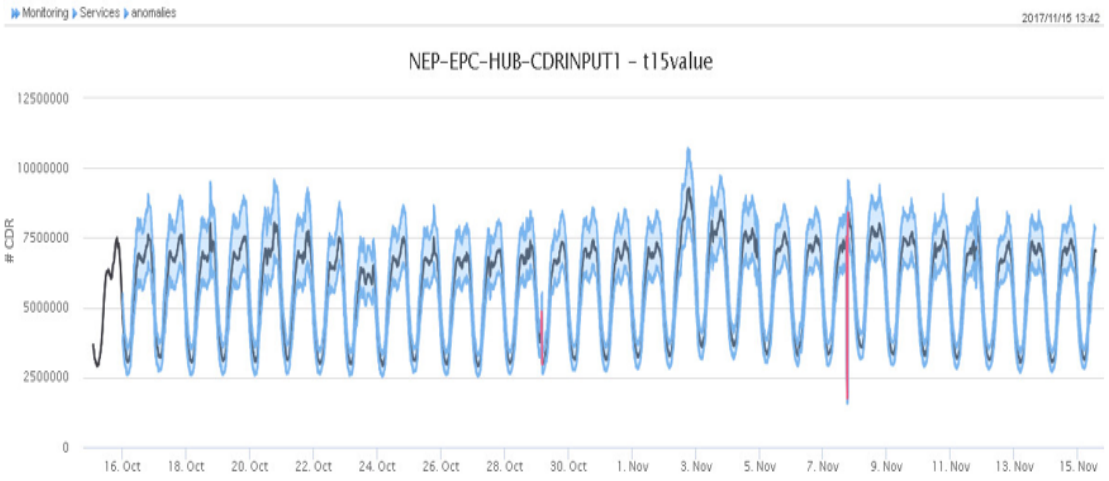


Figure 30: Cockpit™ capture of Nephub™ CDR
Input, at the interface between a Neptune™ probe and the Nephub™.

the corresponding Neptune™ probe eight hours earlier, as shown in Figure 32. When the administrator contacted the customer, they confirmed that a technical maintenance of an MME took place at the time when the anomaly was detected. During the maintenance, the MME was disconnected and the probe was not capturing any traffic. In some cases, by detecting issues in the inputs of the probes, we can locate issues in the cellular network: As the probe passively mirrors the traffic, if the traffic drops abnormally in the monitored equipment, the traffic of the probe drops as well.

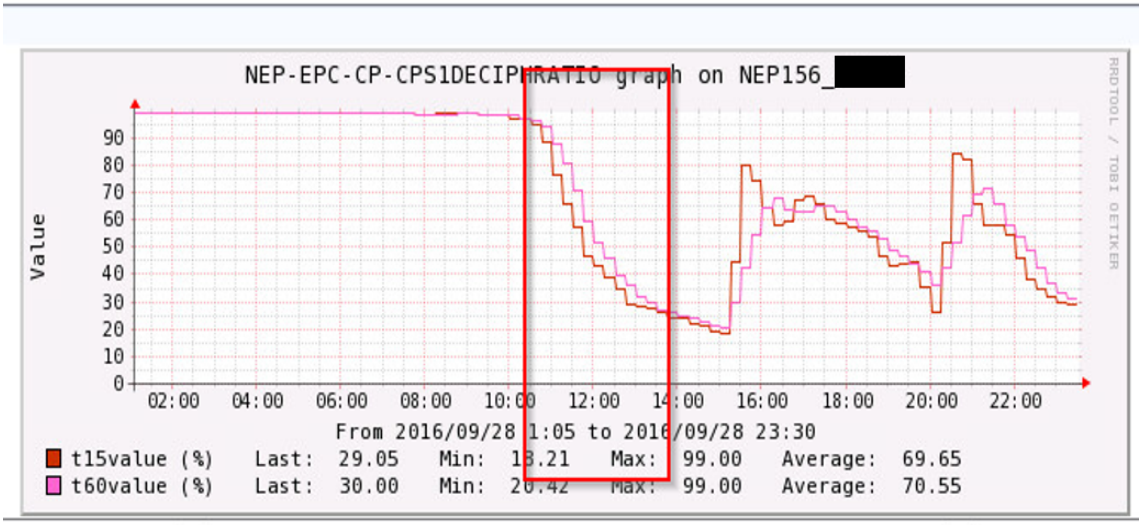


Figure 31: Cockpit™ graph of the deciphering rate of a Neptune™ probe.
An abnormal value drop.

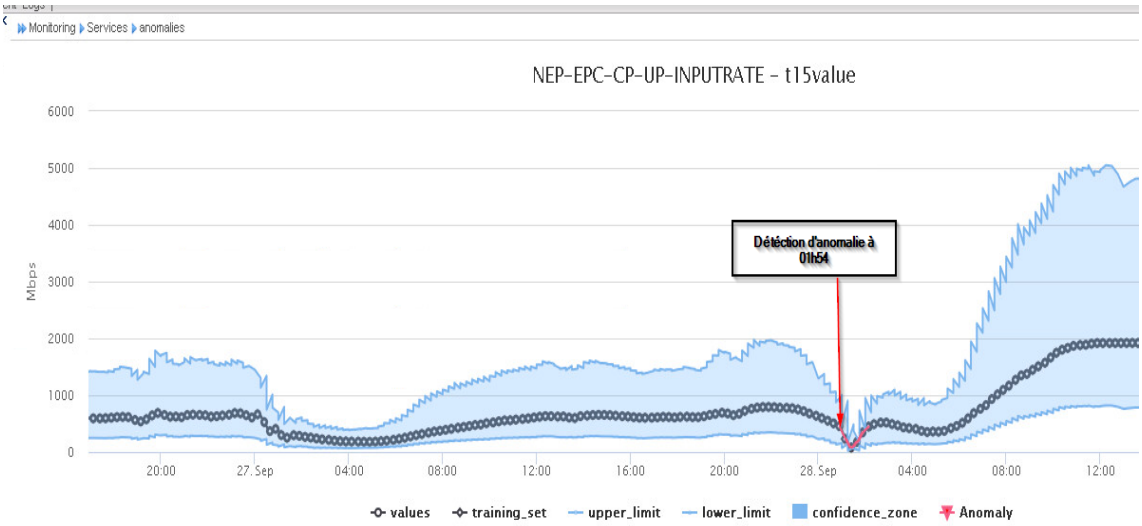


Figure 32: Cockpit™ capture of a Neptune™ probe input rate.
A zoom on a detected anomaly.

6.2 SERVICE QUALITY MANAGEMENT: NOVA ANALYTICS™

In this section, we show how WAD was integrated in Nova Analytics™ to manage service quality and therefore to monitor the cellular network. This implementation is still in test mode. In the following, we go through the scoping phase and the deployment phase of this project.

6.2.1 *The scoping phase*

We show here the scope of this project by introducing the development environment, the available data and the main requirements of this project.

6.2.1.1 *Environment*

Nova Analytics™ is a multi-dimensional Business Intelligence (BI) platform incorporating data from different points of the network to meet multiple needs depicted in Figure 33. Analytics™ offers interfaces to analyze network performance, customer experience, and service quality. Analytics™ is based on a Big Data framework, which centralizes all the data flows. The BI dashboards are created via MicroStrategy, which is a widely used data analysis and visualization tool.

In the context of Service Quality Management (SQM), Nova Analytics™ offers an End to End (E2E) visibility on service quality. This helps to assist Telecom experts on identifying service failures and degradation. By monitoring service quality, Nova Analytics™ contributes to increasing customer satisfaction.

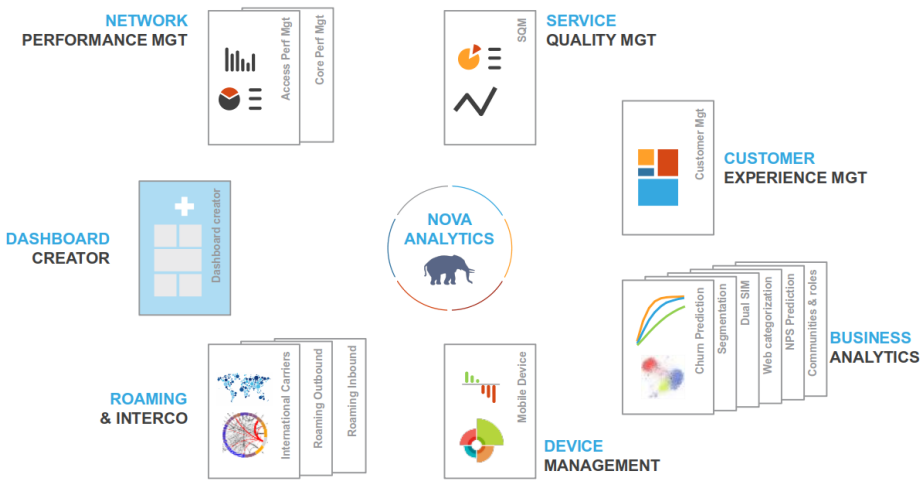


Figure 33: Nova Analytics™ features overview

6.2.1.2 *Accessible Data*

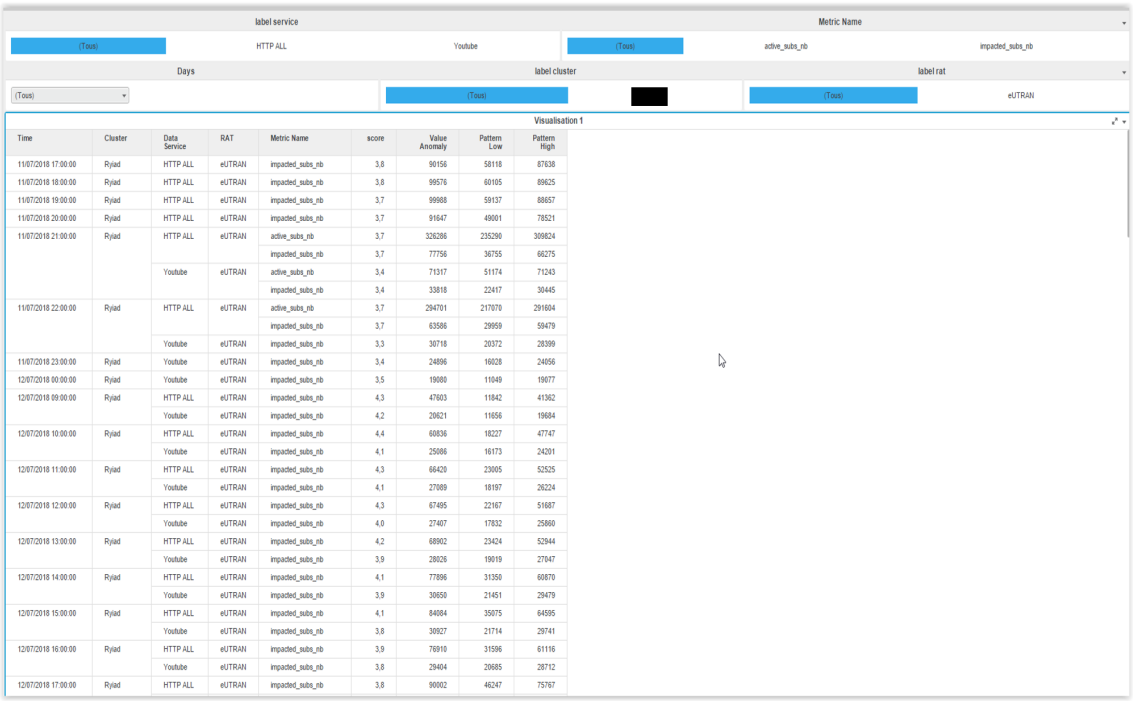
The goal of this implementation is to detect anomalies in the context of SQM. We apply WAD to E2E service quality indicators to detect emerging service issues. We focus on indicators from a user perspective including applications such as Facebook, YouTube,

Instagram, Voice over LTE (VoLTE), Voice over IP (VoIP), Voice over WIFI (VoWIFI), legacy voice as well as special operator services such as Unstructured Supplementary Service Data (USSD). The goal is to detect quality degradation and prioritize service restoration based on the number of affected subscribers. The KPIs covered in by the analysis include up-link/down-link re-transmission ratio per service or application, up-link/down-link throughput per service or application, number of active subscribers per service or application, the number of impacted subscribers by a service issue. The number of affected subscribers is the number of subscribers facing a bad Quality of Service (QoS), e.g., the number of subscribers having a re-transmission ratio higher than 10%. The rules to define bad QoS are set manually based on telecommunication expertise.

6.2.1.3 *Objectives*

The aim of integrating WAD in Analytics™ is to detect anomalies in SQM metrics. The plugin has to process data in real time in a parallel manner (Big Data platform). The configuration should be easy and it should have additional options such as data aggregation and filtering. As for Cockpit™, the plugin should have a low error rate and its maintenance should require reasonable efforts.

6.2.2 *The deployment phase*



Time	Cluster	Data Service	RAT	Metric Name	score	Value Anomaly	Pattern Low	Pattern High
11/07/2018 17:00:00	Ryad	HTTP ALL	eUTRAN	impacted_subs_nb	3,8	90156	58118	87638
11/07/2018 18:00:00	Ryad	HTTP ALL	eUTRAN	impacted_subs_nb	3,8	99576	60105	89625
11/07/2018 19:00:00	Ryad	HTTP ALL	eUTRAN	impacted_subs_nb	3,7	99908	59137	88657
11/07/2018 20:00:00	Ryad	HTTP ALL	eUTRAN	impacted_subs_nb	3,7	91647	49001	78521
11/07/2018 21:00:00	Ryad	HTTP ALL	eUTRAN	active_subs_nb	3,7	326286	235290	309824
				impacted_subs_nb	3,7	77756	36755	66275
		Youtube	eUTRAN	active_subs_nb	3,4	71317	51174	71243
				impacted_subs_nb	3,4	33818	22417	30445
11/07/2018 22:00:00	Ryad	HTTP ALL	eUTRAN	active_subs_nb	3,7	294701	217070	291604
				impacted_subs_nb	3,7	63586	29959	59479
		Youtube	eUTRAN	impacted_subs_nb	3,3	30718	20372	28399
11/07/2018 23:00:00	Ryad	Youtube	eUTRAN	impacted_subs_nb	3,4	24896	16028	24056
12/07/2018 00:00:00	Ryad	Youtube	eUTRAN	impacted_subs_nb	3,5	19080	11049	19077
12/07/2018 09:00:00	Ryad	HTTP ALL	eUTRAN	impacted_subs_nb	4,3	47603	11842	41362
		Youtube	eUTRAN	impacted_subs_nb	4,2	20621	11656	19684
12/07/2018 10:00:00	Ryad	HTTP ALL	eUTRAN	impacted_subs_nb	4,4	60836	18227	47747
		Youtube	eUTRAN	impacted_subs_nb	4,1	25086	16173	24201
12/07/2018 11:00:00	Ryad	HTTP ALL	eUTRAN	impacted_subs_nb	4,3	66420	23005	52525
		Youtube	eUTRAN	impacted_subs_nb	4,1	27089	18197	26224
12/07/2018 12:00:00	Ryad	HTTP ALL	eUTRAN	impacted_subs_nb	4,3	67495	22167	51687
		Youtube	eUTRAN	impacted_subs_nb	4,0	27407	17832	25860
12/07/2018 13:00:00	Ryad	HTTP ALL	eUTRAN	impacted_subs_nb	4,2	68902	23424	52944
		Youtube	eUTRAN	impacted_subs_nb	3,9	28026	19019	27047
12/07/2018 14:00:00	Ryad	HTTP ALL	eUTRAN	impacted_subs_nb	4,1	77896	31350	60870
		Youtube	eUTRAN	impacted_subs_nb	3,9	30650	21451	29479
12/07/2018 15:00:00	Ryad	HTTP ALL	eUTRAN	impacted_subs_nb	4,1	84084	35975	64595
		Youtube	eUTRAN	impacted_subs_nb	3,8	30927	21714	29741
12/07/2018 16:00:00	Ryad	HTTP ALL	eUTRAN	impacted_subs_nb	3,9	76910	31596	61116
		Youtube	eUTRAN	impacted_subs_nb	3,8	29404	20685	28712
12/07/2018 17:00:00	Ryad	HTTP ALL	eUTRAN	impacted_subs_nb	3,8	90002	46247	75767

Figure 34: Anomaly Logs in Nova Analytics™ detected by WAD

The anomaly detection plugin has been deployed in test mode on a live network for six months and the first results are promising. WAD has proven its capability to detect issues in the SQM context. The plugin requires few computational resources and was not interfering with any other Spark jobs in the Big Data cluster. The customer has validated

its interest on having WAD as a part of the analysis solution. Another implementation in an additional customer network is planned in 2019.

Figure 34 shows an example of anomalies detected in the SQM context over 24 hours in July 2018. WAD is applied to cubes from the S1-U interface. The data is aggregated to a value per hour. We applied two filters: {Radio Access Technology (RAT): 4G, service: Hypertext Transfer Protocol (HTTP) or Youtube}. The monitored counters are the number of active subscribers and the number of impacted subscriber by bad QoS (high re-transmission ratio and low throughput).

Figure 35 explains four anomalies on the number of subscribers accessing HTTP service. The upper bar-plot contains the number of subscribers accessing HTTP. The black bars present normal values and the red bars denote anomalies. In "KPIs trend" rectangle, the blue and the gray curves denote respectively the up-link throughput and the down-link throughput. The lower bar-plot in red denotes the up-link Transmission Control Protocol (TCP) re-transmission. In this graph, we see that the first three anomalies (detected on 2018/07/15) coincide with peaks in the up-link re-transmission. This means that the number of subscribers accessing HTTP increased because of an up-link re-transmission issue. The last anomaly (detected on 2018/07/16) coincides with a decrease in the up-link throughput as well as in the down-link throughput. Thus, this anomaly results from a traffic issue. In this figure, the re-transmission bar-plot and the throughput curves confirm that the detected anomalies on the number of impacted subscribers are true positives.

Figure 35: HTTP monitoring.
The number of impacted subscribers compared to the throughput and re-transmission KPIs. Anomalies detected in the number of impacted subscribers.

Figure 36 presents the same KPIs for Youtube service. This figure shows two anomalies on the number of impacted subscribers accessing Youtube (detected on 2018/07/13 and

2018/07/14). These anomalies coincide with dips in the down-link throughput. This confirms that the detected anomalies are true positives resulting from a traffic issue.

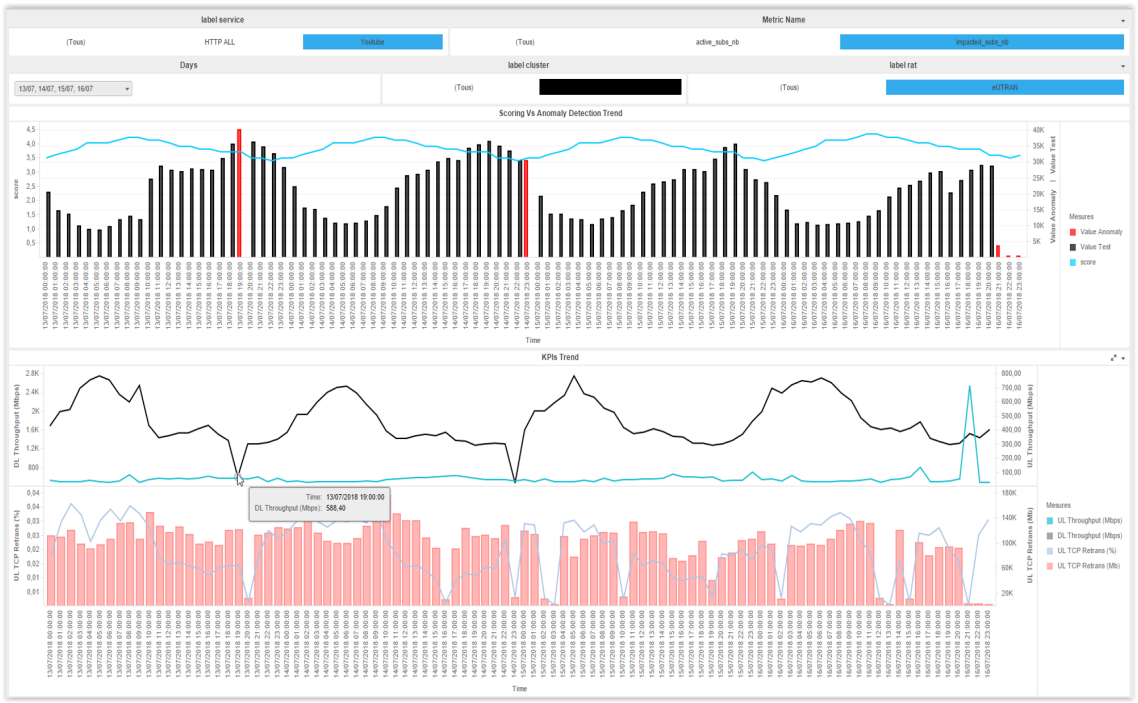


Figure 36: Youtube monitoring.
The number of impacted subscribers compared to the throughput and re-transmission KPIs. Anomalies detected in the number of impacted subscribers.

6.3 ANOMALY DETECTION TIMELINE

The project of anomaly detection started in the April 2015. Figure 37 summarizes the dates of the different phases of this project. We started the design of WAD in June 2015. After validating the main steps of the algorithm, we worked on a prototype starting from January 2016. Then, we took few months to test WAD on multiple data sets collected from different networks. As WAD met performance requirements, we transformed the prototype into a Cockpit™ plugin to detect anomalies in the metrics created by the monitoring system. Later in December 2017, WAD was added to the Analytics™ solution.

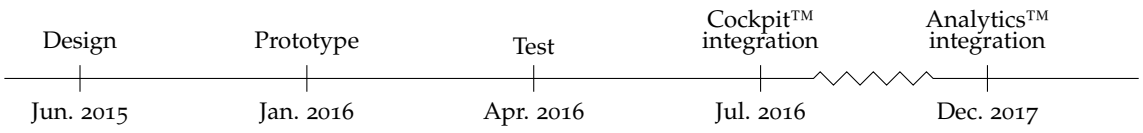


Figure 37: The timeline of WAD development and industrialization

6.4 CONCLUSION

In this chapter, we have showed how WAD was integrated in commercialized products to detect anomalies for two different aims. The first objective is to monitor the monitoring system. The implementation and tests demonstrate that WAD is effective in supervising both hardware and software components of the monitoring system. WAD was also used to monitor the cellular network by detecting anomalies in QoS indicators. The first results are promising. These implementations in industrial products has proved that WAD is a valid solution to detect anomalies in periodic time series in the context of monitoring cellular networks.

Part III

ROOT CAUSE DIAGNOSIS

This part is about the second objective of the thesis: Root cause diagnosis. In this part, we start by a study of the state of the art. Then, we propose a solution for this question and evaluate its performance. We finish by a short description of the industrialization process.

7 STATE OF THE ART

In this chapter, we present the state of the art of root cause diagnosis. We start by stating the problem. Then, we explore the different approaches proposed to deal with this question. Finally, we give an overview of the proposed solutions to perform root cause diagnosis.

7.1 PROBLEM STATEMENT

After detecting anomalies, the natural step towards healing the network is the diagnosis of the anomaly. This step aims at accurately identifying the origin of the error, which allows operators to take the appropriate mitigation actions. In a cellular network, anomalies can be related to the network itself such as radio interference between cells. Anomalies can also be related to subscribers such as Operating System (OS) bugs or to service providers such as routing issues. In the literature, the diagnosis process has taken various denominations in the research history such as "Root cause diagnosis", "Fault diagnosis" and "Fault localization". These titles refer to the process of identifying and delimiting the elements originating the anomaly. The question of root cause diagnosis has been studied in different domains for at least three decades. However, more work need to be carried out as this process still rely on human expertise in a major part. As a matter of fact, identifying the root of an anomaly requires a good knowledge of the domain, the anomaly context and the historical data: In concrete terms, a human expert analyses the context and the historical data to infer the possible causes of an issue. Designing a system capable of carrying out this analysis is very challenging. The system has to have the intelligence to perform a coherent analysis and the memory to learn from past experiences in a dynamic way. The current state of the art shows that many researchers have tried to address this question from different perspective. They proposed a wide variety of approaches differing in may aspects: domain-dependence, level of automation, context-awareness, analysis depth, etc. In this chapter, we aim to examine the main proposed approaches.

As stated in [121], the root cause diagnosis task can be based on different types of information. The different types of data are used by both human experts and monitoring systems to analyse issues and identify their root causes. To start with, network alarms may help in localizing the root cause of issues. The alarms indicate errors in network elements. The element concerned with the alarm is not necessarily the root of the issue and the alarm could be a mere result of an underlying breakdown. Multiple alarms at different point of the network may allow to diagnose the nature of the root problem. Moreover, status monitoring may produce data relevant to the diagnosis task. By analyzing the historical status of each of the network elements, experts are able to delimit the extent of an issue and understand its roots. These data can be found in the network logs/traces. Furthermore, one can use network counters such as Key Performance Indicators (KPIs). By studying the correlation of different counters, one may be able to lo-

calize failure origins. Another interesting information source is communication records such as Call Data Records (CDRs). These records contain detailed information about all elements involved in a communication. These information are useful in identifying the origins of call drops and bad Quality of Service (QoS) in general. In addition, experts use the network topology to understand the propagation of errors throughout the network. By studying the connections between different network elements, one may identify the root of a given issue. Lastly, user data can be useful in the troubleshooting task. For example, the subscription type may help to identify issues with specific services.

The nature of the output of a root cause diagnosis system depends on the approach and the data used as inputs. For instance, one can have a labeled database with classes of issues such as radio interference, Central Processing Unit (CPU) overload, etc. In this case, if a new issue occur, it may be classified and the system returns the type of the issue [74], [121]. On the contrary, if no labeled database is available, one may use the events logged by different network elements, study their causality and identify the root event [107]. In the case where tracking the events through the network is not possible, one may use the performance counters at each elements to identify the root element behind the failure [32]. To conclude, a root cause diagnosis system may return the specific type of the failure, its root event or its root element.

The question of root cause diagnosis is challenging at many levels [93], [132], [144]. Regarding the data aspect [144], the data generated by different network elements may be heterogeneous and stored in different points of the network. The task of collecting and formatting data may be cumbersome. In addition, real world systems may produce inconsistent and incomplete data that the root cause diagnosis system should be capable to handle. The large amount of data produced by mobile networks adds an extra level of complexity to this task. Furthermore, the analysis process may be very difficult in a live systems [132]. Faults arising at different points of the network may be related and identifying causal effect in large complex systems is not straightforward. Added to that, unrelated faults may occur simultaneously and isolating them is not an easy task. Finally, we would like to mention the challenge coming from the dynamic structure of the network. The fast evolution of mobile devices and services prevents network operators from having a knowledge database with all the expected fault types.

7.2 ANALYSIS SCOPE

Since mobile networks and networks in general are very complex, researchers have limited the scope of the diagnosis to specific part or aspects of network. A number of papers focus on diagnosing Radio Access Network (RAN) devices [48], [74], [90], [113]. They work on reporting RAN issues such as congestion, interference, and fading. Some of them go further to diagnose very specific scenarios such as sleeping cells [120] and indoor issues [40], [121]. Other researchers carry out a service-focused diagnosis [56], [63]. They work on issues related to availability such as server overload, to security such as intrusion detection, and to accessibility. Some researches focus on troubleshooting specific services such as video streaming [32] and TV [94]. A few papers focus on the whole network, proposing and End to End (E2E) network diagnosis [41], [68]. They consider both RAN and service issues.

7.3 ANALYSIS APPROACHES

Depending on the type of inputs researchers have, we can group proposed diagnosis solutions into three main approaches: an event based approach, a location based approach, and an error type based approach. In the event based approach, one analyses the sequence of events occurring in the network, studies their causality, and identifies the root event behind a network issues [100], [107]. In this case, the diagnosis is based on sets of event logs. In the location based approach, one studies the connections between network devices in order to identify the network element originating an issue [77], [94], [108]. In the case of error type based approach, the diagnosis systems return the type of the network issue [48], [120]. These systems can be either expert systems based on an expert characterization of each type of error or supervised systems relying on a statistical characterization of errors on the basis of a training database.

7.4 FEATURE DEPENDENCY

The literature on automatic root cause diagnosis is extensive [76]. We distinguish two main approaches, whether the diagnosis is implemented by scrutinizing one feature in particular, or by using dependency analysis.

7.4.1 *Diagnosis on Isolated Features*

Some researchers consider each feature in isolation (e.g., handset type, cell identifier, service), applying statistical inference, Machine Learning (ML) techniques, or expert rules to identify the elements causing network inefficiency. As an example, Gómez-Andrades *et al.* [48] use an unsupervised technique based on Self Organizing Maps and Hierarchical Clustering to identify the cells responsible for network inefficiency. In a more specific use case (ultra dense networks), Rodriguez *et al.* [120] apply a Bayes Classifier combined with expert rules to identify radio inefficiency root causes. For the same use case of determining malfunctioning cells, Palacios *et al.* [113] combine different classification algorithms to create a hybrid system that is more reliable but more complex. Khatib *et al.* [74] apply fuzzy Data Mining techniques to Long Term Evolution (LTE) KPIs to generate association rules. Luengo *et al.* [90] propose a binary classification tree for cells to determine the root cause of poor throughput. This approach focuses mainly on radio issues related to cells. Being a supervised method, it aims to detect only predefined issues. Other studies, which are not limited to the radio context, have an E2E view of the network considering only one feature at a time [41]. For example, Serrano Garcia *et al.* [123] propose a root cause diagnosis a dynamic framework based on weighted correlation. This framework is applicable at any level of the network (e.g. cells, core equipment). Such an approach has also been explored in contexts other than mobile networks. For instance, Zheng *et al.* [149] perform rough root cause location on High Performance Computing (HPC) systems based on software logs to classify issues into three classes of failures: hardware, software and application. In their work, Rodriguez *et al.* [121] proposed a context-aware analysis based on Naive Bayes method applied to KPIs. The additional information of the context depends on the feature. This type of

analysis can be conducted throughout the whole network, feature by feature. Mi *et al.* [100] apply Robust Principal Component Analysis (RPCA) to Cloud Computing logs to detect performance anomalies. They identified anomalous methods and replicas based on the execution time.

This approach, based on analyzing each feature in isolation, can be accurate and easy to understand by end users, but its main drawback is that it does not take into account the dependencies between the features. For instance the cells connected to a malfunctioning Base Station Controller (BSC) may appear as inefficient. The approaches based on considering one feature at a time have also the obvious limitation of ignoring all the problems caused by more than one feature, such as incompatibilities and causal effects. These induced effects cannot be detected unless one uses dependency analysis.

7.4.2 *Dependency-Based Diagnosis*

Some researchers have focused on hierarchical dependencies resulting from the topology of the network, e.g., the content providers of a mis-configured service having their content undelivered. To identify such dependencies, they rely on the topology of the network which is not always straightforward to obtain in an automated way. Jin *et al.* [67] combine multiple classifiers to rank the locations of the issues. Then, they exploit the topology of the wired access network to explain the dependencies between the problems. Mahimkar *et al.* [95] monitor the KPIs to identify the most inefficient elements in the network. Then, they explore the higher level elements in the topological graph of the network to identify the elements impacted by the same problem. Their previous work [94], applied to the specific case of IPTV networks, relies on the same approach using more advanced statistical calculations. By relying on the network topology to identify dependencies, one may miss some relevant occasional dependencies resulting from co-occurrence or coincidence, e.g., a group of cell phone roaming users (tourists) accessing the same cell. These dependencies are not predictable by the experts. To explore both hierarchical and occasional dependencies, different statistical methods have been proposed. Ong *et al.* [110] use Principal Component Analysis (PCA) (applied to categorial data) to create weighted association rules. While this approach is ideal to find incompatibilities, it does not deal efficiently with induced effects related to hierarchical dependencies: we may have multiple association rules pointing to the same problem with no information about the causal effect. Dimopoulos *et al.* [32] use decision trees to create a graph of dependencies. This method requires a database with already solved cases, which is not our case. Furthermore, creating decision trees is accurate and computationally efficient only when dealing with few features which is not our case either. In a similar context, Nagaraj *et al.* [107] propose an unsupervised approach to diagnose distributed systems. They apply the T-test to event logs to identify major events related inefficiencies. Then, they troubleshoot the locations affected by these events. Since several of our features are categorial, we cannot apply the T-test.

These studies, while addressing some of the challenges related to root cause analysis, do not meet all the requirements of a complete and production-ready diagnostic system. First of all, it is nontrivial to automatically and reliably discover the network topology. Solutions that require this as an input are not practical, especially for networks as complex as cellular ones. Second, the aforementioned statistical tools are not well suited to

handle logs with many features (more than one hundred in typical LTE networks), especially when many of these features are categorical. Lastly, for a solution to be viable in a production environment, it must also be scalable and present results that are easy to interpret by human operators.

7.5 TECHNIQUES

In the following we distinguish two types of systems: expert systems and anomaly detection based systems.

7.5.1 *Expert Systems*

Some researchers used expert systems to perform root cause diagnosis [68], [76]. Starting from a set of known issues, they build rules based on KPIs that can determine the nature and location of each problem. As they are based on known issues, they need to be constantly updated as new classes of problems are discovered. This is the norm in mobile networks, given the rapid growth of mobile networks and the multiplicity of actors (equipment vendors, handset manufacturers, software companies). If human experts need to implement these updates, the resulting costs can be prohibitive, given the large volumes of data generated by modern networks. This is why there have been recently some efforts to build more autonomous systems based on ML [76].

7.5.2 *Machine Learning*

There is a vast literature on classification and prediction, including applications to self-healing cellular networks [79]. Even though we are dealing with labeled data, our goal is not to predict the label, which is the goal of supervised ML techniques. Our goal is to identify the root causes of problems that can negatively affect the users of cellular networks. Classification, however, is a potential solution. Some researchers applied clustering techniques to cluster cells or another feature into faulty/non faulty one based on specific KPIs [48], [86], [90], [120]. This approach as, explained in Section 7.4.1, has many limitations. In order to have an E2E diagnosis system detecting different types of issues and considering the dependencies between different features, we need a rule induction algorithm that infers a rule set pointing the different roots of the inefficiencies from data [110]. Decision trees are a good option to generate such rules in this context [32]. However, besides the fact that it requires encoding categorial data (which is not an easy task at large scale), it necessitates a database of solved cases, which would be extremely expensive to generate and to keep up-to-date. On the one hand, human experts are not capable of identifying manually all the major contributors and incompatibilities to train a supervised system. On the other hand, characterizing major contributors and incompatibilities so they can be understood and imitated by the algorithm is complex given the dependencies between features. In general, ML algorithms can learn from labels but not from rules. To address these limitations, one may think of using Reinforcement Learning (RL) to create a self-healing cellular network [103]. RL is a technique that allows to make decisions sequentially. It consists on taking an action in each iteration in order to

maximize a reward function. These actions modify directly the state of the system. In our use case, RL has two major problems. First, we cannot deploy a solution that acts directly on the mobile network without human intervention. Second, it is not straightforward to identify a reward function for major contributors (and incompatibilities) that can be automatically evaluated. Other rule induction solutions based on Rough Set theory such as Learning from Examples Module, version 2 (LEM2) [47], do not require expert training (supervised learning) or a loop back (RL). We will illustrate the limitations of this algorithm further. This is why we claim that we need an ad-hoc solution for root cause diagnosis in mobile networks.

7.6 RECAPITULATION

In this chapter, we have discussed the main approaches adopted by researchers to address the question of root cause diagnosis. As Table 8 shows, different solutions have been proposed in mobile network and other related fields. Some of the proposals deal with a specific part of the network (RAN, core). Others consider the E2E network. Some of the proposed approaches focus on diagnosing specific issues such as indoor access and sleeping cells while others are more generic. A large variety of techniques was used ranging from expert systems to supervised and unsupervised ML algorithms. In the case of unsupervised leaning where there no knowledge base containing error types, diagnosis systems may return the location of the error or its originating event. The data used in the diagnosis varies from communication logs and system logs to KPIs and alarms. Despite these multiple propositions, the question of root cause diagnosis is still not fully addressed. Today, a major part of the diagnosis tasks is carried out by human experts. The complexity and heterogeneity of mobile networks make the diagnosis automation very challenging.

Domain	Focus	Technique	Input	Output	Ref
E2E Mobile networks	E2E fault diagnosis	Pattern recognition	Data records	Fault type	[68]
		Expert system	Subscriber tickets + KPIs	Fault type + location	[41]
	Service diagnosis	Clustering/ statistical testing	Service records	Fault location	[63]
	Video streaming diagnosis	Decision tree	QoE metrics	Fault type	[32]
RAN networks	Faulty cell classification/ clustering	Genetic algorithms + SVM	KPIs	Fault type	[86]
		Fuzzy Logic Controllers			[74]
		Binary tree classification			[90]
		Multiple classifiers + Naïve Bayes			[113]
		SOMs + hierarchical clustering			[48]
	Faulty cell diagnosis	Statistical correlation + expert rules	Network logs + KPIs	Fault type	[123]
	Sleeping cell diagnosis	Bayes classifier + expert rules	UE measurements	Fault type	[120]
	Indoor small cell diagnosis	Naïve Bayes classifier	UE measurements	Fault type	[121]
		Statistical correlation	KPIs + UE reports		[40]

Domain	Focus	Technique	Input	Output	Ref
ISP/IPTV networks	Service fault diagnosis	Correlation + association rules	Service events + metrics	Fault root event	[56]
	Spatio-temporal fault localization	Statistical correlation	Network logs + alarms + tickets	Fault location	[94]
	Maintenance induced issues	MRLS	Network counters	Fault type	[95]
DSL networks	Failure location	Binary classifier	Line measurements	Fault location	[67]
HPC/ Distributed systems	Failure type (soft/ hard/ app)	Probabilistic causality analysis	System logs	Fault type + location	[149]
	Performance issue diagnosis	T-test + dependency networks		Fault root event	[107]
Cloud computing systems	Cloud system diagnosis	RPCA	Cloud traces	Fault root event	[100]
Storage systems/ data centers	Devices troubleshooting	Chauvenet algorithm	Performance metrics	Fault location	[108]
		Expert rules	Performance metrics + alarms		[77]
Manufacturing systems	Alarm diagnosis	PCA + association rules	Alarms	Fault location	[110]

Table 8: Root cause diagnosis: main papers

8 PROPOSED ALGORITHM: ARCD

8.1 INTRODUCTION

Cellular networks have become more complex over the years, with multiple co-existing Radio Access Technologies (RATs) from 2G to 4G and now 5G, multiple core network devices from various vendors, multiple services that go beyond regular telephony, and multiple handsets running various Operating Systems (OSs). This growing complexity makes the task of monitoring the network and identifying the cause of performance degradation more challenging for network operators [147]. Most devices in the network generate *log* messages detailing their operations. Based on these messages it is possible to reconstruct what happened in the network, at least to a certain extent. But, given the sheer number and variety of these messages, it is not feasible for human operators to analyze all of them directly. This is why log messages are usually pre-processed before being scrutinized by human experts who are in charge of identifying and mitigating the issues by appropriate actions. This analysis, while partly automated and aided by ad-hoc tools [112], is often time consuming and inefficient. Network operators would like to increase the automation of this analysis, in particular for cellular networks, in order to reduce the time needed to detect, and fix, performance issues and to spot more complicated cases that are not always detected by human operators.

The data that are generated by the monitoring system are a large number of log *entries* (or simply logs), each of them being the report of what happened during a *session*. The term session depends on the specific service: Call Data Record (CDR) for a regular phone call or Session Data Record (SDR) for a service based on IP. The log takes usually the form of a series of 2-tuples $(feature, value)$. The feature is a name indicating the nature of the corresponding value (for example cell identifier, content provider, handset manufacturer), while the value is what has been collected for this particular session (in

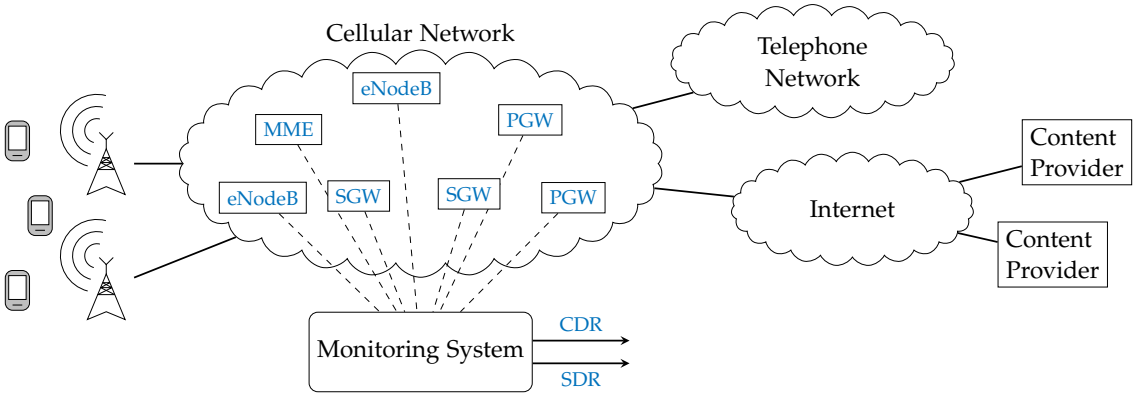


Figure 38: System architecture of an LTE network with a subset of the monitored elements; eNodeB, PGW, SGW, and MME

our example, a number that enables to uniquely identify the cell, the name of a provider, and the name of a manufacturer). The root cause of a network malfunction can be either a certain 2-tuple, or a combination of k 2-tuples. Figure 38 shows a simplified view of an LTE cellular network, including some of its elements and the corresponding monitoring system, which collects data from the different devices to produce the CDRs and SDRs.

Despite an abundant literature related to network monitoring, identifying the root cause of problems in modern cellular networks is still an open research question due to specific requirements related to the nature of this type of networks [102]: First, a diagnosis system should work on various types of logs (voice calls, data sessions, multimedia sessions) because, nowadays, cellular networks carry significant amounts of data traffic as well as voice. Second, a diagnosis solution has to deal with an increasing number of features. Logs can now include features related to the service (e.g., the content provider, the quality and priority classes), to the network (e.g., the RAT and the gateways involved), and to the user (e.g., the handset type and the handset manufacturer). Furthermore, these features can depend on each other due to the architecture of network and services. Third, a diagnosis solution has to address the complex interplay between features, for example, an OS version not supporting a particular service. Both the service and the OS can behave normally in a majority of sessions when scrutinized independently; the issue can only be diagnosed in logs containing both. Finally, the diagnosis solution should focus on problems that have an actual impact on the network performance. A problem that happens *sometimes* in a device that is used by millions of users can have a greater importance than a problem happening *always* in a device used by only a few users. The balance between number of occurrences and inefficiency is a matter of prioritizing mitigation actions.

Our goal is to design a diagnostic system, which addresses three key and challenging features that network operator management teams are waiting for:

- **Identifying Major Contributors**: We call major contributors all the elements that cause a significant decrease of the overall network efficiency. For example, a faulty MME results in the drop of a large number of calls. Our goal is to identify the major contributors in the whole set of logs. Once these major contributors have been identified, it is possible for human experts to scrutinize them in order to resolve the underlying issues.

- **Detecting Incompatibilities**: A consequence of the great variety of devices involved is that incompatibilities between some of them can also result in poor Quality of Experience (QoE) for the users. An incompatibility is a failing combination of two, otherwise properly working, elements. As previously stated, a new release of an OS can typically be incompatible with a service (e.g., voice calls, toll numbers).

- **Forming Equivalence Classes**: Equivalence classes are sets of key-feature values that correspond to the same underlying problem or that are strongly correlated. For instance, if a cell has only a few users and one of them is significantly more active than all the others (e.g., an automated bot making robocalls) and if the corresponding International Mobile Subscriber Identity (IMSI) has an abnormally high failure rate, the corresponding cell might have an abnormally high failure rate

as well, but this is only because the specific IMSI is present in the overwhelming majority of the logs of the calls starting in that cell. We build equivalence classes to prevent the same cause to appear in failing logs under multiple features.

We provide more details in Section 8.3 on the reasons for which these three tasks are relevant and why it is not easy to automate them reliably.

In the following, we present the main algorithms that run our solution, the Automatic Root Cause Diagnosis (ARCD) system. ARCD identifies inefficiency major contributors and incompatibilities. Then, it creates dependency graphs between the identified issues and prunes the graphs to identify root causes. We have evaluated ARCD by analyzing the logs of three different cellular network operators. Our results show that with an automated solution, we can not only carry out the analysis done by experts but we can go to a finer level of diagnosis and point to the root causes of issues with high precision.

8.2 DATA MODEL AND NOTATION

8.2.1 *Data Records*

As specified in the Third Generation Partnership Project (3GPP) specification [1], network operators collect data records to report every mobile communication that is established in the network. A data record contains the technical details of a mobile communication without including its content. These records can be used in network troubleshooting [68]. We call *log* an entry in the data records. A log is a series of 2-tuples $(feature, value)$ where the features can be:

SERVICE RELATED extracted via a Deep Packet Inspection (DPI) module such as service type, Mobile Network Code (MNC), content provider, QoS Class Identifier (QCI).

NETWORK RELATED such as RAT, MME, Radio Network Controller (RNC), cell.

USER RELATED such as IMSI, handset manufacturer, handset type.

In a log, every feature is associated with a value. We show in Table 9 three logs with a few features. Note that every value used in this chapter appears with a hashed value. For example, logs 0 and 2 have the same cell and logs 0 and 1 have the same service. The table shows as well that each log has a feature called *label*, which is a binary value indicating whether the communication was satisfactory or not. The label can be either collected directly by the monitoring system, or it can be computed during a post-processing analysis based on the values of the log.

	first_cell	imsi	tac	service	interface	label
0	a3d2	97c8	c567	ea52	eccb	failed
1	b37a	56ed	ce31	ea52	19c4	successful
2	a3d2	fa3e	c41e	c98e	f487	successful

Table 9: Example of a Data Record on a few features

We consider two types of Data Records in this paper:

CDR the records of voice calls. Each time a subscriber attempts to make a call, a CDR is created. If the call is dropped, the CDR is labeled as failed.

SDR the records created to track every Internet connection in cellular networks. An SDR is created each time a subscriber attempts to use an online mobile application. SDRs are often the summary of multiple Transmission Control Protocol (TCP) connections initiated by the mobile application. Unlike CDRs, SDRs are not labeled. However, it is possible to estimate the QoE for the user and thus to attribute a label, based on the data rate, response time, and retransmission ratio.

Thanks to the success/fail label, it is possible to compute the network *inefficiency*, that is the proportion of failed communications. This proportion can be computed over all the logs, or it can be relative to a specific subset (e.g., all the logs that share a given feature-value pair).

8.2.2 *Notation*

Let E be a set of logs and $f_1, f_2, ..f_n$ be the features of the logs. A log $x \in E$ can also be represented as a vector $x = (x_1, x_2, \ldots, x_n)$ where x_i is the value of the feature f_i. Since every log has a label, we can partition E in two disjoint subsets: S containing the logs labelled as successful and F containing the logs labelled as unsatisfactory (i.e., failed).

We introduce the notion of *signature* to group the logs that have certain similarities. A k-signature s is the set of all logs, irrespective of their label, where k pre-determined features $\{f_{p_1}, f_{p_2}, \ldots, f_{p_k}\}$ $(1 \leqslant p_i \leqslant n, \forall i)$ have k specific values $\{s_{p_1}, s_{p_2}, \ldots, s_{p_k}\}$. We call the parameter k the order of the signature.

For instance, a 2-signature s that groups all logs from cell *ab34* and a mobile phone running the OS *b4e8* can be represented as:

$$\left((\text{first cell}, \text{ab34}), (\text{handset os}, \text{b4e8})\right)$$

We denote by $E(s)$ the set of all logs matching the signature s, regardless of their labels. Similarly, we denote by $S(s)$ (respectively $F(s)$) the set of all successful (respectively failed) logs matching signature s.

We define a few quantities that are useful to characterize signatures. (The operator $|.|$ denotes the cardinality of a set.)

SIGNATURE PROPORTION (π): the proportion of logs matching s:

$$\pi(s) = \frac{|E(s)|}{|E|}$$

COMPLEMENTARY SIGNATURE PROPORTION ($\overline{\pi}$): the proportion of logs that do not match s:

$$\overline{\pi}(s) = 1 - \pi(s)$$

FAILURE RATIO (λ): the proportion of failed logs among those matching s:

$$\lambda(s) = \frac{|F(s)|}{|E(s)|}$$

COMPLEMENTARY FAILURE RATIO ($\overline{\lambda}$): the proportion of failed logs in the data set without considering the logs matching s:

$$\overline{\lambda}(s) = \frac{|F| - |F(s)|}{|E| - |E(s)|}$$

8.3 OBJECTIVES OF THE DIAGNOSTIC SYSTEM

Modern cellular networks are complex systems, containing several devices of different types. Network operators run extensive monitoring systems to keep a constant watch on all these devices, in order to detect misbehaving and faulty ones. Faulty devices are not the only element to influence the QoE of the end-users. For instance, roaming users could be prevented from accessing the network because a misconfiguration of an MME. Users often want to communicate with people (or services) hosted on networks other than the one of the cellular one they are connected to. Because of this, problems in other networks can result in a poor QoE for some of the users of a given cellular network.

8.3.1 *Major Contributors*

One of the consequences of the ever increasing usage of cellular networks is that, even though the overwhelming majority of users have an acceptable QoE, there is no shortage of failed logs in most production networks. Operators would like to be able to identify as quickly as possible, and as efficiently as possible, the major contributors, that is the feature-value pairs (or their combination) denoting elements causing a significant decrease of the network efficiency.

Unfortunately one has to be careful when applying the definition of major contributors, as it can lead to some undesirable results. First, because there is a inherent hierarchy in the cellular network, one can significantly increase the efficiency of the network by excluding all the logs that share a very popular feature-value pair. For instance, we could notice a drastic increase of the efficiency after excluding the logs matching a device in the core network. This does not necessarily mean that the core network element is deficient. If we have multiple user and cell issues, these issues combined may lead to a high failure ratio of the core network element. Thus, the core network element may be diagnosed as faulty whereas it functions correctly.

Second, some elements appear in a statistically significant number of logs, have a high failure ratio, but their inefficiency is *extrinsic*, i.e., caused by other elements. For example, a Base Station Controller (BSC) connected to four cells can be involved in a large number of calls. If two or three of its connected cells are faulty, excluding all the logs referencing this BSC would result in a larger increase in the efficiency than excluding only the logs containing one of the cells. But we are interested in identifying the faulty cells rather than the functioning BSC.

Third, some elements are highly inefficient (they fail often), however they do not appear in a large number of logs. For instance a subscriber can attempt to make a single call, which is dropped. The inefficiency of the IMSI of the subscriber is 1.0 however it is calculated on only one log. This IMSI cannot be considered as a major contributor since its removal has no visible impact on the overall inefficiency. This example illustrates

the tension between major contributors and *weak signals*, i.e., elements that have a high inefficiency but that are shared by a small number of logs.

8.3.2 *Incompatibilities*

While major contributors can potentially affect a non-negligible number of users, incompatibilities often affect fewer users, making them a less urgent issue. For example, even though voice calls using a given technology (4G or 2G) have a very low failure rate, calls started with 4G and then switched to 2G may have an abnormally high failure rate. The reason behind this incompatibility may be an incorrect configuration of the Circuit Switched Fallback (CSFB) procedure or the use of devices from different vendors. Similarly, a new release of a smart phone OS could not be compatible with the Tranport Layer Security (TLS) implementation used by a certain content provider, because of a bug in the implementation of the TLS handshake in the new release.

ARCD detects as incompatibilities all the cases where a combination of two otherwise working elements results in a higher failure rate. Certain cases that fall within this category do not necessarily conform with the definition of the term "incompatibility." For example, consider two neighboring cells that work correctly but such that there is a poorly covered area between the two, so that all the calls started in one cell and then handed over to the other experience a high drop rate. ARCD detects this as an incompatibility, even though, there are no two incompatible devices here. This is, nonetheless, an issue that must be detected and addressed in order to increase customer satisfaction.

It is extremely hard for human operators to detect incompatibilities simply by looking at the logs, unless these affect a significant number of logs. This is why network operators need automated solutions that can identify them with little or no human intervention. As we discuss in Section 8.6.5.3, detecting incompatibilities is more computationally intensive than detecting major contributors, this is why ARCD addresses each problem separately, giving operators the freedom to decide how often to trigger each operation.

8.3.3 *Equivalence Classes*

ARCD detects both types of anomaly (major contributors and incompatibilities) as a set of one, or more, feature-value pair(s). Because logs are meant to accurately represent the widest possible number of cases, they contain between 30 and 60 different features (see Table 10 for the precise number of features present in each data set). It is common to see a strong correlation between one or more features for a set of logs. For example, all SDR corresponding to a given service (feature *service provider*) can have the same value for the IP address of the service (feature *host*), whenever the service provider exposes a single IP address as it does happen for smaller service providers. In this specific example, there is a perfect correlation between the two features, making them completely equivalent for a subset of the logs.

The correlation can also reflect the hierarchy of the cellular network. In the example mentioned above, about a single failing IMSI in a cell with few users, the IMSI feature

is not exactly equivalent to the cell-id feature, but, for the purpose of identifying the underlying problem, they are indeed equivalent.

It is important for ARCD to be able to detect these equivalence classes automatically, in order to present them to the human operators, who could otherwise be overwhelmed by a long unorganized list of feature-value pairs. This is why we build a directed graph where each node contains one or more feature value pair. Such a data structure has the added advantage of being well suited to build interactive visualizations allowing users to easily navigate between the different problems and to drill down into the details of each one when needed.

8.4 MAJOR CONTRIBUTOR DETECTION

We now explain how ARCD processes data records to detect major contributors and how it then filters the results to produce a graph of dependencies between issues occurring in the network. The first step is to label each log as successful or failed if the input logs are not already labeled. Then, it identifies the top signatures responsible for the network inefficiency. These signatures are then classified into equivalence classes, which are groups of signatures corresponding to the same problem. Then it generates a graph outlining the dependencies between all the problems. It finishes by pruning the graph to remove unnecessary nodes denoting false problems (elements appearing as inefficient because they share a part of their logs with malfunctioning ones). Figure 39 gives a graphical representation of these steps, which are detailed below.

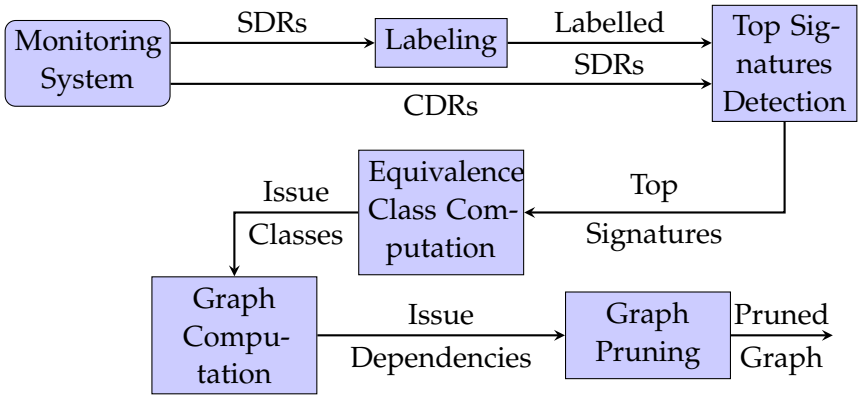


Figure 39: Major contributor detection steps

8.4.1 *Labeling*

The first step consists in labeling the logs. If the data has no success/failure label, we create a binary feature based on standardized criteria specified by the 3GPP. In the case of CDRs, we have a label success/failure based on the Session Initiation Protocol (SIP) messages exchanged between network devices. In the case of SDRs, we assess the Quality of Service (QoS) of TCP connections based on metrics such as mobile response time, server response time and re-transmission ratio. For each metric, we set a lower bound for an acceptable QoS. An SDR with one value below the threshold is labeled as failed.

8.4.2 *Top Signature Detection*

The second step consists in identifying the top 1-signatures contributing to the overall inefficiency of the network. To do so, we start by generating the set of all 1-signatures, that is the set of all possible values taken by each one of the features. Then, for each signature we compute two values: the Complementary Signature Proportion $\overline{\pi}$ and the Complementary Failure Ratio $\overline{\lambda}$. The 1-signatures with the smallest values of $\overline{\lambda}$ correspond to the major contributors: removing all the logs belonging to these signatures results in smallest overall failure ratio for the remaining logs. Some of these signatures match a significant fraction of the logs in the system, for instance because the 1-signature corresponds to a device that handles a lot of traffic with a slightly higher failure ratio than the rest of the network.

There is a trade-off between inefficiency and representativeness. The Complementary Signature Proportion ($\overline{\pi}$) indicates whether a 1-signature matters. The larger is $\overline{\pi}(s)$, the less common is the signature s. Our trade-off is thus as follows: on the one hand we want signatures with the smallest values of $\overline{\lambda}$ but not if the corresponding $\overline{\pi}$ is too small. We achieve this goal by maximizing a linear combination of these two values:

$$\nu(s) = \overline{\pi}(s) - \alpha\overline{\lambda}(s)$$

where α is a parameter to weigh the two aspects mentioned above. For small values of α, the dominant term of the score ν is $\overline{\pi}$. Thus, the signatures with the highest ν will be signatures with few matching logs and a relatively high failure ratio. This way, we highlight "weak signals". However, for large values of α, the dominant term of ν is $\alpha\overline{\lambda}$. In this case, the signatures with high ν will be signatures having large impact on the overall failure ratio in the network. These signatures generally match a large set of logs and are consequently called "major contributors". To have a robust solution, we use several values of α: ten values between 0 and 1 $(0.1, 0.2, 0.3, \ldots, 0.9)$ and then twenty values between 1 and 20 $(1, 2, 3, \ldots, 20)$. The first set of values ($\alpha < 1$) corresponds to the weak signals while the second corresponds to the major contributors.

For each one of these values of α, we compute ν for each 1-signature and we take the twenty signatures with the largest values of ν ("top twenty"). We then compute how many times one of these signatures is in a top twenty. A signature that appears often in the top twenty corresponds to a potential problem. In all our data sets, it is indeed the case that several signatures appear multiple times in the different top twenties. We complete this step by taking the fifty signatures that appear more often in the top twenty. However, we cannot stop here because some of these 1-signatures could correspond to the same underlying problem. That is what the following step addresses.

8.4.3 *Equivalence Class Computation*

This step consists in grouping signatures related to the same problem. As an example, consider a user connecting to a cell, where he is the only active user, with an uncommon handset type. If, for some reason, the user experiences many consecutive bad sessions, the resulting logs are labeled as failed. In this case, the corresponding IMSI, handset type, and the cell-id appear at the top of the signature list that we generated in the previous step. The three signatures point to the same problem rather than three separate

problems and have to be grouped into one 3-signature. In general, two signatures are equivalent (or mutually dependent) when they match the same logs. As an aside, we cannot determine the causal relationship between the features and the failure. In our example, either the phone type, the IMSI, the cell or any combination of these three could be the cause of the failure

We address this problem by computing the *intersection ratios* c_1 and c_2 for each pair of 1-signatures in the list produced by the previous step:

$$c_1 = \frac{|E(s_1) \cap E(s_2)|}{|E(s_1)|}$$
$$c_2 = \frac{|E(s_1) \cap E(s_2)|}{|E(s_2)|}.$$

As $E(s_1)$ and $E(s_2)$ denote respectively the set of logs matching s_1 and s_2, c_1 represents the ratio of logs matching both s_1 and s_2 to all the logs matching s_1. In the same way, c_2 represents the ratio of logs matching both s_1 and s_2 to all the logs matching s_2. If both c_1 and c_2 are larger than a threshold γ, we consider the two signatures as being *equivalent* and we merge them into one 2-signature. We keep repeating this process as long as at least two signatures satisfy these conditions. This process can generate "longer" signatures as it keeps merging shorter ones. For example, it can create a 3-signature by merging a 2-signature with a 1-signature, and so on. We stop iterating when no more signatures are merged. In remainder of this paper, we set $\gamma = 0.9$. The outcome of this step is a set of equivalence classes, where each class denotes one problem.

8.4.4 *Graph Computation*

A *hierarchical* dependency is another case of multiple signatures corresponding to the same underlying problem. For instance, a BSC connected to faulty cells would appear as inefficient even if it is not the cause of the problem. In another case, a normally functioning cell may appear as inefficient because it is connected to a faulty BSC. There is a hierarchical dependency between a cell and a BSC that should be taken into consideration in the analysis. The BSC can be seen as the parent of the cell in a hierarchical representation of the network. In our notation, we say that the cell *depends* on the BSC. In order to highlight this type of dependencies, we create a graph to model the dependencies between equivalence classes created in the previous step. Each equivalence class can be seen as a k-signature. Equivalence classes are presented as the nodes of the graph. To connect the nodes, we need to test hierarchical dependencies between equivalence classes. Therefore, for each k-signature s_1, we find the all the signatures s_2 such that:

$$\frac{|E(s_1) \cap E(s_2)|}{|E(s_1)|} > \gamma$$

which means that the logs matching s_1 are approximately a subset of the logs matching s_2. This way, we find all the parent nodes of s_1. The output of this process is an acyclic direct graph, which is not necessarily connected.

The graph may have superfluous connections, as shown in the example of three signatures s_1, s_2 and s_3 in Figure 40. If s_1 depends on s_2 (s_2 is the parent node s_1) and s_2 depends on s_3 then s_1 depends on s_3. If we generate the graph as explained above, we

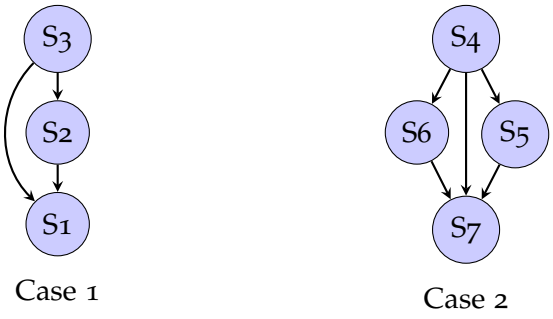


Figure 40: Examples of multiple paths between nodes containing signatures

have two paths between s_1 and s_3: a direct connection and another connection via s_2. In this case, the direct connection between s_1 and s_3 is irrelevant since it does not add any information compared to the connection via s_2. To solve this problem, we use the Depth-first Search algorithm to find all the paths between *every pair of connected nodes* and then we keep only the longest path. Keeping only the longest path does not lead to any information loss.

Consider the second case in Figure 40. There are three paths from s_4 to s_7. Without loss of generality, assume that we have considered the path through s_6 as the longest one. Then, we will keep only this path between s_4 and s_7. But, since s_4 depends on s_5 (and there is only one path between s_4 and s_5), this link is kept as the longest one between s_4 and s_5. The same argument applies to the link between s_5 and s_7. In this case we end up removing only the direct link between s_4 and s_7.

8.4.5 *Graph Pruning*

The structure of the graph allows the exploration of faulty devices and services in a hierarchical way: At the top, we find frequent signature (having a high π) such as core network equipment, popular services and handset manufacturers. At the bottom of the graph, we have less frequent signatures such as user IMSIs, host IP addresses, and the least used cell-ids.

In a well-constructed graph, we need each child node to have extra information compared to its parent nodes. Otherwise, it is irrelevant and it should be removed. In our case, we know that the parent node is inefficient to some extent (all the nodes in the graph are made up of the inefficient signatures selected in step 1). And, as the child node matches a subset of logs of the parent, it is expected to be inefficient as well. Therefore, presenting the child node is meaningful only in the case where it is more inefficient than at least one of its parent nodes. To remove superfluous nodes, we define a measure called *Relative Failure Ratio* λ_r. Suppose that we have two connected nodes (parent and child). λ_r is defined as follows:

$$\lambda_r(s_c, s_p) = \frac{\lambda(s_c) - \lambda(s_p)}{\lambda(s_p)}$$

where s_p is the signature in the parent node and s_c the one in the child node. For each node, we calculate its relative failure ratio with respect to all its parents. We keep the node if at least one of the relative failure ratios is greater than 0.01. Otherwise, we

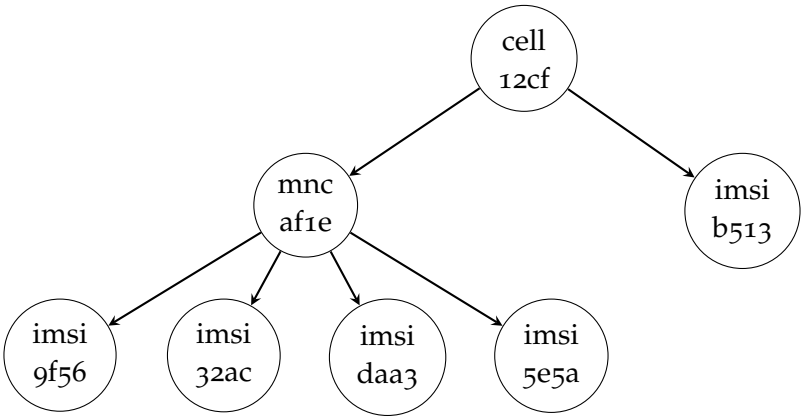


Figure 41: Pruning Scenario

remove it. Every time we remove a node, we connect its ancestors with its successors. After this pruning operation, every child node in the graph is more inefficient than at least one of its parent nodes. In such case, we have two possible scenarios:

- In the first one, the child node presents a separate problem. This could be the case of a user trying continuously to call an unreachable number through a cell having an interference problem. In the graph, we can find the node containing the user IMSI as a child of cell-id node with user IMSI failure ratio being higher than the cell failure ratio. The user calling an unreachable number and the radio interference problem are two separate issues. It is therefore important to keep the two nodes.
- In the second one, the child node is the root of the inefficiency of the parent node. Figure 41 illustrates an example of this scenario. Consider the case of a large group of roaming users (tourists, for example) accessing the network through a small cell (with few resident users). In Figure 41, the id of the small cell is 12cf. The MNC of the home network of the roaming users is af1e. There is a resident user with an IMSI b513 and four roaming users whose IMSIs are 9f56, 32ac, daa3, and 5e5a. The roaming users may experience a bad QoE because of a roaming issue between their home network and the host network. Since the roaming users are the main users of the cell, the cell has a high failure ratio λ. In the graph, we find the MNC of the roaming users as a child node of the cell. The MNC may have a slightly higher λ compared to the cell. In this case, the node containing the cell-id has to be removed since roaming is the real issue.

To deal with the second scenario, we proceed as follows. Consider two connected nodes: a parent node s_p and a child node s_c. Let λ_n be the overall failure ratio of the network.

$$\lambda_n = \frac{|F|}{|E|}$$

Our goal is to know whether the high failure ratio of s_p is due to s_c. To do so, we restrict to the logs matching s_p, $E(s_p)$ instead of using the whole data set. In this subset we calculate the complementary failure ratio $\overline{\lambda}(s_c)$ which is the failure ratio of s_p after removing the logs matching s_c. If $\overline{\lambda}(s_c) \leqslant \lambda_n$, then s_p is a non faulty signature and the parent node is removed. Otherwise, we are in the first scenario and the two nodes

present two different problems. As previously mentioned, each time we remove a node, we connect its ancestors to its successors. With these new connections, we may have other nodes to remove. We repeat the pruning process as long as it has removed at least one node in the previous iteration. This process is guaranteed to terminate as it only removes nodes.

8.5 INCOMPATIBILITY DETECTION

The procedure described in Section 8.4 allows us to detect major contributors that have a high impact on the network inefficiency. In this section, we explain how ARCD finds incompatibilities. By definition, these involve more than one feature-value pair making their detection more computationally demanding than major contributors. For the latter, we only needed to compute all the 1-signatures, that is the set of all logs having the same feature-value pair, for all the different feature-value pair present in the logs. When looking for incompatibilities, we compute all the 2-signatures that is the set of all logs having the same two feature-values for two features, for all the features and their values in the logs. Because of this, finding incompatibilities requires more memory and more processing time. We decided to separate the two problems so that operators can decide when to trigger each detection separately, depending on their needs and the availability of computing resources.

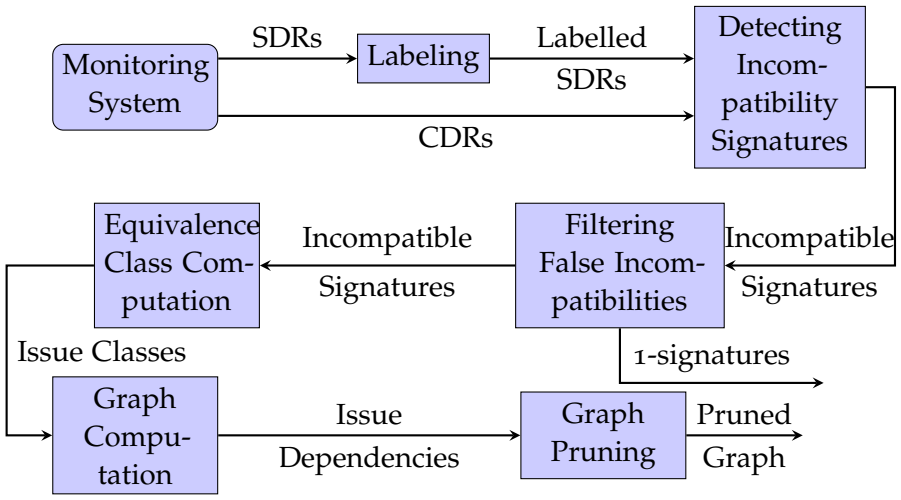


Figure 42: Incompatibility detection steps

Figure 42 describes the main steps of incompatibility detection. We start by labeling data as explained in Section 8.4.1. Then we identify the signatures pointing to incompatibilities throughout the network. Next, we filter false incompatibilities, that is cases where the inefficiency in the logs sharing a given 2-signature is actually explained by a third faulty element that is also present in most of the logs of the 2-signature in question. Finally, similarly to Section 8.4, we compute the equivalence classes.

8.5.1 *Identifying incompatible signatures*

The goal of this step is to find the signatures corresponding to incompatible elements. To do so, we start by generating the sets of all 1-signatures and 2-signatures. For each signature, we measure the failure ratio λ. To decide whether a 2-signature points an incompatibility, we proceed as follows.

Let us consider a 2-signature $s = \{f_i = v_i, f_j = v_j\}$ where v_i and v_j are values taken by the features f_i and f_j, corresponding to two 1-signatures ($s_i = \{f_i = v_i\}$ and $s_j = \{f_j = v_j\}$ respectively). For example, we could have:

$$s_i = (\text{service}, a587) \quad s_j = (\text{handset os}, c255)$$
$$s = ((\text{service}, a587), (\text{handset os}, c255)).$$

To determine whether two feature-value pairs are incompatible, we compute the *gain* of s, defined as:

$$g(s) = \lambda(s) - \max(\lambda(s_i), \lambda(s_j)).$$

The gain allows us to evaluate the impact of combining two 1-signatures on the failure ratio. In other words, we check if the combination of two elements is more inefficient than each one apart. If the gain is larger than a threshold, set to 0.2 in our implementation, we consider that the 2-signature s corresponds to incompatible elements.

8.5.2 *Filtering False Incompatibilities*

A combination with a higher failure ratio than each of its components does not imply automatically that we are dealing with an incompatibility. For example, consider the service type Instant Messaging (IM) and a specific content provider, both having a low failure rate. However, the combination of these two elements has a high failure rate. This combination could be identified in the previous step as an incompatibility. However, by studying the logs matching this combination, we may find a single host delivering IM service in the content provider network. The root of the issue, here, is not an incompatibility between IM and the content provider but rather a malfunctioning host.

In order to filter such cases, we proceed as follows. For each 2-signature selected in the previous step (s), we identify its matching set of logs $E(s)$. In this subset, we find all the 1-signatures t that represent a significant fraction of $E(s)$:

$$\tau(s) = \left\{ t | \frac{|E(t) \cap E(s)|}{|E(s)|} > \gamma \right\}$$

where $\gamma = 0.9$. Then, we re-compute the gain:

$$g(s) = \lambda(s) - \max_{t \in \tau}(\lambda(t)).$$

If the gain remains higher than the threshold (0.2), it means that the combination is a real incompatibility: There is no third element with a high failure ratio co-occurring with the combination. Otherwise, if the gain drops below the threshold, there is a third element that is at the origin of the high failure ratio rather than the combination of the two elements. In this case, the issue is not an incompatibility but a highly inefficient single element. We report this element separately with the appropriate 1-signature and we discard the corresponding 2-signature from the incompatibility list.

8.5.3 *Equivalence Class Computation*

At this point, we have a list of 2-signatures corresponding to incompatibilities. But some of the signatures might actually correspond to the same underlying issue. For instance, suppose that we detected an incompatibility between a handset type and an eNodeB identifier. Suppose as well that this eNodeB has only one static IP address so that we could have detected the same incompatibility twice. In this case we should group the eNodeB identifier and its IP address as an equivalence class that is incompatible with the handset type. The goal of this step is to find for each feature-value pair present in one of the 2-signatures, the classes of elements that are incompatible with it.

Let Ω be the set of all 2-signatures that have passed the filtering step, where each 2-signature (ω_i) is composed of two 1-signatures β_i, δ_i. For each one of these constituent 1-signatures we build a list of all the other 1-signatures with which it is incompatible. In other words, for every 1-signature (ρ) in the union of all the β_i and δ_i we compute $I(\rho) = \{\theta | (\rho, \theta) \in \Omega \text{ or } (\theta, \rho) \in \Omega\}$. As $I(\rho)$ is a set of 1-signatures, we can proceed as in Section 8.4.3 to group $I(\rho)$ into equivalence classes. Recall that this process can create "longer" signatures (2-signatures, 3-signatures, etc.) so that, at the end of this process, for each 1-signature ρ, which was in one of the original 2-signatures in Ω, we have a set of k-signatures that are incompatible with it.

8.5.4 *Graph Computation*

Some incompatibilities may result from other ones. For example, if a a service and an OS are incompatible, all the OS versions will be incompatible with that same service. That is why, to identify the root incompatibility, we need to explore hierarchical dependencies. Therefore, for each signature ρ we use the equivalence classes computed in the previous step to create a graph as we did in Section 8.4.4. Note that we have $2|\Omega|$ graphs, that is a graph for each 1-signature ρ. This graph contains the signatures incompatible with ρ grouped into equivalence classes and connected based on their hierarchical dependencies.

8.5.5 *Pruning*

At this point of the analysis, for each 1-signature ρ, we have a graph of the elements incompatible with ρ. For each pair of connected nodes in this graph, either the parent (θ_p) is the origin of the incompatibility with ρ or the child (θ_c) is the origin. We proceed as in Section 8.4.5 to distinguish between the two cases substituting the failure ratio λ with the gain g. In other words, we check if θ_p (the parent of θ_c) remains incompatible with ρ after removing the logs matching the child (θ_c): we remove the logs matching θ_c; then, we compute the gain of the incompatibility between θ_p and ρ. If the gain is higher than the threshold, this means that the parent θ_p is incompatible with ρ independently of the child θ_c. The incompatibility of the child θ_c with ρ could be seen as a result (inheritance) of the incompatibility of the parent θ_p with ρ. In this case, we remove θ_c from the graph. If, on the contrary, the gain drops below the threshold, this means that the child θ_c is the origin of the incompatibility with ρ. As the logs of the child θ_c

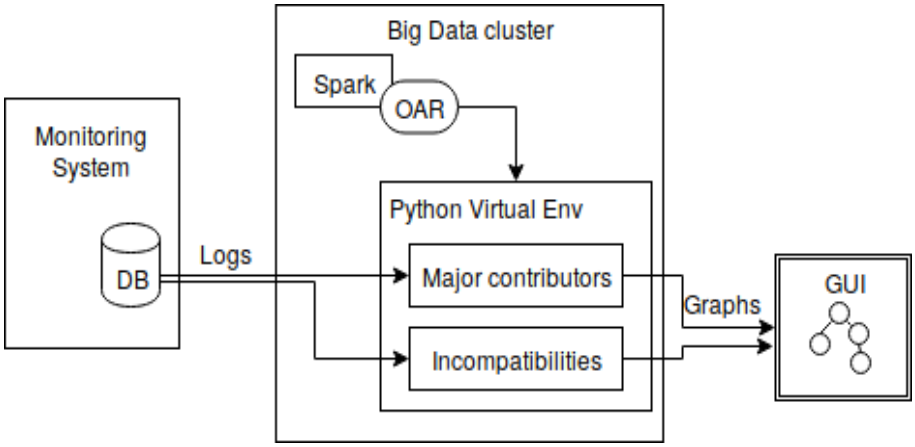


Figure 43: System Architecture

are a subset of the logs of the parent θ_p, θ_p appeared as incompatible with ρ. In this case, we remove θ_p from the graph. Every time a node is removed, its ancestors are connected to its successors. We iterate the pruning until we have only isolated nodes, which correspond to the signatures incompatible with ρ. Note that we execute the whole process for *each one* of the constituent 1-signatures identified in the previous step.

8.6 EVALUATION

We have implemented ARCD in a Big Data cluster and ran major contributor detection and incompatibility detection as Spark jobs scheduled by the task manager OAR as shown in Figure 43. The results are displayed in a Graphical User Interface (GUI). We evaluate the effectiveness and performance of ARCD thanks to three different data sets, collected via Hadoop Distributed File System (HDFS) from three different live cellular networks. After briefly presenting each data set, we analyze the major contributor detection results followed by the incompatibilities results. Finally, we compare the performance of ARCD with Learning from Examples Module, version 2 (LEM2), which is a well known algorithm for deducting decision rules from data.

8.6.1 *Data Sets*

We ran ARCD on three data sets from three different operators. The values of the features in the logs are anonymized by a hashing algorithm.

SET 1: SDRs recording TCP connections during one hour in a European country. A human expert has cleaned and analyzed this data set so that we can use it as a reference in the validation process. She removed samples with missing and inconsistent fields and truncated extreme values.

SET 2: SDRs recording TCP connections during one day from another European operator. This set is a raw one (missing values, inconsistent values). This set was not examined by an expert.

SET 3: CDRs logging voice calls during one day from an Asian operator. This is also a raw data set with no pre-specified root causes but where each entry is labeled as successful/failed.

Table 10 summarizes some of the attributes of the data sets. We used the re-transmission ratio to label each entry for Set 1 and the server response time for Set 2. These metrics were suggested by the experts working with the corresponding operators.

set	number of logs	number of features	number of 1-signatures	failure ratio
1	25 000	48	64 143	0.05
2	10 000 000	31	924 836	0.18
3	1 000 000	35	162 603	0.08

Table 10: Validation Data Sets

8.6.2 *Expert Validation*

We worked with three experts having more than ten years of experience in the domain of mobile network troubleshooting. These experts helped us by providing feedback on the results of our solution. They confirmed that ARCD extracts from the logs actionable information for network troubleshooting. They also appreciated the output being presented as graphs that are straightforward to understand, greatly simplifying and streamlining the analysis of the underlying issues and the planning of the mitigation actions.

8.6.3 *Major Contributor Detection*

To validate our results on Set 1, we compared the outcome of our solution with the list of issues identified by human experts. To validate our results on Sets 2 and 3, we have created an expert emulator, which mimics the manual analysis done by human experts. The emulator analyses a limited number of features (less than ten). For Sets 2 and 3, the experts supervising the networks of the corresponding operators have kindly indicated the features they are focusing on. For each feature, the emulator scans the top ten frequent elements (e.g., top popular services). If one element is more inefficient than the global network, the elements is identified as a major contributor. To pinpoint inefficient elements, the expert emulator calculates the following metrics for each element and compares them to the the values of the whole network: failure ratio for CDRs (Set 3); Retransmission ratio (Set1) and server response time (Set 2). These metrics are the same we used in the data labeling phase of ARCD. The concept of hierarchical dependencies is implicitly included in the expert analysis: Experts start with high level elements (e.g., devices in the core network) and then move down to lower levels (e.g., users).

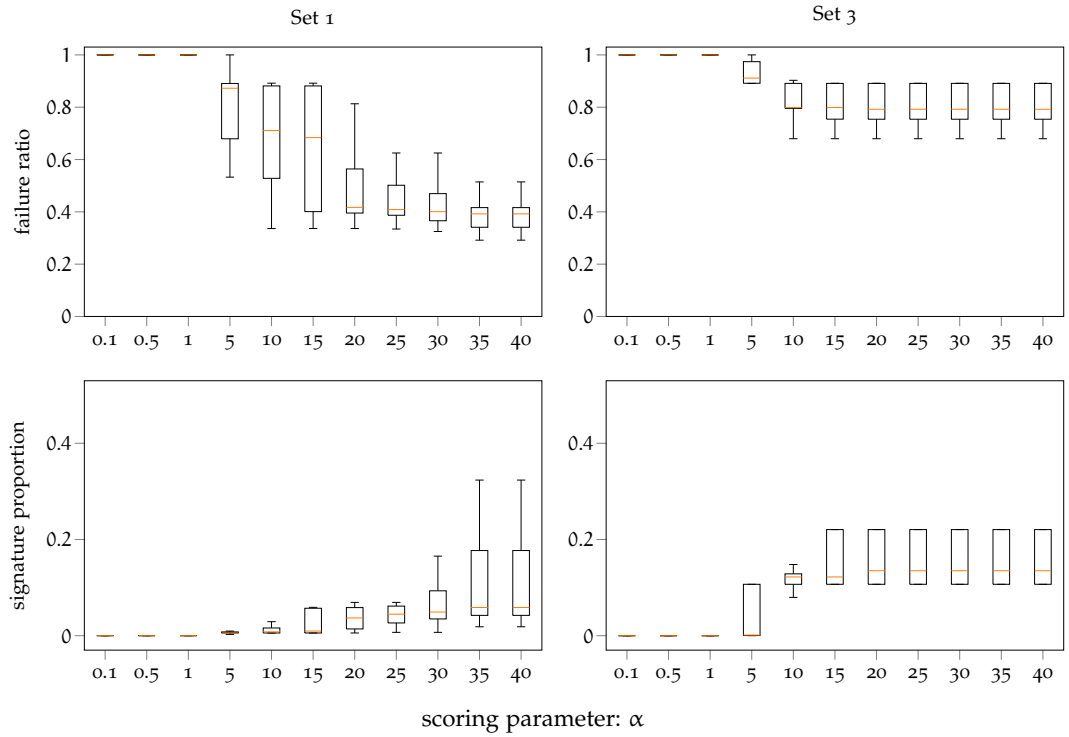


Figure 44: Box-plots of failure ratio and signature proportion as function of scoring parameter for Sets 1 and 3

8.6.3.1 *Parameter Tuning*

As explained in Section 8.4.2, we used a range of values for α to find the top signatures. Figure 44 shows the distribution of the signature proportion and the failure ratio of the top twenty ranked signatures for each value of the scoring parameter α for Sets 1 and Set 3. As we state in Section 8.4.2, the higher α, the higher the signature proportion (corresponding to the most used devices and services). Figure 44 shows also that the smaller α, the higher the failure ratio. This is not surprising as small values of α correspond to the most inefficient elements. As one can notice, for both SDRs and CDRs, the distributions have similar trends. By scanning an interval containing a large range of values for α, we find in practice that we identify the most significant problems, which are feature-value pairs with a sufficiently high number of occurrences to deserves attention and a sufficiently high number of failures suggesting a possible malfunction.

8.6.3.2 *Accuracy*

To evaluate our solution, we select the following metrics:

TRUE POSITIVE (TP): inefficient elements detected by ARCD and validated either by the expert (Set 1) or by the emulator (Sets 2 and 3).

FALSE NEGATIVE (FN): inefficient elements detected either by the expert (Set 1) or the emulator (Sets 2 and 3) but not detected by ARCD.

FALSE POSITIVE (FP): efficient elements detected by ARCD but not detected in the validation process because their inefficiency is no greater than the overall inefficiency of the network.

EXTRA FEATURES (EF): inefficient elements detected by ARCD but not detected in the validation process because of the limited number of features analyzed by experts due to time constraints.

EXTRA VALUES (EV): inefficient elements detected by ARCD but not detected in the validation process because experts analyze only the top 10 frequent elements of each considered feature.

PRECISION: The ratio of inefficient elements correctly identified by ARCD (TP+EF+EV) to the total number of elements identified by ARCD (TP+FP+EF+EV).

RECALL: The ratio of inefficient elements correctly identified by ARCD (TP+EF+EV) to the total number of inefficient elements (TP+FN+EF+EV).

	TP	FN	FP	EF	EV	Precision	Recall
Set 1	11	2	0	38	1	1	0.96
Set 2	5	2	5	30	10	0.9	0.95
Set 3	4	1	0	30	16	1	0.98

Table 11: Major contributor results

Table 11 shows the overall performance of ARCD, which is satisfying in terms of TPs and FNs. The interesting aspect of ARCD is its capability to detect issues that are not identified by experts since they focus only on highly frequent elements (such as handset types, services, and core network equipment) due to time constraints. For this reason, they miss issues occurring at a finer level of granularity, which ARCD does detect, such as roaming issues, bad cell coverage and individual users (bots) submitting a large number of call requests to unreachable numbers. The major contributors identified with ARCD explain respectively 91%, 82% and 65% of the failed logs of sets 1, 2 and 3.

8.6.3.3 *Output as a Graph*

Figure 45 shows an example of the output of ARCD. It is a portion of the graph created based on data from Set 2. The criterion for SDR labeling is the server response time: SDR labeled as failed if response time is larger than 100 ms (the average server response time for the whole network is 60 ms). The nodes contain signatures detected as major contributors. Each node contains the features, a hash of their values (for confidentiality reasons) and two numbers: the number of logs matching the signature and the average response time of the logs matching the signature. We give the response time rather than the failure ratio to ease the interpretation of the graph. The labels on the edges contain the log size and the average response time of the set of logs matching the parent signature and not matching the child signature. The nodes filled in blue are the nodes removed during the pruning process because they denote false problems.

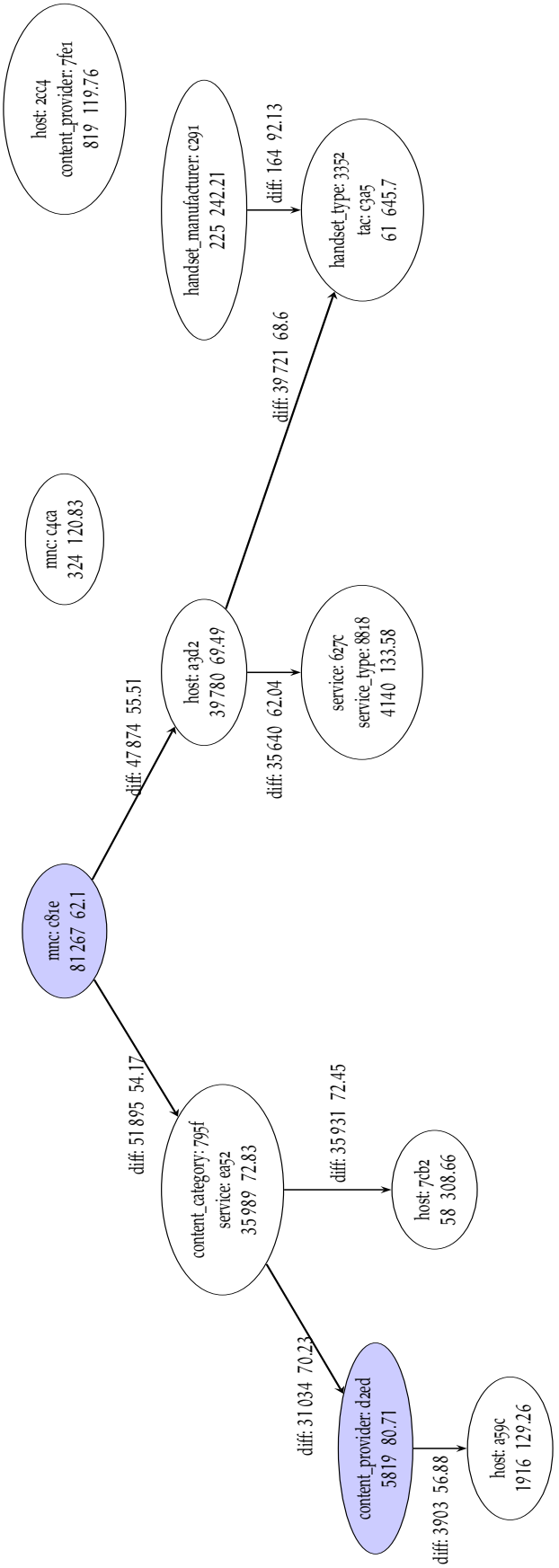


Figure 45: Pruned Graph of Major Contributors for Set 2, nodes in blue are removed during the pruning process

The graph points to two individual problems: a roaming issue (mnc: c4ca); and a content provider issue (host: 2cc4, content_provider: 7fe1). There is also a set of codependent problems with the root of the graph (mnc: c81e). This MNC has a large number of logs and a response time slightly higher than the overall network. By detecting this MNC, one may think of a roaming issue. However by removing one of its child nodes, we see that its average response time drops below the average value of the network. That is why this issue was tagged as a false problem and was removed in the pruning step. The same reasoning applies to the Content Provider: d2ed. The node (handset_type: 3352, tac:c3a5) has two parent nodes: (handset_manufacturer: c291) and (host: a3d2). This node is kept in the graph because it is significantly more inefficient than its two parents.

8.6.4 *Incompatibility Detection*

Validating incompatibilities is more complex than validating major contributors. Incompatibilities are fine-grained issues, which experts do not address on a regular basis. While investigating incompatibilities requires a lot of time and effort, the number of impacted subscribers is generally low. To validate our results, we relied on expert help. After testing our solution on the data sets, we presented our results to network management experts who evaluated the accuracy of the results.

8.6.4.1 *Accuracy*

To evaluate our solution, we use two metrics TP and FP. A TP is an incompatibility detected by ARCD and confirmed by experts to be a real incompatibility, worthy of being reported and investigated. A FP is an incompatibility detected by ARCD whose high inefficiency was actually due to a third feature, detected by the experts. We measure also the precision which is the ratio of TP to (TP+FP).

Set	**TP**	**FP**	Precision
1	10	1	0.91
2	15	2	0.88
3	6	1	0.86

Table 12: Incompatibility results

The overall performance of our solution is satisfying (see Table 12). The issues reported are confirmed by experts to be relevant and deserving attention. In the case of SDRs, we have mainly detected incompatibilities between services/content categories with handset types/OS versions. For CDRs, we have highlighted issues in multi-RAT calls, inaccessible destinations in a specific RAT and issues with the location update procedure in some cells. The incompatibilities detected by ARCD explain respectively 4%, 16% and 7% of the failed logs of sets 1, 2 and 3.

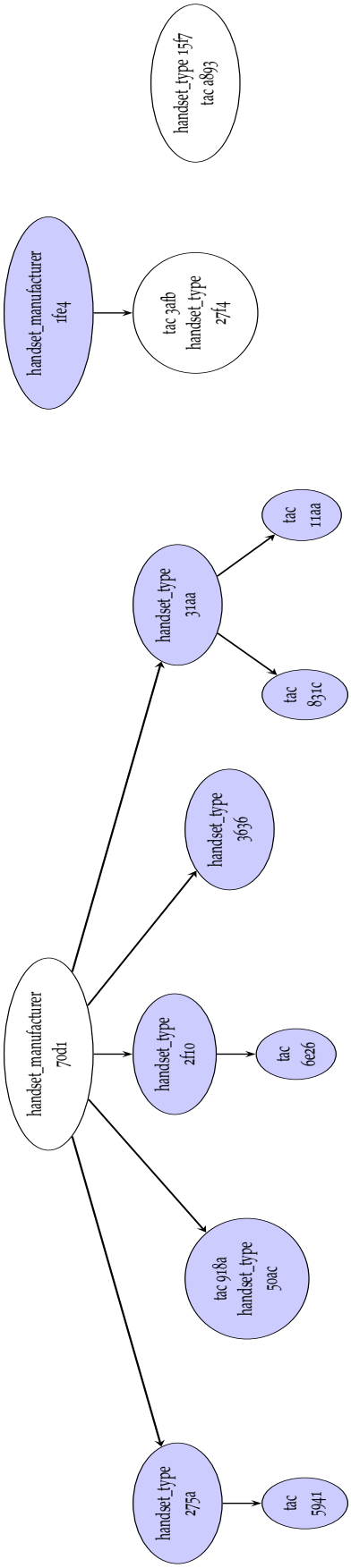


Figure 46: Incompatibilities pruned graph of the 1-signature (content_category, 795f), blue nodes are removed during the pruning step

8.6.4.2 *Output as a Graph*

Figure 46 is an example of part of the output of our solution for the incompatibilities in Set 2. Recall that, for each 1-signature ρ, we generate a graph of the elements incompatible with ρ. Figure 46 is the graph of the elements incompatible with the 1-signature (content_category: 795f). The blue nodes are the nodes removed during the pruning step. The issues highlighted here are incompatibilities between a content category and a handset. The graph shows the three families of handsets (three connected components of the graph) that are incompatible with (content_category: 795f):

- The first connected component starting with (handset_manufacturer,70d1) as incompatible with (content_category: 795f): Many handset types and Type Allocation Codes (TACs) of this same handset manufacturer are incompatible with (content_category: 795f) as well. These handset types and TACs are removed during the pruning process (colored in blue). The root of the incompatibility with (content_category: 795f) is (handset_manufacturer: 70d1). The incompatibilities between the handset types and the TACs with (content_category: 795f) result from this root incompatibility.
- The two connected nodes (handset_manufacturer: 1fe4) and (tac: 3afb, handset_type: 27f4): In this case (tac: 3afb, handset_type: 27f4) is the origin of the incompatibility with (content_category: 795f). The incompatibility between (handset_manufacturer: 1fe4) and (content_category: 795f) results from it.
- The isolated node (handset_type: 15f7, tac: a893): This handset type is simply incompatible with (content_category: 795f).

8.6.5 *Comparison with the LEM2 Algorithm*

To evaluate our solution, we compare it with the LEM2 algorithm [47], which is a rule deduction algorithm. LEM2 creates decision rules from a multidimensional categorial data set. It splits the data set into subsets with regard to the decision variable (the successful/-failed label in our case). For each subset, LEM2 identifies the most frequent signatures that do not match samples (logs in our case) in any other subset in an incremental way. More details are presented in Appendix D. LEM2 is well suited to our use case: First, it handles inconsistent data, with the same feature-value pairs being sometimes successful and sometimes unsuccessful as in our use case. We aim to detect not only breakdowns ($\lambda = 1$) but also inefficient elements ($0 < \lambda < 1$). Second, it generates a minimal and non-redundant rule set. LEM2 identifies the root of the issues and does not report additional rules related to feature dependencies.

8.6.5.1 *LEM2 Results*

We ran the LEM2 algorithm on the same three data sets as ARCD. We have made the following two changes[1] to the LEM2 algorithm in order to obtain meaningful results in a reasonable amount of time:

1 The modified algorithm is available at https://github.com/mdini/pycon_lem2/blob/master/lem2/lem2_naive.py

- We stop generating rules when we have explained more than 90% of the failed logs. The idea behind LEM2 is to generate rules from the most important to the least important. As the execution of LEM2 progresses, the rules cover fewer and fewer logs. At the end of the execution, we may end up with a thousand of rules each matching only one log. By adding this breaking condition, we reduce considerably the execution time. We also discard insignificant rules.
- In the process of generating a rule, we stop adding new elements when the rule matches less than 200 logs. Otherwise, the iterations would lead us to have all the rules with high failure ratios λ but small signature proportions π. We would detect breakdowns but miss inefficiencies.

rule	**count**	λ
[(roaming, e4c2), (first_location_type, 0707), (content_provider, 795f)]	48043	0.67
[(service, ea52), (content_provider, 795f), (handset_type, fb38), (src_node_ip_str, 93e3)]	281	0.49
[(service, ea52), (content_provider, 795f), (dest_node_ip_str, a106)]	21926	0.52
[(mcc_mnc, 6364), (roaming, e4c2), (serving_mcc_mnc, 6364), (last_location_type, 0707), (first_location_type, 0707), (dest_node_ip_str, 2f2a), (host, a59c)]	3074	0.92
[(mcc_mnc, 6364), (first_location_type, 0707), (dest_node_ip_str, 79dd), (host, a59c)]	2557	0.90

Table 13: LEM2 example rules

Table 13 shows some of the rules that LEM2 found in Set 2. Each rule is a k-signature, for which the failure ratio is high. The count column shows the number of logs matching each rule and the last column gives the failure ratio.

8.6.5.2 *Accuracy Comparison*

To evaluate the results of the LEM2 algorithm, we used the same metrics as in Section 8.6.3.2. Table 14 shows the corresponding results. The last column contains the number extra features and values (EF+EV) that LEM2 finds and that were not detected by the experts. We added EF and EV because some of the LEM2 rules contain both EF and EV.

	TP	**FN**	**FP**	**EF+EV**	Precision	Recall
Set 1	5	8	7	13	0.72	0.69
Set 2	7	0	11	43	0.82	1
Set 3	3	2	1	16	0.95	0.9

Table 14: LEM2 accuracy results

The numbers of FP and FN are high due to two reasons: first, to deduct the rules, LEM2 conducts an analysis in the set of failed logs without comparing the set of failed logs and the set of successful logs. For this reason, we have many false positives. Second, during the execution of LEM2, every time a rule is generated, its matching logs are excluded from the analysis. This may be the reason behind false negatives as the sets of logs related to two independent issues may overlap.

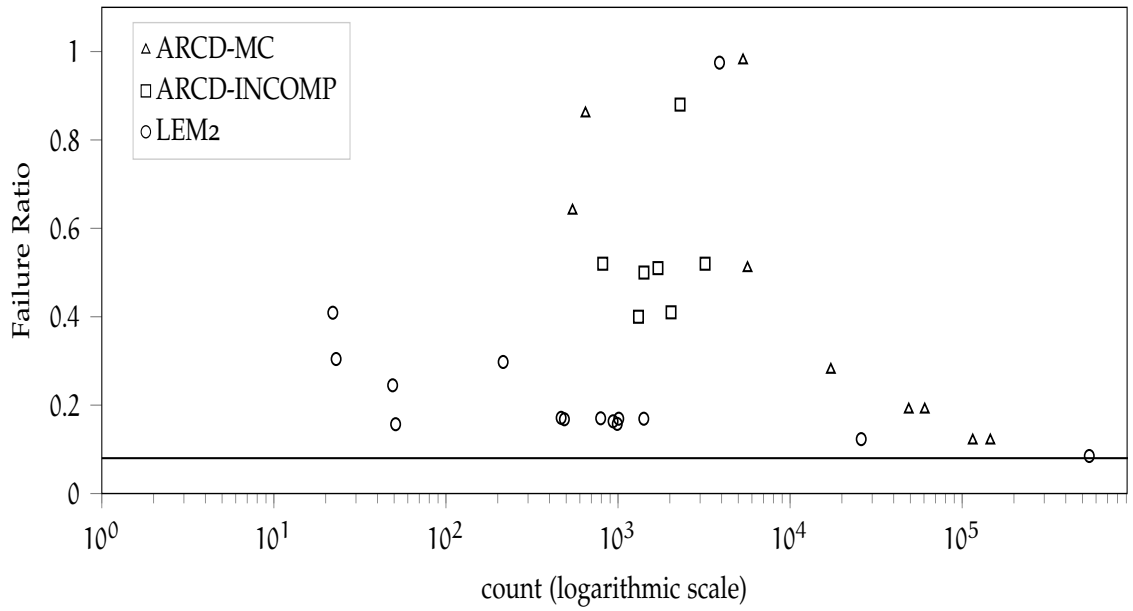


Figure 47: Output from Set 3, signatures identified by ARCD-MC (major contributors), by ARCD-INCOMP (incompatibilities), and LEM2. Each point corresponds to one signature, with its failure ratio and the number of related logs.

Figure 47 shows the results obtained by ARCD and LEM2 on Set 3. Each point is a k-signature, the x value is the number of logs matching that signature and the y value is its inefficiency. ARCD-MC refers to the signatures detected by the major contributor detection process. ARCD-INCOMP refers to the signatures detected in the process of incompatibility detection. LEM2 refers to the signatures identified by LEM2. The horizontal line shows the inefficiency of the whole network. The figure shows that, in the specific use case of root cause diagnosis in mobile networks, ARCD outperforms LEM2: Most of the signatures identified by ARCD have a higher failure ratio and a higher number of logs compared to LEM2 signatures. We can also observe that the identified incompatibility signatures are more inefficient but less frequent, in general, than the signatures of major contributors.

8.6.5.3 *Time Complexity*

Let m be the number of features, k the maximum number of value per feature and n the number of logs. As the dependency analysis (equivalence classes, hierarchical dependencies and pruning) is only performed on a limited number of signatures (top fifty), its complexity is then $O(1)$. Thus, the time complexity of major contributor detection in our solution is equal to the complexity of top signature identification which is equal

to $O(mkn)$. As $m \ll n$, we can approximate the complexity to $O(kn)$. So, in the worst case, where a feature has a different value for each log, the complexity would be $O(n^2)$. Otherwise, in general, it is approximated to $O(n)$. The same reasoning applies to incompatibility detection. Its time complexity is equal to $O(m^2k^2n)$. In the worst case, it is $O(n^3)$ and otherwise, $O(n)$. As explained in [82], the complexity of the LEM2 algorithm is approximated to $O(n^3)$.

9 INDUSTRIALIZATION

In this Chapter, we explain how we industrialized Automatic Root Cause Diagnosis (ARCD) in Nova Analytics™ to carry out online root cause diagnosis. Then, we give an overview of a complete Big Data troubleshooting solution combining Watchmen Anomaly Detection (WAD) and ARCD. The execution of ARCD is triggered by WAD: Once an anomaly is some Key Performance Indicators (KPIs), the diagnosis will run automatically. The implementations covered in this chapter are still ongoing. Nevertheless, we have some interesting preliminary results. We have tested ARCD on local mode and our first results were validated by telecommunication experts.

9.1 ONLINE ROOT CAUSE DIAGNOSIS: NOVA ANALYTICS™

In this section, we give an overview of the implementation of ARCD in a live network. Like WAD, ARCD is implemented as a plugin of Nova Analytics™ to detect both major contributors and incompatibilities. We also show some preliminary results of the implementation.

9.1.1 *The scoping phase*

As for WAD, ARCD is implemented as a plugin of Analytics™. ARCD analyzes Call Data Records (CDRs) and Session Data Records (SDRs) to identify major contributors and incompatibilities in the network. The requirements of parallel processing, low error rate and the ease of maintenance and configuration are also demanded. The plugin is not supposed to run in real time. It will be triggered by the anomaly detection plugin as it will be explained in Section 9.2.

9.1.2 *Preliminary results*

In this paragraph, we present some preliminary results as the implementation is still ongoing. We implemented both major contributor detection and incompatibility detection. The results hereafter are obtained by running the plugin on local mode applied to real data.

MAJOR CONTRIBUTORS Figure 48 shows the major contributors to the network overall inefficiency. Feature 1, 2, 3, 4 are the different levels of the graph generated with ARCD: i.e. feature 2 is the parent of feature 3 and feature 3 is the parent of feature 4. Feature 1 is hidden as it is a common node on the top of the graph added artificially to have a connected graph. This node is named "Impacted Dimensions", see Figure 49, first column. The orange horizontal bar labeled "Countfeat" is the number of SDRs matching the signature and the vertical blue line "DeltaIneff" shows the difference between the

failure ratio of the signature and the overall failure ratio of the network. In this figure, we see that we detect service related issues (hosts, services, content providers). The identified signatures match an important number of SDRs and their failure ratios are higher than the network global failure ratio. The identified signatures match the definition of major contributors and these first results are promising. Figure 49 is another view of the same results. Each column is a level of the graph: in each column, we see parent nodes and their direct children. The thickness of the arrows is based on the importance of the signature (the number of times the signature appears in different α rankings). The thicker the arrow, the higher its contribution to the network inefficiency.

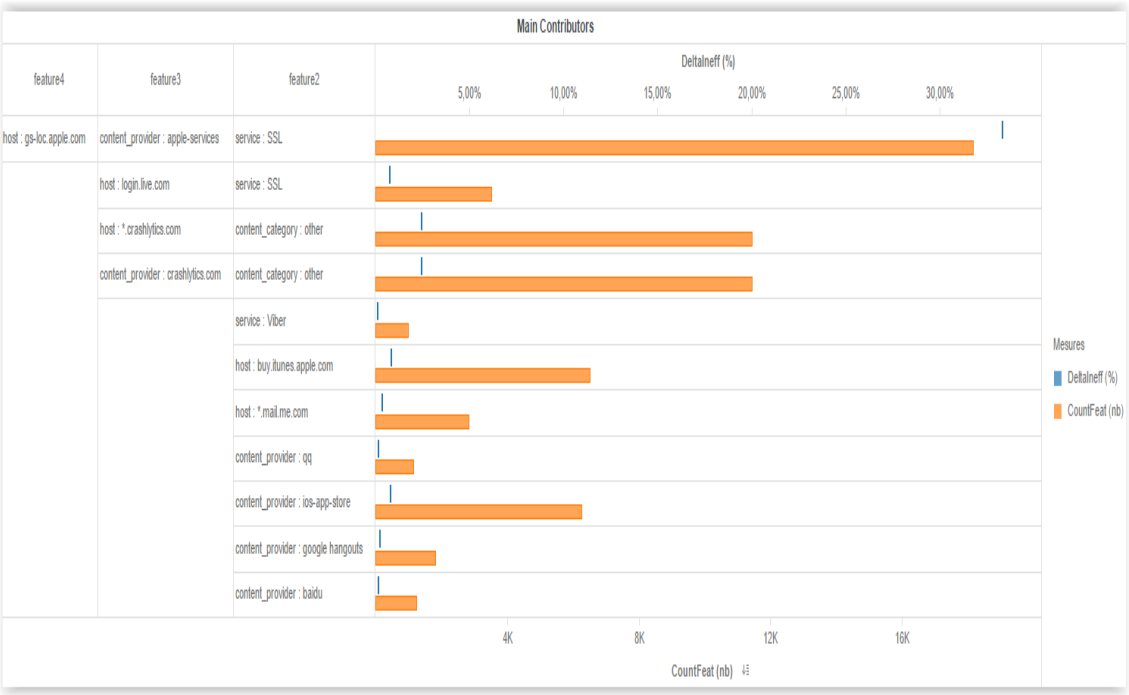


Figure 48: Overview of major contributors to KPI degradation on User Plane

INCOMPATIBILITIES Figure 50 gives a flattened view of detected incompatibilities. The thickness of the arrows is proportional to the gain of the combination. In this figure, we see that we detect different types of incompatibilities:

- handset type/manufacturer - service/service type
- handset type/manufacturer - content provider/category
- service/service type - Mobile Country Code (MCC) Mobile Network Code (MNC)
- content provider/category - OS type/version

These incompatibilities are interesting from a User Plane (UP) perspective as the UP carries the user data (used handset, requested service).

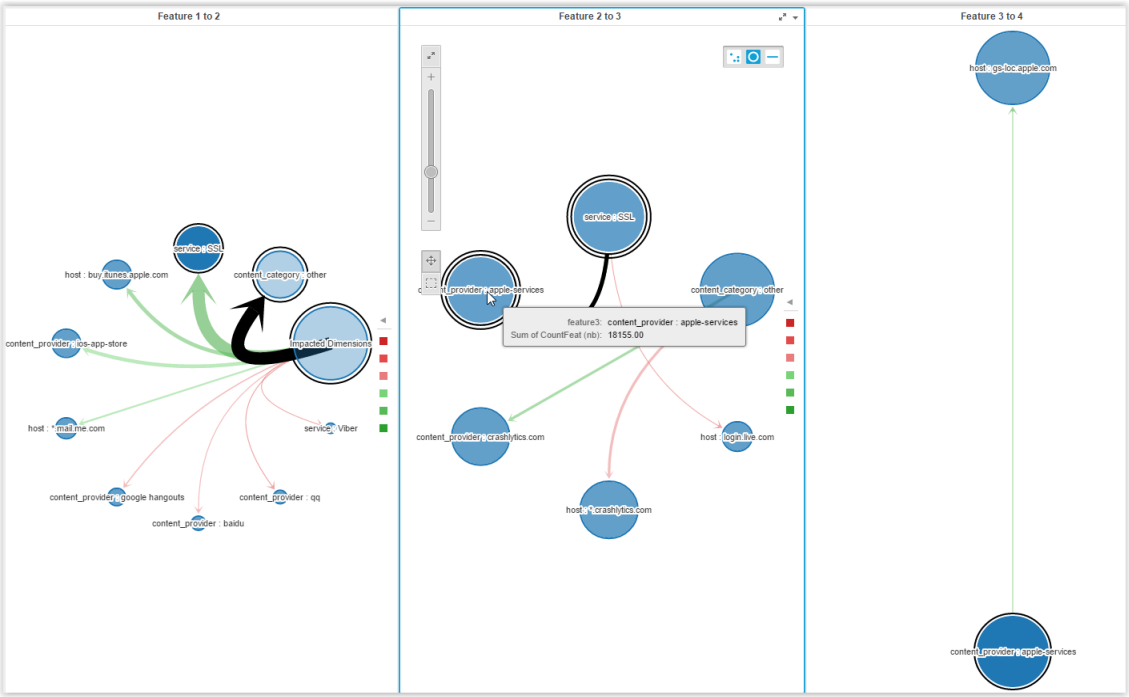


Figure 49: Multi-level view of major contributors to KPI degradation on User Plane

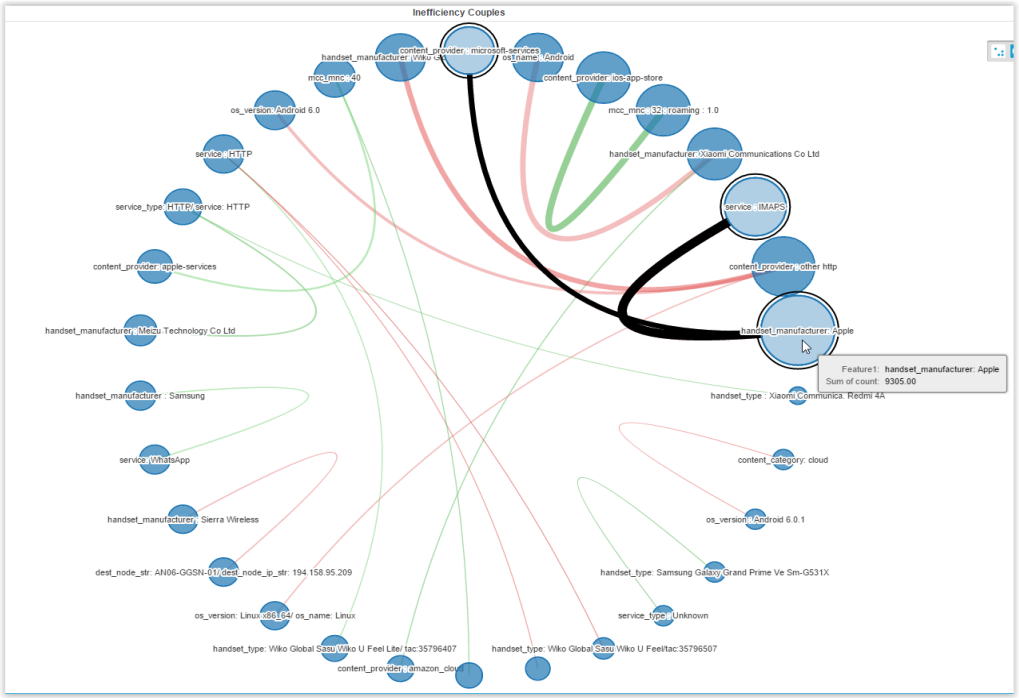


Figure 50: Incompatibilities behind latency issues on User Plane

9.2 A COMPLETE BIG DATA TROUBLESHOOTING SOLUTION: WAD-ARCD

In this section, we explain how we combine WAD and ARCD to have a complete troubleshooting solution in Nova Analytics™ solution. As described in Figure 51, WAD and ARCD are implemented in the Big Data platform that collects data from the monitoring system. The anomaly detection plugin applies WAD to the KPIs computed in the monitoring solution. Once an anomaly is detected, the root cause diagnosis plugin is activated. It identifies the set of major contributors and incompatibilities related to the corresponding KPI. A ticket is then automatically created with faulty elements. The experts are involved, at the end of this process, to take compensation and recovery actions.

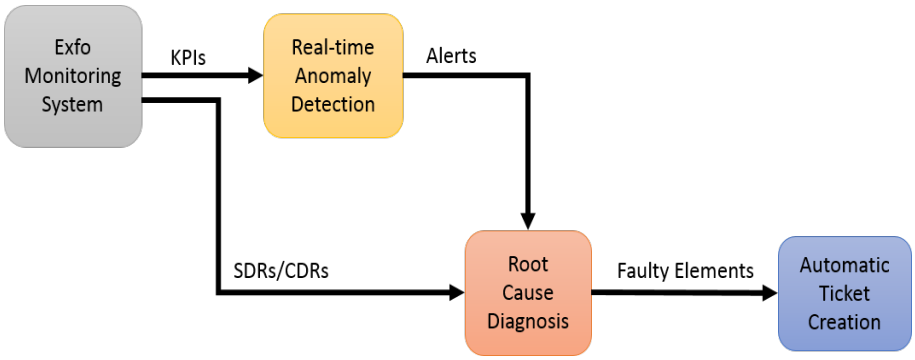


Figure 51: A troubleshooting solution with WAD and ARCD

9.3 ROOT CAUSE DIAGNOSIS TIMELINE

We started the project of root cause diagnosis in April 2017. Figure 52 shows the timeline of this project. In June 2017, we started the design of ARCD. Then, we developed a prototype and proceed to performance testing on few data sets. The industrialization of ARCD as part of the Analytics™ solution started in November 2018. The goal of this integration is to conduct massive testing of ARCD in real networks.

Figure 52: The timeline of ARCD development and industrialization

9.4 CONCLUSION

In this part, we have proposed a solution for root cause analysis, ARCD, to detect not only major contributors to the network inefficiency but also incompatibilities. The implementation of ARCD in a commercialized product is ongoing. The tests we run on local mode has proved that ARCD provides useful information for network troubleshooting. WAD and ARCD will be combined to create a fully automated solution that detects anomalies in the network and analyses their roots.

Part IV

CONCLUSION

10 RESULTS

This thesis consists in the elaboration of two algorithms as part of the implementation of the self-healing process in mobile networks. In the following, we will present the main outcomes of this work.

10.1 CONTRIBUTIONS

This thesis aims at contributing to the implementation of self-healing networks. In this thesis, we have proposed an algorithm of anomaly detection: Watchmen Anomaly Detection (WAD). This algorithm allows the detection of anomalies in periodic times series characterizing the network traffic based on the concept of pattern recognition. WAD is fully unsupervised and learns autonomously from the flow of data. We have also worked on the question of root cause diagnosis in mobile networks and we have elaborated a solution, Automatic Root Cause Diagnosis (ARCD), allowing an End to End (E2E) diagnosis of network inefficiencies. ARCD is composed of two independent processes: a process of identifying major contributors to the network overall inefficiency and a process of detecting incompatibilities. Both WAD and ARCD has proven their effectiveness in automating the healing process, reducing expert effort, and providing more accurate results.

10.1.1 *Anomaly detection*

We have proposed and validated an Anomaly Detection System (ADS) for network monitoring based on pattern recognition principles. WAD builds a dynamic model of periodic time series and detects anomalies in real time. This model is based on a data transformation highlighting abrupt changes in traffic. After transforming the raw data, WAD implements a threshold-based trigger to detect anomalies. We have tested the performance of our system and the results demonstrate that our system is accurate, efficient, and well suited to the type of data that we deal with which is periodic time series. Two operators have tested our solution to supervise their networks and both confirmed that WAD has raised the productivity of their network administrators.

We are working on improving WAD by using the Hurst exponent to automatically compute the error term (tolerance band). We also plan to improve the data transformation. The Difference over Minimum (DoM) transform is finely tuned to amplify the type of anomalies we want to detect for network monitoring. Ideally, we would like to be able to automatically generate data transformation operations that can amplify given events.

10.1.2 *Root cause diagnosis*

We have addressed the problem of automating the system that diagnoses cellular networks based on the data collected from large-scale monitoring systems, proposing a

framework for root cause diagnosis in cellular networks. ARCD is unsupervised: it does not only automate expert analysis, but it carries it to a deeper level as it deals with subtle issues such incompatibilities. Experts can not detect these issues as they require huge time and effort. Our tests, as well as the feedback from experts, show that we have promising results. In comparison to previous work, ARCD can run on a large number of features with categorial values, to identify the complex interplay between various features, and to provide an overview of the main identified malfunctioning devices and services which can easily be double-checked by experts.

The three experts who have studied ARCD were extremely positive about the advantages it brings in their daily tasks. They highlighted the benefits of grouping elements related to the same problem into equivalence classes. They also appreciated the capacity of ARCD to identify occasional dependencies, and the time gain provided by the statistics-based hierarchical dependency discovering instead of manually updating the topology every time the network is modified.

The ARCD solution is also a starting point for a series of significant future studies. One of the first studies to conduct concerns parameter tuning and data labeling. Typically Session Data Records (SDRs) logs still require an expert to decide whether the logs report a successful experience. We aim to improve ARCD by reducing the number of parameters and finding a more efficient way to label data. We would also like to link our root cause diagnosis framework to an anomaly detection system within the same monitoring platform. This way the anomaly detector would trigger the root cause diagnosis process, ideally in real time, this being a key missing element towards an integrated diagnosis system. Finally since some sessions matter more than others, we would like to explore new inefficiency definitions. This selection of open challenges and research questions paves the way toward fully autonomous self-healing mobile networks, having the capacity to diagnose the problem, to identify the root cause, and to fix the problem by reallocating resources or by updating software. We hope that reporting the choices that have made ARCD a successful network management product will inspire other researchers in the area.

10.2 PUBLICATIONS

The project of anomaly detection has led to the publication of a paper [98] in the Symposium on Integrated Network and Service Management. The project of root cause diagnosis resulted in the paper [99] published in the International Conference on Network and Service Management. The latter contains only the solution of major contributor detection. Another paper describing the full ARCD solution, untitled "Towards Self-Healing Mobile Networks: Introducing an Unsupervised Automated Solution for Root Cause Diagnosis" was submitted to the Transactions on Network and Service Management. We have also a patent pending on the topic of root cause diagnosis.

10.3 COMMERCIALIZED PRODUCTS

As explained in Chapter 6, WAD was integrated as a plugin in two different EXFO products: Nova Cockpit™ and Nova Analytics™. In Cockpit™, WAD as a complement to

the monitoring system. In concrete terms, it detects anomalies in the input/output of network probes and other monitoring tools. In Analytics™, WAD is used to manage service quality by detecting anomalies in service quality indicators. In Chapter 9, we have shown briefly how ARCD was integrated in Analytics™ to diagnose issues at different points of the network. This implementation is still ongoing.

11 FUTURE WORK

In this thesis, we have presented two algorithms relevant to the question of self-healing networks. The first algorithm, Watchmen Anomaly Detection (WAD), allows to detect anomalies in periodic time series. The second algorithm, Automatic Root Cause Diagnosis (ARCD), allows to diagnose the network and identify the root causes of issues. Despite the efficiency of our solutions, they still have shortcomings. In the following, we will propose some potential improvements to our algorithms. Then, we will finish with research questions that we started studying in this thesis.

11.1 POTENTIAL IMPROVEMENTS

Regarding anomaly detection, our solution WAD has promising results. However, many improvements are possible. First, WAD has a few parameters to be tuned manually. Automating the parameter setting would make the solution applicable to different use cases without any human intervention. Second, the Difference over Minimum (DoM) transform we have adopted in WAD allows the detection of abrupt changes in time series. Ideally, a complete anomaly detection solution should be able to detect different types of anomalies. Third, our solution works on periodic data and is applicable to constant time series. However, it does not fit chaotic time series. A general solution of anomaly detection in time series would cover both periodic and chaotic time series. An extra work may be to adapt WAD to detect patterns in chaotic time series.

Our solution of root cause diagnosis, ARCD, has proven its efficiency in identifying inefficiency major contributors and incompatibilities. Nevertheless, it is still limited in many aspects. First, ARCD has few parameters that needed to be set by experts and empirical tests. Second, in the case of features having a large number of possible values such as International Mobile Subscriber Identitys (IMSIs), the execution time is polynomial. Some optimizations may be carried out to address this specific case. Third, ARCD detect incompatibilities involving two elements. While the incompatibility detection process can be applicable to identify incompatibilities involving three or more elements, the time complexity may grow exponentially. Last but not least, to identify the root of issues, ARCD explores equivalence classes (mutually dependent elements) and hierarchical dependencies. Other types of dependencies could be added to the process to have a more robust and precise diagnosis.

11.2 NEXT STEPS

As a future work, we will focus on automating the compensation and the recovery processes to have a fully self-healing network. The compensation task consists in determining a quick and temporary operations to circumvent network issues and restore service access as fast as possible. For example, one may modify the route of the traffic

to bypass a non-functional device. The recovery process consists in finding a permanent solution to network issues such as replacing broken equipment. We plan also to work in a more pro-active manner, which means that we would like to predict anomalies and different network issues before they happen. This would be possible if we study the sequence of events preceding network issues. We would also like to extend the scope of our analysis to cloud infrastructures and computer networks in general. We aim to test and validate our solutions on these types of networks. Regarding the data analysis aspect of our work, we plan to extend our work to manipulating unstructured data such as system event logs. Another interesting point is the use of Data Assimilation which combines data analysis with theoretical modeling. It creates dynamic models that do not diverge from the real state of the system by combining a mathematical model with real time observations.

11.3 OPEN RESEARCH QUESTIONS

This thesis contributes to the understanding of some research questions, which are not fully addressed by the scientific community. The main question behind this work is the concept of *Self-healing networks*. Despite the many specifications that describe the functioning of self healing network, current mobile networks still rely importantly on human experts in the troubleshooting tasks. Another question relevant to this work is the *universality of anomaly detection*. While the literature contains hundreds if not thousands of anomaly detection solutions, each domain still has its specificity. Therefore, there is no universal solution applicable independently of the context. The question of *End to End (E2E) network diagnosis* is also tightly related to this work. This question is still open as devices and applications produce heterogeneous data structures that cannot be assimilated and processed by monitoring systems without the help of experts.

Part V

APPENDIX

A

DEVELOPMENT ENVIRONMENT

We describe, in the following, the development environment and the testing environment used in the implementation of Watchmen Anomaly Detection (WAD) and Automatic Root Cause Diagnosis (ARCD).

A.1 GENERAL DESCRIPTION

We created a development environment for the prototyping phase of anomaly detection and root cause diagnosis. The choice of the tools is based on practical considerations. We started with the Linux based Operating System (OS) Ubuntu version 16.04, upgraded then to 18.04. To process data, we used the programming language Python 2.7 and 3.5. We used the Integrated Development Environment (IDE) PyCharm version 2016.3.3. For data exploration, we used the Interactive Python (IPython) based application Jupyter 4.4.0. The data used in the prototyping are sample stored in Comma Separated Values (CSV) files and a MySQL database version 14.14 distribution 5.7.26.

A.2 PYTHON LIBRARIES

The high level programming language Python offers a large number of data analysis libraries. These libraries are open source mostly under an Massachusetts Institute of Technology (MIT) or a Berkeley Software Distribution (BSD) licenses. To install these libraries, we used the package manager PIP Installs Packages (PIP). We created an isolated Python environment via the tool Virtualenv containing all the needed packages and their dependencies. Having a virtual environment allows multiple versions of the same package to coexist in the same system without interfering. A Python virtual environment is also easily replicated. Table 15 contains an overview of the libraries we used in our implementation along with the corresponding versions.

A.3 BIG DATA ENVIRONMENT

To validate our algorithms, we used a Big Data cluster designed for large scale data processing. The parallel processing by the different nodes of the cluster is coordinated by Spark running on the top of Hadoop. The spark jobs are scheduled by the task manager OAR. Spark distributes the multidimensional calculations on several nodes allowing a significant decrease of the execution time. The spark jobs we ran on this platform are python scripts. We ran them either as interactive jobs or passive jobs. In both cases, we use a logging function to track the evolution of the process.

Package	Version	Description
Pandas	0.24.2	Pandas is an efficient high level data analysis package. It offers fast, flexible, and easy to manipulate data structures, namely "DataFrame" and "Series".
Numpy	1.16.3	Numpy is a high level multidimensional scientific computing. It implements calculations on multidimensional arrays and matrices.
Scipy	1.1.0	Scipy is a scientific computing library based on Numpy. It implements linear algebra, statistics and signal processing functions.
Scikit-learn	0.20.3	Scikit-learn is an ML library. It implements regression, classification, and clustering algorithms as well as some preprocessing operations such data cleaning and interpolation.
Networkx	2.1	Networkx is a library used to manipulate complex networks. It implements standard algorithms for graph analysis.
Matplotlib	2.1.1	Matplotlib is a data visualization library. It implements different types of plots (scatter plots, histograms, pie charts, etc). It offers many features to power users to control the style of the plots.
Seaborn	0.9.0	Seaborn is a data visualization library based on Matplotlib. It implements statistical graphics such as joint distributions. Seaborn processes the statistical functions internally and return only the final plot.

Table 15: Data Analysis Python Libraries

B

Symbolic Aggregate Approximation

Symbolic Aggregate Approximation (SAX) is an algorithm that transforms a numerical time series to a sequence of strings. This transformation aims at reducing the size of the data set and removing noise. SAX offers an efficient data representation that can fit classification and forecasting purposes. SAX is applicable on timestamped data in the context of sequential analysis. SAX has two main steps:

- Piecewise Aggregate Approximation (PAA)
- Transforming PAA output into sequences of symbols/letters.

B.1 Piecewise Aggregate Approximation

PAA takes as an input a time series of length n. Then, it splits the time series into w sequences of the same size. We compute afterwards, the mean of each sequence and create a new times series of length w containing the calculated averages. This step has two main advantages. First, it reduce the size of the time series from n to w which alleviates the processing load. Second, it is robust to noise and outliers: The obtained time series is smoother than the original one. It has less significant peaks and dips. The moderation of extreme values at the early stages of the processing makes the data modeling more accurate.

B.2 transformation into symbols

This step consists in converting numerical data into a sequence of symbols. The symbols are equiprobable and their number k is predefined. k controls the sensitivity of the algorithm. We start with the assumption that the values of the time series after the PAA transformation have a continuous Gaussian distribution. We split the Gaussian curve into k equiprobable intervals. Each interval is represented by a symbol: a, b, c.... Then, we map each time series value to the symbol representing its corresponding interval. This process transforms the time series into a sequence of symbols. This sequence could include some repetitive patterns that are more visible with this discretization.

B.3 SAX applications

SAX is useful as a first processing step preceding another clustering/forecasting algorithm. As SAX reduces the size of data and limits the number of possible symbols, it allows the application of complex algorithms that take a large number of iterations before convergence. Another important aspect of SAX is its ability to highlight patterns in the data. These patterns can be clustered for classical classification aims or for anomaly detection. The patterns can also be used to build a tree that models the variations of the

time series. By navigating the tree structure, one may predict the next symbol which is equivalent to the interval of the next value.

C

Principal Component Analysis

Principal Component Analysis (PCA) is a feature extraction technique. It allows to reduce the dimensionality of data with a minimal information loss. PCA transforms the original features into fewer features by the means of linear combinations. The coefficients of the linear combinations are optimally adjusted to retain as much as possible of the original variance of data. Reducing dimensions eases data processing and makes the models more controllable. In addition, the visualization of high dimensional data is a complex task. Reducing the number of features allows to visualize data easily and to explore data patterns. PCA is applicable to numerical data. Depending on the context, PCA can be either a preprocessing step or the core of the processing task.

C.1 STEPS

The first step of PCA is to standardize data. As features have different significance and thus different units and ranges, rescaling the data is essential. Otherwise, features with relatively small values will have a negligible impact on the model. Thus, we normalize the data so each feature has a zero mean and a unit variance. The next step is to create few new features containing the maximum of the variance. To do so, we calculate the covariance matrix of the original data. Then, we compute its eigenvectors and eigenvalues. The eigenvectors are uncorrelated and thus there is no information redundancy. The eigenvalues represent the variance carried by each eigenvector. We rank the eigenvectors by their corresponding eigenvalues and take only the k first ones to be the new features. The data samples are then projected on the basis formed by the new features. This way, we reduce the dimensionality of data with a minimum information loss.

C.2 APPLICATIONS

The PCA can be used in different contexts. As a tool of dimensionality reduction, the PCA allows to reduce the number of features and eliminate information redundancy. Consequently, it is used as a key step in the preprocessing before going to the core of the data processing with complex algorithms such as Support Vector Machine (SVM) and neural networks. Furthermore, PCA can be used as a standalone algorithm serving different purposes. For example, PCA can be used in the case of noise filtering. The eigenvectors with low eigenvalues are the components with low variance. These components contain mainly the noise of the data. As PCA keeps only the eigenvectors with high eigenvalues to be the new features, the noise is removed. Another interesting application is the detection of anomalies. The anomalies are situated in the components with lows variance because of their rarity. Therefore, by studying these components, one may detect anomalies.

D

Learning from Examples Module, version 2

Learning from Examples Module, version 2 (LEM2) is an algorithm of rule induction based on rough set theory. It allows to explain a categorial dependent/decision variable with a set of categorial independent variables. LEM2 transforms the samples of a multi-dimensional data set to a list of feature-value pairs. Then, it highlights the associations of subsets of feature-value pairs with each value of the decision variable. The aim of LEM2 is to create association rules that allow to infer the value of the dependent variable based on the the values of independent variables in a deterministic and accurate way. The extracted rules can be descriptive or predictive. They can be used either to explain a non-trivial classification of data samples or to predict the following values of the decision variable.

D.1 STEPS

LEM2 operates in an incremental way until covering the complete data set. The data set is subdivided into subsets with regards to the decision variable. Then in each subset, LEM2 identifies the most frequent signature (a collection of feature-value pairs) that does not appear in any other subset. If the extracted signature covers all the samples of the subset, then this signature is sufficient to identify the subset. The resulting decision rule would be *signature* $\rightarrow$ *decision variable value*. In the other case, LEM2 identifies the second most frequent signature. This process is iterated until covering all the samples of the subset. It is also replicated in all the subsets. This way, we have a set of association rules for each value of the decision variable. At the end of this process, the rules are optimized by removing uninformative feature-value pairs from the signatures. Superfluous rules are also removed from the rule set.

D.2 APPLICATIONS

LEM2 is applicable to labeled categorial data. It can serve two different purposes: data description and prediction. In the first case, the data may have a label that is not straightforward to understand. LEM2 creates association rules that explain how the label can be inferred from more meaningful variables. LEM2 can also point causalities between variables. In the second case, LEM2 is used to predict the label. The algorithm learns the rules during the training phase. Then, it decides the label of the coming samples on the basis of the deduced rules. LEM2 operates in a greedy way. Thus it may converge to a local or a global optimum. In some contexts, a local optimum is acceptable. Depending on the size and the dimensionality of the data set, the execution time may grow significantly with the nested loops. It is possible to add breaking conditions to the loops of LEM2 to reduce the execution time at the expense of the accuracy.

BIBLIOGRAPHY

[1] "3GPP; Technical Specification Group Services and System Aspects; Telecommunication management; Charging management; Charging Data Record (CDR) parameter description, TS 32.298, v15.5.1," Tech. Rep., 2018.

[2] "5G vision, white paper," Samsung Electronics Co., Tech. Rep., 2015.

[3] "5G; Service requirements for next generation new services and markets, TS 22.261, V15.5.0," 3GPP, Tech. Rep., 2018.

[4] "5G: A technology vision, white paper," Huawei Technologies Co., Tech. Rep., 2013.

[5] P. S. Addison, *The illustrated wavelet transform handbook: introductory theory and applications in science, engineering, medicine and finance*. 2017.

[6] B. Agarwal and N. Mittal, "Hybrid approach for detection of anomaly network traffic using data mining techniques," *Procedia Technology*, 2012.

[7] A. Agresti, *An introduction to categorical data analysis*. 2018.

[8] M. Ahmed, A. N. Mahmood, and J. Hu, "A survey of network anomaly detection techniques," *Journal of Network and Computer Applications*, 2016.

[9] G. A. Akpakwu, B. J. Silva, G. P. Hancke, and A. M. Abu-Mahfouz, "A survey on 5G networks for the internet of things: Communication technologies and challenges," *IEEE Access*, 2018.

[10] S. Axelsson, "Intrusion detection systems: A survey and taxonomy," Tech. Rep., 2000.

[11] P. Barford, J. Kline, D. Plonka, and A. Ron, "A signal analysis of network traffic anomalies," in *Proc. of ACM SIGCOMM Workshop IMW*, 2002.

[12] P. Bereziński, B. Jasiul, and M. Szpyrka, "An entropy-based network anomaly detection method," *Entropy*, 2015.

[13] G. Bergland, "A guided tour of the fast fourier transform," *IEEE spectrum*, 1969.

[14] M. Bianchini and F. Scarselli, "On the complexity of neural network classifiers: A comparison between shallow and deep architectures," *IEEE Transactions on Neural Networks and Learning Systems*, 2014.

[15] H. Bozdogan, "Model selection and Akaike's information criterion (AIC): The general theory and its analytical extensions," *Psychometrika*, 1987.

[16] A. P. Bradley, "The use of the area under the ROC curve in the evaluation of machine learning algorithms," *Pattern recognition*, 1997.

[17] S. Brady, D. Magoni, J. Murphy, H. Assem, and A. O. Portillo-Dominguez, "Analysis of machine learning techniques for anomaly detection in the internet of things," in *Latin American Conference on Computational Intelligence (LA-CCI)*, 2018.

[18] D. Brauckhoff, X. Dimitropoulos, A. Wagner, and K. Salamatian, "Anomaly extraction in backbone networks using association rules," in *Proceedings of the ACM SIGCOMM conference on Internet measurement*, 2009.

[19] P. J. Brockwell, R. A. Davis, and M. V. Calder, *Introduction to time series and forecasting*. 2002.

[20] M. Cairo, G. Farina, and R. Rizzi, "Decoding hidden markov models faster than viterbi via online matrix-vector (max, +)-multiplication," in *AAAI Conference on Artificial Intelligence*, 2016.

[21] R. N. Calheiros, E. Masoumi, R. Ranjan, and R. Buyya, "Workload prediction using ARIMA model and its impact on cloud applications' QoS," *IEEE Transactions on Cloud Computing*, 2015.

[22] G. Casella and R. L. Berger, *Statistical inference*. 2002.

[23] M. Çelik, F. Dadaşer-Çelik, and A. Ş. Dokuz, "Anomaly detection in temperature data using dbscan algorithm," in *International Symposium on Innovations in Intelligent Systems and Applications (INISTA)*, 2011.

[24] V. Chandola, A. Banerjee, and V. Kumar, "Outlier detection: A survey," *ACM Computing Surveys*, 2007.

[25] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM computing surveys (CSUR)*, 2009.

[26] C. Chang and C. Lin, "LIBSVM: a library for support vector machines," *ACM TIST*, 2011.

[27] W. G. Cochran, *Sampling techniques*. 2007.

[28] A. d'Aspremont, F. R. Bach, and L. E. Ghaoui, "Full regularization path for sparse principal component analysis," in *International Conference on Machine Learning*, 2007.

[29] J. Davis and M. Goadrich, "The relationship between Precision-Recall and ROC curves," in *International conference on Machine learning*, 2006.

[30] Day, William H. E. and Edelsbrunner, Herbert, "Efficient algorithms for agglomerative hierarchical clustering methods," *Journal of Classification*, 1984.

[31] "Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); LTE; Network architecture, TS 23.002, V12.5.0," 3GPP, Tech. Rep., 2015.

[32] G. Dimopoulos, I. Leontiadis, P. Barlet-Ros, K. Papagiannaki, and P. Steenkiste, "Identifying the root cause of video streaming issues on mobile devices," in *CoNEXT*, 2015.

[33] J. J. Dongarra and F. Sullivan, "Guest editors introduction to the top 10 algorithms," *Computing in Science and Engineering*, 2000.

[34] Z. Du, L. Ma, H. Li, Q. Li, G. Sun, and Z. Liu, "Network traffic anomaly detection based on wavelet analysis," in *International Conference on Software Engineering Research, Management and Applications, SERA*, 2018.

[35] O. Eluwole, N. Udoh, M. Ojo, C. Okoro, and A. Akinyoade, "From 1G to 5G, what next?" *IAENG International Journal of Computer Science*, 2018.

[36] J. M. Estevez-Tapiador, P. Garcia-Teodoro, and J. E. Diaz-Verdejo, "Anomaly detection methods in wired networks: A survey and taxonomy," *Computer Communications*, 2004.

[37] "Evolved Universal Terrestrial Radio Access (E-UTRA) and Evolved Universal Terrestrial Radio Access Network (E-UTRAN); Overall description, TS 36.300, V9.4.0," 3GPP, Tech. Rep., 2010.

[38] S. Feng and E. Seidel, "Self-organizing networks (son) in 3gpp long term evolution," *Nomor Research GmbH, Munich, Germany*, 2008.

[39] R. Fontugne and K. Fukuda, "A Hough-transform-based anomaly detector with an adaptive time interval," in *Proc. of ACM SAC*, 2011.

[40] S. Fortes, A. Aguilar Garcia, J. A. Fernandez-Luque, A. Garrido, and R. Barco, "Context-aware self-healing: User equipment as the main source of information for small-cell indoor networks," *IEEE Vehicular Technology Magazine*, 2016.

[41] R. Froehlich, *Knowledge base radio and core network prescriptive root cause analysis*, US Patent Application 14/621,101, 2016.

[42] M. M. M. Fuad and P. Marteau, "Towards a faster symbolic aggregate approximation method," *CoRR*, 2013.

[43] S. R. Gaddam, V. V. Phoha, and K. S. Balagani, "K-means+id3: A novel method for supervised anomaly detection by cascading k-means clustering and ID3 decision tree learning methods," *IEEE Trans. Knowl. Data Eng.*, 2007.

[44] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, "Anomaly-based network intrusion detection: Techniques, systems and challenges," *computers & security*, 2009.

[45] A. Garg, "Digital society from 1G to 5G: A comparative study," *International Journal of Application or Innovation in Engineering & Management*, 2014.

[46] G. Giacinto, F. Roli, and L. Didaci, "Fusion of multiple classifiers for intrusion detection in computer networks," *Pattern recognition letters*, 2003.

[47] P. Gogoi, R. Das, B. Borah, and D. K. Bhattacharyya, "Efficient rule set generation using rough set theory for classification of high dimensional data," *IJSSAN*, 2011.

[48] A. Gómez-Andrades, P. M. Luengo, I. Serrano, and R. Barco, "Automatic root cause analysis for LTE networks based on unsupervised techniques," *IEEE Trans. Vehicular Technology*, 2016.

[49] "GSM/EDGE radio access network (GERAN); overall description, TS 43.051, v8.0.0," 3GPP, Tech. Rep., 2009.

[50] F. Guigou, P. Collet, and P. Parrend, "Anomaly detection and motif discovery in symbolic representations of time series," *CoRR*, 2017.

[51] F. Guigou, P. Collet, and P. Parrend, "The artificial immune ecosystem: A bio-inspired meta-algorithm for boosting time series anomaly detection with expert input," in *Applications of Evolutionary Computation*, 2017.

[52] H. Guo and C. S. Burrus, "Wavelet transform based fast approximate Fourier transform," in *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 1997.

[53] M. Gupta, J. Gao, C. C. Aggarwal, and J. Han, "Outlier detection for temporal data: A survey," *IEEE Transactions on Knowledge and Data Engineering*, 2014.

[54] B. Hamzeh, *Network failure detection and prediction using signal measurements*, Patent US9100339B1, 2014.

[55] T. Han, Y. Lan, L. Xiao, B. Huang, and K. Zhang, "Event detection with vector similarity based on fourier transformation," in *Proc. of IEEE ICCSSE*, 2014.

[56] A. Hanemann, "A hybrid rule-based/case-based reasoning approach for service fault diagnosis," in *International Conference on Advanced Information Networking and Applications AINA*, 2006.

[57] F. van Harmelen, V. Lifschitz, and B. W. Porter, Eds., *Handbook of Knowledge Representation*. 2008.

[58] F. Harrou, F. Kadri, S. Chaabane, C. Tahon, and Y. Sun, "Improved principal component analysis for anomaly detection: Application to an emergency department," *Computers & Industrial Engineering*, 2015.

[59] N. A. Heard, D. J. Weston, K. Platanioti, D. J. Hand, *et al.*, "Bayesian anomaly detection methods for social networks," *The Annals of Applied Statistics*, 2010.

[60] J. J. Hildebrand, *System and method for allocating resources based on events in a network environment*, Patent US8788654B2, 2010.

[61] B. Horst and K. Abraham, *Data mining in time series databases*. 2004.

[62] W. Hu, Y. Liao, and V. R. Vemuri, "Robust support vector machines for anomaly detection in computer security," in *Proc. of IEEE ICMLA*, 2003.

[63] Y. Huang, *Service problem diagnosis for mobile wireless networks*, Patent WO2012113436A1, 2016.

[64] M. A. Imran and A. Zoha, "Challenges in 5g: How to empower SON with big data for enabling 5g," *IEEE Network*, 2014.

[65] Z. Jadidi, V. Muthukkumarasamy, E. Sithirasenan, and M. Sheikhan, "Flow-based anomaly detection using neural network optimized with GSA algorithm," in *Proc. of IEEE ICDCS*, 2013.

[66] A. Y. Javaid, Q. Niyaz, W. Sun, and M. Alam, "A deep learning approach for network intrusion detection system," *ICST Trans. Security Safety*, 2016.

[67] Y. Jin, N. G. Duffield, A. Gerber, P. Haffner, S. Sen, and Z. Zhang, "Nevermind, the problem is already fixed: Proactively detecting and troubleshooting customer DSL problems," in *CoNEXT*, 2010.

[68] C. Kane, *System and method for identifying problems on a network*, Patent US20150019916A1, 2015.

[69] W. Kellerer, A. Basta, P. Babarczi, A. Blenk, M. He, M. Klügel, and A. M. Alba, "How to measure network flexibility? a proposal for evaluating softwarized networks," *IEEE Communications Magazine*, 2018.

[70] E. J. Keogh, J. Lin, and A. W. Fu, "HOT SAX: efficiently finding the most unusual time series subsequence," in *Proc. of IEEE ICDM*, 2005.

[71] E. Keogh, S. Chu, D. Hart, and M. Pazzani, "Segmenting time series: A survey and novel approach," in *Data mining in time series databases*, 2004.

[72] S. Khalid, T. Khalil, and S. Nasreen, "A survey of feature selection and feature extraction techniques in machine learning," in *Science and Information Conference (SAI), 2014*, 2014.

[73] L. Khan, M. Awad, and B. Thuraisingham, "A new intrusion detection system using support vector machines and hierarchical clustering," *The VLDB journal*, 2007.

[74] E. J. Khatib, R. Barco, A. Gómez-Andrades, P. M. Luengo, and I. Serrano, "Data mining for fuzzy diagnosis systems in LTE networks," *Expert Syst. Appl.*, 2015.

[75] E. J. Khatib, R. Barco, P. M. Luengo, I. de la Bandera, and I. Serrano, "Self-healing in mobile networks with Big Data," *IEEE Communications Magazine*, 2016.

[76] E. J. Khatib, "Data analytics and knowledge discovery for root cause analysis in lte self-organizing networks," PhD dissertation, University of Malaga, 2017.

[77] D. B. Kiesekamp, T. Pilon, and R. Bolder, *System and method of visualizing most unhealthy network elements within a network or data center*, Patent US20160183109A1, 2016.

[78] G. Kim, S. Lee, and S. Kim, "A novel hybrid intrusion detection method integrating anomaly detection with misuse detection," *Expert Systems with Applications*, 2014.

[79] P. V. Klaine, M. A. Imran, O. Onireti, and R. D. Souza, "A survey of machine learning techniques applied to self-organizing cellular networks," *IEEE Communications Surveys and Tutorials*, 2017.

[80] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *International Joint Conference on Artificial Intelligence, IJCAI*, 1995.

[81] A. Lakhina, M. Crovella, and C. Diot, "Diagnosing network-wide traffic anomalies," in *Proc. of ACM SIGCOMM*, 2004.

[82] O. Leifler, "Comparison of lem2 and a dynamic reduct classification algorithm," Master's thesis, 2002.

[83] F. Li and M. Thottan, "End-to-end service quality measurement using source-routed probes," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2006.

[84] J. Li, W. Pedrycz, and I. Jamal, "Multivariate time series anomaly detection: A framework of hidden markov models," *Appl. Soft Comput.*, 2017.

[85] J. Liu, F. Liu, and N. Ansari, "Monitoring and analyzing big traffic data of a large-scale cellular network with Hadoop," *IEEE Network*, 2014.

[86] X. Liu, G. Chuai, W. Gao, and K. Zhang, "GA-AdaBoostSVM classifier empowered wireless network diagnosis," *EURASIP Journal on Wireless Comm. and Networking*, 2018.

[87] "Looking ahead to 5G, white paper," Nokia networks, Tech. Rep., 2014.

[88] "LTE; Evolved Universal Terrestrial Radio Access Network (E-UTRAN); Self-configuring and self-optimizing network (SON) use cases and solutions, TS 36.902, V9.2.0," 3GPP, Tech. Rep., 2010.

[89] W. Lu and A. A. Ghorbani, "Network anomaly detection based on wavelet analysis," *EURASIP J. Adv. Sig. Proc.*, 2009.

[90] P. M. Luengo, R. Barco, E. Cruz, A. Gómez-Andrades, E. J. Khatib, and N. Faour, "A method for identifying faulty cells using a classification tree-based UE diagnosis in LTE," *EURASIP J. Wireless Comm. and Networking*, 2017.

[91] J. Luo and S. M. Bridges, "Mining fuzzy association rules and fuzzy frequency episodes for intrusion detection," *International Journal of Intelligent Systems*, 2000.

[92] J. Ma and S. Perkins, "Time-series novelty detection using one-class support vector machines," in *Proc. of IEEE-INNS IJCNN*, 2003.

[93] A. Mahapatro and P. M. Khilar, "Fault diagnosis in wireless sensor networks: A survey," *IEEE Communications Surveys and Tutorials*, 2013.

[94] A. A. Mahimkar, Z. Ge, A. Shaikh, J. Wang, J. Yates, Y. Zhang, and Q. Zhao, "Towards automated performance diagnosis in a large IPTV network," in *SIGCOMM*, 2009.

[95] A. Mahimkar, Z. Ge, J. Wang, J. Yates, Y. Zhang, J. Emmons, B. Huntley, and M. Stockert, "Rapid detection of maintenance induced changes in service performance," in *CoNEXT*, 2011.

[96] P. Malhotra, L. Vig, G. Shroff, and P. Agarwal, "Long short term memory networks for anomaly detection in time series," in *European Symposium on Artificial Neural Networks, ESANN*, 2015.

[97] S. Mascaro, A. E. Nicholson, and K. B. Korb, "Anomaly detection in vessel tracks using Bayesian networks," *Int. J. Approx. Reasoning*, 2014.

[98] M. Mdini, A. Blanc, G. Simon, J. Barotin, and J. Lecoeuvre, "Monitoring the network monitoring system: Anomaly detection using pattern recognition," in *IM*, 2017.

[99] M. Mdini, G. Simon, A. Blanc, and J. Lecoeuvre, "ARCD: A solution for root cause diagnosis in mobile networks," in *CNSM*, 2018.

[100] H. Mi, H. Wang, Y. Zhou, M. R. Lyu, and H. Cai, "Toward fine-grained, unsupervised, scalable performance diagnosis for production cloud computing systems," *IEEE Trans. Parallel Distrib. Syst.*, 2013.

[101] H. Z. Moayedi and M. A. Masnadi-Shirazi, "ARIMA model for network traffic prediction and anomaly detection," in *Proc. of IEEE ISCIT*, 2008.

[102] M. A. S. Monge, J. M. Vidal, and L. J. Garcıa-Villalba, "Reasoning and knowledge acquisition framework for 5g network analytics," *Sensors*, 2017.

[103] J. Moysen and L. Giupponi, "A reinforcement learning based solution for self-healing in LTE networks," in *Vehicular Technology Conference, VTC*, 2014.

[104] S. Mukkamala, G. Janoski, and A. Sung, "Intrusion detection using neural networks and support vector machines," in *Proc. of IEEE-INNS IJCNN*, 2002.

[105] A. P. Muniyandi, R. Rajeswari, and R. Rajaram, "Network Anomaly Detection by Cascading k-Means Clustering and c4.5 Decision Tree algorithm," *Procedia Engineering*, 2012.

[106] G. Münz, S. Li, and G. Carle, "Traffic Anomaly Detection Using KMeans Clustering," in *Proc. of In GI/ITG Workshop MMBnet*, 2007.

[107] K. Nagaraj, C. E. Killian, and J. Neville, "Structured comparative analysis of systems logs to diagnose performance problems," in *NSDI*, 2012.

[108] D. Noll, *Identification of storage system elements causing performance degradation*, Patent US9418088B1, 2016.

[109] G. Nychis, V. Sekar, D. G. Andersen, H. Kim, and H. Zhang, "An empirical evaluation of entropy-based traffic anomaly detection," in *Proceedings of the ACM SIGCOMM conference on Internet measurement*, 2008.

[110] P.-L. Ong, Y.-H. Choo, and A. Muda, "A manufacturing failure root cause analysis in imbalance data set using pca weighted association rule mining," *Jurnal Teknologi*, 2015.

[111] M. K. Pakhira, "A linear time-complexity k-means algorithm using cluster shifting," in *International Conference on Computational Intelligence and Communication Networks*, 2014.

[112] D. Palacios, I. de-la-Bandera, A. Gómez-Andrades, L. Flores, and R. Barco, "Automatic feature selection technique for next generation self-organizing networks," *IEEE Communications Letters*, 2018.

[113] D. Palacios, E. J. Khatib, and R. Barco, "Combination of multiple diagnosis systems in self-healing networks," *Expert Syst. Appl.*, 2016.

[114] A. Patcha and J.-M. Park, "An overview of anomaly detection techniques: Existing solutions and latest technological trends," *Computer networks*, 2007.

[115] E. H. M. Pena, M. V. O. de Assis, M. Lemes, and P. Jr, "Anomaly detection using forecasting methods ARIMA and HWDS," in *Proc. of IEEE SCCC*, 2013.

[116] R. Perdisci, D. Ariu, P. Fogla, G. Giacinto, and W. Lee, "Mcpad: A multiple classifier system for accurate payload-based anomaly detection," *Computer networks*, 2009.

[117] K. Potdar, T. S. Pardawala, and C. D. Pai, "A comparative study of categorical variable encoding techniques for neural network classifiers," *International Journal of Computer Applications*, 2017.

[118] H. Ren, Z. Ye, and Z. Li, "Anomaly detection based on a dynamic markov model," *Inf. Sci.*, 2017.

[119] H. Ringberg, A. Soule, J. Rexford, and C. Diot, "Sensitivity of PCA for traffic anomaly detection," in *Proc. of ACM SIGMETRICS*, 2007.

[120] S. F. Rodriguez, R. Barco, and A. Aguilar-Garcıa, "Location-based distributed sleeping cell detection and root cause analysis for 5g ultra-dense networks," *EURASIP J. Wireless Comm. and Networking*, 2016.

[121] S. F. Rodriguez, R. Barco, A. Aguilar-Garcıa, and P. M. Luengo, "Contextualized indicators for online failure diagnosis in cellular networks," *Computer Networks*, 2015.

[122] R. Sekar, A. Gupta, J. Frullo, T. Shanbhag, A. Tiwari, H. Yang, and S. Zhou, "Specification-based anomaly detection: A new approach for detecting network intrusions," in *ACM conference on Computer and communications security*, 2002.

[123] I. Serrano Garcia, R. Barco Moreno, E. Jatib Khatiband, P. Munoz Luengo, and I. De La Bandera Cascales, *Fault Diagnosis in Networks*, Patent Application WO2016169616A1, 2016.

[124] "Services and System Aspects; Telecommunication management; Self-Organizing Networks (SON); Self-healing concepts and requirements, TS 32.541, V11.0.0," 3GPP, Tech. Rep., 2012.

[125] A. Sharma and P. K. Panigrahi, "A neural network based approach for predicting customer churn in cellular network services," *CoRR*, 2013.

[126] T. Shon and J. Moon, "A hybrid machine learning approach to network anomaly detection," *Information Sciences*, 2007.

[127] S. Singh and P. Singh, "Key concepts and network architecture for 5G mobile technology," *International Journal of Scientific Research Engineering & Technology (IJSRET)*, 2012.

[128] M. Sokolova, N. Japkowicz, and S. Szpakowicz, "Beyond accuracy, F-score and ROC: A family of discriminant measures for performance evaluation," in *Australasian joint conference on artificial intelligence*, 2006.

[129] X. Song, M. Wu, C. Jermaine, and S. Ranka, "Conditional anomaly detection," *IEEE Transactions on Knowledge and Data Engineering*, 2007.

[130] V. A. Sotiris, P. W. Tse, and M. G. Pecht, "Anomaly detection through a bayesian support vector machine," *IEEE Trans. Reliability*, 2010.

[131] G. Stein, B. Chen, A. S. Wu, and K. A. Hua, "Decision tree classifier for network intrusion detection with ga-based feature selection," in *Annual Southeast Regional Conference*, 2005.

[132] M. Steinder and A. S. Sethi, "A survey of fault localization techniques in computer networks," *Sci. Comput. Program.*, 2004.

[133] J. Su and H. Zhang, "A fast decision tree learning algorithm," in *National Conference on Artificial Intelligence and Innovative Applications of Artificial Intelligence Conference*, 2006.

[134] T. A. Tang, L. Mhamdi, D. C. McLernon, S. A. R. Zaidi, and M. Ghogho, "Deep learning approach for network intrusion detection in software defined networking," in *International Conference on Wireless Networks and Mobile Communications, WINCOM*, 2016.

[135] T. M. Thang and J. Kim, "The anomaly detection by using dbscan clustering with multiple parameters," in *International Conference on Information Science and Applications (ICISA)*, 2011.

[136] C. Tselios and G. Tsolis, "On QoE-awareness through virtualized probes in 5G networks," in *IEEE International Workshop on Computer Aided Modelling and Design of Communication Links and Networks (CAMAD)*, 2016.

[137] "Universal Mobile Telecommunications System (UMTS); Telecommunication management; Subscriber and equipment trace; Trace control and configuration management, TS 32.422, V6.4.0," 3GPP, Tech. Rep., 2005.

[138] "Universal mobile telecommunications system (UMTS); UTRAN overall description, TS 25.401, v8.2.0," 3GPP, Tech. Rep., 2009.

[139] J. J. Verbeek, N. A. Vlassis, and B. J. A. Kröse, "Efficient greedy learning of Gaussian mixture models," *Neural Computation*, 2003.

[140] P. Viswanath and R. Pinkesh, "l-DBSCAN : A fast hybrid density based clustering method," in *International Conference on Pattern Recognition ICPR*, 2006.

[141] A. Wagner and B. Plattner, "Entropy based worm and anomaly detection in fast IP networks," in *International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprise*, 2005.

[142] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal, *Data Mining: Practical machine learning tools and techniques*. 2016.

[143] M. Woźniak, M. Graña, and E. Corchado, "A survey of multiple classifier systems as hybrid systems," *Information Fusion*, 2014.

[144] Y. Xu, Y. Sun, J. Wan, X. Liu, and Z. Song, "Industrial Big Data for fault diagnosis: Taxonomy, review, and applications," *IEEE Access*, 2017.

[145] M. J. Zaki, "Generating non-redundant association rules," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2000.

[146] J. Zhang, "Advancements of outlier detection: A survey," *ICST Transactions on Scalable Information Systems*, 2013.

[147] J. Zhang and N. Ansari, "On assuring end-to-end QoE in next generation networks: Challenges and a possible solution," *IEEE Communications Magazine*, 2011.

[148] J. Zhang, M. Zulkernine, and A. Haque, "Random-forests-based network intrusion detection systems," *IEEE Trans. Systems, Man, and Cybernetics, Part C*, 2008.

[149] Z. Zheng, L. Yu, Z. Lan, and T. Jones, "3-dimensional root cause diagnosis via co-analysis," in *ICAC*, 2012.

[150] B. Zhu and S. Sastry, "Revisit dynamic ARIMA based anomaly detection," in *Proc. of IEEE PASSAT/SocialCom*, 2011.

[151] H. Zou, T. Hastie, and R. Tibshirani, "Sparse Principal Component Analysis," *Journal of Computational and Graphical Statistics*, 2006.

DECLARATION

I declare that this thesis has been composed solely by myself and that it has not been submitted, in whole or in part, in any previous application for a degree. Except where states otherwise by reference or acknowledgment, the work presented is entirely my own.

Rennes, October 2, 2019

Maha Mdini

COLOPHON

This document is a PhD thesis for the completion of a PhD degree in computer science in the university of IMT Atlantique, France.

Final Version as of October 2, 2019 (`Anomaly Detection and Root Cause Analysis` v1.0).

UNIVERSITE BRETAGNE LOIRE / MATHSTIC

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Titre : Détection d'anomalies et analyse des causes racines dans les réseaux cellulaires

Mots clés : Réseaux cellulaires, Analyse de données, Supervision des réseaux.

Résumé : Grâce à l'évolution des outils d'automatisation et d'intelligence artificielle, les réseaux mobiles sont devenus de plus en plus dépendants de la machine. De nos jours, une grande partie des tâches de gestion de réseaux est exécutée d'une façon autonome, sans intervention humaine. Dans cette thèse, nous avons focalisé sur l'utilisation des techniques d'analyse de données dans le but d'automatiser et de consolider le processus de résolution de défaillances dans les réseaux. Pour ce faire, nous avons défini deux objectifs principaux : la détection d'anomalies et le diagnostic des causes racines de ces anomalies. Le premier objectif consiste à détecter automatiquement les anomalies dans les réseaux sans faire appel aux connaissances des experts. Pour atteindre cet objectif, nous avons proposé un algorithme, Watchmen Anomaly Detection (WAD), basé sur le concept de la reconnaissance de formes (pattern recognition). Cet algorithme apprend le modèle du trafic réseau à partir de séries temporelles périodiques et détecte des distorsions par rapport à ce modèle dans le flux de nouvelles données. Le second objectif a pour objet la détermination des causes racines des problèmes réseau sans aucune connaissance préalable sur l'architecture du réseau et des différents services. Pour ceci, nous avons conçu un algorithme, Automatic Root Cause Diagnosis (ARCD), qui permet de localiser les sources d'inefficacité dans le réseau. ARCD est composé de deux processus indépendants : l'identification des contributeurs majeurs à l'inefficacité globale du réseau et la détection des incompatibilités. WAD et ARCD ont fait preuve d'efficacité. Cependant, il est possible d'améliorer ces algorithmes sur plusieurs aspects.

Title: Anomaly detection and root cause diagnosis in cellular networks

Keywords: Cellular networks, Data analysis, Network monitoring.

Abstract: With the evolution of automation and artificial intelligence tools, mobile networks have become more and more machine reliant. Today, a large part of their management tasks runs in an autonomous way, without human intervention. In this thesis, we have focused on taking advantage of the data analysis tools to automate the troubleshooting task and carry it to a deeper level. To do so, we have defined two main objectives: anomaly detection and root cause diagnosis. The first objective is about detecting issues in the network automatically without including expert knowledge. To meet this objective, we have proposed an algorithm, Watchmen Anomaly Detection (WAD), based on pattern recognition. It learns patterns from periodic time series and detect distortions in the flow of new data. The second objective aims at identifying the root cause of issues without any prior knowledge about the network topology and services. To address this question, we have designed an algorithm, Automatic Root Cause Diagnosis (ARCD) that identifies the roots of network issues. ARCD is composed of two independent threads: Major Contributor identification and Incompatibility detection. WAD and ARCD have been proven to be effective. However, many improvements of these algorithms are possible.