UNIVERSITÉ DE LORRAINE | BIBLIOTHÈQUES UNIVERSITAIRES

AVERTISSEMENT

Ce document est le fruit d'un long travail approuvé par le jury de soutenance et mis à disposition de l'ensemble de la communauté universitaire élargie.

Il est soumis à la propriété intellectuelle de l'auteur. Ceci implique une obligation de citation et de référencement lors de l'utilisation de ce document.

D'autre part, toute contrefaçon, plagiat, reproduction illicite encourt une poursuite pénale.

Contact bibliothèque : ddoc-theses-contact@univ-lorraine.fr
(Cette adresse ne permet pas de contacter les auteurs)

LIENS

Code de la Propriété Intellectuelle. articles L 122. 4
Code de la Propriété Intellectuelle. articles L 335.2- L 335.10
http://www.cfcopies.com/V2/leg/leg_droi.php
http://www.culture.gouv.fr/culture/infos-pratiques/droits/protection.htm

Anomaly Detection and Root Cause Diagnosis for Low-Latency Applications in Time-Varying Capacity Networks

THÈSE

présentée et soutenue publiquement le 29 avril 2025

pour l'obtention du

Doctorat de l'Université de Lorraine

(mention informatique)

par

Joël Roman Ky

Composition du jury

Président :	Nadjib Aitsaadi	Professeur, UVSQ Paris-Saclay, France
Rapporteurs :	Sandrine Vaton	Professeure, IMT Atlantique, France
	Nadjib Aitsaadi	Professeur, UVSQ Paris-Saclay, France
Examinateurs :	Hind Castel-Taleb	Professeure, Télécom SudParis, France
	Yacine Ghamri-Doudane	Professeur, Université de la Rochelle, France
Directrice de thèse :	Isabelle Chrisment	Professeure, Université de Lorraine, France
Co-Directeur :	Raouf Boutaba	Professor, University of Waterloo, Canada
Encadrants :	Abdelkader Lahmadi	Maître de conférences, HDR, Université de Lorraine, France
	Bertrand Mathieu	Chercheur, HDR, Orange Innovation, France
Invitée :	Claudia-Lavinia Ignat	Directrice de recherche, INRIA Nancy, France

Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503

Mis en page avec la classe thesul.

Remerciements

En y repensant, il y a quelque chose d'à la fois amusant et cohérent à finaliser cette thèse. J'ai débuté cette aventure, convaincu que c'était la voie professionnelle qu'il me fallait alors que, trois ans auparavant, je jurais par tous les saints que je ne ferais jamais de thèse. Puis, je repense au gamin que j'étais en primaire, écrivant fièrement sur une copie de rédaction qu'il serait docteur (en médecine) et chercheur scientifique. Alors, me voici aujourd'hui docteur, mais clairement pas en médecine[1]. Bien que je ne sache pas encore si je serai chercheur toute ma vie, je me dis, après ces trois belles années à faire de la recherche, que cela vaut bien le coup de continuer encore un peu.

Ces trois années et demie (soyons précis) ont été marquées par des moments de doute, un syndrome de l'imposteur, des déceptions et des expériences infructueuses..., mais elles ont aussi abouti à ce tapuscrit, à quelques publications et, surtout, à de belles rencontres et de beaux souvenirs. Il me paraît donc tout à fait naturel et juste de commencer ce tapuscrit par des remerciements sincères à toutes ces personnes qui m'ont soutenu et accompagné dans cette aventure.

Lorsqu'on me demande sur quoi repose le bon déroulement d'une thèse, je réponds toujours : *"De bons encadrants !"*. Et je pense pouvoir affirmer avoir eu de bons encadrants, car cette thèse et toutes ces contributions n'auraient jamais vu le jour sans eux. Par ces quelques mots, je tiens à leur exprimer toute ma gratitude pour leur encadrement, leur soutien, et l'autonomie qu'ils m'ont accordée tout au long de cette aventure. Merci à Bertrand Mathieu de m'avoir fait confiance dès le début pour ce sujet. Ton accompagnement, autant humain que professionnel, m'a permis de grandir dans ce parcours exigeant. J'ai énormément appris grâce à tes remarques franches et constructives, ta pédagogie et ta bienveillance, qui m'ont guidé dans mon apprentissage du métier de chercheur. Merci à Abdelkader Lahmadi pour ton soutien, toujours empreint de bienveillance, et pour tes observations pertinentes qui ont énormément enrichi mes travaux. Un immense merci également à Raouf Boutaba et Isabelle Chrisment. Malgré vos agendas bien chargés, vous avez toujours su être disponibles au moment où c'était nécessaire pour faire avancer cette thèse.

Je souhaite également remercier chaleureusement Sandrine Vaton et Nadjib Aitsaadi pour avoir accepté d'être les rapporteurs de cette thèse et pour le temps consacré à l'évaluation de mon travail. Merci à Hind Taleb, Yacine Ghamri et Claudia Ignat pour leur participation au jury. Merci à tous pour votre intérêt pour mes travaux. Merci aussi à Ye-Qiong Song, qui avec Claudia, ont suivi mes travaux depuis le début et m'ont apporté leur accompagnement et leurs précieux conseils.

Un merci tout particulier à Stéphane Tuffin, dont l'engagement a été essentiel pour cette thèse. Merci pour ta franchise, tes conseils avisés et ton soutien logistique. Tu n'as ménagé aucun effort pour m'aider à obtenir les ressources nécessaires à mes expériences, et ton implication fut d'une très grande aide.

J'ai également eu la chance de réaliser cette thèse au sein du projet ANR MOSAICO, où j'ai rencontré des doctorants, ingénieurs, et chercheurs très compétents. Merci à Philippe, Hichem, Marius, Xavier, Thibault, Guillaume, Huu Nghia, Edgardo : les discussions, les échanges, les réunions, les Orange Open Tech Days 2023, et même les moments les plus informels comme les restaurants ou les *social events*, ont rendu ce parcours doctoral aussi enrichissant scientifiquement que socialement. Grâce à cette émulation collective, j'ai pu plonger rapidement dans mon sujet et avancer plus facilement dans mes travaux.

Je souhaite remercier mes collègues de l'équipe ITEQ à Orange Innovation Lannion, merci

[1] Désolé, Maman !

pour votre accueil chaleureux. Je me suis senti intégré dans l'équipe dès les premiers jours et les (longues) pauses café (et surtout les *PPVR*) de l'équipe vont me manquer. Un grand merci à Olivier Carli, en particulier, pour son implication dans ma thèse, depuis le recrutement, mon arrivée dans l'équipe jusqu'à l'organisation de mes déplacements en conférence ou en formation. Ton aide fut précieuse.

Merci aussi à Minqi, Eric et Nicolas pour votre aide lors des tests au labo FA. Ce fut assez laborieux à mettre en place, entre les nouveaux bugs qui apparaissent dès qu'on a résolu les anciens et les allers-retours sur Pégase. Merci de votre aide, car une partie de ces travaux n'aurait pas été réalisée sans vous.

Je tiens également à remercier chaleureusement les chercheur·e·s, ingénieur·e·s et tout le personnel d'Orange Innovation Lannion, qui ont contribué, directement ou indirectement, au succès de cette thèse[2]. J'ai toujours trouvé des personnes disponibles et prêtes à partager leur expertise sur de nombreux sujets tels que les réseaux mobiles ou Wi-Fi, la science des données ou encore pour m'aider à me dépatouiller avec le proxy d'Orange. J'ai pris plaisir à réaliser ma thèse sur le site de Lannion, et merci à chacun d'avoir pris le temps de m'écouter, de répondre à mes questions, et également de votre soutien lors de ma participation au concours *Ma thèse en 3 minutes*.

Merci au LORIA et en particulier aux membres de l'équipe RESIST pour leur accueil lors des derniers mois de ma thèse pour finaliser la rédaction du tapuscrit.

Je ne saurais conclure sans adresser un immense merci à celles et ceux avec qui j'ai partagé mon quotidien (et, avouons-le surtout les galères) durant une bonne partie de ces trois années de thèse. Merci à Ziad mon voisin de bureau pour ta gentillesse (et les pâtisseries libanaises que tu rapportais), à Fatiha pour ta compagnie et nos discussions sur One Piece durant la première année (et surtout les *msemmen*), à Laure pour ton aide après mon opération, à Diane pour les soirées organisées (et ton incroyable absence de second degré), à Thomas qui a toujours la réponse juste à tout problème (et aussi la blague douteuse/contrepèterie appropriée), à Hassina pour ta gentillesse (et aussi tes tiramisus), à Lenaïg pour tes râleries sur les menus végés de la cantine (et tes tacles souvent gratuits), à Vercine pour tes réactions toujours hilarantes et à Régis le geek Lannionnais aux mille et une refs. Merci aussi à Abraham, Marco, Othmane, Khaled, Rim et aux doctorant·e·s et stagiaires rencontrés à Lannion. Merci à tous pour les rires, les potins, les confidences, les dramas, les plaintes sur le doctorat, les parties interminables de Wizard, de ping-pong, les soirées jeux de société, les afterworks, les randos et même ce que je n'ai pas réussi à oublier[3], qui ont apporté un peu de soleil et aidé à apprécier le *si beau temps* de Lannion.

Enfin, je tiens à remercier du fond du cœur ma famille, mon père, ma mère et mes frères. Il aurait été plus logique de commencer ces remerciements par vous, tant votre soutien et vos encouragements ont été importants pour moi. Merci pour vos conseils, vous m'avez toujours poussé à faire de mon mieux, même si vous ne compreniez pas toujours ce que je faisais comme travaux de recherche, ni pourquoi je faisais un doctorat[4]. Promis, c'est mon dernier diplôme, après je commence à travailler[5].

Je termine par toi, Axelle. Merci pour tes remarques toujours pertinentes, de m'avoir soutenu, encouragé et supporté pendant que je me plaignais de mon code, de la rédaction, des commentaires des reviewers ou juste comme ça. Ta patience, ta compréhension et ton amour me rendent tout beaucoup plus facile et juste pour cela merci d'être toujours à mes côtés.

[2]Merci surtout à la cantine Sodexo pour vos plats qui m'épargnaient la charge mentale de la cuisine.

[3]Je parle de ces parties de baby-foot assaisonnées des piques de la bretonne.

[4]J'avoue m'être moi aussi parfois posé la question.

[5]Sauf si je me décide à faire un CAP boulanger ou pâtissier, vu mon amour à peine dissimulé pour le pain et les pâtisseries.

À ma famille,
pour son amour inconditionnel et son soutien indéfectible.
Aux camarades du PMK,
ayant consenti au sacrifice ultime,
que ce travail soit un modeste hommage à leur mémoire et leur courage.

You kinda want to look for the anomalies.
You don't actually want to look for the expected behaviour.
- Keith Rabois

CONTENTS

INTRODUCTION

Motivation

Over recent decades, both wired and wireless networks have undergone remarkable advancements, revolutionizing connectivity and enabling a wide array of use cases across industries and daily life. Wired networks have transitioned from copper-based infrastructure to advanced optical fiber technology, which offers significant advantages. Optical fibers, with their ability to transmit infrared light over long distances with minimal attenuation, provide gigabit-level speeds and serve as the backbone for critical infrastructures, including data centers and enterprise applications. Concurrently, wireless networks have experienced significant growth, with mobile networks evolving from 2G to the high-speed, low-latency capabilities of 5G, and Wi-Fi standards advancing from 802.11b to Wi-Fi 6 and the upcoming Wi-Fi 7 technologies.

In France, substantial efforts have been made to expand fiber and 5G networks. According to the French telecommunications regulator, *Autorité de régulation des communications électroniques, des postes et de la distribution de la presse* [*fr*] (ARCEP) [1], as of the second quarter of 2024, 92% of residents (32 million people) have access to high-speed wired connections, with 70% (23 million) connected via Fiber-To-The-Home (FTTH). Additionally, 5G adoption is rapidly increasing, with 18.5 million active SIM cards—representing 22% of all mobile subscriptions—connected to 5G networks.

These advancements collectively enable networks to deliver impressive performance characterized by gigabit speeds, low latency, and high reliability. This progress is foundational for the emergence of innovative technologies and applications, including real-time remote collaboration, high-quality video streaming and gaming, immersive extended reality (XR) experiences, autonomous systems, and the development of smart cities.

Among the many applications benefiting from advancements in network performance and the widespread deployment of multi-tier clouds, a category known as Low-latency (LL) applications has emerged. This class of applications, which includes cloud gaming (CG), cloud-based virtual reality (VR), tele-robotics, and remote surgery, relies heavily on high-speed, LL connections to deliver seamless user experiences. Cloud gaming has revolutionized the gaming industry by shifting the computational burden from end-user devices to powerful cloud servers. This approach streams high-definition gameplay to users' devices, allowing even low-powered devices like smartphones and tablets to run graphically intensive games. CG eliminates the need for expensive hardware and offers users access to a diverse library of games through subscription-based models, making gaming more accessible and affordable.

Although early CG platforms, such as OnLive (launched in 2009), struggled due to limited network capacity, sparse cloud infrastructure, and insufficient device capabilities, recent advancements have changed the landscape. Today, industry giants like Google, Sony, Microsoft, NVIDIA, and Amazon have embraced CG. These platforms leverage the network performance improvements and the global cloud infrastructure, positioning CG as a fast-growing market. The global CG market is projected to grow from USD 9.71 billion in 2024 to an astounding USD 126.62 billion by 2032 [2], reflecting its widespread adoption and potential.

Similarly, cloud-based VR is driving the VR market, which was valued at USD 28.78 billion in 2023 and is forecasted to reach USD 192.99 billion by 2032 [3]. By offloading intensive processing to the cloud, cloud VR makes immersive experiences more accessible to both consumers and enterprises, eliminating the need for high-cost hardware. This democratization of VR technology

has paved the way for innovative use cases, including interactive virtual classrooms, enabling students to learn in immersive environments; medical training to provide lifelike simulations for surgeons to practice complex procedures; or military training, offering realistic simulations for combat and strategic planning, which reduce costs and risks compared to traditional methods. Key players in the cloud VR market include Meta Platforms, which has heavily invested in its vision of metaverses—shared virtual worlds where people interact as 3D avatars. Through its acquisition of Oculus VR, Meta has developed a range of VR hardware and software to support these initiatives. Other significant contributors to the cloud VR ecosystem include NVIDIA, Samsung, Unity, Sony, and Microsoft, which continue to drive innovation and adoption across various sectors.

Despite the remarkable growth in market size and technological advancements, LL applications still face significant challenges, particularly those arising from network variability. Mobile and Wi-Fi networks are prone to bandwidth fluctuations, increased delays, and jitter, all of which can jeopardize the user experience for latency-sensitive applications. For instance, in CG, video quality and smooth gameplay depend on maintaining a stable downlink bitrate, low network delays, and minimal packet loss. Higher bitrates enable higher resolutions and frame rates, but they can also exacerbate delays due to phenomena like *bufferbloat*, where excessive buffering in network queues leads to increased latency [4].

Moreover, current network architectures remain largely bandwidth-oriented, often neglecting the latency requirements of emerging applications. Latency challenges frequently arise from diverse sources, including network congestion, inefficient routing, and variable capacity links, making it difficult for applications to consistently deliver the expected Quality of Experience (QoE). While advances like latency-based congestion control algorithms, optimized application-layer protocols, and innovative techniques such as *negative latency* [5] have been proposed, their effectiveness is often limited by the underlying variability of time-varying capacity networks.

Although, solutions such as traffic prioritization mechanisms, including network slicing in 5G, offer promising avenues to address latency issues, these approaches raise concerns about net neutrality and the potential degradation of performance for non-prioritized traffic. These aforementioned challenges highlight the need for anomaly and diagnostic solutions for LL applications.

Thesis context and objectives

Context

This thesis is conducted under a *Convention Industrielle de Formation par la REcherche* [*fr*] (CIFRE) agreement [6] with the identification number 2022/0010. It is funded by Orange S.A. and the *Agence Nationale de la Recherche et de la Technologie* [*fr*] (ANRT), and the research is carried out collaboratively in two organizations: Orange Innovation Lannion and the INRIA research center at the University of Lorraine. At Orange Innovation, the work is conducted within the *ITEQ team*, which focuses on network evolution, protocols, and mechanisms to improve connectivity and user experience. At INRIA, the research is conducted within the *RESIST team*, dedicated to developing algorithms and tools for designing elastic, resilient, scalable, and secure networked systems.

[6] `https://www.anrt.asso.fr/fr/le-dispositif-cifre-7844`

Furthermore, this research is carried out within the French research project *ANR MOSAICO* [7] No ANR-19-CE25-0012. The MOSAICO project focus on enhancing the quality of service and security of networks, making them better equipped to support the requirements of LL applications.

Orange S.A. is a leading French (Internet Service Provider (ISP)) offering a broad portfolio of services, including fixed and mobile telecommunications and IT solutions for businesses. Through its research and development branch, Orange Innovation, the company is at the forefront of technological advancements in areas such as 5G, optical fiber networks, artificial intelligence and so on. The telecommunications industry is undergoing a transformative phase marked by rapid advancements like network softwarization, cloudification, artificial intelligence and evolving user demands. These shifts are not only reshaping the technical landscape but also driving the creation of new economic models. In this dynamic environment, Orange want to position itself as a pioneer, aspiring to become the operator of choice for both individual consumers and enterprises by fostering innovation and delivering state-of-the-art connectivity solutions.

The aim of this thesis

Aligned with this strategy, the objective of this thesis is to design novel methodologies for detecting and diagnosing the causes of performance degradation in emerging LL applications.

The outcomes of this research will provide valuable insights into LL applications, enabling Orange to optimize resources and potentially implement network solutions that ensure optimal performance and deliver high-quality experiences to both individual users and industrial clients.

Thesis's challenges

Achieving the objectives of this thesis presents several significant challenges that need to be addressed systematically to ensure robust and actionable outcomes.

The first challenge lies in the lack of existing datasets tailored to LL applications, which are critical for anomaly detection and root-cause diagnosis. Although some studies have explored LL applications, no comprehensive datasets exist that capture the full spectrum of performance metrics under realistic network conditions. Such datasets must account for various scenarios prone to performance degradation to accurately reflect the challenges faced by applications like cloud gaming and cloud-based VR. The collection and preparation of these datasets, monitored at different points in the network, represent a foundational step in the research.

Research Question 1:

How can comprehensive datasets for LL applications be collected to capture diverse performance metrics under realistic network conditions ?

The second challenge is implementing efficient Anomaly Detection (AD) methods. Early AD techniques relied on rule-based systems designed by experts, which were effective in simpler network environments but proved inadequate for modern, complex networks characterized

[7] `https://www.mosaico-project.org/`

by vast amounts of data and diverse Key Performance Indicators (KPIs). The transition to data-driven methods introduced Artificial Intelligence (AI)-based solutions, particularly supervised Machine Learning (ML). While these approaches offer automation and scalability, they rely on annotated datasets, which require extensive labeling by network experts. Given the growing complexity and scale of network environments, this manual labeling process is impractical. Unsupervised ML techniques, which do not require labeled data, have emerged as a solution, but their effectiveness varies significantly depending on the chosen model and dataset.

A related challenge is ensuring the robustness of unsupervised AD models against *data contamination*, where anomalies are inadvertently present in the training data. Such contamination can severely degrade the performance of AD models, making it critical to account for this phenomenon during model training and evaluation. Considering the impact of contaminated data adds an extra layer of complexity to the research.

Research Question 2:

Among the wide array of existing unsupervised AD techniques, which models are best suited for anomaly detection in LL application scenarios in terms of performance, computational efficiency, and robustness to data contamination ?

Research Question 3:

Can we propose an AD model that outperforms existing solutions and remain efficient under data contamination ?

Finally, the thesis aims to identify the root causes of performance degradation in LL applications, a task traditionally performed using expert-defined rules. However, as networks grow in complexity and anomalies evolve, these rule-based methods have become insufficient. ML-based approaches have been proposed for root-cause diagnosis, with most relying on supervised learning. These methods require labeled datasets of anomalous scenarios, which are challenging to obtain, especially for new or unforeseen anomalies. Developing efficient root-cause diagnosis techniques that rely on minimal labeling would be a significant advancement, enabling more scalable and adaptable solutions.

Research Question 4:

How can efficient root-cause diagnosis of performance degradation be achieved with minimal or no reliance on labeled data, while ensuring adaptability to emerging types of anomalies?

Thesis Contributions

To address the research questions outlined in this thesis, several contributions have been made, which are described in this section.

- **C1: Collecting datasets for LL applications under realistic time-varying network conditions.** We collect the datasets necessary for subsequent research tasks. First, 4G network conditions are collected from Orange 4G commercial network to emulate realistic 4G scenarios. Then, we collect KPIs from three commercial CG platforms under the 4G conditions previously collected. Furthermore, we gather KPI data from Cloud VR applications

operating over Wi-Fi networks. These datasets are the basis for evaluating and developing anomaly detection and root-cause diagnosis techniques in the subsequent chapters.

> **An analysis of Cloud Gaming Platforms Behaviour under Synthetic Network Constraints and Real Cellular Networks Conditions.**
> Xavier Marchal, Philippe Graff, Joël Roman Ky, Thibault Cholez, Stéphane Tuffin, Bertrand Mathieu and Olivier Festor.
> *Journal of Network and Systems Management, 2023.* [6]
> **OpenData:** `https://cloud-gaming-traces.lhs.inria.fr/data.html`

- **C2: Evaluation of unsupervised ML models for AD in CG applications over cellular networks.** We evaluate unsupervised ML models for anomaly detection in time series using the CG KPI datasets collected earlier using a consistent framework. The evaluation emphasizes critical aspects such as performance accuracy, robustness to data contamination, and computational efficiency. Various evaluation metrics are employed to ensure comprehensive and reliable conclusions about the models' effectiveness. The findings provide insights and recommendations for selecting the most suitable models for AD on low-latency applications or whatever industrial tasks.

> **Assessing Unsupervised Machine Learning solutions for Anomaly Detection in Cloud Gaming Sessions.**
> Joël Roman Ky, Bertrand Mathieu, Abdelkader Lahmadi and Raouf Boutaba.
> *Workshop on High-Precision, Predictable, and Low-Latency Networking (HiPNet '22), colocated with 18th International Conference on Network and Service Management (CNSM), Thessaloniki, Greece, October 31 - November 4, 2022.* [7]
> **Code:** `https://github.com/joelromanky/cg-ano-detect-eval`

> **ML Models for Detecting QoE Degradation in Low-Latency Applications: A Cloud-Gaming Case Study.**
> Joël Roman Ky, Bertrand Mathieu, Abdelkader Lahmadi and Raouf Boutaba.
> *IEEE Transactions on Network and Service Management, 2023.* [8]
> **Code:** `https://github.com/joelromanky/unsupervised-ml-ad-qoe-deg`

- **C3: Anomaly detection using contrastive learning.** We propose a novel unsupervised model for anomaly detection in time series, termed CATS. Leveraging contrastive learning to improve AD, CATS employs a novel loss function that exploits temporal similarities within time series windows to cluster similar windows while repelling dissimilar ones. This loss function, combined with negative data augmentations, allows CATS to outperform competing methods on public benchmark datasets as well as on the CG KPI datasets collected in this thesis and remains efficient even in presence of data contamination.

> **CATS: Contrastive learning for Anomaly detection on Time Series.**
> Joël Roman Ky, Bertrand Mathieu, Abdelkader Lahmadi and Raouf Boutaba.
> *2024 IEEE International Conference on Big Data (BigData 2024), Washington DC, USA, December 15 - December 18 2024.* [9]

Code: `https://github.com/joelromanky/cats`

- **C4: Two-stage framework for root-cause diagnostic of CloudVR applications over Wi-Fi networks.** We propose a two-stage framework for root-cause diagnosis, applied to Orange Livebox KPIs collected from Cloud VR applications. Building on the CATS anomaly detection model, the framework first detects anomalies and then employs a lightweight classifier to classify the detected anomalies. This approach demonstrates superior performance compared to existing one-stage and two-stage techniques, even with minimal labeled data. This simple but effective solution can diagnose Wi-Fi performance issues in low-latency applications.

RAID: Root cause Anomaly Identification and Diagnosis
Joël Roman Ky, Bertrand Mathieu, Abdelkader Lahmadi, Minqi Wang, Nicolas Marrot and Raouf Boutaba.
Under review at European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD 2025).
Code: `https://github.com/joelromanky/raid`

Thesis outline

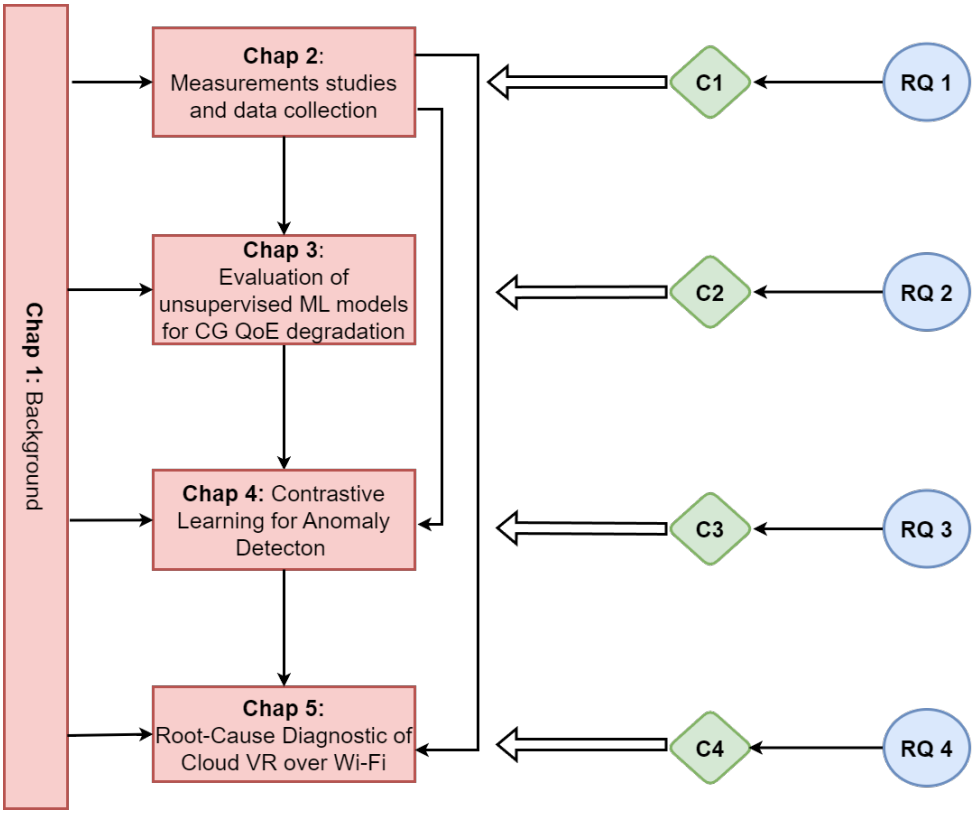


Figure 1: Thesis organization: objectives, contributions and chapters

The manuscript is organized into five chapters, introduction and conclusion excluded. The introductory chapter outlines the context, goals and motivations behind this thesis and provides a summary of the main contributions. Chapter 1 lays the groundwork for the thesis by providing a comprehensive overview of the background and the state of the art in the topics relevant to this research. This includes LL applications, cellular and Wi-Fi networks, machine learning, anomaly detection and root-cause diagnosis.

The following chapters detail the research contributions. More specifically, Chapter 2 describes the experimental setups and methodologies used to collect datasets in realistic network conditions. Chapter 3 evaluates various unsupervised ML models for anomaly detection in time series using the CG KPIs datasets.

In Chapter 4, we introduce CATS model for unsupervised anomaly detection in time series using a novel temporal contrastive learning loss function. Building on this, Chapter 5 proposes a two-stage framework for diagnosis the root causes of performance degradation in Cloud VR applications used over Wi-Fi networks. The manuscript concludes with a synthesis of the findings, highlighting their implications and proposing avenues for future research. A schematic illustration is provided in Fig. 1, to clarify how the chapters interconnect, showing the alignment of the research questions with the contributions and the logical progression of the work.

Publications

The work conducted during this thesis has led to scientific contributions disseminated through peer-reviewed international journal and conferences.

International Journals

- **J. R. Ky**, B. Mathieu, A. Lahmadi, & R. Boutaba. "ML models for detecting QoE degradation in low-latency applications: a cloud-gaming case study". *IEEE Transactions on Network and Service Management*.
- X. Marchal, P. Graff, **J. R. Ky**, T. Cholez, S. Tuffin, B. Mathieu, & O. Festor. An analysis of cloud gaming platforms behaviour under synthetic network constraints and real cellular networks conditions. Journal of Network and Systems Management, 31(2), 39.

International Conferences

- **J. R. Ky**, B. Mathieu, A. Lahmadi, M. Wang, N. Marrot, & R. Boutaba. "RAID: Root cause Anomaly Identification and Diagnosis". Under review at *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases* (*ECML PKDD 2025*).
- **J. R. Ky**, B. Mathieu, A. Lahmadi, & R. Boutaba. "CATS: Contrastive learning for Anomaly detection in Time Series", in *IEEE International Conference on Big Data* (*Big Data*), *IEEE.*, Washington DC, USA, December 15-18, 2024.
- B. Mathieu, O. Dugeon, **J. R. Ky**, P. Graff, & T. Cholez. "Segment Routing for Chaining Micro-Services at Different Programmable Network Levels", in *27th Conference on Innovation in Clouds, Internet and Networks* (*ICIN*), *IEEE*, Paris, France, March 11-14, 2024.
- **J. R. Ky**, P. Graff, B. Mathieu, & T. Cholez. "A hybrid P4/NFV architecture for cloud gaming traffic detection with unsupervised ML", in *IEEE Symposium on Computers and Communications* (*ISCC*), *IEEE*, Gammarth, Tunisia, July 9-12, 2023.

- **J. R. Ky**, B. Mathieu, A. Lahmadi, & R. Boutaba. "Assessing unsupervised machine learning solutions for anomaly detection in cloud gaming sessions", in *Workshop on High-Precision, Predictable, and Low-Latency Networking (HiPNet '22), colocated with 18th International Conference on Network and Service Management (CNSM), IEEE*, Thessaloniki, Greece, October 31 - November 4, 2022.

Chapter 1

Background and Related work

Summary: *This chapter reviews the key concepts underlying this thesis, beginning with an overview of low-latency applications and time-varying capacity networks, such as 4G, 5G, and Wi-Fi. Subsequently, the chapter introduces machine learning paradigms and a background on anomaly detection techniques for time series. Finally, it explores root cause diagnosis approaches for cellular and Wi-Fi networks.*

Contents

1.1 Low-latency applications

LL applications encompass a diverse range of use cases, including CG, VR, video conferencing, tele-robotics, industrial automation, and remote surgery. Although these applications share the common requirement of minimal end-to-end delay, each has unique characteristics and demands in terms of latency, throughput, and network protocols.

Table 1.1 provides a summary of LL applications, highlighting their typical uplink and downlink bitrate requirements as well as the range of end-to-end latency values and compare it to video streaming which is non-interactive application.

LL Application	Uplink Bitrate (Mbps)	Downlink Bitrate (Mbps)	End-to-end Latency (ms)
Video streaming	0.5-3	3-25	$\geq$ 300 (non interactive)
Video conference	1-3	1-4	$\leq$ 300
Cloud Gaming	2-5	3-40	$\leq$ 60-100
Cloud VR	10	25-100	$\leq$ 20
Medical surgery	0.5-10	10-100	$\leq$ 1

Table 1.1: Characteristics of Low-latency applications

In this thesis, we focus on Cloud Gaming and Cloud Virtual Reality, two of the most demanding LL applications in terms of bandwidth and latency requirements. Unlike mission-critical applications such as remote medical surgery, CG and Cloud VR have established commercial implementations and deployments, providing a practical foundation for realistic experiments. These applications also represent significant business opportunities for network operators like Orange, as addressing the challenges they present is crucial for enhancing user experience and meeting the growing demand for LL services.

1.1.1 Cloud gaming

CG (sometimes referred to as game streaming) is a gaming paradigm where the player sends its game commands to a server located in the cloud that executes the game and produces video frames that is streamed back to the player. This gaming paradigm implies that powerful computing resources such as Graphic Processing Units (GPU), are no longer required by gamers as CG can be operated on a wide range of devices such as smartphones or thin clients. CG arise with cloud computing and the first commercial cloud gaming platforms were OnLive and Gaikai [10, 11]. Recent years have seen new entrants in the CG markets. Sony acquired Gaikai in 2014 and launched PlayStation Now (PSN) [8] and were followed by other big industry names such as Google that launched Stadia (STD) [9] (2019), Microsoft with Xbox Cloud (XC) (2020), Nvidia with GeForce Now (GFN) (2020) or Amazon with Luna (2020) [10] [12, 13, 14, 15].

CG platform architecture and network protocols

As shown in Fig. 1.1, a CG architecture consists of a thin client and a cloud game server. The gamer commands acquired through game controllers, keyboards, etc. are encapsulated in network packets and sent to the cloud server. When received by the server, they are processed by the game engine to generate the game state that will be rendered and encoded to a video. This video is streamed to the client where the video frame is decoded, rendered and displayed on the thin client.

Commercial platforms, despite having the same broad architecture, have their specificities in terms of network protocols [16, 6, 17]. For instance, STD and XC rely on WebRTC: game commands are sent with Datagram Transport Layer Security (DTLS) packets, video packets

[8] that have now merged with PlayStation Plus
[9] was shutted down on January 2023
[10] was launched in EU on March 2023

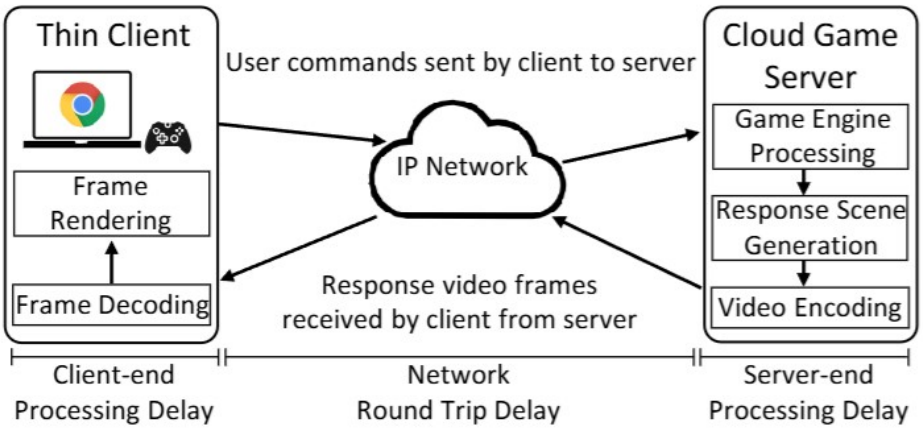


Figure 1.1: Cloud Gaming platform architecture [16]

are encoded using H.264 and VP9 codecs and streamed using Real Time Protocol (RTP) packets. GFN uses User Datagram Protocol (UDP) to send user commands and also RTP for video streaming encoded using H.264 codecs. PSN uses a custom implementation of UDP.

Each platform specifies minimum bandwidth requirements for an optimal cloud gaming (CG) experience. However, they do not provide clear information on the maximum acceptable latency despite its significant impact on user experience, as discussed later in this section. These recommendations are summarized in Tab. 1.2.

CG platforms	Minimum Bandwidth	Resolution	Network protocols
STD	10/20/35Mbps	720p/1080p/4K	WebRTC
GFN	15/25/45Mbps	720p/1080p/4K	UDP
XC	10Mbps	1080p	WebRTC
PSN	5/15/38Mbps	720p/1080p/4K	UDP
Luna	10/35Mbps	1080p/4K	WebRTC

Table 1.2: Commercial CG platforms

Factors impacting QoE in CG platforms

Although CG can remove the need of expensive computing hardware for gaming, streaming high-quality video gaming requires reliable and low-latency connections to avoid a bad user experience. QoE is defined by [18] as *"the degree of delight or annoyance of the user of an application or service. It results from the fulfillment of his or her expectations with respect to the utility and/or enjoyment of the application or service in the light of the user's personality and current state."*

According to ITU-T Rec G.1032 [19], the factors of a CG that can influence the QoE can be classified into three groups:

- Human factors: These are linked to the characteristics of the user, such as gaming experience, vision and auditory capabilities, or intrinsic/extrinsic motivation.

- Context factors: These are related to the user's environment, such as social context or physical surroundings (e.g., room characteristics or use in mobility versus at home).
- System factors: These refer to the properties that *"determine the technically produced quality of an application or service"* [19]. This includes the game being played, the gaming setup (hardware and software), network transmission, and compression.

We focus here only on the network factors that impact the most the QoE of cloud gaming applications.

- **Delay:** In cloud gaming, delay—often referred to as button-to-pixel delay—represents the time elapsed between a user action and the corresponding visual feedback on the screen. This delay (as illustrated in Fig. 1.1) can be broken down into:
  - Network delay: This delay is intrinsic to network communication, as data must travel between the client and server. High delays disrupt responsiveness, rendering games less playable.
  - Client-Side delay: Delays from this end are decreasing with advances in device performance and higher display refresh rates [20].
  - Server processing delay: This delay varies by game and platform, depending on the server's capabilities and load [16].

  The impact of delay on QoE has been been widely studied. Claypool et al. [21] demonstrated that high delays significantly degrade player performance on platforms like OnLive and GamingAnywhere, particularly for genres like racing and first-person shooters, which are more latency-sensitive than role-playing or strategy games. Peñaherrera-Pulla et al. [22] identified latency as one of the key factors to ensure the responsiveness required for CG. Xu et al. [23] highlighted the impact of network congestion, showing that competing traffic (like Transmission Control Protocol (TCP) traffics) can result in buffer bloat and thus increase delays and negatively impact QoE on platforms like STD, GFN and Luna.

- **Jitter:** Jitter is a measure of how inconsistent is the delay between packets. Jitter causes the video to be stuttered instead of being smooth. While video buffering can mitigate this in non-interactive applications like streaming, jitter poses a significant challenge in CG, where it directly impacts the QoE. Aumont et al. [24] demonstrated that jitter significantly impacts the gaming experience on CG platforms like STD and GFN. Oliveira et al. [25] confirmed these findings in their experiments on STD using four different games. They observed that high jitter magnitudes deteriorate the user experience on CG.

- **Bitrate:** The bitrate represents the number of bits transferred per unit of time. In CG, the bitrate (limited by the available bandwidth) determines the maximum frame rate and resolution achievable by the platforms. Some studies outlined that lowering the bitrate (with some bandwidth restrictions for example) leads to lower user experience. Suznjevic et al. [26] demonstrated in a study on GFN that reducing the bandwidth results in lower graphics quality (characterized by decreased resolution and frame rates) and thus resulting in lowers QoE scores. In a study analyzing the behavior of CG platforms under different network conditions, Marchal et al. [6] found that when the available bandwidth decreases, CG platforms adapt by reducing the bitrate. This typically results in a drop in frame rate and resolution (due to a reduction of the quantization factor that is responsible

for video compression). While CG platforms do not adapt the same way to this network disturbance, all experience resolution degradation and some become unplayable when the bandwidth falls below the platform bandwidth requirements. Iqbal et al. [16] observed that under constrained bandwidth conditions, CG platform cannot consistently provide 1080p/60 fps streaming.

- **Packet loss:** occurs when one or more packets traveling a network fail to reach their destination due to network congestion, radio frequency interferences or weak radio signals. Packet loss in CG applications significantly impacts user experience as demonstrated by Jarschel et al. [27] on their experiment with an emulated CG platform. They observed that even a 1% packet loss can reduce the Mean Opinion Score (MOS) score from 5 (excellent) to 2 (poor). Chen et al. [28] observed, using early CG platfors like OnLive and StreamMyGame, that packet loss affects the frame rate and the graphic quality. Thieir study revealed that former CG platforms do not implement robust packet loss concealment mechanisms. Similarly, Marchal et al. [6] found that PSN does not allow to launch game sessions when packet loss exceed 5%. In contrast, it was shown in [24, 6] that modern CG platforms like STD, GFN, XC are slightly impacted by packet loss as they used resilient error correction mechanisms including Forward Error Correction (FEC) and high redundancy.

1.1.2 Cloud VR

VR is a computer-generated experience where users can interact and feel immersed in a virtual environment. Traditional VR applications require high-end computers for the demanding graphics and processing for 3D immersive experiences, as most VR headsets cannot handle all the heavy computations and processing. As a result, the headsets are either tethered to the computer or wireless linked via technologies like AirLink to stream content.

Cloud VR consists in offloading all the heavy computational tasks to cloud servers enabling real-time streaming of rendered VR content to lightweight and less expensive VR headsets. This approach reduces the need for powerful and costly computers, making VR more accessible and affordable. By removing the need on tethered setups, Cloud VR offers more mobility and flexibility enabling diverse use-cases of VR experiences. Cloud VR could also support scalability and fosters collaboration, opening opportunities for enterprises and institutions to develop innovative VR services.

According to [29], Cloud VR applications can be classified into two categories:

- Weak-interaction VR that implies no interaction with the virtual environment. The users are passive and the contents are pre-determined. This includes 360 degree panoramic video, VR streaming or VR live broadcast.

- Strong-interaction VR includes VR services where user can interact in real-time with virtual environments through interactive devices for greater immersion. In these applications, the virtual environment depends on the inputs of the user unlike weak-interaction. This includes VR gaming, VR social networking (metaverses) and so on.

This thesis focuses only on strong interaction VR and Cloud VR will be used to refer to strong-interaction VR unless otherwise stated.

Factors impacting QoE in Cloud VR platforms

As for CG, ITU-T Rec G.1035 [30] classified the factors that can influence the QoE in cloud VR into human, context and system factors. The network factors impacting the Cloud VR experience are categorized based on the experience evaluation factors of Cloud VR, which according to [29] are: **i) the sense of reality** which are determined by the quality of audio and video to allow the users to feel immersed and are directly impacted by the bandwidth; **ii) the sense of interaction** which are impacted by the latency and **iii) the sense of pleasure** that depends on the smoothness and can be impacted by the bandwidth, the latency and the packet loss. Thus, as for CG, the most influencing network factors on QoE for Cloud VR are:

- **Delay:** In VR, the delay, often referred as *motion-to-photon* (MTP) delay, is the time elapsed between a user movement and the resulting change in the headset's field of view after rendering. In Cloud VR, the network latency adds to MTP delay because the VR content is streamed from the cloud. The impact of delay in Cloud VR were discussed in the literature. Song et al. [31] demonstrated that increased latency results in black edge artifacts (that corresponds to black area in the viewport boundary when users turn their heads) affecting QoE. Their study highlighted the sensitivity of users to both the duration and size of these artifacts. Lee et al. [32] found that higher delays or lower frame rates contribute to cybersickness, further degrading the VR experience. Warsinke et al. [33] observed that even moderate additional delays (e.g., 40 ms) significantly affect VR session quality, with impacts varying by game genre.
- **Bandwidth:** Cloud VR applications are highly bandwidth-intensive due to higher resolutions, higher frame rates, coding technology used, extra-perspective rendering and transmission required to reduce the impact of black edge events. Bandwidth constraints can lead to reduce graphics quality and frame rates, affecting the user experience. [29] reported that a minimum bandwidth of 80 Mbps is required for a fair-experience. Li et al. [34] showed that insufficient bandwidth may causes congestion, reducing frame rates and compromising QoE. Their subjective studies confirmed that users notice and react negatively to these degradations. Lee et al. [32] identified a direct correlation between increased bitrate and better visual quality, using the MOS score of their QoE model.
- **Packet loss:** It is recommended to use UDP for Cloud VR but since there is no retransmissions in UDP, packet loss can cause issues like visual quality degradations or black edge events. Li et al. [34] confirmed with objective and subjective studies that packet loss disrupts frame rates and graphics, and affects user experience. Rossi et al. [35] observed in a subjective study, that packet loss rates as low as 12% adversely affect throughput and frame rate, reducing the MOS. Warsinke et al. [33] highlighted that even minimal packet loss (0.3%) significantly impacts VR gaming session quality.

1.2 Time varying capacity networks

1.2.1 Cellular Networks

Since the 1980s and the announcement of the 1st generation (1G) with a data rate of up to 2.4 kbps, each decade has seen the introduction of a new mobile cellular generation. The second generation (2G), launched in the 1990s, brought significant advancements over 1G, including services like Short Message Service (SMS) text messaging. 2G supported communication with data rates of up to 144 kbps.

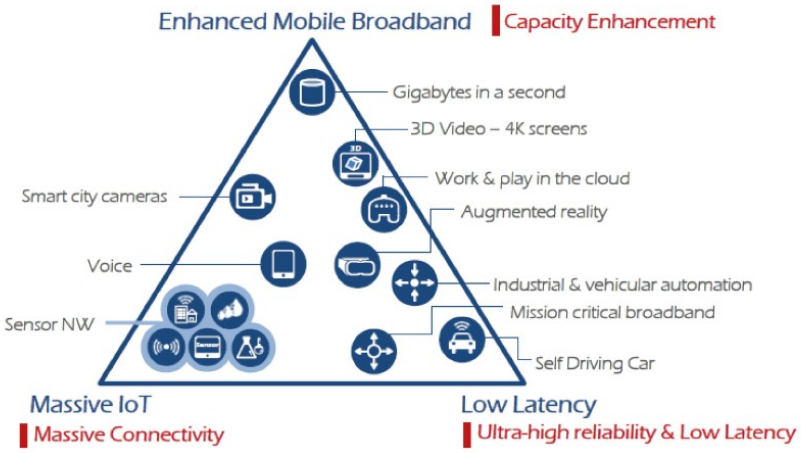


Figure 1.2: ITU-T 5G requirements [36]

The 3rd generation (3G), introduced in the 2000s, offered data rates ranging from 384 kbps to 42 Mbps (with its latest enhancements). 3G enabled new capabilities, such as convenient web browsing, and became the first cellular network widely adopted for diverse applications, including medical devices and fire alarms.

The 4th generation (4G), launched in the 2010s, was expanded by the International Telecommunication Union (ITU) to include the latest evolutions of 3G, such as Long-Term Evolution (LTE), Worldwide Interoperability for Microwave Access (WiMAX), and Evolved High Speed Packet Access (HSPA+). It achieved speeds of up to 150 Mbps and featured reduced latency, improving the user experience for applications like video streaming, video conferencing, and gaming services.

The 5th generation (5G) began deployment in 2019, following its definition by 3rd Generation Partnership Project (3GPP) Release 15, which introduced 5G New Radio (5G NR) Compared to LTE, 5G is expected to deliver up to 20 times the theoretical bandwidth of LTE (1 Gbps) and ultra-low latency down to 1 ms [36].

5G is designed to address three key application scenarios, as illustrated in 1.2:

- Enhanced Mobile Broadband (eMBB), providing faster connections with higher bandwidth (up to 20 Gbps) and moderate latency, suitable for applications like AR/VR, cloud gaming, and 360-degree video streaming.

- Ultra-Reliable Low Latency Communication (URLLC), supporting ultra-low latency (down to 1 ms) and high reliability (99.999%) for mission-critical applications, such as Industry 4.0 processes and remote healthcare.

- Massive Machine Type Communication (mMTC), enabling connectivity for a large number of Internet of Things (IoT) devices (up to 10^6 per square kilometer), albeit with trade-offs in throughput and latency.

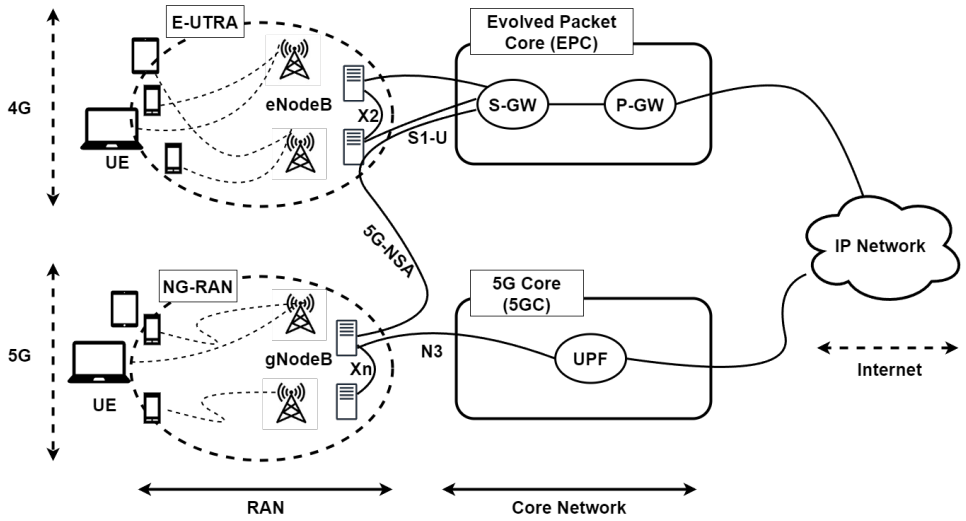


Figure 1.3: Cellular networks architecture (focus on user-plane)

Architecture of 4G/5G networks

The architecture of a cellular network, depicted in Fig. 1.3, consists of three main parts: the User Equipment (UE), Radio Access Network (RAN), and the core network. The core network connects the RAN to external networks (e.g., Internet Protocol (IP) networks), while the RAN provides connectivity between UEs and the mobile core network. In 4G (resp. 5G), the RAN is referred to as Evolved Universal Terrestrial Radio (E-UTRA) (resp. Next Generation RAN (NG-RAN)), and the mobile core network is known as Evolved Packet Core (EPC) (resp. 5G Core (5GC)).

The primary component of the RAN is the Base Station (BS), referred to as evolved NodeB (eNB) in 4G and Next Generation NodeB (gNB) in 5G. UEs connect to the BS through the radio air interface, and the BSs are linked to the core network. In 4G, BSs are interconnected via the X2 interface (resp. Xn interface in 5G) to facilitate handovers between neighboring cells. They are connected to the core network via the S1 interface (resp. N3 interface in 5G).

The 4G EPC comprises several key components. The Mobility Management Entity (MME) handles user authentication, mobility management, and signaling. The Serving Gateway (S-GW) manages the user data plane, while the Packet Data Network Gateway (P-GW) serves as a gateway connecting the network to external IP networks.

In contrast, the 5GC is designed to be more flexible and cloud-aligned using a Service-Based Architecture (SBA), where network functions are decoupled and deployed as microservices using Nerwork Function Virtualization (NFV). Key components of the 5GC include the Access and Mobility Management Function (AMF), which replaces the MME in 4G and handles connectivity and mobility management; the User Plane Function (UPF), which assumes the data routing roles of the SGW and PGW; and the Session Management Function (SMF), which manages session establishment and Quality of Service (QoS).

The 5G architecture introduces advanced concepts such as network slicing, Multi-Access Edge Computing (MEC), and the use of a wider range of frequency bands (e.g., mmWave) to achieve higher throughputs and lower latency, supporting next-generation applications like low-latency services.

Early 5G deployments used the Non-Standalone (NSA) architecture, which leverages 5G NR

in the RAN to enhance data speeds while relying on the 4G EPC for control plane functions. This approach enabled faster initial 5G rollouts by requiring fewer infrastructure changes. However, NSA cannot deliver the full potential of 5G, such as ultra-low latency and network slicing. Full Standalone (SA) deployments, which use 5G NR along with the cloud-native 5G core, are expected to realize the complete range of capabilities promised by 5G networks.

1.2.2 Wi-Fi networks

Wi-Fi is a trademark of the Wi-Fi Alliance, commonly used to refer to wireless Local Access Network (LAN)s based on IEEE 802.11 standards. Wi-Fi is used to connect devices such as laptops, smartphones, IoT devices, and gaming consoles to the Internet via a wireless router. It enables short-range wireless communication in environments such as campuses, homes, and restaurants. Any device capable of connecting to a wireless network is referred to as a station and is equipped with a wireless network interface controller. Stations are categorized into two types: Access Point (AP)s and clients.

Architecture of Wi-Fi networks

Wi-Fi networks are primarily deployed in infrastructure mode, where all communications pass through an AP. In this mode, a Wi-Fi network architecture comprises one or more APs, one or more client devices connected to the AP, and a distribution system (typically a wired Ethernet connection) that links the AP to the Internet. The AP broadcasts to the clients a Service Set Identifier (SSID) to identify the network, using beacon packets. To access the Internet, a client sends an association request to the AP, which authenticates the client using security protocols such as Wi-Fi Protected Access (WPA) (e.g., WPA2/WPA3). Once authenticated, the client is assigned an IP address by the Dynamic Host Configuration Protocol (DHCP) server on the AP, enabling Internet connectivity.

Evolution of WiFi standards with performance characteristics

The first Wi-Fi standard, IEEE 802.11, was released in 1997. Operating at 2.4 GHz, it offered data rates of up to 2 Mbps. However, it faced limited adoption due to interoperability issues, insufficient throughput, and weak security. In 1999, IEEE 802.11b (Wi-Fi 1) improved upon this by achieving theoretical data rates of up to 11 Mbps at 2.4 GHz and a range of up to 300 meters in clear environments. Wi-Fi 1 gained popularity for basic Internet browsing and email but suffered from interference issues due to sharing the 2.4 GHz band with other wireless technologies like Bluetooth and microwave ovens.

IEEE 802.11a (Wi-Fi 2) was introduced alongside Wi-Fi 1, operating on the less-crowded 5 GHz band with data rates of up to 54 Mbps. However, its higher frequency required more expensive antennas to maintain range, and it was not interoperable with Wi-Fi 1.

In 2003, IEEE 802.11g (Wi-Fi 3) combined the strengths of 802.11b and 802.11a by operating on the 2.4 GHz band while offering data rates of up to 54 Mbps. It was backward compatible with 802.11b, making it a popular choice for commercial use.

Released in 2009, IEEE 802.11n (Wi-Fi 4) brought significant improvements in speed and reliability. It introduced Multiple Input Multiple Output (MIMO) technology, allowing multiple antennas to transmit and receive data simultaneously, achieving data rates of up to 600 Mbps. It also supported dual-band support (2.4 and 5 GHz), offering better range and reduced interference.

Built on the advancements of 802.11n, IEEE 802.11ac (Wi-Fi 5), launched in 2013, further enhanced Wi-Fi performance by operating exclusively on the 5 GHz band, offering gigabit speeds of up to 3.5 Gbps. Features such as advanced MIMO configurations, beamforming, and wider channel bandwidths made it ideal for applications like High Definition (HD) and 4K video streaming, gaming, and video conferencing. It remained backward compatible with previous standards.

IEEE 802.11ax (Wi-Fi 6 and later Wi-Fi 6E) improved performance, scalability, and energy efficiency with Orthogonal Frequency Division Multiple Access (OFDMA) technology. Wi-Fi 6 operates on the 2.4 GHz and 5 GHz bands, while Wi-Fi 6E adds the 6 GHz band, providing additional spectrum and reducing congestion. With speeds of up to 9.6 Gbps, this standard is ideal for environments with many connected devices, such as smart homes and IoT ecosystems.

The upcoming IEEE 802.11be (Wi-Fi 7) standard aims to further enhance performance, delivering data rates of up to 46 Gbps and ultra-low latency. It is designed to support advanced applications like AR/VR, 8K streaming, and immersive experiences.

The evolution of Wi-Fi standards is summarized in Tab. 1.3.

	IEEE 802.11 protocol	**Release Date**	**Frequency Band (GHz)**	**Channel Bandwidth (MHz)**	**Max Throughput**
	802.11	1997	2.4	22	2 Mbps
Wi-Fi 1	802.11b	1999	2.4	22	11 Mbps
Wi-Fi 2	802.11a	1999	5	20	54 Mbps
Wi-Fi 3	802.11g	2003	2.4	20	54 Mbps
Wi-Fi 4	802.11n	2009	2.4	20/40	600 Mbps
Wi-Fi 5	802.11ac	2013	5	20/40/80/160	3.5 Gbps
Wi-Fi 6	802.11ax	2019	2.4/5	20/40/80/160	9.6 Gbps
Wi-Fi 6E	802.11ax	2020	2.5/5/6	20/40/80/160	9.6 Gbps
Wi-Fi 7	802.11be	2024 (expected)	2.5/5/6	20/40/80/160/320	46.1 Gbps

Table 1.3: Evolution of Wi-Fi standards

1.2.3 LL applications performance under time-varying capacity network

Low-latency applications such as CG and Cloud VR require high-performance networks to ensure acceptable QoS/QoE. However, their performance is significantly affected by time-varying network capacity. This section synthesizes insights from the literature on how CG and Cloud VR perform under such network conditions.

Performance on cellular networks

Kamarainen et al. [20] found that LTE networks deliver stable network delays under strong signal conditions (RSRP between -50 and -70 dBm). However, noticeable delay fluctuations occur with weaker signals (-100 dBm or more), and latency can increase 2-4 times under poor conditions. Domenico et al. [17] examined cloud gaming performance on 3G and 4G networks using STD as a benchmark. They observed that STD struggles to stream on 3G, even under optimal conditions. In contrast, 4G networks can support resolutions up to 4K with excellent signal quality, although poor 4G conditions lead to abrupt streaming interruptions. Tan et al. [37]

highlighted LTE's signaling inefficiencies, such as inter-protocol incoordinations and single-protocol overhead, which contribute to network latency often exceeding the 25 ms threshold required for acceptable VR performance. Bhuyan et al. [38] demonstrated that 5G networks can handle 4K CG streams with high bitrates (up to 50 Mbps). However, frame drops occur due to larger game frame sizes and buffering-induced packet loss. These frame drops are significantly reduced when the bitrate is lowered to 30 Mbps. The study also revealed that 5G is 46% more energy-intensive than Wi-Fi, mainly due to its use of higher frequencies and the power demands of mmWave antennas. Penaherrera et al. [39] validated that standalone 5G (5G SA) networks effectively support Cloud VR applications. However, combining 5G SA with WiFi 6 increases latency due to Wi-Fi's higher uplink latency in WiFi. In mobility scenarios, 5G outperforms WiFi by offering higher throughput and more stable latencies.

Performance on WiFi networks

Maiorano et al. [40] demonstrated that the performance of Cloud VR applications, such as NVIDIA's CloudXR, deteriorates as the distance between the headset and router increases. Low bandwidth reduces frame rates while signal attenuation increases delay. Zhang et al. [41] revealed that Wi-Fi 5 struggles with Cloud VR due to high Round Trip Time (RTT) of up to 228 ms and jitter levels that exceed the recommended thresholds for real-time applications. Jansen et al. [42] showed that signal attenuation on Wi-Fi networks causes bandwidth fluctuations, destabilizing throughput and affecting Cloud VR performance. The 5 GHz band offers higher bandwidth than the 2.4 GHz band but is more susceptible to signal attenuation. Penaherrera et al. [39] demonstrated that Wi-Fi 6 provides superior latency performance for Cloud VR compared to 5G in stationary scenarios. However, its uplink performance remains a limiting factor. Burbano et al. [43] emphasized that even Wi-Fi 6 struggles to maintain stable Cloud VR sessions under high traffic congestion, particularly on the 2.4 GHz band.

Key Observations Across Networks

Studies such as [17, 38] emphasize the importance of adequate bandwidth in ensuring the performance of low-latency applications. While 5G offers superior capacity, Wi-Fi remains limited by its inherent constraints, especially in older protocols. Cellular networks, particularly 5G, excel in providing low-latency connections. However, inefficiencies in LTE protocols and signal attenuation in Wi-Fi can significantly degrade performance. Additionally, [38] highlights that Wi-Fi is more energy-efficient than 5G for CG and Cloud VR, making it more suitable for stationary use cases. Finally, the performance of both 4G and Wi-Fi is heavily dependent on signal quality, with poor conditions exacerbating latency, jitter, and packet loss.

1.3 Machine Learning paradigms

ML is a branch of AI that uses algorithms capable of learning from data and generalizing to unseen data to perform specific tasks. Tom Mitchell (1997) proposed an engineering-oriented definition of ML: "*A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E.*"

Deep Learning (DL) is a subset of ML based on multi-layered neural networks (hence the term *deep*) to model complex patterns in large datasets. DL has demonstrated remarkable success in various fields, such as computer vision and natural language processing, due to its ability to automatically extract relevant features from raw data.

The simplest neural network architecture, the perceptron, was invented by Frank Rosenblatt in 1957. It was sufficient for solving linearly separable tasks but limited in handling non-linear problems. Neural networks were subsequently improved to address increasingly complex tasks. Examples of key advancements include:

- **Multilayer Perceptron (MLP)** (1958): consists of multiple layers of neurons including at least one hidden layers, an input layer and an output layer. With backpropagation, MLPs was capable to solve non-linear problems.
- **Convolutional Neural Network (CNN)**: are space-invariant neural networks that learns features using convolution kernels (or filters) sliding across input features. CNNs gained popularity for image processing tasks but can also be applied to signal processing.
- **Recurrent Neural Network (RNN)**: designed for sequential data processing, such as time series or text, RNNs maintain a memory of previous inputs. They were enhanced with Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) architectures to effectively capture long-term dependencies.
- **Autoencoders**: consist of an encoder that compresses input data into a latent space and a decoder that reconstructs the input from its latent representation. Autoencoders are primarily used in unsupervised learning for dimensionality reduction and later for representation learning and generative tasks with multiple variants.
- **Generative Adversarial Network (GAN)** (2014): comprised of two networks with opposite goals- a generator and a discriminator- that contest in a zero-sum game resulting in a generator capable of generating highly realistic data.
- **Transformers** (2017): an architecture that has revolutionized many tasks including Natural Language Processing (NLP), Computer Vision (CV), and more, utilizing multi-head attention mechanisms to focus on different part of a sequence simultaneously. Transformers are at the core of architectures like Vision Transformers (ViT), and Large Language Model (LLM) achieving state-of-the-art results across various domains and benchmarks.

1.3.1 Learning paradigms

In ML, various learning paradigms are tailored to specific tasks and the availability of training data. Among these, only the paradigms relevant to this thesis are discussed, as depicted in Fig. 1.4.

Supervised learning

Supervised Learning (SL) is the most fundamental and widely used learning paradigm. In SL, the training dataset $\mathcal{D}_{train} = \{(x_1, y_1), ..., (x_N, y_N)\}$ consists of a sufficient and representative amount of labeled samples (input-label pairs) that serve to learn a function f_θ approximating the true underlying relationship between x_i and y_i.

The common SL tasks are classification tasks where we learn how to map the input x_i to a discrete label (also category or class) and regression tasks where we map the input to a continuous value.

While effective for certain tasks, SL has notable limitations:

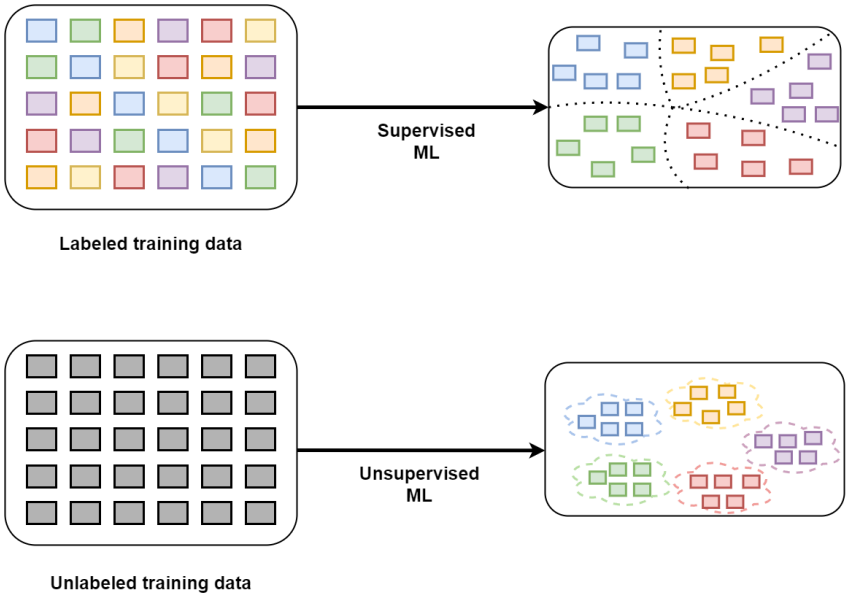


Figure 1.4: Machine learning paradigms

- **Amount of labels**. SL requires a large amount of labeled data for training. However, the acquisition of these labels is often time-consuming and resource-intensive and may demand expert knowledge in domains such as networking or healthcare.
- **Label quality**. If the labels are erroneous, (which is possible since annotation process is done by humans), it can negatively impact the model performance.
- **Generalizability and overfitting**. SL models may require complex architectures to capture the underlying patterns in the training data. This complexity can lead to overfitting, causing poor performance on unseen data or data with different distribution.
- **Scalability**. The performance of SL improves with larger datasets. However, increasing the dataset size requires substantial computational resources and higher annotation costs, which may become impractical.

Unsupervised learning

Unsupervised Learning (UL) is a learning paradigm in which, unlike SL, the learning process is conducted on unlabeled data. The model uncovers hidden patterns or relationships in the data without explicit guidance. UL can involve tasks such as discriminative, dimensionality reduction, or generative tasks. Generative UL focus on modeling the data distribution by learning a parametric approximation p_θ of the true data distribution p enabling the generation of data samples $x_i \in \mathcal{D}$. Dimensionality reduction involves representing high-dimensional datasets in low-dimensional feature spaces while preserving intrinsic dataset properties. It is useful for tasks like noise reduction, data visualization, and reducing computational time and memory usage, especially before applying ML algorithms that struggle with high-dimensional data. Discriminative UL identifies relationships or boundaries between samples based on their similarities and differences. This can be achieved through clustering, which organizes samples into groups (clusters) based on these relationships, or through representation learning. Representation learning aims to construct feature extractors that identify the most meaningful and

insightful representations from input data, enhancing downstream tasks and generalization to unseen data.

Self-supervised learning

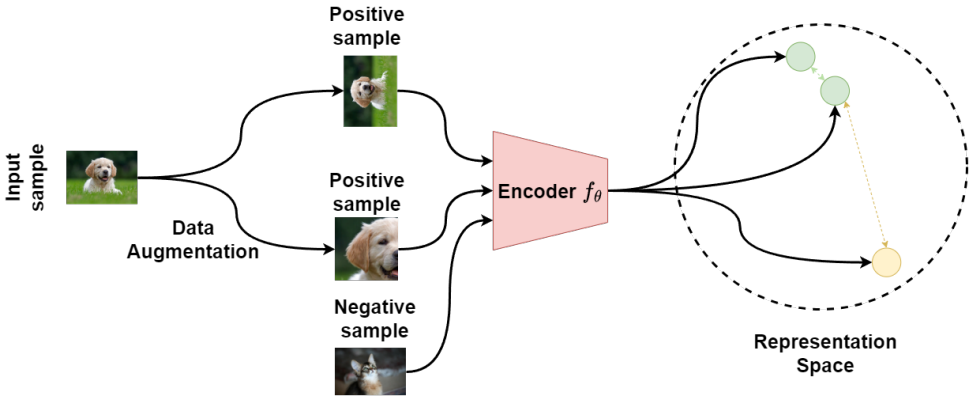


Figure 1.5: Contrastive learning

Self-supervised Learning (SSL), a subset of UL, has gained immense popularity due to its remarkable performance across various tasks in NLP and CV. SSL leverages pseudo-labels derived from the inherent structure of unlabeled data to train models via pretext tasks. By solving these pretext tasks, the model generates self-supervision signals, enforcing it to learn task-agnostic features that capture essential patterns in the data (representation learning). Pretext tasks are designed based on the intended downstream tasks and typically include: i) **input completion** which consists in masking parts of the input and predicting the missing segments (e.g., predicting the next word in a sentence or reconstructing a portion of an image); ii) **transformation predictions** which consists in predicting transformations applied to the input (e.g., determining an image's rotation) or iii) **transformation invariance** which consists in learning invariant and meaningful features common across different representations of the same input after various transformations (e.g., using data augmentations).

Among SSL methods, Contrastive Learning (CL) stands out as one of the most efficient and widely adopted frameworks. Its core idea, illustrated in Fig. 1.5 involves training a feature extractor to cluster similar data (*positive pairs*) in the feature space while pushing apart dissimilar data (*negative pairs*). The key elements of CL include:

- **Pair Selection:** Positive pairs and negative pairs are often generated using data augmentation. For example, in image classification, positive pairs can be created by applying different random crops or rotations to the same image, while negative pairs are generated by randomly selecting different images from the dataset.

- **Contrastive loss functions:** These include contrastive loss [44], triplet loss [45], and InfoNCE loss [46]. The most commonly used is the Normalized Temperature-scaled Cross Entropy (NT-Xent) loss [47]. The most commonly used is NT-Xent loss [47] which is defined as follows.

  Given a mini-batch $\mathcal{B} = \{x_1, x_1^+, x_2, x_2^+, ..., x_N, x_N^+\}$ where x_i^+ is an augmented version of x_i, an encoder function f_θ, NT-Xent loss will maximize the similarity between a representation of a sample $z_i = f_\theta(x_i)$ with its positive z_i^+ while minimizing its similarity with the

$2N-2$ negatives $(z_j)_{\forall j \neq i \in \|1;N\|}$. The similarity is often measured with cosine similarity measure and the NT-Xent loss $\mathcal{L}_{NTXent}$ can be expressed as follows:

$$\mathcal{L}_{NTXent} = -\frac{1}{2N} log \frac{exp(sim(z_i, z_i^+)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neg i]} exp(sim(z_i, z_k)/\tau)} \tag{1.1}$$

with τ a temperature hyperparameter that controls the relative importance of distance between pairs of points.

1.3.2 Training DL model

Implementing a DL model involves a series of steps aimed at determining and optimizing the model's parameters by minimizing a loss function, ultimately achieving maximum performance on a given task. The process, summarized in Fig. 1.6 is described as follows:

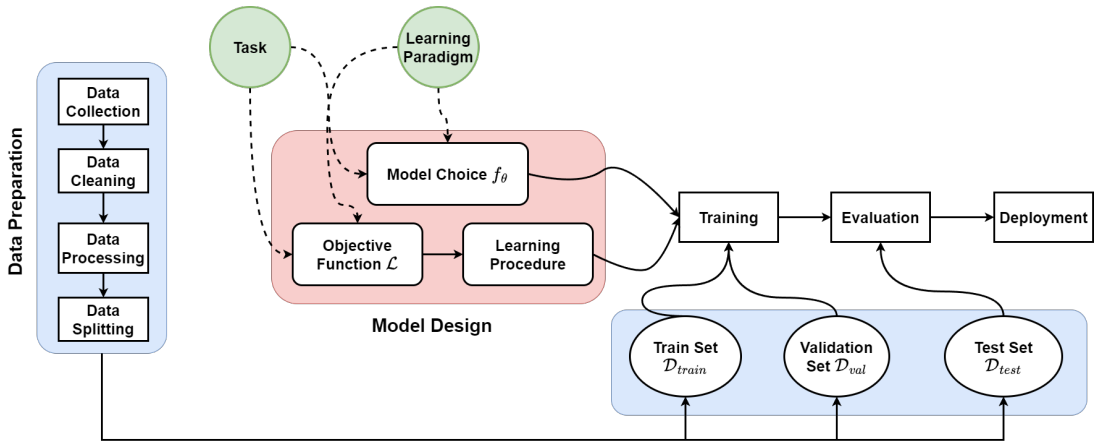


Figure 1.6: Deep learning implementation pipeline

Dataset preparation

The performance of a deep learning model is closely linked to the quality of the dataset. Preparing the dataset involves the following steps:

- **Data collection :** Gather datasets that represent the problem domain. For this thesis, the focus is on time-series datasets for cloud gaming (CG) anomaly detection and virtual reality (VR) root-cause diagnosis.
- **Data cleaning :** Identify and remove missing, inconsistent or erroneous values in the dataset to ensure high-quality data.
- **Data processing :** Prepare data for model input through the following:
  - Standardization/Normalization: Scale numerical features to a range (e.g., [0, 1]) or normalize them to have zero mean and unit variance.
  - Categorical feature encoding: Apply one-hot or ordinal encoding for categorical features.

  - Feature selection: Remove irrelevant or redundant features to improve learning efficiency.

- **Data splitting :** the data should be split in three (03) sets. The training set used for the training of the model, the validation set to evaluate the model's performance during the training and the test set to assess the model's final performance and its generalizability.

Design stage

This stage involves selecting or defining the neural architecture (f_θ) appropriate for the task and data type. To train the model, it is necessary to specify a loss function ($\mathcal{L}$), which quantifies the discrepancy between the model's outputs and the desired outputs (e.g., true labels in supervised learning). Common choices for the loss function include cross-entropy for classification tasks in supervised learning and reconstruction errors for autoencoders in unsupervised learning. The subsequent step is to choose an optimization technique to enhance the model's convergence speed and performance. The most widely used optimizers are Stochastic Gradient Descent (SGD) and Adaptative Momentum Estimation (Adam).

Training stage

The training stage is crucial as it aims to find the parameters that optimize task performance while ensuring generalizability. This involved solving an optimization problem that iteratively updates the model's parameters using only first-order derivatives. However, due to the large datasets typically used, calculating gradients across the entire dataset is impractical. Therefore, training is conducted using mini-batches (i.e., small subsets of training instances) and iterated over the entire dataset. The training process includes the following steps:

- Sample a mini-batch $\mathcal{B} = x_1, \ldots, x_{N_B}$ from the training dataset.
- Perform a forward pass through the neural network to compute the outputs $f_\theta(x_i)$ for each sample (x_i) in the mini-batch.
- Calculate the average loss over all instances in the mini-batch: $\frac{1}{N_B} \sum_i^{N_B} \mathcal{L}(y_i, f_\theta(x_i))$.
- Compute gradients for model parameters with respect to the loss using backpropagation.
- Update the model parameters using an optimizer (e.g., SGD, Adam).

This setup is standard for supervised learning. For unsupervised or contrastive learning tasks, additional steps may be required. Hyperparameter tuning (e.g., learning rate, batch size, and number of epochs) is often necessary to enhance performance, alongside regularization techniques to mitigate overfitting.

Evaluation stage and deployment

After training, the model's performance is assessed on a test set using task-specific evaluation metrics. If the model demonstrates sufficient generalizability, it can be deployed in production environments. Post-deployment, it is essential to monitor the model's performance on new data to ensure continued effectiveness, and retraining may be necessary if performance declines.

1.4 Anomaly Detection in time series

Large amounts of multivariate time series data are generated across various domains, such as monitoring systems, finance, healthcare, networking, and security. The presence of anomalies in these time series can indicate malicious activity, fraud, cyber threats, system malfunctions, or diseases in healthcare. Consequently, detecting anomalies in time series data is becoming increasingly critical and has become a prominent area of research. Before delving into existing anomaly detection methods, we define the problem and introduce some key definitions.

Definition 1.4.1 (Time series)**.** A time series is a sequence of data points chronologically collected and equally-time spaced. Time series can be univariate or multivariate.

A multivariate time series is formally defined as $X = \{x_1, x_2, ..., x_T\}$ where $x_t \in \mathbb{R}^m$ denotes a m-dimensional vector corresponding to the observations of the m-features at a specific time t. A univariate time series is the special case when $m = 1$. In this thesis, we focus on multivariate time series.

Definition 1.4.2 (Anomaly)**.** "*An anomaly is an observation that deviates considerably from some concept of normality.*" [48].

Chandola et al. [49] proposed a taxonomy of anomalies based on their nature:

- **Point anomalies.** An individual data point that significantly differs from the rest of the data. This is the simplest type of anomaly. For instance, in fraud detection, an unusually high transaction amount compared to typical spending patterns is a point anomaly.
- **Contextual anomalies.** A data point, anomalous in one context but normal in another. For instance, a temperature of 30°C is normal in the summer but anomalous in winter.
- **Collective anomalies.** A set of data points that, when considered together, represent an anomaly. Individually, each point may not be anomalous, but as a group, they indicate abnormal behavior. For instance, in a network monitoring system that typically processes 200–500 packets per minute, a steady rate of 450 packets per minute over 10 minutes might signal a collective anomaly.

AD encompasses techniques for identifying data that deviate from normal patterns. While the terms *anomaly*, *outlier*, and *novelty detection* are often used interchangeably, they have distinct meanings depending on the context. According to Ruff et al. [48], given the distribution $\mathbb{P}^+$, of normal samples, an anomaly is an observation from a distribution different from $\mathbb{P}^+$. An outlier is a rare or a low-probability instance drawn from $\mathbb{P}^+$ while a novelty is an instance drawn from a new region of a non-stationary distribution of $\mathbb{P}^+$. For example, if $\mathbb{P}^+$ denotes the distribution of dogs, a cat would be an anomaly, a rare breed of dog an outlier and a newly discovered dog breed would be a novelty. Despite their differences, these concepts share common detection methods, and this thesis collectively refers to them under the term anomaly detection.

For a multivariate time-series dataset $X = \{x_1, x_2, ..., x_T\}$ with $x_t \in \mathbb{R}^m$, AD involves learning a model f_θ that assigns, for each unseen observation $\tilde{x_t}$, a label $y_t \in \{0, 1\}$ indicating whether the new observation is anomalous ($y_t = 1$) or not ($y_t = 0$). Depending on the learning setting, time series AD algorithms can be supervised, semi-supervised or unsupervised. This thesis focuses on reviewing unsupervised and self-supervised AD algorithms as supervised and semi-supervised algorithms require labels which is unavailable or expensive to obtain in real-world scenarios.

1.4.1 Unsupervised AD

Unsupervised anomaly detection is the most commonly used approach due to the challenges of obtaining sufficient labeled normal and anomalous samples for supervised training. In the unsupervised setting, we assume that the training data is *free* of anomalies (i.e., the training data consists solely of normal data). Since the model is trained exclusively on normal data, anomalies are expected to deviate significantly from normal data. This deviation is quantified using an anomaly score; if the score exceeds a predefined threshold, the data is classified as anomalous. However, this assumption may be violated in practice due to *data contamination*, where some anomalous samples may be present in the training set.

Unsupervised AD methods can be broadly categorized into machine learning (ML)-based and deep learning (DL)-based approaches.

ML-based

- **Traditional statistical algorithms.** Statistical techniques operate on the assumption that normal data lie in high-probability regions of a stochastic model, while anomalies are located in low-probability regions [49]. Parametric methods assume that normal data follows a Gaussian distribution [50] or an underlying linear statistical model, such as Vector AutoRegression (VAR) or AutoRegressive Integrated Moving Average (ARIMA) [51]. Non-parametric methods use kernel functions to estimate the probability density function, as in Kernel Density Estimation (KDE) [52] or are based on Kalman filters [53].

- **Distance-based algorithms.** These algorithms detect anomalies by measuring the distance between data points. The core idea is that sequences with large distances from the majority of the data are considered anomalous. Methods like k-th Nearest Neighbor (KNN) [54] calculate the distance between a time series instance and its nearest neighbors, using this distance as the anomaly score. Alternatively, Local Outlier Factor (LOF) [55] measures the relative density of each time series instance, considering instances in low-density neighborhoods as anomalous and those in high-density neighborhoods as normal. Some distance-based methods perform calculations relative to clusters, rather than individual neighbors. Clustering-based approaches, such as k-Means [56] and Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [57], are typically faster than nearest-neighbor methods, as the number of clusters is smaller than the number of instances.

- **Miscellaneous algorithms.** This category includes AD techniques based on spectral analysis, such as Principal Component Analysis (PCA), RobustPCA [58], and Independent Component Analysis (ICA) [59]. These algorithms project data into a lower-dimensional space defined by the principal components (PCA) or decompose time series into statistically independent components (ICA). Anomalies are identified based on reconstruction errors (PCA) or the degree of non-Gaussianity (ICA). Isolation Forest (IF) [60] is an isolation-based technique that recursively isolates anomalies from normal data using randomly features selection and split values. One-Class Support Vector Machine (OC-SVM) [61] learns the smallest hypersphere that encapsulates all normal data, with anomalies being points lying outside the hypersphere.

DL-based

- **Reconstruction-based algorithms.** These techniques identify anomalies by learning to reconstruct normal data. The underlying assumption is that a model trained on normal data will exhibit high reconstruction errors when attempting to reconstruct anomalous data, using these errors as anomaly scores. The most common deep learning architecture in this category is the Autoencoder (AE), and numerous AE-based AD techniques have been proposed in the literature. Deep Autoencoding Gaussian Mixture Model (DAGMM) [62] combines a deep AE with a Gaussian Mixture Model (GMM), which estimates sample likelihoods based on their low-dimensional representations. The model computes an energy score, which, along the reconstruction error, identifies anomalies. Mult-Scale Convolutional Recurrent Encoder-Decoder (MSCRED) [63] addresses temporal patterns in time series using an attention-based convolutional LSTM encoder to construct multi-scale signature matrices that the decoder reconstructs. These signature matrices capture intercorrelations between different pairs of time series, enhancing AD. UnSupervised Anomaly Detection (USAD) [64] employs adversarial training on two AEs with a shared encoder. The first AE reconstructs normal data, while the second attempts to distinguish the reconstructions, forcing the first AE to improve its reconstructions.

  Variational Autoencoder (VAE), a probabilistic extension of AE, integrates to autoencoders, Bayesian inference modeling latent spaces as a probability distribution. Donut [65] is a VAE-based AD method introducing techniques like modified Evidence Lower Bound (ELBO), missing data injection, and Markov Chain Monte Carlo (MCMC) imputation, outperforming previous VAE-based AD algorithms. LSTM-VAE [66] extends AEs with a denoising autoencoder that utilizes LSTM to capture temporal dependencies and Bayesian inference to reconstruct input data based on a learned latent distribution. Omni-Anomaly [67] combines a VAE with a stochastic RNN to model temporal dependencies and uses planar normalizing flows to handle non-Gaussian latent spaces distributions.

  Transformer architectures known for their ability to handle long sequences, have also been adapted for time series AD. Anomaly Transformer [68] adapts the transformer architecture for time series AD by leveraging attention mechanisms to capture both prior and series associations, combined with a min-max optimization strategy that highlights the association discrepancy between normal and anomalous data. TranAD [69] combines a transformer-based encoder-decoder architecture with adversarial training. It introduces self-conditioning to enhance robust feature extraction, training stability, and better generalization.

  Generative models like GAN have shown great promise in generating realistic images and time series, and have been applied to time series AD as well. MADGAN [70] employs an LSTM-RNN as the core model in a GAN framework for time series AD. It introduces a novel anomaly score based on reconstruction error and discrimination loss to identify anomalies. BeatGAN [71] applies GANs to perform robust time series reconstruction for AD using adversarial regularization.

- **Forecasting algorithms.** These approaches predict future values based on past dat and identify anomalies by comparing the predicted values with and actual observations. LSTM-PRED [72] uses LSTM-RNN to learn temporal dependencies and applies a cumulative sum method for anomaly detection. Telemanom [73] employs LSTM-RNN for forecasting and introduces a dynamic thresholding scheme for anomaly identification. AD-LTI [74] uses a GRU network to forecast seasonal trends in time series and identifies anomalies

based on a probability score calculated using the Local Trend Inconsistency (LTI) metric, which quantifies deviations between the predicted and actual sequence. DeepAnt [75] predicts future timestamps using a deep CNN network, detecting anomalies based on the Euclidean distance between predicted and actual values.

- **Miscellaneous algorithms.** Deep Support Vector Data Description (Deep-SVDD) [76] is a one-class classification algorithm that leverages deep learning for one-class classification, overcoming the limitations of kernel and feature selection in traditional OC-SVMs. Deep Isolation Forest (DIF) [77] utilizes deep learning to learn random representations from input data, applying a novel isolation method for non-linear partitioning on subspaces to detect anomalies. Calibrated One-class classification for Unsupervised Time series Anomaly detection (COUTA) [78] proposes a calibrated one-class classifier based on uncertainty modeling and the injection of synthetic anomalies, which reduces the impact of data contamination and improves detection performance.

1.4.2 Self-supervised AD

The growing success of SSL across various tasks and domains has led many research efforts to adopt these techniques to improve the performance of anomaly detection (AD) algorithms. This section focuses on contrastive AD approaches, as they are widely adopted and studied in the literature.

Contrastive AD

Contrastive One-Class Anomaly (COCA) framework [79] introduces a contrastive one-class framework for anomaly detection that does not rely on negative samples. It trains on positive pairs consisting of low-dimensional time series representations and their reconstructed counterparts (via an LSTM-based Seq2Seq model). The combined contrastive and one-class loss functions mitigate collapse issues and enhance AD performance.

ContrastAD [80] defines both positive and negative temporal transformations. Positive pairs are formed by the embeddings of a subsequence and its positive transformation, while negative pairs consist of the embeddings of the positive transformations of two different subsequences, along with a window's embedding and its negative transformations. These positive/negative pairs help ContrastAD achieve improved AD performance across various benchmark datasets.

Contrastive Time series Anomaly Detection (CTAD) [81] employs time series data augmentation to generate two views of a time series window: a semantic-preserving view and synthetic anomalies. By contrasting each window segment with 2N-1 positive samples and 2N negative samples, CTAD enhances representation learning, even with a simple encoder architecture.

TimeAutoAD [82] leverages AutoML to automatically tune the framework's hyperparameters. It also employs negative samples generation and a contrastive loss that leverages these generated negative samples to improve AD performance.

CARLA [83] is a two-stage framework. The first stage learns representations from triplets (original, previous, and anomaly-injected windows). The second stage uses a self-supervised classification loss that forces the model to distinguish between normal samples and anomalies using a custom loss function, where positive pairs are nearest neighbors and negative pairs are furthest neighbors.

DCdetector [84] uses a multi-scale dual attention network to capture both spatial and temporal dependencies. Using a negative-sample-free contrastive loss based on representations of

different views of the time series, it identifies anomalies by applying a representation discrepancy criterion.

1.4.3 Discussion

Table 1.4 summarizes the time series anomaly detection (AD) models reviewed in this thesis. It provides an overview of the contributions of each method and categorizes them accordingly.

The AD techniques discussed in this thesis span various approaches, including ML, DL and SSL. Despite their effectiveness, these techniques exhibit several limitations that impact their performance in real-world environments. Traditional statistical and ML-based methods are lightweight, interpretable, and efficient for small, simple datasets. However, parametric methods that assume Gaussian distributions can struggle to detect anomalies when the data deviate from these assumptions, a common challenge in real-world applications. Additionally, distance-based methods, such as KNN and LOF, as well as parametric approaches like ARIMA, tend to become computationally expensive as the dataset grows, leading to inefficiencies when applied to high-dimensional data.

Deep learning-based approaches leverage neural network architectures to handle large dimensional data for anomaly detection. While these methods are powerful, they demand significant computational resources and memory, which may limit their applicability in real-time systems. Furthermore, DL techniques, like their ML counterparts, are typically trained on normal data and may be compromised by the presence of anomalies in the training set. Another key limitation is their *black box* nature, which makes it difficult for practitioners (e.g., network experts) to interpret the reasoning behind the model's predictions, reducing their transparency and trustworthiness in real-world scenarios.

Contrastive learning techniques have been introduced to enhance AD performance by leveraging self-supervised learning strategies. However, these approaches heavily rely on data augmentation to generate positive and negative pairs for training. Bad choices of data augmentation techniques may result in poor performances. Furthermore, while some of the models reviewed in this thesis incorporate neural architectures that attempt to capture temporal dependencies, the contrastive learning process itself does not explicitly consider the temporal nature of the data, which may hinder their ability to detect anomalies that exhibit temporal correlations.

In this thesis, we propose in Chapter 4, a novel contrastive learning-based approach that integrates temporal similarity to address these limitations and improve anomaly detection in time series data while being more robust to data contamination.

Table 1.4: Time series anomaly detection models

Category	Sub-Category	Method	Main contributions
ML-based	Statistical	Gaussian model [50]	Method assuming data follows a Gaussian distribution.
		VAR/ARIMA [51]	Method assuming data follows a linear model.
		KDE, [52] Kalman Filter [53]	Method based on probability densities estimations.
	Distance-based	KNN [54]	Method that uses the distance to nearest neighbors as an anomaly score.
		LOF [55]	Method that uses the density to the neighborhood.
		k-Means [56], DBSCAN [57]	Clustering methods that group data for faster AD.
	Miscellaneous	PCA/RobustPCA [58], ICA [59]	Spectral analysis methods using dimensionality reduction or independent components.
		IF [60]	Method that recursively isolates anomalies through random splits.
		OC-SVM [61]	Method that separates normal data from anomalies using a hypersphere.
DL-based	Reconstruction	DAGMM [62]	Combines AE and GMM to detect anomalies using reconstruction error and energy score.
		MSCRED [63]	Utilizes attention-based convolutional LSTM to capture time series correlations.
		USAD [64]	Adversarial AEs trained to reconstruct normal data and identify anomalies.
		Donut [65]	Probabilistic AE that models latent spaces as distributions for AD.
		LSTM-VAE [66]	VAE-based model combined with LSTM to capture temporal dependencies.
		Omni-Anomaly [67]	VAE combined with stochastic RNN and normalizing flows for AD.
		Anomaly Transformer [68]	Leverages attention mechanisms to capture associations for AD.
		TranAD [69]	Transformer with adversarial training and self-conditioning.
		MADGAN [70]	GAN-based time series reconstruction with LSTM-RNN for AD.
		BeatGAN [71]	GAN-based time series reconstruction using adversarial regularization.
	Forecasting	LSTM-PRED [72]	Forecasting-based LSTM combined with cumulative sum method for AD.
		Telemanom [73]	LSTM-RNN forecasting method with dynamic thresholding.
		AD-LTI [74]	GRU for seasonal trends prediction and AD based on local trend inconsistency.
		DeepAnt [75]	CNN-based forecasting method comparing predicted and actual values for AD.
	Miscellaneous	Deep-SVDD [76]	DL-based one-class classification to overcome kernel selection limitations.
		DIF [77]	Isolation method using random representations and nonlinear partitioning for AD.
		COUTA [78]	Calibrated one-class classifier with uncertainty modeling and synthetic anomaly injection.
SSL-based	Contrastive	COCA [79]	Contrastive one-class framework for time series AD.
		ContrastAD [80]	Uses positive and negative temporal transformations for contrastive AD.
		CTAD [81]	Augments time series data for contrastive learning using positive and synthetic anomalies.
		TimeAutoAD [82]	AutoML-based method with contrastive loss for AD.
		CARLA [83]	Two-stage framework with triplet-based representation learning and anomaly classification.
		DCdetector [84]	Multi-scale dual attention network using representation discrepancy for AD.

1.4.4 Evaluation metrics for time series anomaly detection

It is crucial to evaluate AD algorithms to assess their performance. The commonly used metrics are listed below.

- **Precision.** The fraction of anomalies correctly detected among all the instances classified as anomalies by the model. It indicates the accuracy of the model for detecting anomaly.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{1.2}$$

- **Recall.** The fraction of actual anomalies that were correctly detected. It indicates the ability of model to detect all existing anomalies.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{1.3}$$

- **F1-Score.** It is the harmonic mean of precision and recall indicating a balance between both metrics.

$$\text{F1} = 2.\frac{\text{Precision.Recall}}{\text{Precision} + \text{Recall}} \tag{1.4}$$

- **MCC.** Matthews Correlation Coefficient (MCC) [85] is a binary classification metric, similar to the Pearson coefficient, that addresses the class invariance limitation of the F1-score (i.e., if the positive class becomes the negative class and vice versa) as highlighted and discussed in [86]. It evaluates model performance by equally rewarding accurate predictions for both the positive and negative classes.

$$MCC = \frac{TP.TN - FP.FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{1.5}$$

- **TPR.** True Positive Rate (TPR) is equivalent to recall.

- **FPR.** False Positive Rate (FPR) is the fraction of normal instance that are classified as anomalies.

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}} \tag{1.6}$$

- **AU-ROC.** Area Under Receiver Operating Characteristic Curve (AU-ROC) indicates the area under the ROC curve which is a chart that visualizes the trade-off between the TPR and FPR. A value close to 1 suggests better performance while a value of 0.5 implies random guessing.

$$\text{AU-ROC} = \int_0^1 \text{TPR}(t) \frac{d(\text{FPR}(t))}{dt}\, dt \tag{1.7}$$

- **AU-PR.** Area Under Precision-Recall Curve (AU-PR) correspond to the area under the Precision-Recall curve and is more informative than AU-ROC for imbalanced datasets. The PR curve visualizes the trade-off between precision and recall.

$$\text{AU-PR} = \int_0^1 \text{Precision}(t) \frac{d(\text{Recall}(t))}{dt}\, dt \tag{1.8}$$

1.5 Root-Cause diagnosis in time varying capacity networks

While detecting anomalies is crucial for maintaining network performance, Root-Cause Diagnosis (RCD) represents the essential next step for enabling network providers to design and optimize networks that support low-latency applications effectively. Root-cause diagnosis, often referred to as "root-cause analysis" or "fault diagnosis" in the literature, focuses on identifying the specific causes of performance degradation or anomalies.

To provide a comprehensive understanding of root-cause diagnosis, in this section, we will present for cellular network and WiFi networks respectively, the common factors leading to performance issues in these networks and review the approaches for root-cause diagnosis in the literature.

1.5.1 Causes of performance degradation on time varying networks

RCD in time-varying capacity networks presents significant challenges due to following factors. Performance degradation can arise from diverse causes such as congestion, interference, signal attenuation, hardware failures, or software misconfigurations. These issues often manifest through similar symptoms, including increased packet loss, higher delays, or reduced throughput, making it difficult to pinpoint the root cause. Time varying networks also generate an enormous volume of KPIs from various network elements, including the RAN, APs, core network or user devices. We present in the following the common causes of performance degradation in time varying network.

Cellular networks

Performance degradation in cellular networks arises from a variety of sources, some of which are highlighted below:

- **Congestion :** Congestion in cellular networks is often caused by bufferbloat, which occurs when excessive queuing delays arise due to packets arriving at a node faster than it can process them. This issue is particularly prevalent in the RAN, where multiple flows are aggregated, often resulting in large queues [4, 87]. Congestion leads to increased delays and packet loss, significantly impacting network performance.

- **Handovers:** High-speed mobility scenarios, such as users traveling in vehicles or trains, can result in frequent handovers. These handovers cause fluctuations in link capacity and introduce additional latency in the RAN [37, 88, 89].

- **Coverage:** Network coverage is influenced by factors such as the distance from the base station, environmental obstacles, and weather conditions. Poor coverage often results in higher delays due to retransmissions and reduced throughput [88].

- **Interference:** Cellular networks are susceptible to interference, with inter-cell interference being a primary factor. This occurs when the same frequency is reused across multiple cells, leading to degraded signal quality and degrease the throughput [90].

- **Other Factors:** Additional sources of performance degradation include various types of delays, as categorized in the taxonomy by Briscoe et al. [91] (e.g., propagation and scheduling delays), device-induced delays [20], and software bugs in the RAN.

WiFi networks

Several phenomena can lead to performance degradation in WiFi networks, impacting QoS parameters such as bandwidth, latency, and packet loss. The most common sources of degradation mentioned in the literature include:

- **WiFi Interference:** Interference can be caused by nearby APs using the same or adjacent channels. This can lead to collisions and degraded signal quality [92, 93, 94].
- **Non-WiFi Interference:** Non-WiFi devices operating in the 2.4 GHz (e.g., Bluetooth devices, microwaves, baby monitors) or 5 GHz (e.g., media converters) frequency bands can cause interference, leading to packet loss [95, 93, 96].
- **Congestion:** High traffic volumes from many devices connecting to the same AP or the use of bandwidth-intensive applications can cause congestion, resulting in increased latency and packet loss [97, 92, 98].
- **Signal Attenuation:** Signal attenuation can occur due to the distance between the device and the AP (the greater the distance, the weaker the signal) or obstacles like walls, furniture, and humans that absorb the signal and reduce its strength. This can lead to packet loss and retransmissions, causing higher delays [99, 92, 97, 98, 94].
- **Hidden Terminal Problem:** This problem occurs when two devices can communicate with an AP but cannot directly communicate with each other due to being out of range or obstructed by physical barriers. This can lead to simultaneous transmissions to the AP, causing collisions (when two devices transmit data at the same time and the data gets corrupted) which lead to retransmissions and/or packet loss [97, 94].
- **Contention:** When multiple devices compete for access to the same wireless channel, contention arises, leading to collisions, increased latency, and reduced throughput [99, 95, 94].

1.5.2 Root cause diagnosis techniques

Traditionally, RCD relies on expert domain expertise to identify potential causes. While these techniques are effective for diagnosing well-understood and recurring issues, they face significant challenges in dynamic and modern cellular and Wi-Fi networks. The primary limitation lies in their scalability: expert-based rules must be updated manually to account for new faults, which is impractical given the large volume and complexity of KPIs generated by these networks. This limitation has driven the adoption of automated, data-driven approaches that can adapt to evolving network environments. Data-driven techniques, particularly those leveraging machine learning, have become the dominant approach for root-cause diagnosis in cellular networks. These methods analyze vast amounts of telemetry data to identify patterns and correlations indicative of network faults.

RCD techniques on cellular networks

Diagnosis techniques for cellular networks are diverse and span a range of methodologies, targeting different parts of the network, such as the Radio Access Network (RAN) or the core network. These techniques can broadly be categorized into expert-based approaches and data-driven methods.

Expert-based techniques rely on predefined rules and domain knowledge to analyze KPIs or network topology for identifying impairments. For example, Watanabe et al. [100] and Kane et al. [101] used such methods to identify fault scenarios based on rule-based systems.

Data-driven techniques are mainly based on ML techniques. For instance, Dimopoulos et al. [102] conducted root-cause analysis for video streaming QoE using supervised ML models trained on features collected across multiple network layers. Chen et al. [103] combined neural networks and SVM to detect fault causes on a 4G network based on network KPIs. Liu et al. [104] proposed a framework for QoE anomaly detection and root-cause analysis for LTE networks. Their method utilized QoE-driven Key Quality Indicators (KQIs) extracted from RAN KPIs to diagnose faults affecting applications such as video streaming and instant messaging. Shi et al. [105] developed NeTExp, a dual-classifier system based on deep learning, which used Graph Neural Network (GNN) to analyze network-side KPIs and CNNs for UE-side KPIs. Fida et al. [106] proposed a two-stage method for bottleneck identification in cloudified mobile networks. Their approach combined VAE for anomaly detection with MLPs for classifying faults based on KPIs collected from UE, RAN, and network nodes. Hasan et al. [107] proposed Simba, a model combining GNN and transformers to capture spatiotemporal dependencies in time-series KPI data to diagnose faults in 5G RAN environments.

RCD techniques on WiFi networks

There are in the literature numerous works on troubleshooting on WiFi networks. Primary works leveraged user-level network probes for WiFi impairments detection. Kim et al. [95] proposed WiSlow to detect channel contention and non-WiFi interference. Kanuparthy et al. [97] employed active probing techniques to identify impairments like congestion, hidden terminals, and low signal through access delay and packet loss analysis.

Statistical and clustering methods were also employed. Da Hora et al. [108] utilized K-means clustering to identify poor QoE for web browsing, video streaming, and WebRTC applications. Rayanchu et al. [99] adopted a statistical-based method to differentiate packet loss causes, such as collisions or weak signals.

Recent works leveraged ML for causes classification using KPIs. Salinas et al. [98] identified common WiFi impairments, such as interference and congestion, and proposed a supervised ML tool to diagnose these issues in home networks. Similarly, Salik et al. [96] developed ML models to detect non-WiFi interference in the 2.4GHz and 5GHz bands, which significantly impact WiFi performance. Syrigos et al. [94] proposed ML-based tools to detect common impairments, including hidden terminals and low signal strength.

1.5.3 Discussion

The RCD techniques reviewed in this paper, summarized in Table 1.5, represent significant advancements in network troubleshooting. However, they also present several limitations that can affect their scalability and performance in diverse network environments. Expert-based techniques, which rely on predefined rules and domain knowledge, struggle to adapt to evolving network conditions or new fault types. These methods often require frequent manual updates to remain effective. Additionally, techniques that depend on user-level probes or active measurements may introduce latency, reducing their ability to accurately detect network impairments.

Data-driven techniques address some of the shortcomings of expert-based methods, yet they introduce their own challenges. Most RCD approaches are supervised and rely on large, labeled datasets, which are not always readily available or sufficiently diverse to capture the full spec-

Table 1.5: RCD Techniques for Cellular and WiFi Networks

Network	Paper	Year	Summary
Cellular networks	Watanabe et al. [100]	2008	Statistical rule-based system used for fault identification in cellular networks.
	Kane et al. [101]	2015	Expert-based technique to identify network faults using system rules.
	Dimopoulos et al. [102]	2015	Supervised ML for video streaming QoE root-cause analysis using multi-layer network features.
	Chen et al. [103]	2019	Neural networks and SVM used for fault detection in 4G networks based on KPIs.
	Liu et al. [104]	2019	QoE-driven KQI framework for fault diagnosis in LTE networks, targeting video and messaging apps.
	Shi et al. [105]	2022	NeTExp, a dual-classifier (GNN and CNN) system for analyzing network and UE-side KPIs.
	Fida et al. [106]	2023	Two-stage method with VAE and MLPs for bottleneck detection in cloudified mobile networks.
	Hasan et al. [107]	2024	Simba, a GNN and transformer-based model for spatiotemporal KPI data analysis in 5G RAN.
Wi-Fi networks	Rayanchu et al. [99]	2008	Statistical method to differentiate packet loss causes like collisions and weak signals in WiFi networks.
	Kanuparthy et al. [97]	2012	Active probing techniques to diagnose WiFi impairments.
	Kim et al. [95]	2014	WiSlow, an expert-based tool to detect channel contention and non-WiFi interference in WiFi networks.
	Da Hora et al. [108]	2018	K-means clustering to detect poor QoE for web browsing, streaming, and WebRTC apps.
	Salinas et al. [98]	2018	Supervised ML tool to diagnose WiFi impairments like interference and congestion in home networks.
	Syrigos et al. [94]	2019	ML-based tools for detecting common WiFi impairments such as hidden terminals and low signal strength.
	Salik et al. [96]	2023	ML model to detect non-WiFi interference in 2.4GHz and 5GHz bands affecting WiFi performance.

trum of potential network faults. This limits their generalizability. Moreover, many of these techniques demand substantial computational resources for both training and inference, which can hinder their application in real-time and large-scale network environments.

In this thesis, Chapter 5 proposes a two-stage approach tailored for RCD of Cloud VR applications. By leveraging contrastive anomaly detection techniques, our approach achieves strong diagnostic results, even with minimal labeled data, while maintaining reasonable training and inference times.

Takeaways

- LL applications like cloud gaming, Cloud VR, video conferencing, and remote medical surgery share a need for minimal end-to-end delay, but their bandwidth and latency requirements differ. CG and Cloud VR, the focus of this thesis, demand high network reliability to deliver interactive and immersive experiences. These applications involve complex architectures and protocols, with network factors like delay, jitter, bitrate, and packet loss significantly impacting user experience as highlighted by many studies in the literature.
- Cellular networks have evolved from 1G to 5G, with 5G offering ultra-low latency, enhanced bandwidth, and novel concepts like network slicing and edge comput-

ing. Similarly, Wi-Fi standards have progressed from IEEE 802.11 to Wi-Fi 7, delivering higher throughput and reduced latency. Despite these advancements, both network types face multiple challenges resulting in bandwidth unavailability and latency instability, which impact performance of LL applications.

- ML and DL have become essential for addressing complex problems across various domains. DL leverages neural networks, which excel in capturing intricate patterns, to design inference functions that solve tasks through various learning paradigms, including supervised, unsupervised, and self-supervised learning. Building a DL solution involves several key steps such as data preparation, model design, training, evaluation, and deployment, ensuring the performance and generalizability required for deployment in real-world scenarios
- AD techniques in time series involves identifying deviations from normal patterns, often crucial in monitoring systems like networking. Multivariate time series data pose unique challenges, addressed by unsupervised methods leveraging statistical models, clustering, reconstruction-based approaches, or forecasting. Recent advancements in self-supervised learning, particularly contrastive learning, have further improved detection capabilities. Evaluating AD techniques relies on metrics like F1-score, MCC or AU-PR to better assess the AD accuracy.
- RCD aims to pinpoint specific factors causing performance degradation in networks, such as congestion, interference, signal attenuation... Traditional expert-based methods are increasingly complemented by automated, ML-driven approaches. Time varying network RCD often employs clustering, statistical methods, supervised ML and deep neural networks to analyze vast telemetry data and classify impairments. These techniques enhance network performance by addressing underlying issues efficiently.

Link to the thesis objectives

This thesis focuses on anomaly detection and root-cause diagnosis for the performance of low-latency (LL) applications, with a specific emphasis on CG and Cloud VR. These applications were chosen due to their growing business relevance for Orange and their sensitivity to network performance issues. In Chapter 2, we conduct data collection using commercial CG platforms and experimental Cloud VR systems under time-varying network conditions (4G and Wi-Fi networks). These conditions are designed to replicate real-world scenarios prone to the network factors which can cause performance anomalies in LL applications. The resulting datasets provide the foundation for evaluating anomaly detection techniques in Chapters 3 and 4. Furthermore, they are used to evaluate our proposed solution for root-cause diagnosis of LL application performance issues, detailed in Chapter 5.
We follow the same DL implementation procedure when designing ML solutions, whether for anomaly detection (Chapters 3, 4) or root-cause diagnosis (Chapter 5).
In this thesis, we adopt the unsupervised learning paradigm for AD to address the limitations of supervised learning, which requires labeling all collected data—a process that would be impractical. This approach enables us to evaluate and compare existing UL-

based AD solutions for detecting QoE degradation in Chapter 3. Furthermore, in Chapter 4, we propose a UL-based solution, specifically leveraging contrastive learning for time-series AD, which outperforms existing methods on our CG datasets and other benchmark datasets. Building on this approach, we also propose in Chapter 5, a technique for RCD in Cloud VR applications operating on Wi-Fi networks.

Chapter 2

Measurements studies and data collection

Summary: *This chapter presents the methodologies and experiments used to collect datasets for training machine learning models for anomaly detection and root cause diagnosis in low-latency applications. We conduct three data collection experiments, each performed in a lab environment, with careful attention to ensuring they were both reproducible and as realistic as possible. The experiments focus on 4G networks, cloud gaming applications under emulated 4G conditions, and Cloud VR applications in Wi-Fi scenarios. These datasets form the foundation for the machine learning models developed and evaluated in subsequent chapters.*

Contents

Contributions

The main contributions of this chapter can be summarized as follows:

- We collect 4G network conditions using the Saturator tool, generating transmission opportunity (txops) files that emulate realistic, time-varying network behaviors.
- We collect cloud gaming datasets under emulated 4G conditions by leveraging the collected txops files, Mahimahi-LinkShell, and DECAF tools to capture detailed Quality of Experience (QoE) and Quality of Service (QoS) metrics.
- We extend the data collection to Cloud VR applications operating under Wi-Fi network conditions. Experiments are conducted to collect Cloud VR datasets under both normal and impaired Wi-Fi conditions, simulating scenarios such as interference and signal attenuation.

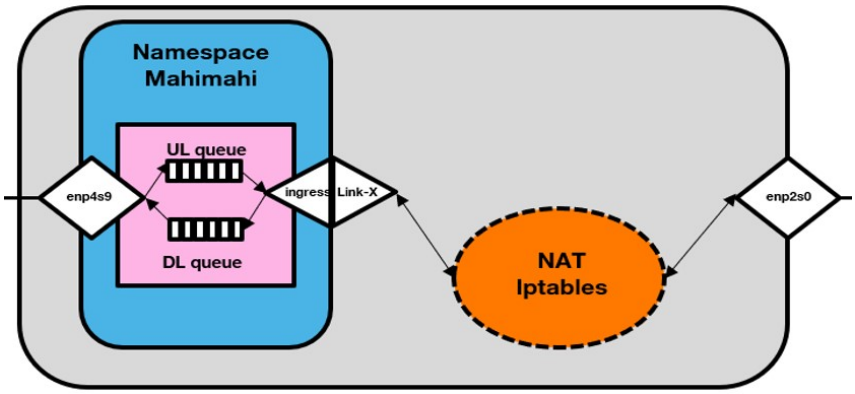


Figure 2.1: LinkShell component.

2.1 Measurements and data collection of 4G network conditions on Orange commercial network

2.1.1 Motivation

Controlled experiments under realistic network conditions is crucial in networking. In this thesis where performance of low-latency applications under time-varying capacity networks like cellular networks, it was important to be able to perform reproducible and realistic experiments on LL applications under cellular networks. To this end, MIT researchers developed a network emulation tool called Mahimahi This tool offers the ability to reproduce the behavior of time-varying capacity networks using recorded transmission opportunities (*txops*). However, the txops used by Mahimahi are old and not representative of current cellular network capacities. They were collected on Verizon and TMobile LTE network in 2016 with a downlink throughbout about 5-10 Mbps while the rapport of ARCEP [109] of 2021 report that in France, the downlink throughput on average whatever the location or the network operator used were about 71 Mbps. Based on this observation, it was important to collect more recent txops file to perform better evaluations.

2.1.2 Measurement methodology

Mahimahi-LinkShell

Mahimahi [110] is a framework designed to capture and replay traffic from HTTP-based applications under emulated network conditions. It operates by creating a set of Linux shells, each contained within its own network namespace, meaning that each shell has an isolated network stack with its own routes, firewall, and network devices. Mahimahi includes 04 tools such as LinkShell, RecordShell, DelayShell, and LossShell. Compared to earlier traffic capture and replay tools, Mahimahi offers several advantages: it enables multi-server emulation for web applications and provides network namespace isolation in each shell. This isolation eliminates variability caused by interference from cross-traffic, thus ensuring reproducible experiments. Additionally, Mahimahi's composability and extensibility make it a versatile tool for network research.

In this thesis, the Mahimahi component of particular interest is LinkShell. LinkShell, depicted in Fig. 2.1, allows for the emulation of network links, either with a fixed rate or with

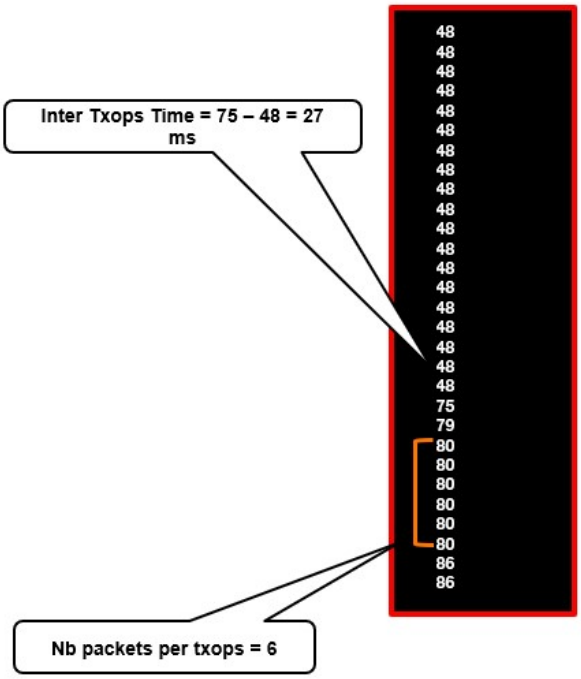


Figure 2.2: An example of a txop file.

time-varying conditions, such as those found in cellular networks. To achieve this, LinkShell relies on two transmission opportunity (txops) files: one for uplink and one for downlink network conditions. Each line in a txops file represents a transmission opportunity, defined by a time (in milliseconds) when a Maximum Transmission Unit (MTU)-sized packet (1500 bytes) can be transferred through the emulated link. Since actual packet sizes can often be smaller than 1500 bytes, a transmission opportunity may allow for the transfer of multiple packets whose combined size equals 1500 bytes. LinkShell queues incoming packets (separately for uplink and downlink) and transmits them during the corresponding transmission opportunity. If no packet is available during a transmission opportunity, that opportunity is wasted.

An example of a txops file is depicted in Fig. 2.2

Saturator Tool

Since the existing txops files are outdated, it was necessary to collect new ones that more accurately reflect the current capacity of cellular networks. For this purpose, we utilized the Saturator tool [11], which captures network trace samples by saturating the radio link between a User Equipment (UE) and the base station, enabling precise measurement of transmission opportunities. The operation of the Saturator tool is illustrated in Fig. 2.3. Our setup consisted of two Xiaomi Mi 10 5G UEs, one Ubuntu 20.04 machine acting as the client, and one Ubuntu 18.04 machine as the server. The client saturated the uplink of the USB-tethered UE, while the server saturated the downlink. Acknowledgements were exchanged between the client and server through a non-saturated WiFi-tethered UE, ensuring accurate communication without interfer-

[11] `https://github.com/keithw/multisend/blob/master/sender/saturatr.cc`

Table 2.1: Characterization of the six txops captured

Network Conditions	Mean (Mbps)	Std (Mbps)	% of time ≥ 20 (Mbps)	% of empty period of 33ms	Location (France)	Hour
Trace E	230	48.7	99.9	2.6	Orange Lannion	2pm
Trace D	173.3	43.4	99.8	3.8	Orange Lannion	11am
Trace C	141.6	49.5	97.4	0.4	Brélévenez	9am
Trace B	78.8	20.3	99.5	8.0	Brélévenez	1pm
Trace A	45	7.4	99.3	5.8	Plemeur-Bodou	3pm
Highway	43.7	31.5	70.5	17.7	Guingamp - Lannion	10am

ing with the uplink and downlink saturation. During the process, the client sent timestamped packets to the server, which computed the round-trip time (RTT) upon receiving acknowledgements. The RTT information was then used to adapt the client's congestion window, ensuring continuous saturation of the radio link. A similar process occurred on the server side to keep the radio link between the base station and UE fully saturated.

Once the capture was completed, the raw data collected by the client and server was aggregated and processed to generate two txops files (one for uplink and one for downlink). These files are used by Mahimahi's LinkShell to emulate time-varying cellular networks, replicating the behavior observed during the experiment.

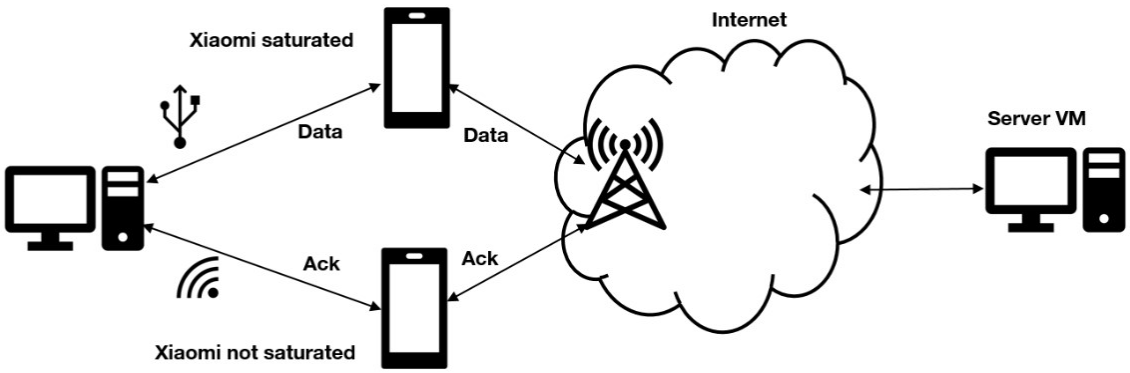


Figure 2.3: Saturator tool functioning.

Experiments and results

We conducted txops captures using this setup on the Orange 4G network in January 2022. Each capture lasted for 10 minutes, under varying network conditions, including differences in average bandwidth, bandwidth variability, and time gaps, to ensure the generated txops files would be representative of typical network scenarios. Additionally, we captured txops data during a mobility experiment on a highway between Guingamp and Lannion (France), traveling at 110 km/h. This capture reflected highly variable coverage, with strong signal near urban areas and

weak coverage in rural zones.

As a result, six txops files (covering both uplink and downlink) were collected, representing diverse conditions of the Orange 4G network. The experimental conditions and key statistics of the recorded traces are presented in Tab. 2.1. Further details on the six txops files can be found in the appendix A.

Takeaways

We collect in this experiment new txops files that will be used in the rest of our work for data collection on CG platforms but could also serve for other use-cases like performance evaluation of web applications under cellular network conditions [111] or for congestion control evaluation.
As such, the datasets are available as OpenData : `https://cloud-gaming-traces.lhs.inria.fr/ANR-19-CE25-0012_Mahimahi_Orange_4G_traces_January_2022.zip`

2.2 Data collection of user-KPI time series data for cloud gaming sessions over 4G networks

2.2.1 Motivation

Using the six transmission opportunity (txops) files generated from the previous experiments, we conducted a series of measurements to collect datasets for cloud gaming (CG) applications operating under realistic 4G network conditions. These datasets are essential for developing machine learning (ML) models aimed at detecting QoE degradations. By leveraging both representative network conditions and commercial CG platforms, we ensured that the collected data closely reflected real-world scenarios, making the results applicable to actual deployment environments.

2.2.2 Methodology

Our methodology for data collection relies on the integration of the Mahimahi-LinkShell tool (described earlier) and the DECAF tool, enabling comprehensive QoE and QoS measurement for CG platforms.

DECAF tool

DECAF [16] is a tool designed to measure delay and streaming performance for CG applications. While DECAF includes capabilities for game delay measurement, we focused exclusively on its streaming performance features in this work. This decision was made because the game delay measurement module is not generalizable across the range of video games used in our experiments.

The streaming measurement module of DECAF is built on an instrumentation of the WebRTC API in Chromium, extending its capabilities to collect additional metrics beyond those natively provided by the API. This enhancement allowed us to extract detailed time-series Key Performance Indicators (KPIs) relevant for QoE degradation detection,

Testbed

The testbed used for data collection is illustrated in Figure 2.4. It consists of two main components:

- Windows 10 laptop client: This device, equipped with the Chromium browser, is used for running and playing the games from CG platforms.
- Ubuntu 20.04 laptop as a network emulator: This machine runs the Mahimahi platform, serving as a controlled environment for emulating realistic 4G network conditions based on the txops files generated earlier. The network emulator connects to CG platform servers via a Fiber-To-The-Home (FTTH) Internet connection, ensuring that the bottleneck remains at the emulator rather than the Wide Access Network (WAN) interface.

During gameplay sessions, incoming uplink and downlink packets are queued within the network emulator and released by Mahimahi's LinkShell in accordance with the txops files. This configuration faithfully reproduces the time-varying behavior of 4G network conditions captured during the earlier measurements.

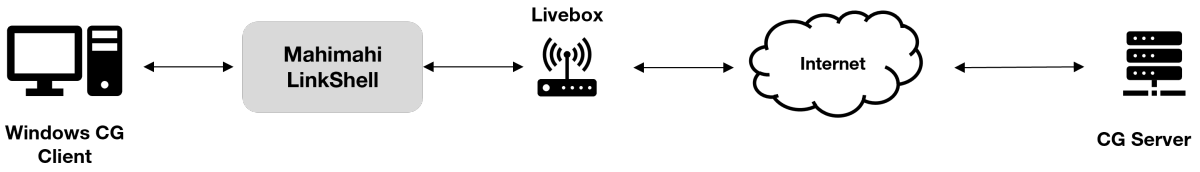


Figure 2.4: Cloud Gaming measurements testbed.

We performed measurements on three major CG platforms compatible with Chromium and available in Europe during the experiment period: Google Stadia (STD), Nvidia GeForce Now (GFN), and Microsoft XCloud (XC). To maintain consistency across experiments, we selected racing games that were available on each platform:

- *Dirt 5* for STD
- *TrackMania* for GFN
- *F1 2021* for XC

Each game session lasted at least 5 minutes and was conducted under each of the six distinct 4G network conditions represented by the txops files. This ensured a diverse dataset covering various scenarios, including static and mobility-based network conditions.

During the gameplay sessions, we collected time-series features representing both QoS and QoE metrics. These included data such as bandwidth, frame rate, latency, packet loss, and other performance indicators.

Takeaways

The collected datasets are used in subsequent chapters of this thesis (Chapters 3 and 4) for ML techniques for anomaly detection in multivariate time series for QoE degradation detection in CG applications.

The datasets are available as OpenData : `https://cloud-gaming-traces.lhs.loria.fr/ANR-19-CE25-0012_std_gfn_xc_cg_webrtc_metrics.7z`

2.3 Data collection of CloudVR data over Wi-Fi networks

2.3.1 Motivation

Previous experiments on CG applications conducted over cellular networks provided foundational datasets for detecting anomalies related to QoE degradation in CG sessions. However, these datasets were not suitable for performing root cause analysis, as the underlying causes of the anomalies could not be identified. This limitation stemmed from the complex and intrinsic phenomena occurring in real cellular networks, which make it challenging to isolate and attribute specific causes to each observed anomaly. Additionally, while cloud gaming is a significant low-latency application for network operators such as Orange, it primarily serves individual clients. In contrast, industry clients, who represent a major segment of the market, are more likely to leverage Cloud VR for their business needs.

Given this context, we chose to focus our data collection efforts on Cloud VR applications over Wi-Fi networks. This approach allows us to generate datasets that are specifically designed to facilitate root cause analysis for Cloud VR performance issues. Wi-Fi networks offer the advantage of replicating real-world scenarios in controlled laboratory environments. Using Faraday cages, we can emulate various network impairments, enabling us to systematically label the collected datasets for ML training and analysis.

2.3.2 Methodology

In this section, we outline the testbed developed for CloudVR experiments, which leverages the NVIDIA CloudXR architecture to deliver immersive virtual reality experiences over wireless networks. We begin by providing an overview of the CloudVR application framework, followed by a detailed description of the testbed infrastructure and automation components.

Testbed Infrastructure

The testbed is built upon the NVIDIA CloudXR SDK, an XR streaming platform that utilizes edge computing and cloud-based GPU acceleration to deliver high-fidelity VR experiences. The CloudXR platform streams OpenVR-based applications from a remote server to XR client devices with limited graphical capabilities, such as the untethered Head-Mounted Displays (HMDs) Meta Quest 2 and Meta Quest 3. These devices connect to the server through various network interfaces, including Ethernet, Wi-Fi, or cellular connections.

The CloudXR architecture consists of the following key components:;

- CloudXR Driver: This server-side driver streams audio and video content from XR applications (e.g., SteamVR) to the client device using network protocols like UDP.
- CloudXR Client: The client-side software enables the HMD to receive and render the streamed VR content while maintaining low latency and high-quality visuals.

The CloudVR testbed infrastructure, illustrated in Fig. 2.5, is designed to facilitate controlled experiments under realistic network conditions. It consists of i) a server, a high-performance

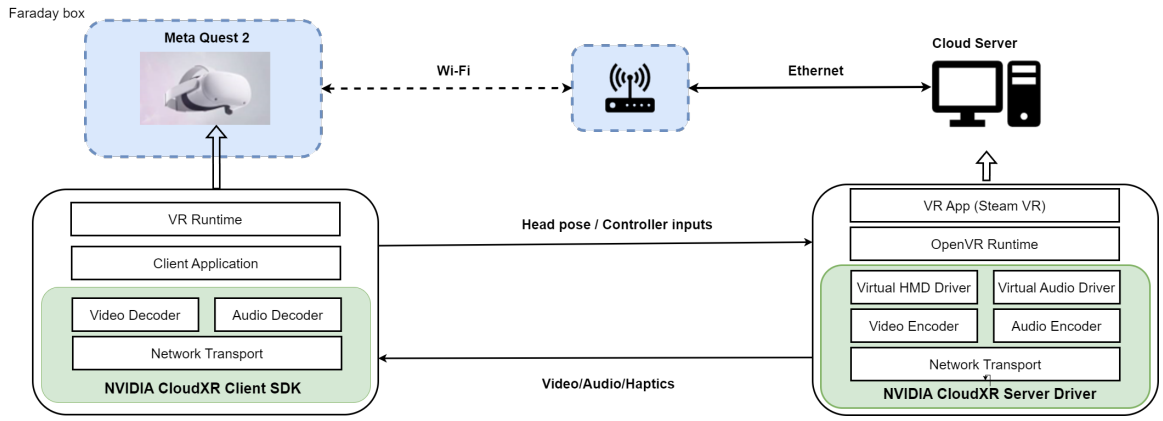


Figure 2.5: Cloud XR Architecture

machine equipped with an Intel(R) Xeon(R) W2235 CPU @ 3.8GHz with 32 GB of RAM, NVIDIA GeForce RTX 3090 Ti with 24GB, running the CloudXR server software to render and stream VR content; ii) a thin client, the Meta Quest 2 HMD, which acts as the client device, connected wirelessly to the network and iii) a Wi-Fi AP, a Orange commercial Livebox router that serves as the wireless bridge between the server and the HMD.

Wi-Fi Testbed for Controlled Experiments

Wi-Fi networks are critical for ensuring seamless connectivity for CloudVR applications, but they are inherently susceptible to performance degradation due to obstacles, interference, and competing traffic. To study these challenges systematically, we designed a controlled Wi-Fi testbed, as illustrated in Fig. 2.6. This setup allows for the reproduction of real-world scenarios and controlled network impairments in laboratory conditions, enabling detailed analysis of their impact on CloudVR services.

The testbed consists of two primary layers:

- **Infrastructure layer:** This layer includes up to four Faraday cages used to isolate equipment from external electromagnetic interference. The cages are allocated as follows: one for the VR headset, two for the APs, and one for a station used in interference scenarios. The layer also includes two access points (AP1 and AP2): AP1 is utilized for normal and coverage experiments, while AP2 introduces interference in specific scenarios. The Faraday cages are interconnected using coaxial cables to transmit Wi-Fi signals, and Radio Frequency (RF) attenuators [12] are employed to simulate variations in signal strength during experiments. Two servers are part of this setup: the first server (see Fig. 2.5) runs the CloudVR software and is connected to AP1 via Ethernet, while the second server acts as a traffic generator connected to AP2, simulating competing downlink traffic using UDP.

- **Control and Automation layer:** This layer includes a VR PC controller connected to the VR headset via USB, responsible for managing headset settings and collecting performance metrics, such as KPIs from OVR Metrics tools[13] or quality-of-service (QoS) statistics from CloudXR. The controller also automates game sessions using the Meta Quest

[12] https://adauratech.com/attenuators/
[13] https://developers.meta.com/horizon/downloads/package/ovr-metrics-tool/

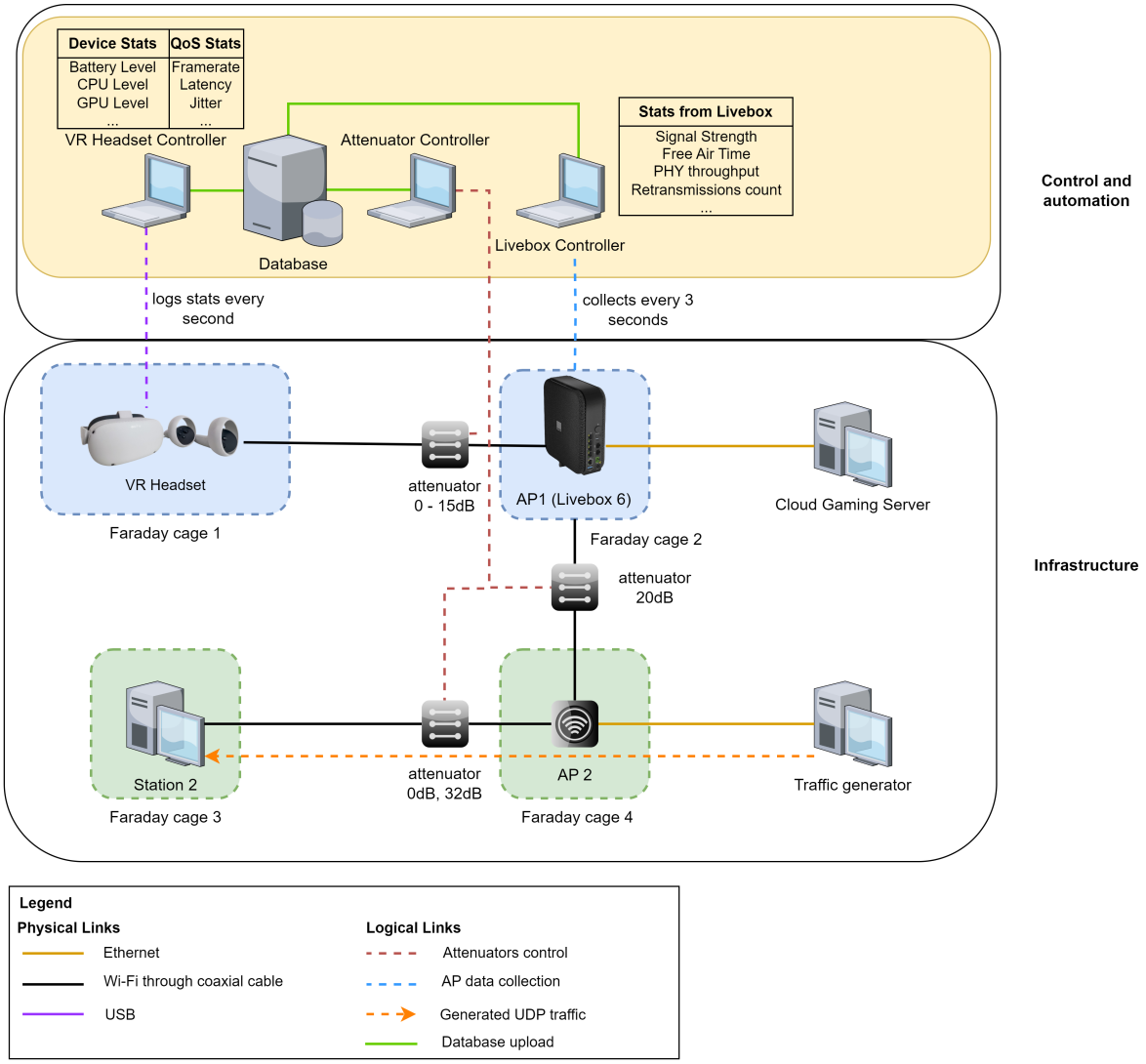


Figure 2.6: Wi-Fi testbed architecture

Autodriver[14]. Additionally, this layer features an attenuator controller that configures and manages RF attenuators via APIs, enabling automated signal attenuation adjustments through FastAPI [15]. Furthermore, the Livebox controller manages the Livebox via Telnet to collect Wi-Fi KPIs every 3 seconds. A local ELK Stack [16] database aggregates data from all controllers for post-experiment analysis.

Scenarios considered

We collect Cloud VR data across several controlled scenarios using the Beat Saber game as a benchmark. These scenarios were executed on both the 2.4 GHz and 5 GHz frequency bands,

[14] https://developers.meta.com/horizon/documentation/unity/ts-autodriver
[15] https://fastapi.tiangolo.com/
[16] https://www.elastic.co/fr/elastic-stack

as detailed below:

- **Normal scenarios :** CloudVR sessions were conducted under ideal conditions, without any Wi-Fi impairments. The VR headset was positioned in an optimal coverage area and connected to the primary access point (AP1) for each frequency band. A total of 5 sessions, each lasting 300 seconds, were performed for both the 2.4 GHz and 5 GHz bands.
- **Coverage scenarios :** To evaluate the impact of signal strength variations, an RF attenuator was used to simulate different received signal strength indicator (RSSI) values during VR sessions agter launching the game. RSSI levels of {-55 dB, -60 dB, and -65 dB} in 2.4 GHz and RSSI levels of {-80 dB, -85 dB, and -90 dB} in 5GHz. Each configuration was tested in three sessions, with each session lasting 100 seconds.
- **Interference scenarios :** Interference was introduced by a neighboring station connected to a secondary access point (AP2) operating on the same frequency and bandwidth as AP1. The interference was simulated by occupying a portion of the transmission opportunities (txops) available on AP1 after starting the game. Interference levels leaving {10% and 15%} of txops in 2.4 GHz band and interference levels leaving {9%, 10%, and 11%} of txops in 5GHz for AP1. Each configuration was tested in three sessions, with each session lasting 100 seconds.

The testbed configuration ensures a highly controlled environment for Cloud VR experiments. By utilizing Faraday cages, we eliminate uncontrolled radio interference, and by deploying local servers, we remove fluctuations from external network segments. Automation allows for the seamless execution of test scenarios and comprehensive data collection of Cloud VR applications.

Data collected

During the Cloud VR experiments, three distinct types of data were collected to comprehensively capture application performance and network behavior:

- **Application-Level Metrics**: Metrics were obtained directly from the Oculus OVR Metrics Tool, which provides detailed insights into the performance of the VR application on the headset. These include frame rates, resolution, CPU and GPU levels and usage, and other statistics related to VR streaming. These metrics help evaluate the QoE from the perspective of the end user. Additionally, QoS metrics were retrieved from the NVIDIA CloudXR stack, which facilitates the streaming of high-quality VR content from the server to the headset. These metrics include network round trip delay, client queuing time, packet loss, and jitter.
- **Livebox-Level Metrics**: The collected Wi-Fi KPIs include metrics from the Wi-Fi physical layer, such as Received Signal Strength Indicator (RSSI), background noise levels, free airtime, Modulation and Coding Scheme (MCS) index, channel bandwidth, and physical layer bitrates, as well as metrics from the network layer, such as data bitrates, dropped and retried packets, and packet counters for the Basic Service Set (BSS). These metrics were directly collected from the proprietary Livebox router to monitor and evaluate the Wi-Fi conditions experienced during the Cloud VR sessions.
- **Raw Traffic Captures**: Packet Capture (PCAP) data was collected using a probe placed in the Faraday cage near the VR headset. These raw traffic captures provide a detailed

view of packet-level interactions between the VR client and the server, including timestamps, protocol usage, packet sizes, and packet inter-arrival times, which may be useful for detecting potential anomalies at the transport and application layers.

Takeaways

For the experiments presented in this work, the RCD focused solely on the Livebox-level metrics because they are easily accessible by network operators, who own and manage the Livebox. These metrics were used to detect and classify anomalies affecting the Wi-Fi performance during Cloud VR sessions. The application-level and raw traffic metrics, while not utilized in the current analysis, present opportunities for future work. They could enhance diagnostic performance by providing richer context and additional data points for identifying the underlying causes of performance degradation.
The datasets are available as OpenData: `https://cloud-gaming-traces.lhs.loria.fr/ANR-19-CE25-0012_cloud_vr_wifi_metrics.zip`

Chapter 3

Evaluation of unsupervised ML models for QoE degradation detection in CG applications

Summary: *This chapter evaluates eight unsupervised ML models for AD using the real-world cloud gaming KPI datasets collected on Chapter 2, focusing on robustness, efficiency, and computational complexity. Beyond traditional F1-score metrics, the study employs Matthews Correlation Coefficient (MCC) for a nuanced comparison. It introduces a novel Window Anomaly Decision (WAD) approach to address shortcomings in existing window-based techniques and provides recommendations tailored to application requirements.*

Contents

3.1 Introduction

Recent years have witnessed the deployment of high performance networks, e.g., FTTH and 5G mobile networks, to support the stringent and requirements in terms of latency, bandwidth, reliability, and jitter of emerging applications. For instance, remote surgery requires reliability and low-latency for video streaming and control feedback; AR/VR applications and the Metaverse require high-bandwidth and low-latency to allow human interaction with the environment. Cloud Gaming (CG) applications also face challenges in terms of delay and bandwidth consumption which are hardly met by current cellular network architectures, exposing CG users to network impairments that deteriorate their QoE. ISPs need efficient monitoring techniques to detect QoE deterioration that their customers may experience.

A primary approach widely adopted in networking to detect users' QoE degradation is to rely on AD methods which are either applied manually or rely on rule-based techniques [112]. However, the increasing complexity of network infrastructures tend to make these techniques impractical and inefficient. The recent advances in Machine Learning (ML) have been leveraged in various domains, including networking where large amounts of data are available and could be used to build classification and prediction models. The popularity of ML is mainly due to the success of supervised learning which requires a plethora of labeled data during the training phase. However, data labeling has to be done by domain experts and has proven to be a long and tedious process. To circumvent the need for labeled data, unsupervised learning techniques are increasingly adopted, in particular for anomaly detection.

Many anomaly detection models based on ML techniques have been proposed in the literature, including distance-based algorithms [60, 113], predictive algorithms, reconstruction-based algorithms [62, 64], one-class algorithms [61, 76], generative algorithms [114], etc. The performance of these models is usually evaluated using the F1-score as the key performance metric. The majority of these studies also used a *point-wise* approach to compute the F1-score, i.e., they consider only anomalous observations. The issue with the point-wise (PW) approach is that it disregards the fact that degradations can occur in the form of consecutive anomalous observations. To address this limitation, new approaches [65, 115, 73] are proposed to better evaluate the performance of the AD models. These approaches consider degradations as *windows* of anomalies, aggregate the predictions following different criteria and compute the F1-score accordingly. Another issue presented by many of existing ML models is the difficulty in comparing them for a given application since each model is evaluated using a different methodology, benchmark datasets and evaluation metrics. Each method reports higher F1-score than its competitors making it tricky for domain experts to choose the most appropriate approach that fits their needs.

In this chapter, we present a comprehensive comparison and analysis on the performance of unsupervised ML models through experiments while relying on a consistent evaluation methodology. In particular, we focus on the detection of QoE degradation in low-latency applications using a real-world multivariate time-series dataset of Key Performance Indicators in CG sessions collected under different 4G network conditions. The game sessions are recorded using the public Google Stadia CG platform. We objectively evaluate the ability of each model to detect anomalies that can lead to QoE degradation for CG users. The experiments conducted under 4G conditions and the conclusions drawn from them remain applicable to 5G networks, as the QoE degradation we aim to detect are related to features collected at the CG application-level. In addition, we provide insights and offer recommendations to network operators on the most appropriate AD model that meets their application requirements for anomaly detection, including real-time or offline inference, detection accuracy, robustness to data contamination

rate due to different environments, etc.

Contributions

The main contributions of this chapter can be summarized as follows:

- We demonstrate, based on synthetic models, the limitations of existing *window* approaches for evaluating the performance of AD models and then propose our Window Anomaly Detection (WAD) approach.
- We perform an exhaustive evaluation of ML models to assess (i) their robustness by injecting anomalies in their training sets (i.e., data contamination), and (ii) their capability to detect short and longer users' QoE degradation by varying the windows size.
- Based on our experiments, we offer recommendations to network operators and network management experts on the best models for different application requirements.

3.2 Background and Related work

In this section, we first discuss CG applications and the impact of latency on their performance. Then, we review existing unsupervised learning models and window-based approaches used for anomaly detection using time series data.

3.2.1 Cloud Gaming applications

Cloud games are processed and executed on cloud servers and the rendered scenes are streamed over the Internet to client devices removing the need for having powerful gaming computers or consoles [116]. Several works [6, 16, 17] focused on studying the behavior of CG platforms and analysed their respective network protocols. For instance, Google Stadia uses WebRTC for its service and relies on RTP to stream its audio and video contents. Marchal et al. [6] showed that link capacity and latency are driving Stadia platform to adapt its bitrate and resolution.

On the other hand, several works [21, 117, 118, 119] addressed the impact of network latency on the performance of gamers in cloud and non-cloud games. They showed that the performance of gamers drops with the increase of latency. Raaen et al. [117] and Vlahovic et al. [119] also include in their subjective studies that most of the gamers could not tolerate a delay above 100ms and the most demanding gamers are able to perceive delays below 40ms. These network perturbations can lead to QoE degradation, which this work aims to detect using unsupervised ML algorithms.

3.2.2 Unsupervised Learning models for anomaly detection

Anomaly detection is a popular and a well-covered research topic with numerous surveys [49, 120, 121, 122] proposing different taxonomies and categories of techniques, including existing machine learning models for anomaly detection. Some studies [123, 48, 124, 125] have specifically focused on the use of deep-learning for anomaly detection due to its efficiency in handling multivariate and high-dimensional data. Schmidl et al. [120] categorized anomaly detection

models for time-series data depending on the way they determine anomalies. Reconstruction-based algorithms detect anomalies by learning a model from *normal* training data. They encode the training features into a low-dimensional (i.e., latent) space and reconstruct the input features from the latent features. An anomaly score is computed by comparing the reconstructed data to the input data in order to detect anomalies. Since the model is built on *normal* data only, anomalous time-series cannot be well-reconstructed and will have a high anomaly score (i.e., above a pre-determined threshold). This class of algorithms, include Principal Component Analysis (PCA) and neural network algorithms such as AutoEncoder, LSTM-VAE [66], DAGMM [62], OmniAnomaly [67], Donut [65], and USAD [64].

One-class classification models detect anomalies by learning a hypersphere that encloses the representation of normal data. Any points that remain outside the learned hypersphere are classified as anomalies. The most popular algorithm from this category is the OC-SVM [61] and its neural network variant Deep-SVDD [76].

Furthermore, it is worth mentioning that there are other families of unsupervised learning models for time-series data, including Isolation methods that build an ensemble of isolation trees to isolate anomalies. Isolation methods such as Isolation Forest [60] assume that anomalies are easy to *isolate* since they are fewer in the data and are starkly different from normal instances. Distance-based methods (e.g., KNN [113], LOF[55]) detect anomalous instances based on their larger distances from normal instances. Predictive methods (e.g.,ARIMA) predict time-series sequences and compare them to original time-series to differentiate the anomalies. Generative methods (e.g., GAN [114]) train a model to generate new (i.e., normal or anomalous) data based on real data distribution and a discriminator that learns how to discriminate real data from generated data.

The aforementioned anomaly detection models were evaluated in different studies [126, 127, 122] either with broad or domain-specific benchmark datasets. Our work provides a comprehensive and consistent experimental evaluation of anomaly detection models for QoE degradation detection, while taking into account the impact of data contamination and window size.

3.2.3 Window-based approaches for anomaly detection

Chandola et al. [49] categorize anomalies into three classes: *point anomalies* which are individual observations that deviate from normal ranges; *collective anomalies* that represent a group of anomalous observations and *contextual anomalies* that represent a group of observations that can be considered as anomalous in a specific context. Point anomalies are the most widely addressed anomalies in the literature.

In most time-series, anomalies occur as contiguous anomalous observations (*i.e., collective anomalies*) rather than individual *point anomalies*. Assessing the performance of AD models for time-series data using PW approaches is ineffective as it disregards the contiguous nature of anomalies. To overcome this limitation, point-adjust (PA) approach is proposed by Xu et al. [65] and is often employed [67, 64] to deal with windows of anomalies. PA approach considers that all the anomalies of a window are correctly detected by a AD model if any of the anomalous observations in the window is correctly detected. However, it was shown in [128, 79, 129] that PA approach presents limitations as well and may overestimate the performance of an AD model. The authors showed that a well-trained and efficient algorithm provides the same F1-score as a random model with PA approach. Hence, with the PA, it is difficult to conclude that a model outperforms others.

To address the limitations of PA approach, revised point-adjust (RPA) [73] and PA%K [128] were proposed. Unlike PA, RPA is less tolerant to high false-positive rates by severely penaliz-

ing them. Other approaches such as Numenta Anomaly Benchmark (NAB) [115] and range-based Precision/Recall [130] have also been proposed, but they are are too complex to be widely adopted. In this chapter, we propose the WAD approach, which addresses the aforementioned limitations and aims at fairly assessing the performance of anomaly detection models.

3.3 Methodology

In this section, we describe our general methodology for evaluating ML models to detect anomalies in cloud gaming sessions. We first formulate the anomaly detection task for anomalous windows in CG sessions and introduce the collected datasets. Our new approach, called Window Anomaly Decision (WAD), proposed to address the limitations of existing window approaches, is then presented. This is followed by an introduction on the unsupervised ML models used in our comparative evaluation.

3.3.1 Problem statement

Our goal is to detect the end-users' QoE degradation in CG sessions. For this, we decided to base our detection analysis using windows of observations, instead of individual points of observation. Indeed, QoE degradation lasting 5ms is not perceptible by human beings for whom the perception of latency is around 150ms [131]. Therefore, we evaluate the ML models with 3 window sizes $p \in \{10, 20, 30\}$ to perform a detection within $50, 100, 150$ ms respectively, and have a representative time of perception (i.e., very reactive people to less reactive ones, via mean value).

As such, we can formalize our problem as follows: we denote the time-series of our datasets as $x = \{x_1, x_2, ..., x_T\}$, where T is the length of x and $x_t \in \mathbb{R}^m$ denotes a m-dimensional vector corresponding to the values of our m features at time t. For the representation of observations in windows, we split x into sequences of windows $W = \{w_1, w_2, ..., w_{T-p+1}\}$ with stride 1 (5ms) where $w_t = \{x_t, x_{t+1}, ..., x_{t+p-1}\}$, p being the window size.

Given an unsupervised anomaly detection model $\mathcal{M}$, a set of parameters $\mathcal{W}$ is learnt to output an anomaly score $s(\tilde{x_t})$ for each unseen observation $\tilde{x_t}$. From this anomaly score and a carefully chosen threshold δ, a binary variable $\tilde{y_t} \in \{0, 1\}$ is assigned for each observation as follows:

$$\tilde{y_t} = \begin{cases} 1, & \text{if } s(\tilde{x_t}) > \delta \\ 0, & \text{otherwise.} \end{cases} \tag{3.1}$$

3.3.2 Data collection

We rely on the CG datasets collected in Chapter 2 [17] which contain time-series features of QoE and QoS collected while playing racing games on public cloud gaming platforms:

- *Dirt 4* for Google Stadia (STD);
- *TrackMania* for Nvidia GeForce Now (GFN);
- *F1 2021* for Microsoft Xbox Cloud (XC).

[17] Datasets are available as OpenData : `https://cloud-gaming-traces.lhs.loria.fr/ANR-19-CE25-0012_std_gfn_xc_cg_webrtc_metrics.7z`

Table 3.1: Description of collected features

Features	Description
Network RTT	Network RTT computed during game session.
Decoding delay	Delay to decode each video frame.
Jitter buffer delay	Delay between the time, the first packet of a video frame enters the jitter buffer and the time the whole frame exits the jitter buffer.
Video rendering jitter	Time between two consecutive video frames.
Uplink Bitrate	Number of bits-per-second (bps) sent by the client.
Downlink Bitrate	Number of bits-per-second (bps) received by the client.
Frame rate	Frames received per-second (FPS).
Height resolution	Number of pixels in the frame height.
Width resolution	Number of pixels in the frame width.
Freeze	A freeze is count if an inter-frame delay is greater than a value defined as $max(3 * avgInterFrameDelay, avgInterFrameDelay + 150ms)$.
Frames dropped	Number of video frames dropped before decoding step.
Frames decoded	Number of video frames decoded.
Packets received	Number of packets received.
Downlink throughput	Max downlink capacity allowed by the 4G network conditions.

A set of 14 features (cf. Table 3.1) were collected through the Chromium WebRTC API adapted by the DECAF tool [16]. The measurements were performed under 6 different 4G network conditions, with 5 differing in the average downlink throughput and 1 in a mobility scenario on a highway.

The collected datasets are unlabeled (i.e., with no ground-truths). Consequently, evaluation of the models performance with well-known ML evaluation metrics, such as Precision, Recall or F1-score is not possible. We hence create ground-truth $(y_t)_{t\in\|1,T\|}$ (i.e., labels) for evaluation purposes (and intrinsically to split the data), but the labels are not used by ML algorithms during training. According to the CG platform's recommendations for high quality streaming,[18] [19] we define the ground-truths based on the following criteria ($\gamma = 1080$ for STD and XC and $\gamma = 768$ for GFN):

[18] https://support.google.com/stadia/answer/9607891?hl=fr/
[19] https://www.nvidia.com/en-us/geforce/products/geforce-now/system-reqs/

Table 3.2: Datasets summary

Datasets	Train normal windows	Contamination set windows	Test windows	Test anomalies ratio (%)
STD	80486	59480	169706	52.57
GFN	27415	22667	61417	55.36
XC	83611	17918	110487	24.32

$$y_t = \begin{cases} 1, & \text{if } \begin{cases} resolution(x_t) < & \gamma \text{ or} \\ frameRate(x_t) < & 60 \text{ or} \\ freeze(x_t) = & 1 \end{cases} \\ 0, & \text{otherwise.} \end{cases} \tag{3.2}$$

We would like to stress here that the primary objective of this chapter is to compare the performance of unsupervised ML models in detecting anomalies on a CG case study. The objective is not to detect degradation using simple rules, but rather to utilize such rules to establish ground-truths required for computing the ML metrics. Although being simple, there rules present a significant challenge for our tasks, particularly when they are unknown a-priori as shown later in the chapter our results in Section 3.4.

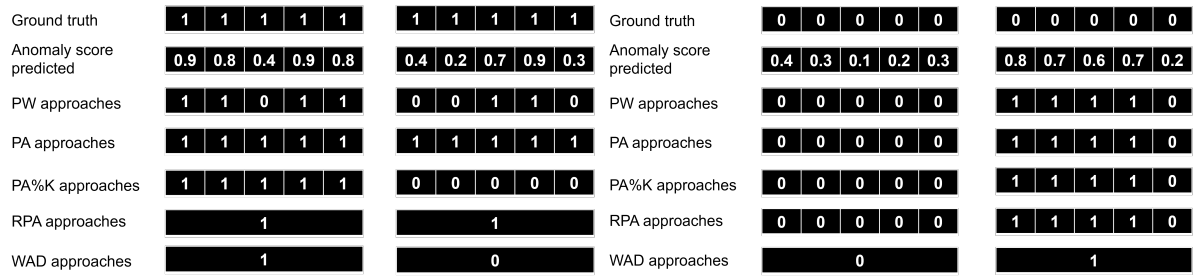
Using these ground-truth labels as a reference, we define a ground-truth window as anomalous if at least 80% of the labels of the window are anomalous.

3.3.3 Data processing and splitting

The features of the datasets are resampled to have a fixed time-step of 5ms and they are normalized before training. We build the training and testing sets following the splitting strategy proposed by Zong et al. [62] which ensures consistency over different experiments and allow to perform a fair evaluation. The entire dataset is split as follows: 50% of the normal samples are associated to the training set while the remaining 50% are considered as a test set. The test set also contains 60% of the anomalous samples. The remaining 40% of the anomalous samples compose a set called the contamination set. The anomalous samples in the latter set, are actual instances of anomalies that occurred during gameplay, and were subsequently collected in our datasets. They serve to *contaminate* the training set with anomalous samples (with a ratio $c \in \{0\%, 4\%, 8\%, 12\%, 20\%\}$) and study the robustness of ML algorithms to data contamination. The rationale behind this splitting strategy is to maintain consistency across various experiments and ensure fairness in evaluating model performance under conditions of data contamination. Table 3.2 shows the summary of our datasets after the splitting step with the number of normal/anomalous windows in the train and test sets.

3.3.4 Existing window evaluation approaches

As mentioned in Section 3.2.3, PW approaches are unsuitable for AD in CG sessions and window approaches are preferred. PA, PA$\%K$ and RPA approaches were proposed in [65] [128] and [73] respectively.



(a) Prediction on anomaly windows **(b) Prediction on normal windows**

Figure 3.1: Illustration of PW, PA, RPA and WAD approaches. 0 is normal and is 1 anomalous. The anomaly score threshold to decide if an observation is anomalous or not is $\delta = 0.5$. Window size $p = 5$, $\alpha = 0.8$.

PA approach assumes that if any observation in a ground-truth anomaly segment is correctly detected, all the observations in this segment are considered as anomalous and correctly detected (i.e., p true positives are recorded). If none of the observations in the ground-truth anomaly segment is correctly detected, p false negatives are recorded. Observations that are not in a ground-truth anomaly segment are treated as point-wise.

RPA approach behaves in the same way as PA but for ground-truth anomaly segment: instead of recording p true positives, it records only 1 true positive for the whole window if any observation in a ground-truth anomaly segment is correctly detected. It records 1 false negative if none of the observations in the ground-truth anomaly segment is correctly detected. The observations that are not in a ground-truth anomaly segment are treated as point-wise.

PA$\%K$ approach aims to minimize overestimation errors in the PA approach by using a hyper-parameter K to adjust the anomaly prediction threshold. Specifically, PA$\%K$ considers an anomaly segment to be correctly detected if at least $K\%$ of its observations are accurately detected (i.e. p true positives are recorded if so and p false negatives otherwise.). The observations outside of a ground-truth anomaly segment are treated as point-wise.

Analysing this behavior (and later demonstrated by our experiments in Section 3.4.1), we can first observe that the PA approach is more tolerant to high false positive rates in a large anomaly windows and hence leads to high scores, thereby overestimating the performance of a model. In contrast, the RPA approach gives lower scores due to its differential and unfair evaluation between anomaly windows and normal windows. Classifying a window as anomalous, based on only one observation correctly detected (as done by PA/RPA approach) can result in many false alarms, which can have extra unnecessary costs and makes the detection model unpractical. Moreover, the three aforementioned approaches necessitate having access to accurate ground-truth segments in order to evaluate the model.

Given the limitations and shortcomings of these approaches when comparing various ML models, we therefore propose a Window Anomaly Decision (WAD) approach to accurately evaluate ML models for window-based anomaly detection.

3.3.5 Window Anomaly Decision (WAD) approach

WAD approach is designed to fairly evaluate anomalous and normal windows. Furthermore, compared to the previous approaches, which require a ground-truth to compute their scores, the WAD approach only uses the model output to classify a window as anomalous or normal.

Hence, WAD is not only an evaluation approach, but also a decision approach.

The WAD computes the score for the whole window based on the model output for each observation. If more than a rate α of observations are classified as anomalous, the window itself will be classified as anomalous. Otherwise, the WAD approach will classify the window as normal. Specifically, WAD works as follows: given an anomalous window, a true positive is recorded if the model correctly detects a rate of α anomalous observations, otherwise a false negative is recorded. For a normal window, a true negative is recorded if less than a rate of α anomalous observations is detected, otherwise a false positive is recorded.

The approach is formulated as follows, where $\tilde{w_t}$ denotes an unseen window of observations, $\mathbb{1}_{\tilde{y_i}=1}$ is the characteristic function that equals 1 if $\tilde{y_i} = 1$ and 0 otherwise, and p is the size of the window.

$$WAD(\tilde{w_t}) = \begin{cases} 1, & \text{if } (\sum_{i \in \{1..p\}} \mathbb{1}_{\tilde{y_i}=1}) \geq int(\alpha.p) \\ 0, & \text{otherwise,} \end{cases} \tag{3.3}$$

with

$$\mathbb{1}_{\tilde{y_i}=1} = \begin{cases} 1, & \text{if } \tilde{y_i} = 1 \\ 0, & \text{otherwise,} \end{cases} \tag{3.4}$$

In this manner, WAD gives a higher score to a model that can detect at least a rate α of all the anomalous observations in the window and lower scores to a model that cannot.

We can use the parameter α to adjust the desired accuracy of the model by making WAD more or less tolerant to fault (i.e., error of the ML model or error with the measured metric). If the goal is to evaluate only perfect models, we can configure $\alpha = 1$. A smaller value can be chosen if a larger tolerance is desired. We leave to the domain experts the possibility to adjust the WAD α value according to their domain-specific considerations. In our evaluation study, we consider that a rate $\alpha = 0.8$ is reasonable since under this rate, a window does not contain enough anomalous observations to be considered as anomalous with respect to CG end-users' QoE degradation detection.

In order to prevent the possibility of missing anomalies spread across two consecutive windows, our approach should be used with overlapping consecutive windows as done in this study wherein a stride of 1 was employed.

Following the previous explanations, Fig 3.1 presents an example of how the PA, PA%K, RPA and WAD approaches work given anomalous and normal windows. There are four scenarios (two on anomalous windows and two on normal windows), each with the anomaly score predicted by an unsupervised model for some input windows of $p = 5$ observations, whose ground-truths are depicted. Given the anomaly score and a threshold $\delta = 0.5$, each approach assigns a binary variable either at the observation level (for PW, PA and RPA) or at the window level (WAD and RPA). WAD approach is used with $\alpha = 0.8$ and PA%K with $K = 80$ for fair comparison.

3.3.6 Performance evaluation metrics

The performance of the unsupervised ML algorithms in our experiments is assessed using the F1-score (Eq. 1.4) and MCC (Eq. 1.5), as described in Chapter 1. We exclude the AUC score (Eq. 1.7), which can produce misleading results when dealing with imbalanced datasets [132, 7]. MCC is preferred for a more balanced assessment of the model's ability to predict both anomalous and normal instances.

In AD tasks, the negative class (here the number of correctly detected normal instances) is often considered less relevant, as the primary goal is to detect anomalies (the positive class). However, we experimentally demonstrate that disregarding the negative class while relying solely on the F1-score can result in unreliable conclusions about the model's performance.

3.3.7 Unsupervised anomaly detection models

We present below the descriptions of eight unsupervised AD models that are used in our comparative evaluation. We only focus on reconstruction-based, one-class classification and isolation-based algorithms because they are the most widely studied in the literature, while generative and predictive algorithms suffer from limitations such as higher computational complexity and expensive training due to instability and reproduction issues [121].

The eight unsupervised models evaluated in this chapter have been selected since they are used in many evaluations studies or to benchmark new approaches for various anomaly detection tasks [133, 134, 135].

- **PCA:** Principal Component Analysis is often used as a baseline for AD tasks. It performs a lossy reconstruction using the principal components computed with the Singular Value Decomposition (SVD). We choose a number of principal components that preserve 90% of the variance in the data in our Scikit-Learn implementation.

- **iForest:** Isolation Forest uses *isolation* trees to recursively isolate anomalies. Its performance relies on the number of trees t, the sub-sampling size ϕ, and the assumed amount of contamination in the dataset. We use the default values of these hyper-parameters in the Scikit-Learn library.

- **OC-SVM:** One-Class SVM [61] is a popular and efficient shallow ML model. OC-SVM uses a hyper-parameter, ν, which is an upper bound on the fraction of outliers in the dataset. We use Scikit-Learn implementation of OC-SVM with *rbf* function and $\nu = 0.1$, which is the default parameter used in [61]. Due to the computational complexity of OC-SVM, we process the training inputs with PCA by retaining 70% explained variance.

- **Deep-SVDD:** Deep Support Vector Data Description [76] can be seen as a deep learning implementation of OC-SVM. Deep-SVDD benefits from the efficiency of deep learning on large, high-dimensional data. We use the *soft-boundary objective* function which assumes that the training data may contain a ratio ν of anomalies. We use the PyTorch implementation of Deep-SVDD on Github.[20]

- **AE:** AutoEncoder (AE) is a neural network architecture composed of an encoder, that encodes input data into a low-dimensional space and a decoder that reconstructs input data from the low-dimensional features. We choose an AE with feed-forward network and *Tanh* activation function for our custom implementation in PyTorch.

- **LSTM-VAE:** It combines a neural network designed for time-series, the LSTM, to an autoencoder with bayesian inference for reconstruction of input data [66]. The reconstruction error is used as anomaly score and we provide a custom implementation of LSTM-VAE in PyTorch based on TensorFlow implementation on Github.[21]

[20] `https://github.com/lukasruff/Deep-SVDD-PyTorch`
[21] `https://github.com/paya54/Anomaly_Detect_LSTM_VAE`

- **DAGMM:** Deep Autoencoder Gaussian Mixture Model combines an autoencoder and a gaussian mixture model, where the representation given by the autoencoder is feed to the gaussian model to produce an energy used as anomaly score. We process the training inputs with PCA by retaining 90% explained variance to decorrelate the features for DAGMM and avoid runtime issues. Our implementation is based on the PyTorch implementation on Github.[22]

- **USAD:** UnSupervised Anomaly Detection [64] adversely trains two autoencoders sharing the same encoder under two objectives: (i) reconstruct input data, and (ii) discriminate real data from reconstructed data. Our implementation of USAD is based on the PyTorch implementation on Github.[23]

The aforementioned reconstruction-based algorithms require a threshold that needs to be carefully chosen. We found in our previous work [7] that using the 3σ rule-of-thumb as a thresholding rule may lead to poor scores, due to a threshold too low to detect all the anomalies in the test set (i.e., low recall scores). In this chapter, we use a different strategy: we randomly select 20% of the test set and select the threshold δ that gives the best F1-score on this sample of the test set. This threshold is then kept and used for evaluation on the remaining 80% of the test set. This thresholding strategy often used in AD [62, 64] reports the best performance that the model can achieve.

We do not perform any hyper-parameters tuning in this study. Instead, we adopt the parameter settings documented in the respective papers of each model, as these configurations have been validated across multiple datasets and determined to be the optimal choices. We then train the neural network models using Adam optimizer with a learning rate of 10^{-3} and a batch size of 128 during 100 epochs. Early stopping is applied to avoid overfitting and longer training time, i.e., training is stopped when the validation loss do not decrease during 10 consecutive epochs. For each experiment, the models are evaluated five times to draw reliable conclusions except for OC-SVM which is run once due to its computational complexity. The details on our implementations are available on Github.[24]

3.4 Experimental Evaluation

In this section we first validate the proposed WAD approach by showing that it produces more accurate results compared to the existing approaches. Furthermore, we experimentally demonstrate with synthetic datasets that F1-score presents some limitations. We then perform comparative evaluations of the eight aforementioned models while studying the impact of data contamination c and window size p.

3.4.1 Comparison of WAD with existing approaches

In this section, we show that the WAD approach can yield more accurate performance results compared to existing approaches. We then define a synthetic anomaly detection model that produce a rate β of detection errors on the test set. Specifically, this AD model has an incorrect prediction for $\beta * 100$ observations over a set of 100 observations. We refer to this model as $(1-\beta)$-*perfect detector*, which is perfect when $\beta = 0$ and is always wrong when $\beta = 1$. We select

[22] `https://github.com/danieltan07/dagmm`
[23] `https://github.com/manigalati/usad`
[24] `https://github.com/joelromanky/unsupervised-ml-ad-qoe-deg`

Table 3.3: Comparison of WAD, PA, PA%K and RPA approaches with MCC and F1 score.

Metric	0.05	0.1	0.15	0.2	0.25	0.3	0.5
MCC-Score							
PA	$95.31_{(\pm 0.02)}$	$90.70_{(\pm 0.03)}$	$86.13_{(\pm 0.01)}$	$81.60_{(\pm 0.01)}$	$77.11_{(\pm 0.02)}$	$72.62_{(\pm 0.02)}$	$54.11_{(\pm 0.03)}$
RPA	$87.01_{(\pm 0.06)}$	$76.34_{(\pm 0.07)}$	$67.20_{(\pm 0.04)}$	$59.17_{(\pm 0.08)}$	$51.97_{(\pm 0.09)}$	$43.35_{(\pm 0.09)}$	$21.96_{(\pm 0.07)}$
WAD$_{\alpha=0.8}$	$97.93_{(\pm 0.02)}$	$91.36_{(\pm 0.03)}$	$80.80_{(\pm 0.07)}$	$68.60_{(\pm 0.11)}$	$56.48_{(\pm 0.20)}$	$45.23_{(\pm 0.15)}$	$0.13_{(\pm 0.13)}$
PA%$K_{K=80}$	$93.38_{(\pm 0.03)}$	$81.94_{(\pm 0.02)}$	$65.74_{(\pm 0.07)}$	$47.36_{(\pm 0.10)}$	$28.66_{(\pm 0.16)}$	$10.50_{(\pm 0.11)}$	$-46.37_{(\pm 0.07)}$
WAD$_{\alpha=0.9}$	$90.62_{(\pm 0.07)}$	$74.56_{(\pm 0.08)}$	$59.19_{(\pm 0.08)}$	$46.18_{(\pm 0.05)}$	$35.51_{(\pm 0.06)}$	$26.78_{(\pm 0.09)}$	$0.15_{(\pm 0.06)}$
PA%$K_{K=90}$	$85.61_{(\pm 0.05)}$	$63.81_{(\pm 0.07)}$	$42.11_{(\pm 0.08)}$	$22.09_{(\pm 0.03)}$	$3.96_{(\pm 0.06)}$	$-12.09_{(\pm 0.09)}$	$-51.71_{(\pm 0.06)}$
WAD$_{\alpha=1}$	$64.30_{(\pm 0.08)}$	$44.79_{(\pm 0.11)}$	$32.07_{(\pm 0.09)}$	$23.03_{(\pm 0.05)}$	$16.40_{(\pm 0.09)}$	$11.39_{(\pm 0.08)}$	$0.04_{(\pm 0.19)}$
PA%$K_{K=100}$	$58.97_{(\pm 0.05)}$	$33.37_{(\pm 0.09)}$	$13.37_{(\pm 0.09)}$	$-03.02_{(\pm 0.06)}$	$-16.08_{(\pm 0.10)}$	$-26.28_{(\pm 0.11)}$	$-52.06_{(\pm 0.09)}$
F1-Score							
PA	$98.06_{(\pm 0.01)}$	$96.19_{(\pm 0.01)}$	$94.36_{(\pm 0.01)}$	$92.60_{(\pm 0.01)}$	$90.89_{(\pm 0.02)}$	$89.23_{(\pm 0.02)}$	$82.99_{(\pm 0.02)}$
RPA	$89.29_{(\pm 0.06)}$	$80.31_{(\pm 0.07)}$	$72.63_{(\pm 0.06)}$	$66.02_{(\pm 0.09)}$	$60.28_{(\pm 0.10)}$	$55.23_{(\pm 0.10)}$	$39.88_{(\pm 0.08)}$
WAD$_{\alpha=0.8}$	$99.04_{(\pm 0.01)}$	$95.72_{(\pm 0.02)}$	$89.34_{(\pm 0.05)}$	$79.95_{(\pm 0.09)}$	$68.05_{(\pm 0.19)}$	$54.56_{(\pm 0.17)}$	$10.00_{(\pm 0.07)}$
PA%$K_{K=80}$	$97.31_{(\pm 0.01)}$	$92.44_{(\pm 0.01)}$	$84.81_{(\pm 0.04)}$	$74.63_{(\pm 0.06)}$	$62.60_{(\pm 0.13)}$	$49.79_{(\pm 0.11)}$	$12.15_{(\pm 0.07)}$
WAD$_{\alpha=0.9}$	$95.20_{(\pm 0.04)}$	$84.42_{(\pm 0.06)}$	$70.10_{(\pm 0.08)}$	$54.23_{(\pm 0.06)}$	$38.91_{(\pm 0.08)}$	$25.75_{(\pm 0.12)}$	$2.11_{(\pm 0.04)}$
PA%$K_{K=90}$	$93.72_{(\pm 0.03)}$	$82.14_{(\pm 0.05)}$	$67.79_{(\pm 0.06)}$	$52.78_{(\pm 0.03)}$	$39.04_{(\pm 0.07)}$	$27.74_{(\pm 0.08)}$	$7.48_{(\pm 0.05)}$
WAD$_{\alpha=1}$	$74.89_{(\pm 0.08)}$	$51.60_{(\pm 0.16)}$	$32.85_{(\pm 0.13)}$	$19.40_{(\pm 0.06)}$	$10.69_{(\pm 0.10)}$	$5.49_{(\pm 0.06)}$	$0.19_{(\pm 0.02)}$
PA%$K_{K=100}$	$75.93_{(\pm 0.06)}$	$54.66_{(\pm 0.13)}$	$38.24_{(\pm 0.12)}$	$26.74_{(\pm 0.06)}$	$19.25_{(\pm 0.08)}$	$14.53_{(\pm 0.09)}$	$7.18_{(\pm 0.07)}$

different values for $\beta \in \{0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.5\}$ and compute F1-score and MCC score accordingly. Table 3.3 depicts the F1 and MCC score of the $(1-\beta)$-perfect detector for the WAD approach.

We notice that both MCC and F1-scores decrease for each window approach as the error rate of the synthetic model increases. On the one hand, when β varies from 0.05 to 0.25, PA still gives high scores to the model (from a F1-score of 98% to 90%), while RPA results in low scores (from a F1-score of 89% to 60%). On the other hand, WAD$_{\alpha=0.8}$ and PA%$K_{K=80}$ strike a balance to provide a more accurate reflection of model quality. For instance, MCC score decreases from 98% to 56% for WAD and from 93% to 28% for PA%K. Both approaches increase their penalization as β increases. However, when β reaches 0.5, which corresponds to random predictions, WAD approach reports an MCC score of 0%, while PA%K continues to penalize the model and produces negative MCC scores (worse than a random model). The reason behind this behavior of PA%K is related to its way of adjusting differently for anomalous and normal segments prediction compared to WAD approach, which handles both segment types similarly.

As for the PA%K approach, the score reported with WAD also depends on a tolerance parameter, here α. Increasing α from 0.8 to 1, results in low scores, even lower than those reported with RPA on a near-perfect model. For instance, a model with a low error rate ($\beta = 0.05$) and WAD with tolerance $\alpha = 1$, achieves an F1-score of 74%, while the PA and RPA approaches report a F1-score of 98% and 89%, respectively. We obtain such scores because with a tolerance $\alpha = 1$, only models that can detect all anomalies in the window get high scores.

Takeaways

Unlike PA, PA%K and RPA, the WAD approach offers a more accurate evaluation of model performance by equally adjusting the prediction for anomalous and normal seg-

Table 3.4: Comparison between F1-score and MCC computed on synthetic AD models.

Model	Window approach	Precision	Recall	F1-score	MCC
Zero-rule	**PW**	58.38	100	73.72	0.0
	PA	58.38	100	73.72	0.0
	RPA	19.06	100	32.02	0.0
	WAD$_{\alpha=0.8}$	54.34	100	70.41	0.0
	PA$\%K_{K=80}$	58.38	100	73.72	0.0
	WAD$_{\alpha=1}$	52.57	100	68.92	0.0
	PA$\%K_{K=100}$	58.38	100	73.72	0.0
Random	**PW**	58.37	49.97	53.89	0.0
	PA	73.97	96.20	82.99	54.10
	RPA	28.61	77.77	39.88	21.95
	WAD$_{\alpha=0.8}$	54.32	5.47	9.94	-0.01
	PA$\%K_{K=80}$	19.70	8.74	12.11	-46.40
	WAD$_{\alpha=1}$	52.19	0.10	0.20	-0.03
	PA$\%K_{K=80}$	12.40	5.05	7.18	-52.06

ments and the score obtained by a model is proportional to its ability to accurately detect a sequence of anomalous observations long enough to be severe.

3.4.2 Comparison between F1-score and MCC

We compare in this section the F1-score and MCC metrics on synthetic anomaly detection models. We first consider a *zero-rule* model that always outputs $\tilde{y}_t = 1 \; \forall \; \tilde{x}_t$ (i.e., this model always classifies any input $\tilde{x}_t$ as an anomaly). This model, if deployed, triggers false alarms and may introduce high anomaly mitigation costs. We report the F1-score and the MCC of the *zero-rule* model on our datasets with each window approach strategy in Table 3.4.

The results of Table 3.4 show that F1-score reports higher scores ($\sim 70\%$) for each window approach (except for RPA), while the MCC reports a score of 0% (i.e., worse than a tossing coin classifier) regardless of the approach. High F1-scores are mainly due to perfect recall scores, i.e., the model detects all the anomalies in the datasets. But these F1-scores are misleading as the model identifies many normal instances as anomalous and then outputs a high number of false positives but no true negatives, which are not taken into account the F1-score. Consequently, when evaluating models having a low rate of true negatives, the F1-score metric should be carefully used.

Furthermore, we make the same observations in Table 3.4 with a random guessing model. Indeed, we notice that F1-score is higher than MCC score, particularly for each approach. For instance, F1-score reported with PA (resp. WAD$_{\alpha=0.8}$) is 83% (resp. 9.9%), while the MCC score is 54% (resp. 0%). Since PA is known to overestimate the model performance, we also compute the F1-score and the MCC with the PW approach which are 53% and 0% respectively. The high F1-score obtained with this model regardless of the approach used asserts the observations made with zero-rule model.

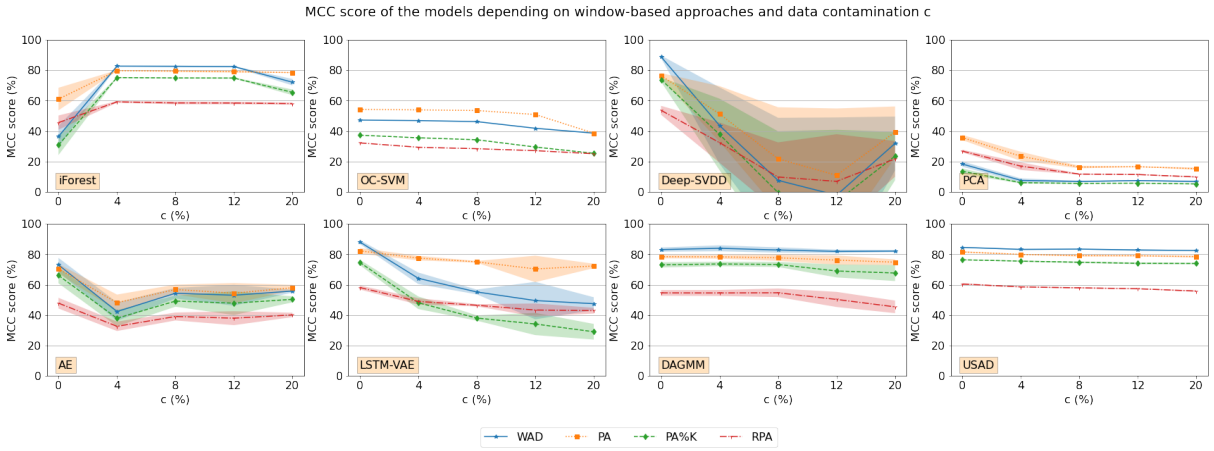


Figure 3.2: MCC score of unsupervised ML models with STD dataset according to the evaluation window-based approaches and the data contamination ratio c.

Takeaways

If considering all conditions (TP, TN, FP, FN) of the models is required, the MCC score should be preferred over the F1-score. The results of this experiment confirm the conclusions drawn by Chicco et al. [86]. In general, we recommend to use the MCC score and F1-score metrics together to avoid unreliable conclusions.

3.4.3 Data contamination impact on unsupervised models

This section presents the evaluation results for the eight unsupervised ML models. We first present the impact of data contamination on the performance of these models while using window-based approaches (WAD, PA, PA%K and RPA). Fig. 3.2 depicts the variation of their MCC scores using data contamination ratio $c \in \{0\%, 4\%, 8\%, 12\%, 20\%\}$ on the STD dataset. We choose these values for data contamination as there are only few anomalies encountered in production networks, and these values are those encountered in previous works [136, 126]. We go up to 20% to stress the models.

As depicted in Fig 3.2, data contamination impacts the ML models differently depending on the model family. Isolation-based models appear to benefit from data contamination. iForest shows low MCC score when there is no data contamination. Its performance largely increases with data contamination $c = 4\%$ and slightly decreases as data contamination ratio increases. This is the expected behavior of iForest which assumes that anomalies are present in the training set but in few quantity and starkly differ from normal instances. The presence of anomalies helps iForest during training but with the increasing anomalies in training set, its performance drops when c reaches 20%. Therefore, the iForest can not efficiently isolate anomalies when there are a lot of anomalies in the training set.

One-class classification models, OC-SVM and Deep-SVDD, have their best performance with no data contamination (MCC_{WAD} of 47.24% and 88.65%, respectively for instance). However, their performance decreases when anomalies are included in the training set. MCC score of OC-SVM slightly decreases and remains at a similar level. Surprisingly, Deep-SVDD perfor-

Table 3.5: Overall performance (mean and standard deviations over 5 runs) according to the training set's data contamination ratio c and using the WAD approaches on the 3 datasets. Bold values indicate the best score for each model.

			STD		GFN		XC	
	Models	c (%)	**F1**	**MCC**	**F1**	**MCC**	**F1**	**MCC**
Isolation-based	**iForest**	**0**	48.68$_{(\pm 9.33)}$	36.45$_{(\pm 6.97)}$	25.65$_{(\pm 2.19)}$	19.20$_{(\pm 1.74)}$	**54.36**$_{(\pm 3.26)}$	**48.60**$_{(\pm 2.82)}$
		4	**90.99**$_{(\pm 0.12)}$	**82.70**$_{(\pm 0.19)}$	49.89$_{(\pm 0.89)}$	37.26$_{(\pm 0.72)}$	49.69$_{(\pm 3.40)}$	44.34$_{(\pm 2.96)}$
		8	90.93$_{(\pm 0.15)}$	82.59$_{(\pm 0.40)}$	**50.15**$_{(\pm 0.85)}$	**37.49**$_{(\pm 0.36)}$	48.66$_{(\pm 3.20)}$	44.34$_{(\pm 2.43)}$
		12	90.82$_{(\pm 0.19)}$	82.39$_{(\pm 0.36)}$	48.58$_{(\pm 0.25)}$	36.42$_{(\pm 0.38)}$	46.01$_{(\pm 2.26)}$	42.37$_{(\pm 2.28)}$
		20	83.77$_{(\pm 1.48)}$	72.13$_{(\pm 2.11)}$	48.00$_{(\pm 0.46)}$	36.16$_{(\pm 0.36)}$	43.39$_{(\pm 1.46)}$	40.98$_{(\pm 1.34)}$
One-class classification	**OC-SVM**	**0**	**77.67**	**47.24**	**69.88**	**29.41**	**66.27**	**51.79**
		4	77.35	46.93	70.37	30.36	65.15	48.51
		8	76.79	46.23	68.68	28.24	64.18	46.60
		12	74.20	41.87	67.02	26.19	63.32	44.98
		20	71.77	38.66	63.47	21.78	62.03	42.65
	Deep-SVDD	**0**	**90.91**$_{(\pm 0.59)}$	**88.85**$_{(\pm 0.83)}$	66.07$_{(\pm 7.34)}$	8.69$_{(\pm 20.27)}$	49.61$_{(\pm 14.19)}$	31.72$_{(\pm 19.7)}$
		4	74.05$_{(\pm 10.99)}$	43.64$_{(\pm 25.11)}$	67.49$_{(\pm 4.78)}$	12.07$_{(\pm 13.27)}$	46.57$_{(\pm 17.96)}$	23.9$_{(\pm 26.11)}$
		8	56.48$_{(\pm 19.1)}$	7.63$_{(\pm 41.11)}$	65.03$_{(\pm 6.16)}$	5.82$_{(\pm 16.13)}$	**62.87**$_{(\pm 13.46)}$	**46.74**$_{(\pm 19.20)}$
		12	53.08$_{(\pm 23.06)}$	-2.13$_{(\pm 51.12)}$	67.44$_{(\pm 6.68)}$	11.69$_{(\pm 19.17)}$	37.07$_{(\pm 7.59)}$	13.64$_{(\pm 9.18)}$
		20	68.82$_{(\pm 7.65)}$	31.9$_{(\pm 17.65)}$	**68.3**$_{(\pm 7.69)}$	**13.34**$_{(\pm 21.11)}$	34.8$_{(\pm 10.89)}$	7.70$_{(\pm 15.14)}$
Reconstruction-based	**PCA**	**0**	**61.79**$_{(\pm 0.90)}$	**18.33**$_{(\pm 1.73)}$	**63.59**$_{(\pm 0.52)}$	**1.89**$_{(\pm 0.81)}$	**36.38**$_{(\pm 0.29)}$	**10.96**$_{(\pm 0.25)}$
		4	58.26$_{(\pm 0.21)}$	7.64$_{(\pm 1.29)}$	63.13$_{(\pm 0.70)}$	-2.40$_{(\pm 0.97)}$	33.32$_{(\pm 0.24)}$	6.63$_{(\pm 0.35)}$
		8	58.80$_{(\pm 0.24)}$	6.87$_{(\pm 0.49)}$	63.56$_{(\pm 0.26)}$	-2.19$_{(\pm 0.52)}$	33.85$_{(\pm 0.66)}$	7.58$_{(\pm 0.79)}$
		12	58.92$_{(\pm 0.13)}$	7.50$_{(\pm 0.25)}$	63.62$_{(\pm 0.28)}$	-1.96$_{(\pm 0.58)}$	34.53$_{(\pm 0.43)}$	8.45$_{(\pm 0.41)}$
		20	58.66$_{(\pm 0.14)}$	6.87$_{(\pm 0.58)}$	63.13$_{(\pm 0.16)}$	-3.25$_{(\pm 0.24)}$	35.43$_{(\pm 0.39)}$	9.42$_{(\pm 0.58)}$
	AE	**0**	**86.73**$_{(\pm 2.04)}$	**73.05**$_{(\pm 4.61)}$	**79.44**$_{(\pm 4.08)}$	**47.17**$_{(\pm 10.72)}$	**76.84**$_{(\pm 6.48)}$	**68.12**$_{(\pm 8.43)}$
		4	74.09$_{(\pm 2.56)}$	42.38$_{(\pm 6.03)}$	65.38$_{(\pm 1.65)}$	10.98$_{(\pm 4.32)}$	49.23$_{(\pm 10.70)}$	30.97$_{(\pm 15.82)}$
		8	79.01$_{(\pm 1.85)}$	54.44$_{(\pm 3.47)}$	65.11$_{(\pm 1.29)}$	10.02$_{(\pm 2.25)}$	40.77$_{(\pm 2.66)}$	18.98$_{(\pm 2.63)}$
		12	78.29$_{(\pm 3.32)}$	53.1$_{(\pm 6.84)}$	64.34$_{(\pm 2.42)}$	8.36$_{(\pm 4.97)}$	45.74$_{(\pm 8.53)}$	25.25$_{(\pm 11.12)}$
		20	79.67$_{(\pm 0.86)}$	55.85$_{(\pm 0.99)}$	64.48$_{(\pm 2.11)}$	6.59$_{(\pm 6.83)}$	42.98$_{(\pm 2.75)}$	21.15$_{(\pm 3.36)}$
	LSTM-VAE	**0**	**94.44**$_{(\pm 0.57)}$	**88.13**$_{(\pm 1.30)}$	**79.30**$_{(\pm 0.63)}$	**50.68**$_{(\pm 1.65)}$	**64.15**$_{(\pm 12.44)}$	**54.95**$_{(\pm 11.11)}$
		4	81.9$_{(\pm 2.02)}$	64.23$_{(\pm 3.78)}$	76.87$_{(\pm 0.45)}$	43.28$_{(\pm 0.84)}$	63.50$_{(\pm 2.31)}$	51.90$_{(\pm 3.41)}$
		8	76.51$_{(\pm 0.93)}$	55.26$_{(\pm 1.56)}$	74.94$_{(\pm 1.23)}$	38.09$_{(\pm 3.25)}$	59.05$_{(\pm 4.17)}$	45.65$_{(\pm 4.99)}$
		12	74.14$_{(\pm 5.12)}$	49.61$_{(\pm 12.3)}$	73.00$_{(\pm 1.58)}$	34.56$_{(\pm 4.25)}$	57.75$_{(\pm 0.53)}$	44.44$_{(\pm 0.65)}$
		20	71.91$_{(\pm 2.62)}$	47.55$_{(\pm 4.23)}$	70.28$_{(\pm 0.86)}$	31.24$_{(\pm 1.88)}$	53.29$_{(\pm 1.82)}$	39.16$_{(\pm 2.85)}$
	DAGMM	**0**	91.23$_{(\pm 0.71)}$	83.11$_{(\pm 1.37)}$	**74.31**$_{(\pm 1.86)}$	**32.08**$_{(\pm 3.64)}$	**68.10**$_{(\pm 2.82)}$	**55.76**$_{(\pm 3.82)}$
		4	**91.84**$_{(\pm 0.77)}$	**83.95**$_{(\pm 1.96)}$	76.72$_{(\pm 1.81)}$	38.08$_{(\pm 3.87)}$	66.16$_{(\pm 1.34)}$	53.18$_{(\pm 2.13)}$
		8	91.50$_{(\pm 0.79)}$	82.83$_{(\pm 1.74)}$	75.09$_{(\pm 3.04)}$	34.47$_{(\pm 7.15)}$	63.02$_{(\pm 2.98)}$	49.17$_{(\pm 3.80)}$
		12	91.33$_{(\pm 0.48)}$	81.92$_{(\pm 1.12)}$	73.86$_{(\pm 2.9)}$	31.32$_{(\pm 7.31)}$	63.04$_{(\pm 2.48)}$	49.14$_{(\pm 3.20)}$
		20	91.43$_{(\pm 0.3)}$	82.12$_{(\pm 0.65)}$	71.66$_{(\pm 3.16)}$	24.80$_{(\pm 6.55)}$	60.10$_{(\pm 3.14)}$	45.54$_{(\pm 4.05)}$
	USAD	**0**	**92.79**$_{(\pm 0.11)}$	**84.57**$_{(\pm 0.20)}$	**76.73**$_{(\pm 0.41)}$	**38.77**$_{(\pm 1.03)}$	**73.27**$_{(\pm 1.52)}$	**62.30**$_{(\pm 1.87)}$
		4	92.09$_{(\pm 0.20)}$	83.23$_{(\pm 0.41)}$	76.64$_{(\pm 0.66)}$	38.04$_{(\pm 1.56)}$	65.74$_{(\pm 1.56)}$	52.60$_{(\pm 2.03)}$
		8	91.78$_{(\pm 0.14)}$	83.44$_{(\pm 0.28)}$	75.25$_{(\pm 0.77)}$	34.31$_{(\pm 2.80)}$	65.16$_{(\pm 3.45)}$	51.77$_{(\pm 4.31)}$
		12	91.63$_{(\pm 0.08)}$	82.82$_{(\pm 0.55)}$	75.06$_{(\pm 0.38)}$	33.69$_{(\pm 1.03)}$	61.24$_{(\pm 4.49)}$	47.97$_{(\pm 4.65)}$
		20	91.36$_{(\pm 0.17)}$	82.48$_{(\pm 0.13)}$	74.64$_{(\pm 1.30)}$	32.23$_{(\pm 2.99)}$	61.06$_{(\pm 5.74)}$	46.38$_{(\pm 7.02)}$

mance (which has the best MCC score among all the models without data contamination), drops significantly and becomes even worse than a random classifier when $c = 12\%$ (i.e.,

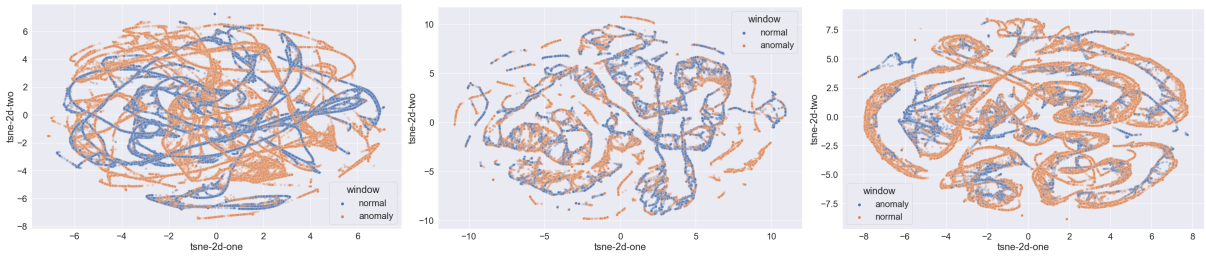
$\text{MCC}_{WAD} = -2\%$) and presents high standard deviations errors (i.e., 51%). Deep-SVDD, although being a neural implementation of OC-SVM with the same objective function, presents high variability in results from one run to another. Moreover, it is difficult to contrast these results on Deep-SVDD with previous works as there is, to the best of our knowledge, either no work on data contamination impact with time-series data using Deep-SVDD or the existing works focus on image datasets.

Reconstruction-based models present two different behavior when facing data contamination. On one hand, we note that PCA, AE and LSTM-VAE are less robust to the presence of anomalies in the training set. PCA, which is the less efficient model for anomaly detection, achieves a MCC_{WAD} score of 18% when $c = 0\%$ which drops to 7% when $c = 4\%$. From this contamination level, increasing the rate of anomalies seems to have no significant impact on the PCA model, as its performance remains at the same level. AE and LSTM-VAE report a MCC_{WAD} score of 73% and 88%, respectively which decreases to 55% and 47%, respectively, when data contamination ratio reaches $c = 20\%$. AE and LSTM-VAE are the most impacted models to data contamination since a contamination ratio of $c = 4\%$ is enough to deteriorate their performance by up to 30%. On the other hand, the impact of data contamination on the performance of DAGMM and USAD models is negligible. USAD presents its highest score at $c = 0\%$ with the lowest standard deviation. The performance of DAGMM model peaks at 83% MCC_{WAD} with $c = 4\%$. Unlike our previous work [7], the performance of reconstruction-based models does not collapse with data contamination. This is the consequence of the thresholding rule used in this work, which is better than 3σ thresholding rule. F1-score reported for AE and USAD in this work are in the same magnitude as those presented in previous works [126, 64]. The only surprising results are those with DAGMM which contradict those in previous works. The DAGMM model in this study is less impacted by data contamination, while in [62, 126, 136] the performance drop is between 10-50% with a data contamination rate of 12%. A reason behind the better performance of DAGMM in this work is the datasets used that present correlated features. DAGMM model use Cholesky matrix factorization to compute the anomaly score. However, this factorization fails with our datasets because matrices computed from our features present negative eigenvalues when Cholesky factorization requires strictly positive eigenvalues. Hence, to run DAGMM model on our datasets, a PCA processing step is required to decorrelate the features.

Table 3.5 presents the F1-score and MCC score obtained using the WAD window approach on the three datasets (STD, GFN and XC). Our study show that the impact of data contamination observed with the STD dataset is consistent across all three datasets. In particular, our analysis reveals the following key findings:

- iForest model benefits from moderate data contamination when applied to the GFN dataset but its performance declines when applied to the XC dataset;
- OC-SVM model's performance decreases with increasing levels of data contamination while Deep-SVDD model's performance exhibits more fluctuation with varying degrees of data contamination when applied to GFN and XC datasets, as well as STD dataset;
- Reconstruction-based models' performance significantly decline as data contamination increases.

However, the models performance with GFN and XC datasets are not as impressive as those achieved with the STD dataset. For instance, LSTM-VAE without data contamination achieves a MCC of 50.68% for GFN and 54.95% for XC. Similarly to the results on STD dataset, iForest,



(a) STD (b) GFN (c) XC

Figure 3.3: Visualization of normal and anomalous windows for the three datasets in a two-dimensional space by t-SNE. Normal windows in blue and anomalous windows in orange.

DAGMM and USAD models remain robust to data contamination in GFN and XC datasets, but their performance levels are lower. To further understand this difference, we use t-SNE projection [137] to visualize in a two-dimensional representation the normal and anomalous windows of each dataset. As depicted in Fig. 3.3, the GFN and XC datasets have numerous overlaps between normal and anomalous windows, while the STD dataset's normal windows cluster in the center of the figure. Consequently, detecting anomalies in the former datasets may be more challenging since anomalous and normal windows are indistinguishable.

Takeaways

Based on our evaluation, we can conclude that while isolation-based models may benefit from data contamination up to a certain moderate level, reconstruction and one-class models show a decrease in their performance when faced with anomalies in the training sets.

3.4.4 Impact of window size

We also analyze the impact of window size on model performance. The window size represents the duration of the anomalous events that can occur during CG sessions. We choose 3 different values for window size ($p \in \{10, 20, 30\}$) to have window length representative of user perceptions (50ms, 100ms and 150ms) and to analyze if the models can efficiently detect short or long anomalous events in CG sessions. Fig. 3.4 depicts the MCC score of each model along with the standard deviations with STD dataset. For each model, we represent the impact of the window size p and the contamination ratio c on its performance.

It is worth noting that increasing the window size seems to slightly improve the performance of iForest, DAGMM and USAD models. For instance, the iForest model shows a +30% increase with $c = 0\%$ when the window size varies from 10 to 30. DAGMM and USAD show a moderate increase in performance (i.e., +4%). Their performance remain better with higher window size regardless of the data contamination ratio. Conversely, OC-SVM, PCA and LSTM-VAE models achieve high performance value with a window size of 10, which decreases as the window size increases. The high variability of Deep-SVDD and the performance variation of AE do not allow to observe a significant pattern. Our findings remain consistent upon studying the impact of window size on both the GFN and XC datasets: increasing the window size may improve very slightly the performance of iForest and USAD but not the other models.

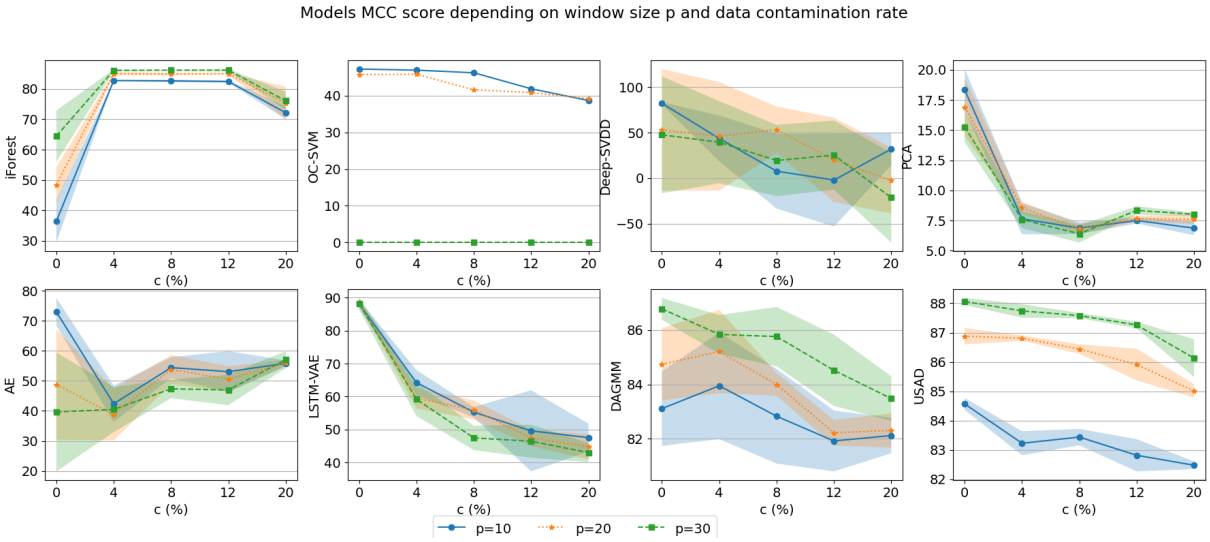


Figure 3.4: MCC score of unsupervised ML models with STD dataset according to data contamination ratio c and window size p.

Apart from the USAD model, to the best of our knowledge, there are no studies on the impact of window size on the AD learning models. Moreover, Audibert et al. [64] showed that increasing the window size has insignificant impact on the performance of USAD, while, in our case, it leads to an increase in model performance. We attribute the performance improvement of iForest and DAGMM (and in the same way the performance degradation of OC-SVM, PCA and AE) to the fact that increasing window sizes leads to the presence of anomalous observations in the training set that may improve (or deteriorate for OC-SVM, PCA and AE) the performance of the models.

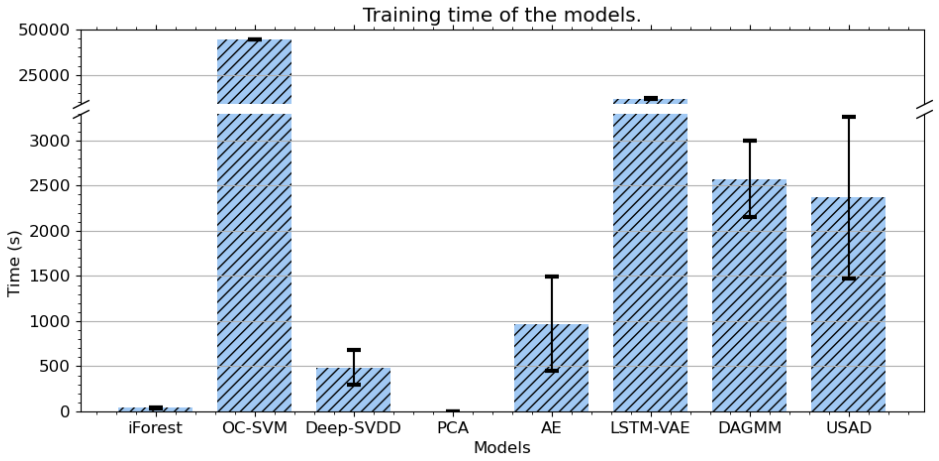
Takeaways

The takeaway from these experiments is that detecting longer anomalous events is less efficient for most of the unsupervised models except for iForest, DAGMM and USAD which are able to better learn anomalies over larger windows.

3.4.5 Computational time performance

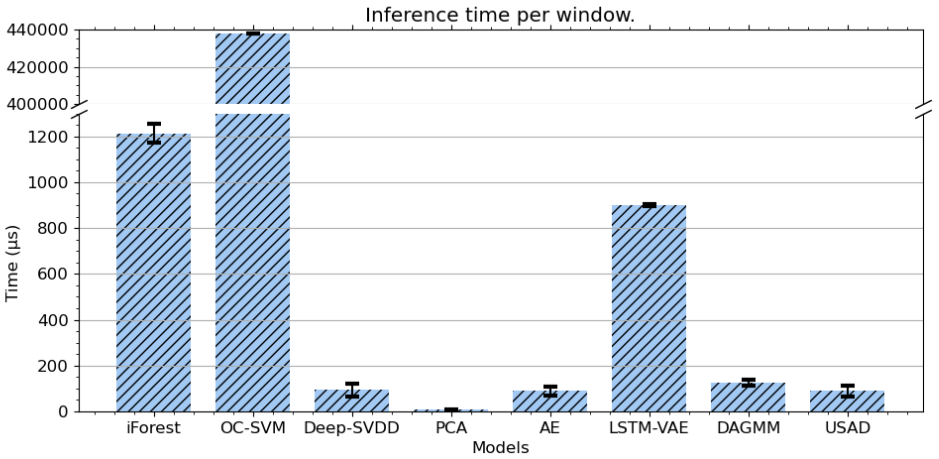
This section analyzes the computational performance of unsupervised ML models. We measure the training time for each model on the training set and compute the average time taken by each model for inference given a window size of 10. The models are trained and tested on a Google Cloud Platform (GCP) VM with 8 CPUs and 30GB RAM. To accelerate the training of LSTM-VAE, a NVIDIA T4 GPU is used. Fig. 3.5 presents the training and inference times.

PCA takes the lowest training time (i.e., 2s) while iForest training time is 20x longer. Neural networks based models present longer training time ranging from 970s for AE to 2500s for DAGMM. Despite the use of GPU, LSTM-VAE takes 4,6 times longer than DAGMM to complete its training. The high variability in their training times is due to early stopping strategy that prevents the models from overfitting. The most time consuming model is OC-SVM that takes



(a)

Training time



(b)

Inference time per window

Figure 3.5: Train and inference time.

44700s for its training. Compared to other shallow models such as iForest, OC-SVM require 1000x longer to train. We make similar observations when comparing the inference time for a window of observations of size $p = 10$. Deep learning based models have inference time between 89 and 126µs (except for LSTM-VAE that goes up to 900µs with GPU). Unlike neural network models, shallow models like iForest and OC-SVM take much more inference time, i.e., iForest takes 1213µs and OC-SVM takes 360x longer.

Takeaways

Algorithms with the lowest training/inference time like PCA or with highest training/inference time like OC-SVM or LSTM-VAE do not provide as good qualitative results as iForest, DAGMM and USAD models. The latter present good performance and robustness with reasonable training and test times.

Model	Performance	Robustness	Deployment	Explainability
iForest	+	++	-	-
OC-SVM	-	+	- -	- -
Deep-SVDD	++	- -	++	- -
PCA	- -	++	++	++
AE	+	-	++	- -
LSTM-VAE	++	- -	-	- -
DAGMM	++	++	+	- -
USAD	++	++	+	- -

++: good; +: somewhat good; -: somewhat bad; - -: bad.

Table 3.6: ML models recommendation

3.5 Discussion

3.5.1 ML Models recommendation

Based on our comparative study, we highlight the advantages and benefits of the models and provide recommendations to network operators according to the following requirements (summarized in Tab 3.6):

Performance: Deep-SVDD and LSTM-VAE are the models with the best performance without data contamination. They achieve near-optimal detection of anomalies in the dataset with low false alarms. However, they present performance degradation when anomalies are present in the training set. The performance instability of Deep-SVDD in the face of anomalies makes it unreliable. The LSTM-VAE model based on LSTM neural networks has high computing cost during training and testing (e.g., requiring more GPUs) incurring high costs in energy consumption in deployment.

Robustness: There is no guarantee that data from production network environments are free of anomalies and removing them for training the models is a difficult task. Therefore, we recommend the use of algorithms such as iForest, DAGMM and USAD in such situations since they are robust to data contamination and will ensure reasonable performance when data contamination is not too high.

Deployment: For deployment recommendations, we consider the training and inference time. All the models used in this work are trained *offline*. Neural networks are trained during several epochs and take a long time for their training phase while PCA and iForest are fast to train. In production, deep learning models like DAGMM and USAD can be used for *real-time* prediction since they have low inference latency. Real-time inference with iForest or LSTM-VAE can be difficult due to their high inference time which also exacerbates scalability issues for the former and resource requirements for the latter.

Explainability: Machine-learning techniques are known to suffer from the lack of transparency and explainability. Network experts in practice need to better understand the decisions taken by ML models to efficiently monitor and manage network systems. None of the algorithms used in this study achieve the trade-off between efficiency and explainability. iForest and neural networks-based algorithms (LSTM-VAE, DAGMM, USAD) are seen as *black-box*

and do not allow for easy interpretation of their anomaly detection.

3.5.2 Limitations

This work presents some limitations worth discussing. First, the thresholding strategy used in this evaluation, which selects the threshold that lead to the best score, is not applicable in practice since it requires the use ground-truth test sets. Further investigations should be carried out to automatically determine and select the best threshold values. Some works [138, 139] advocate for the use of evaluation metrics insensitive to the selected threshold. Second, the chosen criteria to introduce anomalies in the collected datasets may result in introducing too many of them (cf. 3.2) which can raise questions worthy of further investigation. Additionally, as discussed in Section 3.4.3, a PCA processing step was employed to decorrelate the features before utilizing the DAGMM model. It is worth noting that the performance of this model may differ when applied to datasets with correlated or uncorrelated features. Moreover, as mentioned in Section 3.3.7, we could not include in our evaluation study, generative and predictive models since they may suffer from computational constraints.

Furthermore, in our methodology, we used the DECAF tool [16] to collect the QoE/QoS time-series features on the client-side. However, this tool is only compatible with CG platforms based on WebRTC and cannot be used with other CG platforms to collect the time-series features employed in our study. Finally, network operators may not always have the possibility to gather such information at the client-side. Consequently, they may need to develop QoE degradation detection algorithms based on features that can be readily collected at the network edge, such as network packets.

3.6 Conclusion

In this chapter, we performed a comparative evaluation of eight unsupervised ML models applied to the detection of users' QoE degradation in CG sessions. Our evaluation showed that the F1-score metric, which is widely used for model evaluation, has limitations and should be combined with the MCC score for more accurate model evaluation. Moreover, existing window-based approaches, used to cover sequences of events including anomalies, lead to erroneous conclusions regarding model performance. To address these limitations, we proposed the WAD approach to allow for a fair and better assessment. The WAD approach offers the possibility of parameter calibration to detect more or less severe anomalies. As practical considerations in the use of ML models in networking, we also showed that data contamination has a considerable impact on unsupervised ML models, and revealed their disparity with respect to their computational performance. Our study has shown that the use of models such as those included in our evaluation in an industrial context requires further investigation of their applicability and calibration. Highly performing state-of-the art ML models have not been necessarily designed with industrial considerations in mind such as robustness, energy consumption, explainability and likely others. Many of these considerations can be conflicting with commonly used performance evaluation metrics. In summary, this chapter emphasizes the importance of employing a consistent methodology and appropriate metrics when evaluating ML models. In particular, we found no one-size-fits-all solution as some solutions may be preferred to others depending on the requirements of the operational environment under consideration. These insights will serve as the foundation for subsequent chapters, where we further propose novel anomaly detection and root-cause diagnosis methods in the context of low-latency applications.

Contrastive learning for anomaly detection in time series

Summary: *This chapter introduces CATS, a novel approach leveraging temporal similarity measures and anomaly-informed contrastive learning (CL) to address limitations of existing CL frameworks for time series AD. By incorporating negative data augmentation, CATS generates realistic anomaly distributions and enhances representation learning. Experiments on six real-world datasets demonstrate CATS's superior performance, making it a robust solution for time series AD in diverse big data contexts.*

Contents

4.1 Introduction

In various application domains such as networking or cyber-security, large volumes of time series datasets are continuously generated and collected that are valuable for monitoring systems performance. Accurately detecting anomalies within these datasets are essential for identifying networks faults, device/system malfunctions, security breaches or service degradations that can significantly impact operations and user quality of experience (QoE). However, labeling

real-time series data for anomaly detection is a time-consuming and challenging task due to the large amount of available time-series data and the scarcity of labeled anomalies in production systems.

As a result, unsupervised time series anomaly detection (AD) has received significant attention in machine learning research community and several papers have proposed different techniques for this purpose. The proposed AD techniques can be categorized into five groups: reconstruction-based [64, 66, 67, 65], distance-based [55], one-class classification-based [76, 61], isolation-based [60], and generative-based [140, 71]. These techniques aim to learn the patterns of normality exclusively from normal data and detect anomalies based on deviations from the learned normality. The deviations are computed using metrics such as reconstruction error or distance to normal centroids. However, these techniques have some inherent limitations. Firstly, they struggle to discriminate anomalies that are close to normal samples because they do not leverage information about the anomalous class which are not used during the training phase. Secondly, the presence of unknown anomalies in the training sets introduces the challenge of *data contamination*, which can negatively impact the performance of these models.

To enhance the performance of time series AD, contrastive learning (CL) [44] emerges as a promising approach, increasingly applied to various classification and forecasting tasks [141, 142, 143, 144]. CL has gained popularity across multiple disciplines including computer vision and natural language processing (NLP). Recently, its application in time series analysis has attracted attention. The fundamental concept of CL is to learn data representations by contrasting positive views and negative views. Specifically, through data augmentation, input data is transformed into multiple views, which are then contrasted in the latent space. This contrastive process aims to cluster similar views while repelling dissimilar ones. Following the training phase, the encoder is employed for downstream tasks. The effectiveness of contrastive learning stems from its ability to learn transformation-invariant properties through data augmentations.

However, in time series analysis tasks, the exploration of data augmentation techniques has not yet been as extensive as in the field of computer vision. Additionally, some previous works in this field have used CL for anomaly detection. Nevertheless, they did not utilize a similarity function specifically designed for time series data, resulting in an inefficient exploitation of the temporal aspect inherent in multivariate time series data, which are nevertheless crucial for modeling. To address these limitations, we propose in this chapter an innovative end-to-end method called contrastive learning for anomaly detection in time series (CATS). We evaluate this method using large and several time series datasets including cloud gaming and system monitoring applications.

Contributions

Specifically, the main contributions of this chapter can be summarized as follows:

- Using Dynamic Time Warping (DTW) similarity, we propose a novel DTW-based temporal contrastive learning loss to efficiently model multivariate time series .
- We employ negative data augmentations to generate synthetic anomalies to establish a realistic out-of-order distribution that contrasts with normal instances in the training set.
- We conduct extensive empirical experiments on large real-world and popular benchmark time series datasets. Furthermore, we conduct experiments on time

series datasets in the networking domain, collected from cloud gaming platforms. We demonstrate the effectiveness of our proposed framework and its generalization capabilities compared to its counterparts.

- We conduct ablation studies to evaluate the effectiveness of each component of the proposed method, assess the impact of data augmentation, data contamination and the influence of hyperparameters.

4.2 Contrastive learning for Time-Series

Contrastive Learning has emerged as a prominent self-supervised learning technique demonstrating a great potential across various domains, including computer vision and natural language processing. Traditional contrastive learning models [145, 47, 146] construct positive sample pairs i.e. augmented views of the same instance and negative pairs i.e. augmented views of other instances or a dictionary queue to facilitate representation learning with data augmentation techniques. The effectiveness of these approaches relies on key factors such as data augmentation, efficient sampling of negative pairs, and the use of large batch sizes [47]. Recent advancement in contrastive learning architectures such as [147, 148] have shown that meaningful representations can be learned without the explicit need of negative pairs or large batch sizes. These techniques have demonstrated remarkable performance in computer vision domain's downstream tasks.

In the domain of time series analysis, recent techniques have been developed to learn representations from time-series data. [142] introduced a triplet loss and a temporal negative sampling strategy to construct pairs for contrastive learning. [149] leveraged the local smoothness of time series to define neighborhoods and incorporates de-biased contrastive learning. [141] utilized temporal and contextual contrastive learning on different views of time-series data generated with weak and strong data augmentation. Similarly, TS2Vec [143] employed a hierarchical contrastive learning approach, using augmented context views to capture multi-scale information. Information theory-based adaptive data augmentation is proposed by [150] in their InfoTS method for contrastive learning. [151] utilized time domain and frequency domain contrastive losses to learn disentangled seasonal and trend representations for time series forecasting. [144] employed a siamese network without negative pairs for time series forecasting. These aforementioned methods have demonstrated robust performance across various downstream tasks, notably in forecasting and classification.

CL techniques have also been applied to anomaly detection in time-series data. [152] employed deterministic contrastive learning with learnable transformations to generate diverse views. [79] combined negative sample-free contrastive learning with one-class classification to leverage the representation learning capabilities of contrastive learning and the normality assumption of one-class classification for effective anomaly detection. [81] also incorporated one-class classification scheme to contrastive learning, utilizing negative data augmentations. [80] utilized temporal transformations to generate anomalies from normal windows, thereby enhancing contrastive learning. [82] employed AutoML techniques to automatically configure the anomaly detection pipeline and tune hyperparameters of the contrastive learning loss, enabling discrimination between original samples and generated negative samples. [83] proposed a two-stage framework. The first stage leverages time series representation learning by injecting generated anomalies to learn similar representations for temporally close windows. The final

stage classifies window time series based on their nearest and furthest neighbors in the feature space, enhancing anomaly detection.

While some studies suggest that negative pairs are not essential for contrastive learning, we believe that negative pairs, constructed through negative data augmentations, bring some learning knowledge that may enhance AD tasks, unlike their impact on other downstream tasks. In contrast to the aforementioned AD approaches, our approach CATS considers temporal similarity using a temporal loss to better discriminate anomalous time series windows.

4.3 Proposed Method: CATS

In this section, we first formulate the multivariate time series anomaly detection problem. We then present the architecture of CATS and provide a detailed description of each of its components. In particular, we explain the data augmentation techniques employed in our work, introduce and describe the temporal contrastive loss, and the global contrastive loss. Finally, we define the computation of the anomaly score, which is utilized for the detection of time series anomalies.

4.3.1 Problem Formulation

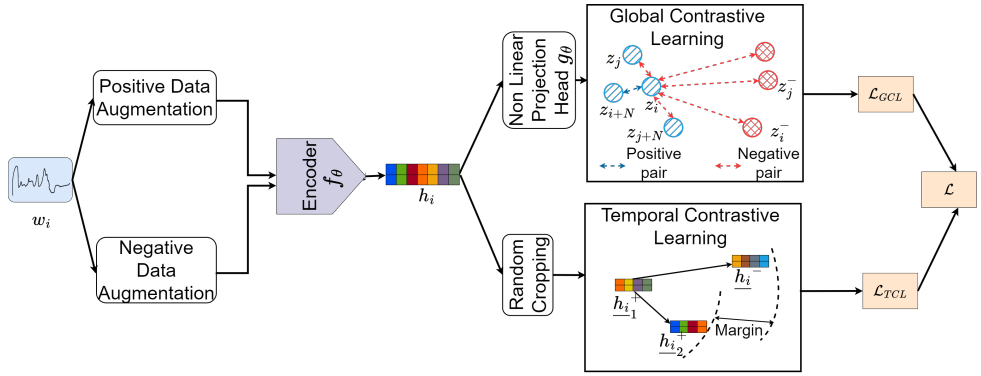
Given a multivariate time series dataset $X = \{x_1, x_2, ..., x_T\}$, where T is the length of X and $x_t \in \mathbb{R}^m$ denotes a m-dimensional vector corresponding to the values of our m features at time t. The dataset is sliced into sequences of time series windows $W = \{w_1, w_2, ..., w_{T-p+1}\}$ with stride 1 where $w_t = \{x_t, x_{t+1}, ..., x_{t+p-1}\} \in \mathbb{R}^{p \times m}$, p being the window size.

Time series anomaly detection aims at training an unsupervised anomaly detection model $\mathcal{M}$ that given an unknown window time series $\tilde{w}_t$ at inference time will output an anomaly score $s(\tilde{w}_t)$. By using this anomaly score and a threshold η, a binary label $\tilde{y}_t \in \{0, 1\}$ is computed which indicates whether a window is anomalous ($\tilde{y}_t = 1$) or not ($\tilde{y}_t = 0$).

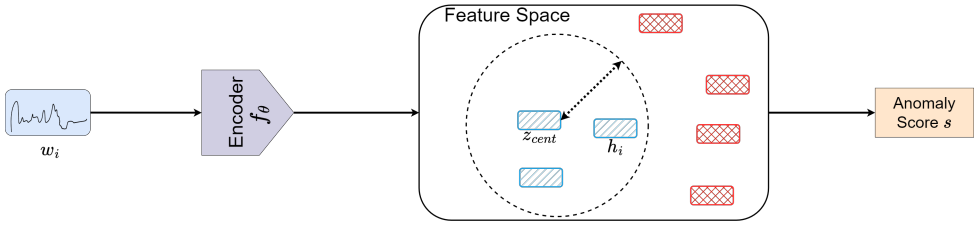
4.3.2 Model Architecture

The overall architecture of CATS is shown in Fig. 4.1. Our architecture comprises the following components:

- A stochastic data augmentation module that transforms each input window time series w_i to three views: two positive views $\{w_i^+; w_{i+N}^+\}$ and a negative view w_i^-.
- A siamese network encoder f_θ that learn representations from augmented views of the input window time series $h_i = f_\theta(w_i) \in \mathbb{R}^{p \times d}$ with d the feature size.
- A projection head g_θ that projects the latent representations h_i to a projection space $z_i = g_\theta(h_i)$ due to its importance for contrastive learning [47].
- A global contrastive loss (GCL) that performs contrastive learning on projection space giving the set of (z_i) from the batch using an improvement of NT-Xent loss [153].
- Random cropping is applied on feature space representations h_i to built temporal pairs for the temporal contrastive learning.
- A temporal contrastive loss (TCL) to enforce temporal similarity between cropped versions ($\underline{h_i} \in \mathbb{R}^{k \times d}$) ($k$ the crop size) of latent vectors and a triplet loss.



(a) Training



(b) Inference

Figure 4.1: The proposed architecture of CATS. (a) is the training phase which consists in time series representation learning with temporal contrastive learning (TCL) and global contrastive learning (GCL). (b) is the inference phase where given an unknown window time series $\tilde{w}_i$ its anomaly score is computed as the distance between its latent representation $\tilde{h}_i$ and the centroid of training samples z_{cent}.

The model is trained using a contrastive loss model combining the temporal and the global contrastive losses defined as follows:

$$\mathcal{L} = \alpha.\mathcal{L}_{TCL} + \beta.\mathcal{L}_{GCL} \tag{4.1}$$

where α and β are two hyper-parameters representing the relative weight of each loss.

At inference time, the projector g_θ is discarded and a anomaly score is obtained by computing the distance between an unseen window and the center of latent representations.

4.3.3 Data augmentation

Data augmentation plays a crucial role in contrastive learning, facilitating the learning process. While diverse data augmentation techniques are available in the computer vision domains, selecting appropriate data augmentation methods for time series remains challenging. Building upon previous works that have utilized or evaluated data augmentation for time series [47, 154], we define a set of positive data augmentations $\mathcal{D}^+_{aug} = \{d^+_1, \ldots, d^+_{n^+_{aug}}\}$ which are as follows, given a window time series $w = (x_1, \ldots, x_p)$:

- Jitter transformation adds a i.i.d Gaussian noise with zero mean and variance σ^2 to the

time series.

$$d_{jitter}(w) = w + \mathcal{N}(0, \sigma^2) \tag{4.2}$$

- Scaling transformation changes the global magnitude of a portion P of the time series by multiplying all the values by a scaling factor α.

$$d_{scaling}(w) = \alpha_{scaling} * w_P \tag{4.3}$$

In the context of unsupervised learning for anomaly detection, the focus is typically on modeling normality using only normal instances during training, and anomalous instances are only encountered during inference. We use negative data augmentations to incorporate weak supervision during the representation learning stage, thus providing prior information about what does not constitute normal data [83, 81, 80, 155]. This approach enhances the learning process by incorporating knowledge of abnormal patterns during training. We thus define a a set of negative data augmentations $\mathcal{D}^-_{aug} = \{d^-_1, \ldots, d_{n^-_{aug}}\}$ which are as follows:

- Mask transformation consists in randomly masking some points of the time series.

$$d_{mask}(w) = (\widehat{x_1}, \ldots, \widehat{x_p}) \text{ with } \widehat{x_i} = 0 \tag{4.4}$$

- Trend transformation applies a linear drift w_{drift} to the time series to simulate a shift in the trend of the time series.

$$d_{trend}(w) = w + w_{drift} \tag{4.5}$$

Thus, given a time series window w_i from a batch $\mathcal{B}$ of size N, we generate two positive views w^+_i and w^+_{i+N} and one negative view w^-_i resulting in a batch of positive samples $\mathcal{B}^+ = (w^+_i)^{2N}_{i=1}$ and a batch of negative samples $\mathcal{B}^- = (w^-_i)^N_{i=1}$.

However, the range of anomalies present in time series data is usually large, and it is not feasible to expect negative data augmentations to generate all possible anomalies. Therefore, similar to previous studies [83, 81, 80], random data augmentation is employed. Specifically for each window w_t, and a selected data augmentation $d^-_i \in \mathcal{D}^-_{aug}$, diversity is enforced through the hyperparameters of d_i that control the augmentation process. These hyperparameters include the ratio of features n_{feat} that will be augmented and the ratio of time points n_t that will be augmented per features. Given n_{feat} (resp. n_t), and a given time series window w_t, a random subset of features (resp. time steps) will be chosen for augmentation. This approach allows for increased variability and diversity in the generated augmented samples, enhancing the learning process for anomaly detection in multivariate time series data.

4.3.4 Temporal Contrastive Learning

One shortcoming of previous anomaly detection approaches using contrastive learning is that they do not exploit temporal dependencies for contrasting. We address this limitation by proposing a novel Temporal Contrastive Loss (TCL). TCL aims at learning temporal properties, and clusters time series window that are temporally similar while pushing away windows that are dissimilar. To achieve that representation in the feature space, we use DTW (Dynamic Time Warping) [156] as a time series similarity measure that is more suited for time series forecasting or clustering than classical Euclidean distance.

DTW similarity measure aims at minimizing the Euclidean distance between aligned time series under all possible temporal alignments. However, DTW measure is not differentiable and

is not suitable for gradient-based algorithms. To overcome this limitation, Soft-DTW [157] was proposed to smooth DTW and make it differentiable everywhere and then can be used as a loss function or similarity measure. In this work, we choose a Soft-DTW variant called Soft-DTW divergence [158] that unlike the former is positive and minimized when the time series are equal. Specifically given two time series x_i and x_j, Soft-DTW divergence is defined as follows:

$$\begin{aligned} D^{\gamma}(x_i, x_j) = & DTW^{\gamma}_{Soft}(x_i, x_j) \\ & - \frac{1}{2}(DTW^{\gamma}_{Soft}(x_i, x_i) + DTW^{\gamma}_{Soft}(x_j, x_j)) \end{aligned} \tag{4.6}$$

with $DTW^{\gamma}_{Soft}(.)$ being the Soft-DTW measure and γ a smoothing parameter.

To enhance the temporal contrastive learning, we build triplets using cropped versions (subsets of time series windows as defined in [143]) of the two positive views and a crop version of the negative view. Specifically, we apply random cropping on a positive view $h_i^+ \in \mathbb{R}^{p \times d}$ to generate two positive subseries $\underline{h_i}_1^+$ and $\underline{h_i}_2^+ \in \mathbb{R}^{k \times m}$ where $k < p$. We do the same process on a negative view h_i^- to obtain a negative subseries $\underline{h_i}^-$. The rationale behind building triplets in this way, instead of using the views resulting from positive and negative data augmentations to build them, is to ensure that the two positive subseries will be *temporally* similar and *temporally* distant from the negative subseries. Since we consider only normal time series windows to build positive pairs, we avoid the limitation of the cropping strategy raised by [143].

Given this similarity measure, and a triplet of latent representations $\{\underline{h_i}_1^+; \underline{h_i}_2^+; \underline{h_i}^-\}$ TCL is defined as follows:

$$\mathcal{L}_{TCL} = \frac{1}{N} \sum_{i=1}^{N} max(D^{\gamma}(\underline{h_i}_1^+ - \underline{h_i}_2^+) - D^{\gamma}(\underline{h_i}_1^+ - \underline{h_i}^-) + m; 0) \tag{4.7}$$

where $D^{\gamma}(.)$ is the Soft-DTW divergence measure, m the margin (minimum distance that must be kept between positive samples and negative samples). One advantage of Soft-DTW divergence measure is that it can be applied to time series of different sizes and then our TCL can be computed using the cropped versions of the latent vectors. However, the computation of Soft-DTW has a quadratic time complexity which can increase the training time.

4.3.5 Global Contrastive Learning

We define a Global Contrastive Loss to learn representations at the instance level. This loss improves the NT-Xent loss by considering more negative pairs for contrastive learning. Traditionally, given an instance z_i in a batch of size N, CL models using this loss contrast one positive pair (z_i, z_i^+) i.e., two views from the same instance to $N-1$ negative pairs (z_i, z_j^+) $\forall j \neq i$ i.e., pairs with views of different instances.

However, in anomaly detection tasks that are performed on normality assumptions, the training data mostly belong to the same class, i.e., normal data. To enhance, the representation learning, we include views coming from negative data augmentation. Specifically, we form a negative pairs by computing the similarity between an instance and all negative views of all the instances in the batch (z_i, z_j^-) $\forall j \in [|1; N|]$. Consequently, we get one positive pair and $2N-1$ negative pairs for each z_i in the batch. Then, the GCL is expressed as follows:

$$\mathcal{L}_{GCL} = \frac{1}{2N} \sum_{i \in \mathcal{B}^+} log \frac{exp(sim(z_i, z_{i+N})/\tau)}{\sum_{j \in \mathcal{B} \text{ and } j \neq i} exp(sim(z_i, z_j)/\tau)} \tag{4.8}$$

Table 4.1: Datasets summary.

	Dataset	Train	Test	Dimensions	Anomalies (%)
Benchmark	SMD	708405	708420	28*38	4.16
	SMAP	135183	427617	55*25	13.13
	MSL	58317	73729	27*55	10.72
Cloud Gaming	STD	80486	169706	14	52.57
	GFN	27415	22667	14	55.36
	XC	83611	17918	14	24.32

with $\mathcal{B} = \mathcal{B}^+ \cup \mathcal{B}^-$, N the batch size, τ the temperature hyperparameter and $sim(.)$ the cosine similarity.

$$sim(z_i, z_j) = \frac{z_i.z_j^T}{\|z_i\|\|z_j\|} \tag{4.9}$$

4.3.6 Anomaly score

After model training, we discard the projection head g_θ and we only use the encoder f_θ for our downstream task since h_i feature have more information for contrastive learning than z_i [47]. To identify anomalies, we follow the same assumptions as one-class classifiers and we expect the normal data to be clustered and anomalies to lies away from that cluster. Hence, given a window from the test set $\tilde{w}_t$, we define the anomaly score $s(.)$ as follows:

$$s(\tilde{w}_t) = \mathcal{D}(f_\theta(\tilde{w}_t), z_{cent}) = \mathcal{D}(\tilde{z}_t, z_{cent}) \tag{4.10}$$

$$z_{cent} = \frac{1}{N_{train}} \sum_{i=1}^{N_{train}} w_i \tag{4.11}$$

where $\mathcal{D}(.)$ is the L2-distance measure function, z_{cent} is the centroid of latent features of the training set and N_{train} is the size of the training set.

One advantage of using Eq. 4.10 as anomaly score is that z_{cent} can be computed offline and stored allowing lower inference time.

4.4 Experiments

This section begins by outlining the experimental setup, including the datasets and anomaly detection (AD) models used for comparison. The experiments focus on evaluating CATS's accuracy in anomaly detection against other models across different datasets. Additionally, we conduct an ablation study on CATS's components, examine its robustness to data contamination, and analyze the effects of hyper-parameters.

4.4.1 Dataset description

We evaluate the model on the cloud gaming datasets collected in Chapter 2 over three cloud gaming platforms: i) STD (Stadia from Google), ii) GFN (GeForceNow from NVIDIA) and iii) XC (Xbox Cloud from Microsoft). These datasets consist of QoE and QoS time series.

We also evaluate the performance of our technique on three public anomaly detection benchmark datasets to confirm that our approach maintains its effectiveness when applied to them. The selected datasets are listed as follows: iv) SMD dataset (Server Machine Dataset) which consists of 38 sensors continuously monitored during 10 days collected on 28 servers. v) MSL dataset (Mars Science Laboratory) and vi) SMAP (Soil Moisture Active Passive) are two datasets collected from a monitoring system. The benchmark datasets (SMD, MSL and SMAP) are multi-entity datasets (contain different subdatasets).

4.4.2 Benchmark AD models

We compare our models against state-of-the-art algorithms or traditional algorithms mostly used in previous studies for anomaly detection in time series data.

- **Shallow machine learning algorithms**: we use Isolation Forest (IF) [60] that isolate anomalies using features values.
- **Unsupervised time-series anomalies detection**: We select i) Deep-SVDD [76], ii) Auto-Encoder (AE), iii) UnSupervised Anomaly Detection (USAD) [64] that reconstruct normal data and use reconstruction error to detect anomalies.
- **Contrastive learning algorithms**: We use the following contrastive learning architectures from i) SimCLR [47], ii) SimSiam [147], iii) TS2Vec [143].

4.4.3 Implementation details

We choose as encoder f_θ, the same encoder architecture as TS2Vec, which consists of a dilated CNN module with ten residual blocks of 1D convolutional layers. The projection head g_θ is a three-layer MLP with ReLU activation. The embedding size and projection size are 100 and 50 for use-case datasets and 128 and 128 for benchmark datasets respectively. CATS is trained for 100 epochs with a batch size of 512, using Adam optimizer with a learning rate of 10^{-3}, a weight decay of 10^{-5} and the learning rate is decayed using cosine decay schedule. GCL loss temperature parameter τ is set to 0.1, TCL margin to 5 and TCL smooth parameter γ to 1 in all experiments. We use $\alpha = \beta = 0.5$. The positive data augmentations are *jitter* and *scaling* and the negative data augmentations used are *trend* and *mask*.

To allow a fair comparison with contrastive learning benchmark methods, we adopt the same encoder to learn representations, the same positive data augmentation and the same anomaly score procedure for all benchmarks (cf. in Section 4.3.6). All deep learning architectures are trained using the same aforementioned hyperparameters.

All experiments are performed on a workstation with the following specifications: Ubuntu 22.04, Intel(R) Xeon(R) W-2235 CPU @ 3.8GHz with 32 GB of RAM, NVIDIA GeForce RTX 3090 Ti with 24GB, Python 3.10.6, PyTorch 2.2.0 and CUDA 12.1. The datasets and the code to reproduce all the experiments are provided.[25]

[25] `https://github.com/joelromanky/cats`

Table 4.2: Performance comparison on the datasets. Mean and standard deviation computed over all entities for benchmark datasets and over five runs for case-study datasets. Bold values indicate best results and underlined values the second best results.

	Models	IForest	Deep-SVDD	AE	USAD	SimCLR	SimSiam	TS2Vec	CATS
STD	**AUPR**	75.16$_{(\pm 1.28)}$	95.24$_{(\pm 0.59)}$	97.57$_{(\pm 0.16)}$	97.55$_{(\pm 0.04)}$	97.46$_{(\pm 0.21)}$	79.79$_{(\pm 8.14)}$	97.65$_{(\pm 0.99)}$	**98.72**$_{(\pm \mathbf{0.09})}$
	AUC	74.57$_{(\pm 1.63)}$	91.19$_{(\pm 1.08)}$	96.04$_{(\pm 0.27)}$	96.09$_{(\pm 0.08)}$	95.78$_{(\pm 0.39)}$	75.65$_{(\pm 11.3)}$	95.63$_{(\pm 1.94)}$	**97.93**$_{(\pm \mathbf{0.13})}$
	F1	75.79$_{(\pm 1.42)}$	87.18$_{(\pm 1.24)}$	90.35$_{(\pm 0.51)}$	90.02$_{(\pm 0.24)}$	90.15$_{(\pm 0.52)}$	74.21$_{(\pm 9.22)}$	92.83$_{(\pm 1.92)}$	**94.06**$_{(\pm \mathbf{0.45})}$
	MCC	39.56$_{(\pm 3.66)}$	71.83$_{(\pm 2.77)}$	78.93$_{(\pm 1.14)}$	77.89$_{(\pm 0.36)}$	78.48$_{(\pm 1.17)}$	39.31$_{(\pm 19.8)}$	84.33$_{(\pm 4.12)}$	**86.72**$_{(\pm \mathbf{0.88})}$
GFN	**AUPR**	76.97$_{(\pm 0.58)}$	87.81$_{(\pm 1.51)}$	88.60$_{(\pm 0.40)}$	88.65$_{(\pm 0.14)}$	90.19$_{(\pm 0.65)}$	84.49$_{(\pm 3.33)}$	89.63$_{(\pm 1.99)}$	**93.60**$_{(\pm \mathbf{0.52})}$
	AUC	61.97$_{(\pm 0.87)}$	71.78$_{(\pm 3.41)}$	74.05$_{(\pm 0.84)}$	74.84$_{(\pm 0.42)}$	78.50$_{(\pm 1.95)}$	67.07$_{(\pm 3.25)}$	74.91$_{(\pm 4.32)}$	**84.35**$_{(\pm \mathbf{1.23})}$
	F1	74.12$_{(\pm 0.71)}$	75.51$_{(\pm 2.11)}$	74.05$_{(\pm 0.84)}$	77.80$_{(\pm 0.38)}$	81.20$_{(\pm 2.61)}$	74.25$_{(\pm 2.93)}$	76.76$_{(\pm 2.71)}$	**82.88**$_{(\pm \mathbf{0.96})}$
	MCC	17.07$_{(\pm 1.27)}$	24.26$_{(\pm 6.56)}$	28.08$_{(\pm 0.14)}$	31.40$_{(\pm 1.22)}$	37.46$_{(\pm 3.87)}$	17.86$_{(\pm 6.27)}$	28.19$_{(\pm 8.39)}$	**47.27**$_{(\pm \mathbf{1.49})}$
XC	**AUPR**	67.19$_{(\pm 1.61)}$	61.99$_{(\pm 7.61)}$	84.21$_{(\pm 3.24)}$	83.34$_{(\pm 0.31)}$	80.55$_{(\pm 2.93)}$	76.44$_{(\pm 13.4)}$	**95.01**$_{(\pm \mathbf{1.75})}$	93.65$_{(\pm 0.41)}$
	AUC	78.71$_{(\pm 1.13)}$	67.32$_{(\pm 6.52)}$	89.18$_{(\pm 2.31)}$	89.97$_{(\pm 0.26)}$	85.81$_{(\pm 3.17)}$	83.35$_{(\pm 10.6)}$	**96.96**$_{(\pm \mathbf{1.36})}$	96.10$_{(\pm 0.41)}$
	F1	63.33$_{(\pm 1.18)}$	50.83$_{(\pm 7.69)}$	75.94$_{(\pm 3.30)}$	77.59$_{(\pm 0.58)}$	70.58$_{(\pm 3.45)}$	69.09$_{(\pm 13.4)}$	**89.60**$_{(\pm \mathbf{2.03})}$	86.69$_{(\pm 0.83)}$
	MCC	43.42$_{(\pm 2.43)}$	27.40$_{(\pm 11.4)}$	63.95$_{(\pm 4.63)}$	65.35$_{(\pm 0.72)}$	56.59$_{(\pm 4.89)}$	52.30$_{(\pm 21.3)}$	**84.07**$_{(\pm \mathbf{2.94})}$	79.67$_{(\pm 0.11)}$
SMD	**AUPR**	27.96$_{(\pm 21.7)}$	30.34$_{(\pm 21.6)}$	42.55$_{(\pm 26.5)}$	44.14$_{(\pm 26.5)}$	43.97$_{(\pm 26.4)}$	38.47$_{(\pm 25.9)}$	38.75$_{(\pm 26.4)}$	**46.89**$_{(\pm \mathbf{26.1})}$
	AUC	77.10$_{(\pm 11.9)}$	75.31$_{(\pm 14.5)}$	81.33$_{(\pm 13.2)}$	81.08$_{(\pm 12.5)}$	80.83$_{(\pm 14.7)}$	77.26$_{(\pm 14.9)}$	74.25$_{(\pm 16.6)}$	**82.21**$_{(\pm \mathbf{14.3})}$
	F1	29.88$_{(\pm 20.6)}$	34.75$_{(\pm 21.5)}$	46.00$_{(\pm 24.3)}$	46.62$_{(\pm 26.3)}$	46.51$_{(\pm 25.7)}$	41.82$_{(\pm 25.3)}$	43.18$_{(\pm 25.9)}$	**50.65**$_{(\pm \mathbf{23.6})}$
	MCC	29.62$_{(\pm 20.8)}$	36.25$_{(\pm 22.0)}$	47.00$_{(\pm 24.1)}$	47.98$_{(\pm 25.1)}$	48.06$_{(\pm 24.2)}$	43.24$_{(\pm 24.9)}$	44.95$_{(\pm 24.7)}$	**50.85**$_{(\pm \mathbf{23.6})}$
MSL	**AUPR**	19.17$_{(\pm 19.9)}$	24.88$_{(\pm 24.3)}$	22.02$_{(\pm 22.0)}$	21.62$_{(\pm 21.7)}$	22.12$_{(\pm 22.8)}$	20.94$_{(\pm 22.3)}$	24.93$_{(\pm 23.0)}$	**25.44**$_{(\pm \mathbf{23.7})}$
	AUC	56.94$_{(\pm 14.1)}$	61.38$_{(\pm 17.1)}$	62.30$_{(\pm 16.1)}$	63.31$_{(\pm 14.3)}$	61.09$_{(\pm 15.4)}$	62.07$_{(\pm 14.3)}$	63.95$_{(\pm 15.0)}$	**64.98**$_{(\pm \mathbf{15.7})}$
	F1	21.24$_{(\pm 21.4)}$	27.93$_{(\pm 25.6)}$	26.02$_{(\pm 22.9)}$	27.16$_{(\pm 23.0)}$	25.72$_{(\pm 23.1)}$	23.78$_{(\pm 23.2)}$	28.43$_{(\pm 24.5)}$	**29.15**$_{(\pm \mathbf{24.2})}$
	MCC	11.09$_{(\pm 21.8)}$	19.24$_{(\pm 29.2)}$	16.49$_{(\pm 24.4)}$	17.33$_{(\pm 24.8)}$	16.30$_{(\pm 25.1)}$	14.11$_{(\pm 24.0)}$	19.86$_{(\pm 24.8)}$	**20.14**$_{(\pm \mathbf{27.8})}$
SMAP	**AUPR**	18.80$_{(\pm 22.4)}$	26.22$_{(\pm 32.0)}$	26.55$_{(\pm 30.8)}$	**26.80**$_{(\pm \mathbf{30.6})}$	25.66$_{(\pm 29.0)}$	25.10$_{(\pm 29.5)}$	26.62$_{(\pm 30.8)}$	26.40$_{(\pm 29.8)}$
	AUC	56.98$_{(\pm 17.3)}$	62.52$_{(\pm 19.1)}$	**64.30**$_{(\pm \mathbf{19.6})}$	61.11$_{(\pm 19.4)}$	63.99$_{(\pm 17.7)}$	62.12$_{(\pm 17.1)}$	61.42$_{(\pm 20.3)}$	64.07$_{(\pm 18.6)}$
	F1	22.80$_{(\pm 27.2)}$	29.20$_{(\pm 33.0)}$	28.93$_{(\pm 33.5)}$	**30.10**$_{(\pm \mathbf{33.1})}$	28.23$_{(\pm 32.2)}$	27.46$_{(\pm 33.2)}$	28.26$_{(\pm 33.2)}$	29.07$_{(\pm 29.07)}$
	MCC	11.38$_{(\pm 29.0)}$	23.93$_{(\pm 33.1)}$	23.96$_{(\pm 34.0)}$	23.66$_{(\pm 34.9)}$	22.52$_{(\pm 32.2)}$	21.44$_{(\pm 32.5)}$	23.5$_{(\pm 32.8)}$	**24.28**$_{(\pm \mathbf{32.7})}$

4.4.4 Performance comparison

Table 4.2 summarizes the anomaly detection performance of CATS in comparison to other methods, using the AUPR (Eq. 1.8), AUC (Eq. 1.7), F1 (Eq. 1.4), and MCC (Eq. 1.5) metrics for evaluation. Although AUC have limitations in the context of imbalanced datasets, it is included to facilitate comparison with prior studies. We have chosen not to employ the Point Adjust (PA) method in our evaluation, despite its widespread use in several time series anomaly detection studies, as it has been shown to overestimate model performance [83, 81, 8]. Additionally, the WAD window-based approach introduced in Chapter 3 is not used here in order to maintain a fair comparison with previous works. These results are reported in Appendix B.

Cloud Gaming KPIs datasets

The reported outcomes on the CG datasets represent the average of five independent runs, ensuring fair comparisons across methods using the same evaluation metrics as previously mentioned. CATS achieves the best performance on all datasets, except for the XC dataset, where it ranks second.

Benchmark datasets

Since the benchmark datasets consist of multiple sub-datasets, we report the average performance of each model across all sub-datasets. On average, CATS demonstrates superior performance on SMD and MSL, and ranks second-best on the SMAP. The results in this study are reported using the standard F1-score instead of the more commonly used F1-PA metric, which has a tendency to inflate performance estimates [83, 81, 8]. Despite yielding lower values, the standard F1-score offers a more reliable and consistent basis for comparison across datasets. Consequently, our reported scores are below those documented in previous works on time series anomaly detection which used F1-PA metric. However, they are consistent with those of other studies that have used the standard F1-score [83]. Moreover, the benchmark datasets contain sub-datasets with varying levels of anomaly detection difficulty (e.g., models report a standard F1-score of 0 on some sub-datasets, while achieving scores as high as 90% on others). Consequently, averaging results across all sub-datasets leads to overall scores with large standard deviations.

The experimental results show that our model performs on average better than the other methods on all datasets. It also suggests that traditional contrastive learning only is not sufficient for AD since unsupervised models like AE and USAD perform on average better than SimCLR or SimSiam. However, by considering temporal dependencies and introducing knowledge of the anomaly class, the AD performance is enhanced as shown through CATS results.

Takeaways

CATS achieves superior performance across benchmark and CG KPIs datasets, outperforming other methods on most datasets and ranking second on SMAP and XC. By integrating temporal dependencies and anomaly class knowledge, CATS surpasses traditional contrastive learning and unsupervised ML baselines.

4.4.5 Ablation studies

We conduct two ablation studies. First, we assess the effectiveness of the two loss components $\mathcal{L}_{TCL}$ and $\mathcal{L}_{GCL}$. The results are shown in Table 4.3. We compare our $\mathcal{L}_{GCL}$ to NT-Xent loss $\mathcal{L}_{NTXent}$ that performs contrastive learning without using negative augmentations and also compare the performance by removing random cropping $\mathcal{L}_{w/o-crop}$

The results show that each loss individually achieves lower score than the combined loss in CATS. On the one hand, GCL outperforms the NT-Xent loss on AD tasks as indicated by the MCC score which fairly reflects the model's ability to distinguish well between normal instances and anomalous instances. On the other hand, TCL individually achieves slightly lower performance than GCL, but when combined with GCL contributes to an overall performance enhancement. This disparity may be attributed to TCL having only one positive pair and one negative pair for each window time series, whereas GCL incorporates one positive pair and $2N - 1$ negative pairs. Using multiple negative pairs to enhance the discriminative capacity of TCL comes at the expense of longer training times ($\mathcal{O}(n^4)$) due to the quadratic time complexity of the Soft-DTW divergence. Thus, we opt for using only one negative pair to maintain reasonable training times. Moreover, the results show that removing the random cropping stage negatively impacts the AD performance.

Furthermore, we evaluate the impact of different negative data augmentation strategies. The results in Fig. 4.2 show that there is no one-size-fits-all data augmentation solution. In

Table 4.3: Ablation study on loss components.

	GFN		XC	
Loss	**F1**	**MCC**	**F1**	**MCC**
$\mathcal{L}_{NTXent}$	$81.20_{(\pm 2.61)}$	$37.46_{(\pm 3.87)}$	$70.58_{(\pm 3.45)}$	$56.59_{(\pm 4.89)}$
$\mathcal{L}_{GCL}$	$82.52_{(\pm 1.77)}$	$40.73_{(\pm 2.32)}$	$85.68_{(\pm 2.52)}$	$78.31_{(\pm 3.67)}$
$\mathcal{L}_{TCL}$	$79.93_{(\pm 2.69)}$	$38.12_{(\pm 8.32)}$	$76.57_{(\pm 5.68)}$	$65.71_{(\pm 8.34)}$
$\mathcal{L}_{GCL} + \mathcal{L}_{TCL}$	$\mathbf{82.88}_{(\pm \mathbf{0.96})}$	$\mathbf{47.27}_{(\pm \mathbf{1.49})}$	$\mathbf{86.69}_{(\pm \mathbf{0.83})}$	$\mathbf{79.67}_{(\pm \mathbf{0.11})}$

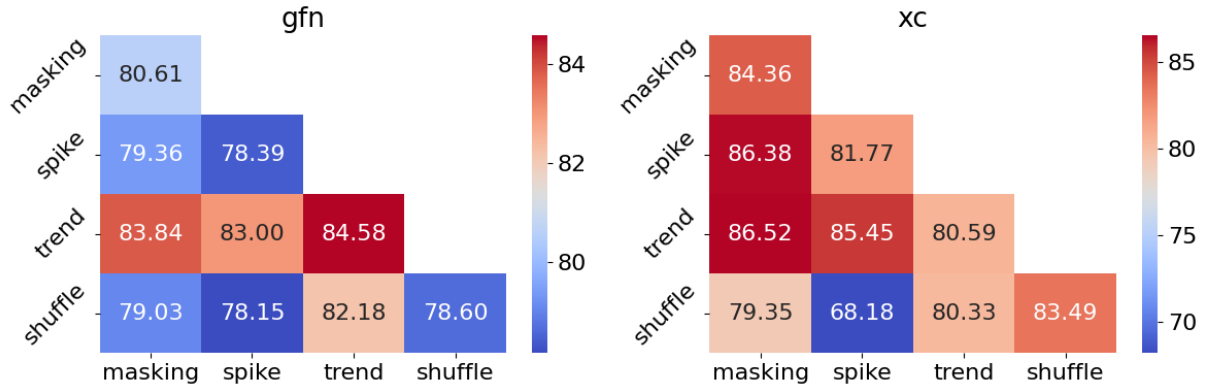


Figure 4.2: Ablation of negative data augmentations using F1-score for GFN and XC datasets.

fact, d_{trend} yields the highest scores with GFN dataset while for XC dataset the best results are achieved by combining d_{trend} with d_{mask}. To optimize the performance of CATS, we recommend fine-tuning the selection of negative data augmentation techniques tailored to each specific dataset.

Takeaways

The ablation studies demonstrate that the combined loss $\mathcal{L}$ significantly improves CATS' performance, with each component playing a distinct role: GCL surpasses NT-Xent due to its use of multiple negative pairs, while TCL provides additional enhancements when integrated with GCL. Random cropping is a critical component of the framework, and fine-tuning data augmentation strategies for specific datasets further optimizes performance.

4.4.6 Data contamination impact

To assess the robustness of our model to data contamination, we introduce various levels of anomalies, denoted as $c \in \{0, 4, 8, 12, 20\}$, into the training set. These contamination rates are chosen to reflect realistic scenarios. Fig. 4.3 illustrates the F1-score of CATS in comparison to other models on the GFN dataset, considering different levels of data contamination.

Among the compared models, with the exception of the iForest model, the performance of all models deteriorates as the data contamination rate increases. Although CATS is slightly

more affected than some models, it consistently demonstrates the highest performance across all contamination levels when compared to the other models even with high contamination rates of up to 20%.

We attribute CATS's behavior to the use of negative data augmentation during the learning process, which helps the model recognize anomalies. However, the extent of this improvement is limited due to the use of synthetic anomalies. Further experiments should be conducted in future work to explore the potential enhancements in robustness that can be achieved by incorporating more data-specific synthetic anomalies.

Takeaways

The robustness evaluation shows that while all models (except iForest) experience performance degradation as data contamination increases, CATS consistently outperforms its counterparts across all contamination levels, maintaining the highest F1-scores even at a contamination rate of 20%. This robustness of CATS is attributed to the use of negative data augmentation.

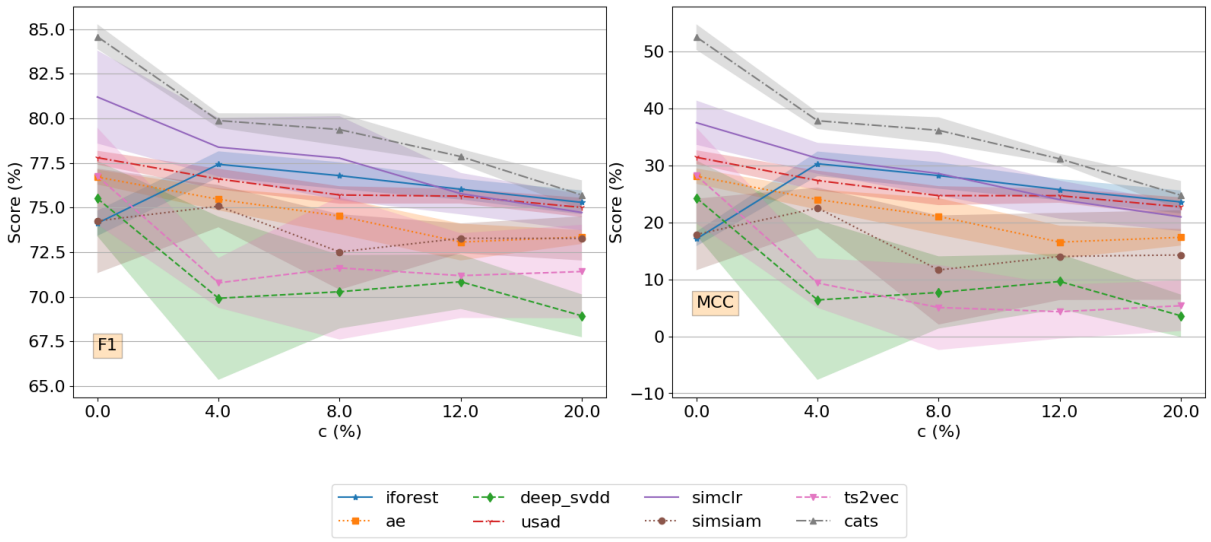


Figure 4.3: CATS robustness to data contamination.

4.4.7 Hyperparameters sensitivity

We study here how some CATS hyperparameters, namely the temperature τ, Soft-DTW smoothing parameter γ, the margin m, the embedding size, the projection size and the batch size may impact the performance. We illustrate that with Fig. 4.4 on GFN dataset. Our experimental results reveal that the temperature parameter τ and the margin m are the most influential hyperparameters. Lower τ values enable the model to enhance its learning by focusing on hard-negatives i.e., negative samples that are closer to the positive samples [47]. Conversely, the results show that larger margins lead to suboptimal performance. This is because larger margins penalize negatives that are not distant enough from the anchors and the positives.

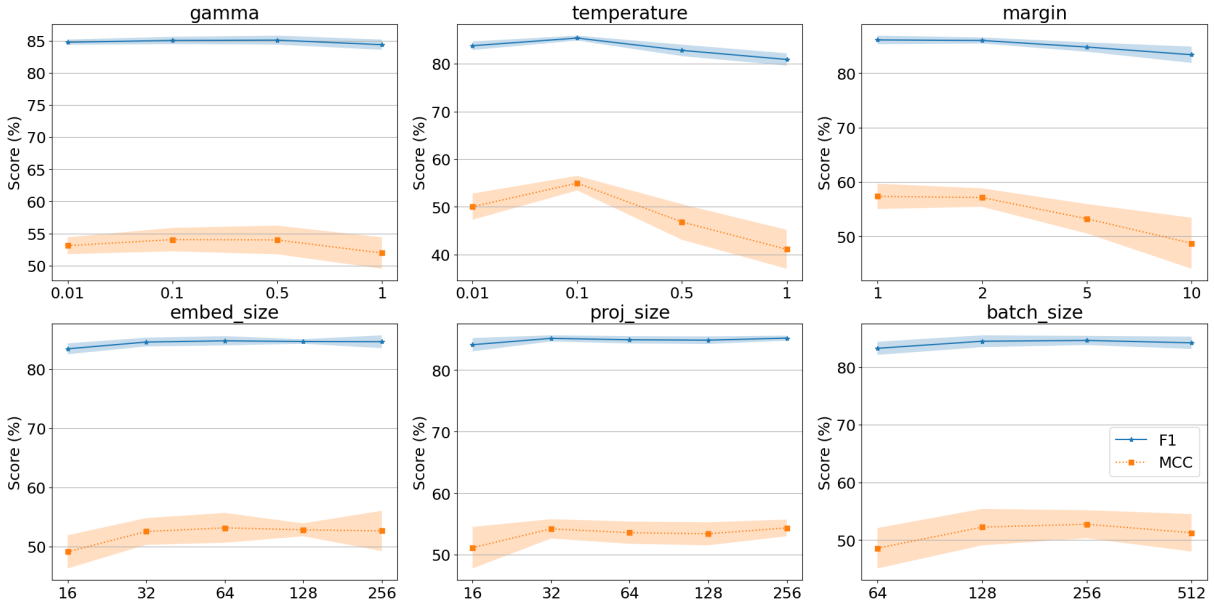


Figure 4.4: CATS sensitivity evaluation to its hyperparameters using F1-score and MCC metrics.

Furthermore, our findings indicate that lower embedding sizes, projection sizes and batch sizes reduce the efficacy of CATS model to learn a good representation for AD. These results highlight the importance of the hyperparameters selection for CATS model.

Takeaways

The hyperparameter sensitivity analysis reveals that the temperature parameter τ and margin m have the most significant impact on CATS' performance.

4.5 Conclusion

This chapter introduces CATS, a novel end-to-end contrastive learning framework for anomaly detection in time series data. CATS addresses the limitations of traditional contrastive learning methods through two key components: temporal similarity and negative data augmentation. By incorporating negative data augmentation, synthetic anomalies are generated, significantly enhancing the model's capability to effectively detect anomalies. Additionally, CATS introduces a novel DTW-based temporal loss, enabling efficient time series representation learning by capturing the temporal patterns inherent in time series data. Empirical evaluations conducted on benchmark datasets and use-case datasets demonstrate the significant improvements achieved by CATS in anomaly detection compared to competing unsupervised models. Importantly, CATS demonstrates superior performance even in challenging scenarios involving data contamination, showcasing its practical applicability to real-world datasets.

In summary, this chapter highlights the significant advancements offered by CATS in the domain of anomaly detection in time series data, establishing it as a versatile and effective framework for addressing diverse real-world challenges.

Root Cause Diagnosis of Cloud VR applications over Wi-Fi networks

Summary: *This chapter presents RAID, a novel approach for identifying network issues in Cloud VR sessions over Wi-Fi networks. It introduces a two-stage framework that combines contrastive anomaly detection with supervised classification to detect and diagnose performance impairments. The chapter used network time series KPIs collected on Wi-Fi testbed presented in Chapter 2 for evaluating the proposed model. Extensive empirical experiments compare the solution against state-of-the-art time series classification models, demonstrating its effectiveness, even in low-label scenarios. Additionally, the chapter highlights the model's suitability for real-time deployment due to its balance between training efficiency and inference speed.*

Contents

5.1 Introduction

The recent evolution of network technologies has led to the rise of low-latency applications such as cloud gaming, telemedicine, cloud robotics, and cloud virtual reality (VR). Cloud VR eliminates the need for powerful local processing hardware by offloading computation to remote servers, enabling the use of lightweight and cost-effective VR headsets. This architecture opens

avenues for diverse innovations, including immersive gaming experiences, enhanced virtual collaboration, and advanced training simulations across various domains. Cloud VR has particularly gained significant importance in recent years [3] due to its ability to deliver immersive, high-resolution VR content to end-users. Cloud VR applications, however, come with stringent technical requirements. They typically require high-resolution content streaming (up to 4K-8K) with high bandwidth demands (≥ 80 Mbps) and ultra-low latency (≤ 20 ms) to ensure a seamless and interactive user experience. Meeting these requirements is essential to avoid issues like lag, reduced interactivity, and user discomfort, which can culminate in cybersickness [29].

These demanding requirements pose significant challenges for Cloud VR applications when delivered over wireless networks, particularly Wi-Fi networks. Although recent Wi-Fi standards (e.g., Wi-Fi 6) enable higher bandwidth and lower latency under ideal line-of-sight conditions, real-world performance is hindered by several limitations. These include bandwidth variability, jitter, additional latency, and packet loss, which are exacerbated by environmental factors such as 1) signal attenuation that occurs due to obstacles that absorb the Wi-Fi signal (e.g., walls, furniture); 2) interference that may arise from nearby access points (APs) or coexisting IoT devices using Wi-Fi or Bluetooth protocols, particularly in dense urban environments; or 3) network congestion that can result from high traffic loads on the same AP or simultaneous access by multiple users. These factors degrade the QoE for end-users, causing delays, black-edge events, or poor synchronization between movements and visuals, all of which undermine the immersive nature of VR applications [34, 32, 35]

Given the growing importance of Cloud VR applications, it is imperative to detect and diagnose performance degradation issues to ensure consistent and reliable sessions. Accurate root-cause diagnosis can help improve network architectures to support emerging low-latency applications or enable ISP) to develop intelligent APs capable of diagnosing and mitigating impairments dynamically. Traditionally, root-cause diagnosis has been performed using expert-based rules [159, 160]. In this approach, network experts analyze KPIs using predefined heuristics, thresholds, or conditions to identify anomalies and determine their root causes. While effective in simple scenarios, this method is manual, time-consuming, and does not scale well to modern, complex network environments where vast amounts of KPIs are generated continuously.

In recent years, advancements in machine learning (ML) have revolutionized root-cause diagnosis by automating the analysis of large-scale KPI data. Machine learning models can automatically process time-series KPI data to perform root-cause diagnostic efficiently. Time Series Classification (TSC) techniques, in particular, have emerged as effective tools for this task, as they can capture temporal patterns and relationships essential for identifying each cause. Several categories of TSC techniques have been proposed, including distance-based (e.g., DTW), kernel-based (e.g., SVM), shapelet-based approaches, tree-based algorithms or deep learning models leveraging MLPs, CNNs, RNNs, and GNNs [161, 162, 163]. However, supervised TSC methods rely on annotated datasets, which are often expensive and time-consuming to generate, as they require domain expertise to label vast amounts of KPI data. To address the challenges of annotation, researchers are increasingly shifting toward SSL for unsupervised time-series classification. SSL has achieved remarkable performance in fields like computer vision and NLP by learning meaningful representations of data without relying on labeled samples. Unlike supervised methods, self-supervised approaches define pretext tasks (e.g., contrastive learning or self-prediction) to extract underlying features from raw data. One of the most popular SSL frameworks is CL, which uses data augmentation to generate multiple views from a single instance. Positive pairs (augmented views of the same instance) are brought closer together in the feature space, while negative pairs (views of different instances) are pushed apart. Contrastive learning has demonstrated significant success in representation learning for time-series

data and forms the foundation of many state-of-the-art unsupervised TSC methods.

Inspired by these advancements, we propose Root cause Anomaly Identification and Diagnosis (RAID), a two-stage machine learning framework for root-cause diagnosis of performance degradation in Cloud VR applications over Wi-Fi networks: in the first stage, an anomaly detection model uses contrastive learning to differentiate between normal and anomalous time-series KPIs. In the second stage, a lightweight supervised classifier that distinguishes between different types of anomalies detected in Stage 1. We empirically demonstrate that this two-stage framework outperforms existing competing methods, even when trained with limited labeled data. The evaluations are conducted on KPI datasets collected from the real-world Cloud VR testbed, presented on Chapter 2, featuring a Cloud VR application built on NVIDIA CloudXR architecture. The system streams VR content over a Wi-Fi connection using a commercial access point and an Oculus Quest 2 headset. Our proposed solution offers a scalable and efficient method for root-cause diagnosis, paving the way for improved network support for low-latency applications like Cloud VR.

Contributions

Specifically, the key contributions of this chapter are as follows:

- We introduce a novel two-stage framework that combines contrastive learning-based anomaly detection with supervised classification to effectively detect and diagnose Wi-Fi impairments in Cloud VR environments.
- We perform extensive empirical evaluations using time series KPI datasets collected from our testbed that replicates real-world network impairments, comparing our proposed solution with state-of-the-art time series classification models.
- Our solution demonstrates strong performance even in low-label scenarios, highlighting its ability to generalize with minimal supervision. Additionally, it offers a balanced trade-off between moderate training time and low inference latency, making it well-suited for real-time deployment in practical applications.

5.2 Related work

5.2.1 ML-based network root cause diagnosis

RCD aims to identify the underlying reasons behind network anomalies, such as degraded performance or failures. The advent of machine learning (ML) has driven the development of numerous data-driven solutions for RCD, leveraging ML models to analyze network data and pinpoint the sources of issues. Below, we discuss notable research efforts in this domain, focusing on various supervised and unsupervised techniques tailored for network environments.

Dimopoulos et al. [102] introduced a supervised ML framework to detect the root cause of QoE issues in video streaming. The metrics were collected from three key vantage points—mobile devices, routers, and content servers—to train supervised models. This work highlights the effectiveness of incorporating diverse data sources in diagnosing video streaming issues, enabling targeted interventions. Kawasaki et al. [164] focused on root cause analysis in NFV environments. The study applied supervised ML algorithms such as MLP, SVM, and Random Forests for fault classification. By comparing multiple ML models, the authors demonstrate the feasibility of using supervised approaches for diagnosing issues in complex, virtualized networks.

Salik et al. [96] detected non-WiFi interference and estimated its impact on WiFi throughput for both 2.4GHz and 5GHz bands. The authors developed supervised models, including Random Forest, CatBoost, and MLP, using network metrics collected at the WiFi edge. This approach effectively identified and mitigated interference from non-WiFi sources, improving network reliability in congested environments. Syrigos et al. [94] developed ML-based tools to troubleshoot common WiFi impairments such as contention, interference, hidden terminals, and low signal strength. Supervised ML models were trained on curated datasets to classify network impairments, enabling actionable diagnostics. The framework provided a systematic way to enhance WiFi network performance by pinpointing specific issues. Dötterl et al. [165] classified home network problems using advanced deep learning techniques. A transformer-based architecture is employed to analyze outputs from network monitoring tools like ping and dig. It demonstrated that transformer models can effectively classify complex network problems, offering precise diagnostics for home networks.

Fida et al. [106] proposed a two-stage framework for bottleneck identification in cloudified 5G networks: in Stage 1, an unsupervised VAE detects anomalies in network telemetry data while in Stage 2, an MLP classifier identifies the specific root cause of the bottlenecks. The distributed telemetry framework enabled real-time diagnosis in 5G networks, demonstrating scalability and adaptability in modern mobile environments.

Our work builds on recent advancements in RCD by addressing the limitations of existing solutions that rely exclusively on supervised methods. We introduce a novel approach specifically tailored for low-latency applications operating in Wi-Fi network environments. By leveraging contrastive learning, our solution enhances diagnostic accuracy, particularly in scenarios with limited labeled data. While our approach shares similarities with the method proposed in [106], we differentiate ourselves by employing contrastive learning for anomaly detection, which results in improved performance.

5.2.2 Time series Classification

TSC has been a prominent research topic for decades due to its applicability in numerous real-world domains, including cybersecurity and network management. A variety of solutions have been proposed in the literature, with methods ranging from classical approaches to modern deep learning techniques.

According to Bagnall et al. [166], classical TSC methods can be categorized into several groups including distance-based methods amoung those one of the most prominent examples is the 1-NN-DTW algorithm, which performs one-nearest-neighbor classification using dynamic time warping (DTW) as the distance metric; interval-based methods like Time Series Forest (TSF) [167] which leverage random forests and extract summary statistics from intervals of time series data to improve classification accuracy; shapelet-based methods like binary shapelet transformation methods [168] which use shapelets that are phase-independent subseries that capture discriminative patterns within time series data for TSC or ensemble-based methods like Collective of Transformation-based Ensembles (COTE) [169] that combines classifiers trained on different data representations. This approach highlights the importance of transforming data into subspaces where features are more easily discriminable.

In recent years, deep learning techniques have gained significant traction in TSC due to their ability to learn complex and hierarchical features directly from raw data. Various architectures have been proposed, including MLPs, CNNs, TCNs or RNNs [170, 171, 172]. These techniques were supervised learning techniques and were successful in time series classification over diverse domains [161]. However, the rise of SSL have come with a jump in performance of TSC

models. In fact, SSL has gained significant interest for its ability to learn meaningful representations from unlabeled data through carefully designed pretext tasks on computer vision or natural language processing. These tasks aim to uncover the intrinsic structure of the data, enabling the learned representations to be effectively utilized in downstream tasks, including TSC. Among the various SSL approaches, CL has emerged as particularly effective. It works by encouraging similar samples (positive pairs) to cluster in the feature space while pushing apart dissimilar samples (negative pairs). For time series data, numerous CL frameworks have been proposed, demonstrating strong performance on TSC tasks. T-Loss [142] introduces a triplet loss that constructs positive pairs using subseries extracted from the same time series, while negative pairs are formed using subseries from other time series. Coupled with a SVM, the learned representations perform exceptionally well on multivariate TSC tasks, demonstrating robust feature extraction. TNC (Temporal Neighborhood Coding) [149] constructs pairs based on the concept of temporal neighborhoods, leveraging statistical tests to identify segments of a time series that share similar underlying properties. It achieves excellent results on simulated and real-world datasets, such as ECG data, by capturing temporal dependencies. TS-TCC (Time-Series Temporal and Contextual Contrastive Learning) [141] employs cross-view prediction using data augmentations and integrates temporal and contextual contrastive learning modules. This design ensures the model learns robust, discriminative representations and demonstrates superior accuracy on TSC tasks across diverse domains by effectively handling temporal dynamics. CLOCS [173] specifically designed for ECG signals, applies data transformations to learn patient-specific representations that are invariant to spatio-temporal variations. Focused on healthcare, this method excels in classifying ECG time series by capturing unique and invariant features. TF-C (Time-Frequency Contrastive Framework) [174] encourages time series with time and frequency similarity to cluster in the time-frequency space thanks to the introduction of novel frequency-based augmentation techniques to generate augmented views of the data. It achieves competitive results in TSC across a wide range of domains, showcasing the effectiveness of incorporating frequency-domain information. TS2Vec [143] proposes a hierarchical framework to learn both temporal-wise and instance-wise similarities at multiple semantic levels. The contextual consistency-based pair selection enhances representation learning for diverse time series patterns. It demonstrates versatility and efficiency in multivariate TSC and other related tasks, further establishing the value of contextual representation learning.

We tackle RCD in Cloud VR as a time series classification problem. Unlike aforementioned approaches to time series classification, our method employs a two-stage framework: first, it identifies normal scenario time series, and then it classifies faulty scenarios, thereby improving the overall accuracy and effectiveness of the classification process.

5.3 Proposed Method: RAID

In this section, we present our approach to root cause diagnosis for Cloud VR applications as a time series classification problem. The dataset of T multivariate time series KPIs is represented as $\mathcal{D}_{train} = \{(w_1, y_1), (w_2, y_2) \ldots, (w_T, y_T)\}$ where $w_t \in \mathbb{R}^{p \times m}$ is a $p \times m$-dimensional vector corresponding to the KPIs values for time steps $t, t+1, \ldots, t+p-1$, and y_t is a one-hot encoded label vector of size K. $\forall j \in [|1, K|]$, $j = 1$ if w_t belongs to class j and $y_{t,j}$ 0 otherwise.

The task of root cause diagnosis is to learn a classification model f_θ on $\mathcal{D}_{train}$ that can assign the root cause y_t for any newly observed time series $\tilde{w}_t$. To achieve this, we propose RAID, depicted in Fig. 5.1, a two-stage architecture comprising i) an **anomaly detection stage** and ii)

a **root cause classification stage**.

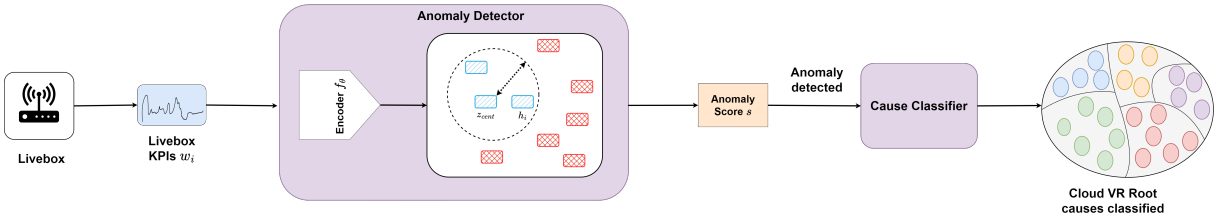


Figure 5.1: RAID Framework: The KPIs w_i collected from the Livebox are fed into the anomaly detector, which generates an anomaly score s. If the anomaly score exceeds a predefined threshold, indicating an anomaly, the KPIs w_i are then passed to the root cause classifier to determine the corresponding root cause class.

5.3.1 Anomaly detection stage

For the anomaly detection stage, we adopt the CATS framework, described in Chapter 4, which we briefly recall here. CATS leverages temporal contrastive learning to detect anomalies without labeled data. It enhances performance through synthetic anomaly generation and a combined loss of global and temporal contrastive learning. The CATS framework (depicted in Fig. 5.2) includes data augmentation, an encoder for latent representations, and contrastive losses (Temporal Contrastive Loss and Global Contrastive Loss) to capture both temporal and global similarities. Anomalies are detected by comparing the distance between the latent representation of new data and the centroid of normal representations.

5.3.2 Root Cause Classification

Once an anomaly is detected, the next step is to determine its underlying root cause. This stage is framed as a supervised classification problem, where the objective is to map each detected anomaly to a predefined class of root causes $\{cause_1, cause_2, \ldots, cause_{K-1}\}$.

To ensure efficiency and simplicity, we use a shallow classifier, a Support Vector Machines (SVM). Despite the numerous techniques for supervised TSC proposed in the literature, this model is efficient enough for our task given the anomaly detection stage previously executed.

SVM are supervised learning algorithms designed to find the optimal hyperplane that separates data into distinct classes in a feature space. In the context of root cause classification, the input of time series flagged as anomalies $(x_i, y_i)_{i=1}^{N_{train}}$ are fed as input to the SVM classifier. The SVM employs a kernel trick (a kernel function k satisfying $k(x_i, x_j) = \phi(x_i)\phi(x_j)$) to map inputs into a higher-dimensional space where a linear separation is possible. We use the Radial Basis Function (RBF) kernel function ($k(x_i, x_j) = \exp(-\gamma|x_i - x_j|^2)$). The classifier next identifies the hyperplane that maximizes the margin between the classes by solving the dual problem.

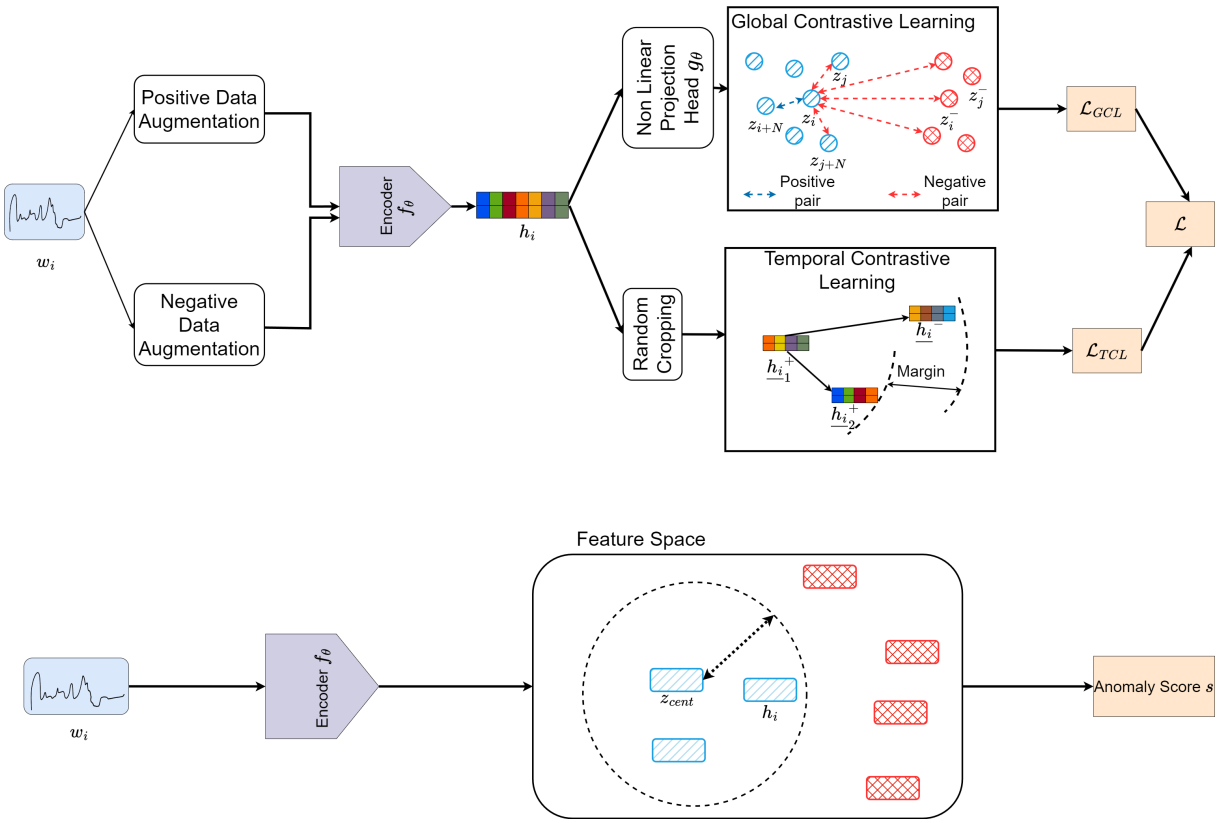


Figure 5.2: The anomaly detector stage: Training with contrastive learning (GCL and TCL) and at the inference step, an anomaly score is computed as the distance between the centroid and the latent representation.

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2}\sum_{i,j=1}^{N_{train}} \alpha_i \alpha_j y_i y_j k(x_i, x_j) - \sum_{i=1}^{N_{train}} \alpha_i, \\ \text{s.t} \quad & 0 \leq \alpha_i \leq C, \quad \forall i = 1, \ldots, N_{train} \\ & \sum_{i=1}^{N_{train}} \alpha_i y_i = 0 \end{aligned} \tag{5.1}$$

where α_i are the Lagrange multipliers obtained by solving the dual optimization problem; $C > 0$ is the regularization parameter that controls the trade-off between maximizing the margin and minimizing classification errors; y_i are the labels of the training data points and x_i are the feature vectors of the training data.

Each input $\tilde{x}$ is then assigned to one of the predefined cause $\tilde{y}$ as follows:

$$\tilde{y} = \text{sign}\Big(\sum_{i=1}^{N_{train}} \alpha_i y_i K(x_i, \tilde{x}) + b\Big), \tag{5.2}$$

For multi-class classification with $K > 2$, SVM is extended using one-vs-one strategies

where $\frac{K(K-1)}{2}$ binary classifiers are trained, each distinguishing between two classes and the class with the highest votes is chosen.

5.4 Evaluation setup

5.4.1 Dataset Description

To evaluate our proposed solution, we utilize the time series datasets collected from the experimental testbed described in Chapter 2. Although our setup gathers KPIs from both the VR headset and the CloudXR stack, which provide insights into QoS and QoE during VR sessions, this study focuses exclusively on data retrieved from the Livebox. This choice is motivated by the practical accessibility of these metrics for network operators, who own and manage the Livebox. Leveraging these metrics for RCD allows for the development of smarter APs and more intelligent network management solutions, aligning closely with the operational needs of network operators.

The dataset consists of 112 time series features extracted from the Livebox. These features include signal strength indicators (e.g., RSRP, RSSI), transmission performance metrics (e.g., txops), channel utilization measures (e.g., air time), among others, offering a comprehensive view of Wi-Fi performance in various conditions. Monitoring was performed at a frequency of one sample every three seconds. To facilitate analysis, the data is structured into overlapping time series windows, each spanning 10 time steps (30 seconds per window).

In total, the dataset contains 13,657 time series windows, which are partitioned into training and testing subsets using a 70:30 split ratio. The training set contains 9,524 windows, while the test set consists of 4,133 windows. The dataset is further categorized into three classes, corresponding to distinct experimental scenarios during data collection:

A summary of the dataset, including the class-wise breakdown of training and test samples, is presented in Table 5.1.

Table 5.1: Dataset Summary

Class	Train Size	Test Size	Number of Features	Number of Time Steps
Normal	4718	1924	112	10
Coverage	2984	1270	112	10
Interference	1822	939	112	10
Overall	9524	4133	112	10

5.4.2 Competing Solutions

To demonstrate the effectiveness of RAID, we compare it against several baselines. These include both one-stage and two-stage time series classification (TSC) models.

One-Stage Models

For the one-stage models, we include the following:

- **1-NN-DTW:** The nearest neighbor classifier paired with a distance function. When combined with Dynamic Time Warping (DTW) as the distance measure, this model has proven to be a strong baseline for time series classification across numerous benchmarks [166].

We also evaluate self-supervised time series representation learning methods, which have demonstrated efficiency in TSC. Following the protocol outlined in [142], these models undergo a self-supervised pre-training stage, after which an SVM classifier with an RBF kernel is trained on the learned representations for downstream classification. The selected methods are:

- **T-Loss [142]:** A self-supervised learning approach that uses a novel triplet loss with time-based negative sampling to obtain representations general enough for downstream tasks, even when dealing with variable-length, multivariate time series.
- **TS-TCC [141]:** A self-supervised framework for time series representation learning that employs weak and strong time series augmentations to create different views. These views are used in temporal and contextual contrastive modules to learn robust and discriminative representations.
- **TS2Vec [143]:** A framework designed to learn both instance-wise and temporal-wise information for robust time series representations. It utilizes a hierarchical contrastive objective applied to augmented views of time series, achieving excellent results in TSC tasks.

Two-Stage Models

For the two-stage models, we replace the CATS model in the first stage with unsupervised anomaly detection methods. The selected models are:

- **iForest [60]:** An isolation-based unsupervised anomaly detection model. It isolates anomalies by recursively performing random splits on the feature values. Anomalous samples are easier to isolate due to their significant deviation from the normal data, requiring fewer splits.
- **USAD [64]:** UnSupervised Anomaly Detection is a deep learning-based approach employing two autoencoders in a min-max game. The first autoencoder learns to reconstruct input data, while the second attempts to differentiate between true data and reconstructions.
- **SimCLR [47]:** A contrastive learning framework originally proposed for computer vision and adapted for time series. It learns representations from augmented views of data and can be used for anomaly detection following a procedure similar to the CATS framework.

5.4.3 Evaluation Metrics for Cause Classification

We evaluate the performance of our root-cause diagnosis models using well-known multi-class classification metrics which differ from the AD metrics presented in Chapter 1. These metrics include weighted Precision (P), weighted Recall (R), weighted F1-score (F1), Accuracy (Acc), and Normalized Accuracy (N-Acc). These metrics are defined as follows:

- **Precision:** The macro-weighted precision is the weighted average of precision values computed for each class, w_i being the proportion of class i:

$$P = \sum_{i=1}^{K} w_i \times P_i, \quad P_i = \frac{TP_i}{TP_i + FP_i} \tag{5.3}$$

- **Recall:** The macro-weighted recall is the weighted average of recall values computed for each class, w_i being the proportion of class i:

$$R = \sum_{i=1}^{K} w_i \times R_i, \quad R_i = \frac{TP_i}{TP_i + FN_i} \tag{5.4}$$

- **F1-Score:** The macro-weighted F1-score is the harmonic mean of macro-weighted Precision and Recall. This metric coupled P and R are suitable for imbalanced datasets.

$$F1 = 2 \times \frac{P \times R}{P + R} \tag{5.5}$$

- **Accuracy:** The fraction of correctly classified samples over the total number of samples. This metric is widely used and easy to interpret.

- **Normalized Accuracy (N-Acc):** This metric adjusts the balanced accuracy ($bac = \frac{1}{K} \sum_{i=1}^{K} w_i \times R_i$) which is the weighted average of the recall of each class with respect to the accuracy of random guessing (bac_{RG}), ensuring that random predictions score 0 while perfect predictions score 1. It is more interpretable and suitable for imbalanced datasets.

$$\text{N-Acc} = \frac{bac - bac_{RG}}{1 - bac_{RG}} \tag{5.6}$$

5.4.4 Implementation details

The datasets collected are normalized and splits into training and test sets. For our detector solution, we use the same implementation details as CATS architecture. This consists in a dilated CNN module with ten residual blocks of 1D convolutional layers as encoder; a three-layer MLP with ReLU activation as projection head. The data augmentation techniques include *jitter*, *scaling* for positive views and *masking* and *trend* for negative views. The anomaly detection framework is trained with temporal and global contrastive loss optimized using the Adam optimizer with a learning rate of 10^{-3}, the batch size is 512 and training is conducted for 100 epochs. The

For competing solutions, we rely on publicly available implementations provided on Github. We use the same epochs, optimizer, learning rate and batch size. After pretraining of SSL competing solutions, classification is performed using a SVM with an RBF kernel. Hyperparameters of the SVM are fine-tuned using a grid search procedure to achieve optimal performance.

Experiments are performed on a Ubuntu 22.04 with AMD Ryzen 9 5900X 12-Core Processor and a NVIDIA RTX 3090 Ti of 24GB. Python 3.10.12 is used as programming language, PyTorch 2.2.0 as the deep learning framework and CUDA 12.1 for GPU acceleration. The code and datasets to reproduce all the experiments are provided [26]. All details, implementation and hyper-parameters are included.

[26] `https://github.com/joelromanky/raid`

Table 5.2: Performance comparison on the datasets. Mean and standard deviation computed over five runs for Cloud VR datasets. Bold values indicate best results and underlined values the second best.

Models	Metrics	Accuracy	N-Accuracy	Precison	Recall	F1-score
One-stage	1-NN-DTW	51.54$_{(\pm 0.11)}$	26.36$_{(\pm 0.17)}$	56.74$_{(\pm 0.09)}$	51.54$_{(\pm 0.11)}$	52.96$_{(\pm 0.10)}$
	T-Loss	79.47$_{(\pm 4.39)}$	**75.22**$_{(\pm \mathbf{5.58})}$	**83.98**$_{(\pm \mathbf{4.74})}$	79.47$_{(\pm 4.39)}$	79.60$_{(\pm 4.53)}$
	TS2Vec	70.12$_{(\pm 5.28)}$	55.71$_{(\pm 6.53)}$	75.42$_{(\pm 3.22)}$	70.12$_{(\pm 5.28)}$	70.49$_{(\pm 4.88)}$
	TS-TCC	73.78$_{(\pm 6.38)}$	66.17$_{(\pm 8.12)}$	79.29$_{(\pm 6.07)}$	73.78$_{(\pm 6.38)}$	73.86$_{(\pm 6.68)}$
Two-stage	iForest	72.48$_{(\pm 2.69)}$	62.22$_{(\pm 4.69)}$	72.26$_{(\pm 3.14)}$	72.48$_{(\pm 2.69)}$	72.24$_{(\pm 2.99)}$
	USAD	72.22$_{(\pm 0.80)}$	63.39$_{(\pm 1.36)}$	72.72$_{(\pm 0.97)}$	72.22$_{(\pm 0.80)}$	72.38$_{(\pm 0.84)}$
	SimCLR	57.76$_{(\pm 3.25)}$	37.59$_{(\pm 4.84)}$	61.01$_{(\pm 2.60)}$	57.76$_{(\pm 3.25)}$	58.65$_{(\pm 3.06)}$
	RAID	**81.83**$_{(\pm \mathbf{2.96})}$	74.80$_{(\pm 4.19)}$	81.85$_{(\pm 3.02)}$	**81.83**$_{(\pm \mathbf{2.96})}$	**81.60**$_{(\pm \mathbf{3.05})}$

5.5 Results

5.5.1 Performance Evaluation

Table 5.2 summarizes the evaluation results of our solution compared to competing TSC methods using various performance metrics, including accuracy, normalized accuracy, precision, recall, and F1-score. The results highlight the superiority of our approach over both one-stage and two-stage methods.

Evaluation of One-Stage Models

One-stage models, including 1-NN-DTW, T-Loss, TS2Vec, and TS-TCC, directly perform root cause classification without a preliminary anomaly detection step. Among these models, 1-NN-DTW exhibits the lowest overall performance, with an accuracy of 51.54% and an F1-score of 52.96%. Despite being a strong baseline for TSC, it struggles to handle the complex time-series data encountered in CloudVR scenarios.

Contrastive learning-based SSL models outperform 1-NN-DTW. Among them, T-Loss emerges as the most effective technique, achieving the highest normalized accuracy (75.22%) and precision (83.98%) within this category. This demonstrates its capability to learn meaningful representations for RCD tasks. TS-TCC follows with an accuracy of 73.78% and an F1-score of 73.86%, while TS2Vec achieves an accuracy of 70.12% and an F1-score of 70.49%.

Evaluation of Two-Stage Models

Two-stage models incorporate a preliminary anomaly detection step, enabling better focus on relevant patterns before root cause classification. iForest and USAD achieve comparable performance, with accuracy scores of 72.48% and 72.22%, respectively. Both models demonstrate strong F1-scores around 72%, yet they fall short of advanced one-stage approaches like T-Loss.

Meanwhile, SimCLR performs suboptimally with an accuracy of 57.76% and an F1-score of 58.65%.

Our proposed solution significantly outperforms all competing methods across most metrics. It achieves the highest accuracy (81.83%), recall (81.83%), and F1-score (81.60%), demonstrating robustness and effectiveness for CloudVR RCD. While T-Loss marginally outperforms in normalized accuracy and precision, our model achieves the best balance across all metrics, establishing it as the most reliable approach in this evaluation.

The superior performance of our solution can be attributed to the efficiency of its anomaly detection stage. As shown in Fig. 5.3, our solution outperforms other two-stage techniques in detecting anomalies across various AD metrics (cf Chapter 1). RAID achieves the best overall anomaly detection performance, which directly contributes to its effectiveness in RCD tasks.

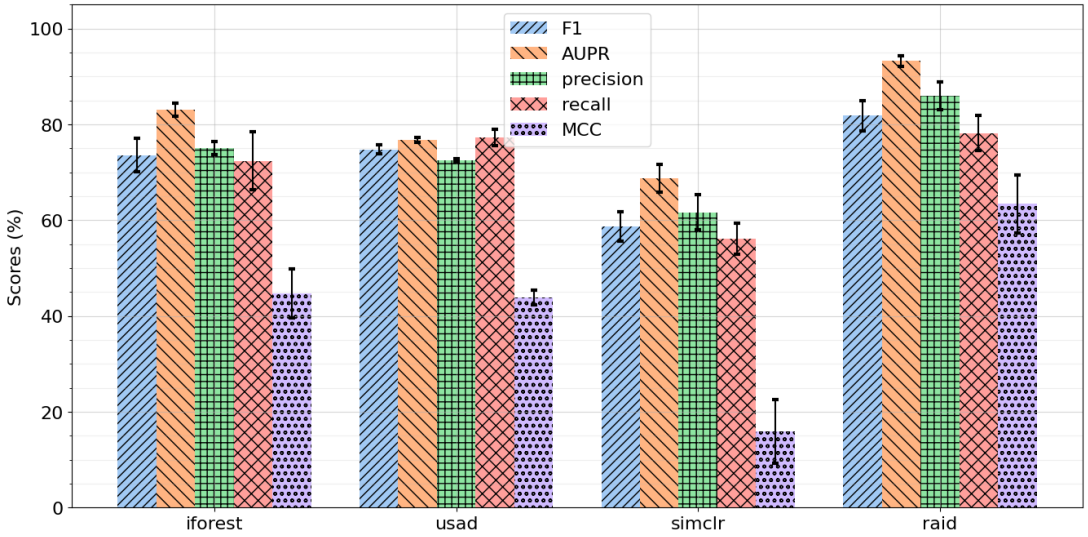


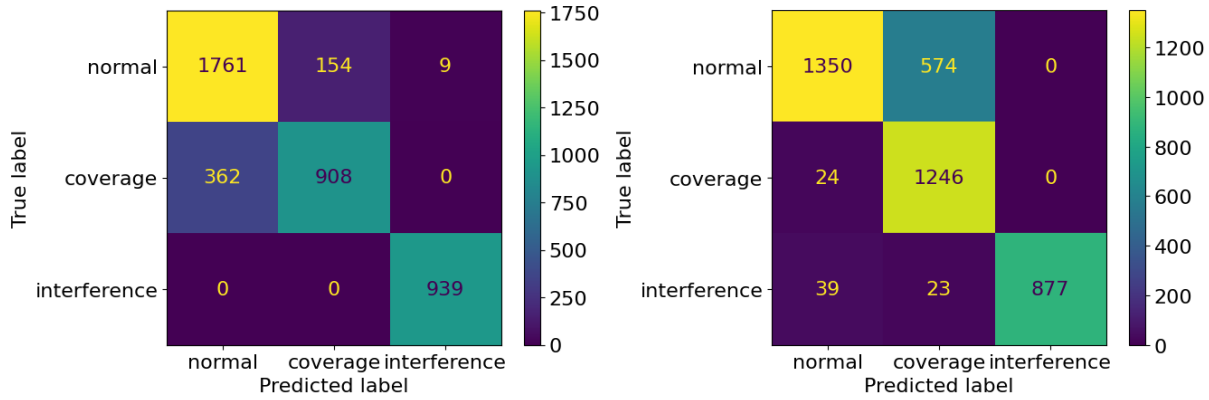
Figure 5.3: Results of anomaly detectors of two-stage models.

Per-Class Performance Analysis

Figures 5.4 and 5.5 provide a detailed comparison of the per-class performance metrics for T-Loss and RAID. Our approach demonstrates a significant advantage in efficiently distinguishing normal scenarios from both coverage and interference scenarios.

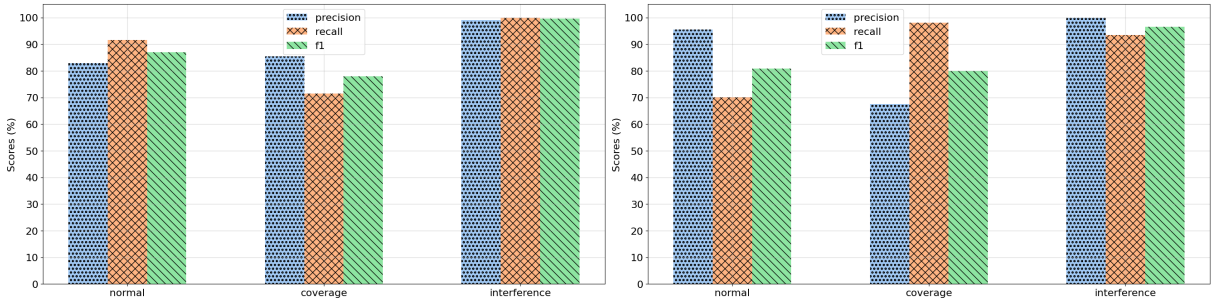
For normal scenarios, our solution achieves a notably lower misclassification rate compared to T-Loss, with 1,761 correctly classified normal samples versus 1,350 for T-Loss. This represents a substantial improvement in detecting normal behavior. Additionally, our solution attains perfect classification for interference scenarios, with a recall of 100%, highlighting its robustness in detecting distinct anomaly patterns such as interference.

However, Fig. 5.5 also reveals the limitations of our solution. It struggles to discriminate coverage scenarios, with a considerable number of coverage windows misclassified as normal. This indicates challenges in capturing the subtle variations and transitional patterns between normal and coverage states. In contrast, T-Loss, while less accurate overall, shows a more balanced performance in handling coverage scenarios. Since interference states contributed the most to user experience degradation during our data collection (this will need confirmation when a Cloud VR QoE model is developed), the impact of this limitation is reduced in practical terms.



(a) **RAID** (b) **T-Loss**

Figure 5.4: Confusion matrix



(a) **RAID** (b) **T-Loss**

Figure 5.5: Per-class precision, recall and F1-score.

This trade-off underscores the strengths and weaknesses of our model: it is highly effective in detecting clear-cut anomalies but requires further refinement to enhance its sensitivity to nuanced variations between normal and coverage states. Future work could focus on addressing this limitation by incorporating advanced feature extraction techniques or domain-specific data augmentation strategies.

Takeaways

RAID model demonstrates superior performance in root cause diagnosis for Cloud VR scenarios, achieving the highest performance compared to both one-stage and two-stage approaches. Its robust anomaly detection stage significantly enhances its effectiveness, outperforming other methods in detecting distinct patterns such as interference. However, the model struggles with subtle variations in coverage scenarios, suggesting room for improvement in handling nuanced transitions.

5.5.2 Efficiency with Few Labels

Fig. 5.6 illustrates the evolution of model performance as the percentage of labeled data increases. The left subplot depicts the accuracy scores across various label ratios, while the right subplot presents the corresponding F1-scores.

At the lowest label ratios (1%-5%), most models exhibit limited performance, reflecting the inherent difficulty of accurate root cause diagnosis (RCD) with minimal supervision. However, T-Loss and RAID stand out by achieving relatively higher accuracy and F1 scores, showcasing their ability to generalize effectively even with sparse labeled data. T-Loss benefits significantly from its triplet-based pretraining strategy, which efficiently captures meaningful representations from the unlabeled dataset, thereby enhancing fine-tuning performance. Similarly, the pretraining stage of RAID contributes to its robustness in low-label scenarios by effectively leveraging the anomaly detection process to prioritize relevant patterns.

As the label ratio increases, all models demonstrate steady improvement in performance, highlighting the benefits of additional labeled data. Notably, RAID and T-Loss consistently lead in performance, with our solution exhibiting a steady performance boost. This consistency underscores the robustness of RAID across varying levels of supervision. While T-Loss initially competes closely, its performance shows a slight decline between the 5% and 20% label ratios, coupled with increased variability, indicating potential sensitivity to the quality or distribution of labeled data in these ranges.

The findings from Fig. 5.6 highlight the efficiency of RAID in leveraging limited labeled data, making it an ideal solution for real-world scenarios where labeling is both expensive and time-consuming. Its performance with sparse labels, along with its stable scalability as more labeled data becomes available, firmly establishes RAID as the most suitable model in this evaluation.

Takeaways

RAID also demonstrates strong performance in scenarios with limited labeled data thanks to its anomaly detection stage that effectively prioritize relevant patterns. While T-Loss competes closely in low-label scenarios, its performance exhibits greater variability as label ratios increase.

5.5.3 Time complexity

Fig. 5.7 presents the training time (in seconds) and inference time per time series (in milliseconds) for each of the RCD models. The model with the longest training time is Triplet, which takes approximately 300 seconds, while the fastest training model is 1-NN-DTW, completing training in 500 milliseconds. In terms of inference time, 1-NN-DTW significantly outpaces other models, with the highest inference time of 1800 milliseconds. In contrast, models such as Triplet or TS-TCC, achieve inference times as low as 0.5 milliseconds.

Our proposed solution, RAID, demonstrates a moderate training time of 200 seconds and an inference time of 3.5 milliseconds. While this inference time is the second highest among the models compared, it is still well-suited for real-time deployment, especially in our testbed where data is collected at frequent intervals (e.g., every 3 seconds). This makes RAID an excellent choice for RCD, as it balances moderate training overhead with sufficiently low inference latency, allowing for continuous monitoring and fast anomaly detection. Additionally, being a two-stage model, RAID offers a key advantage: when new causes or anomalies are detected, only the supervised classifier requires retraining. Most of the training time originates from the

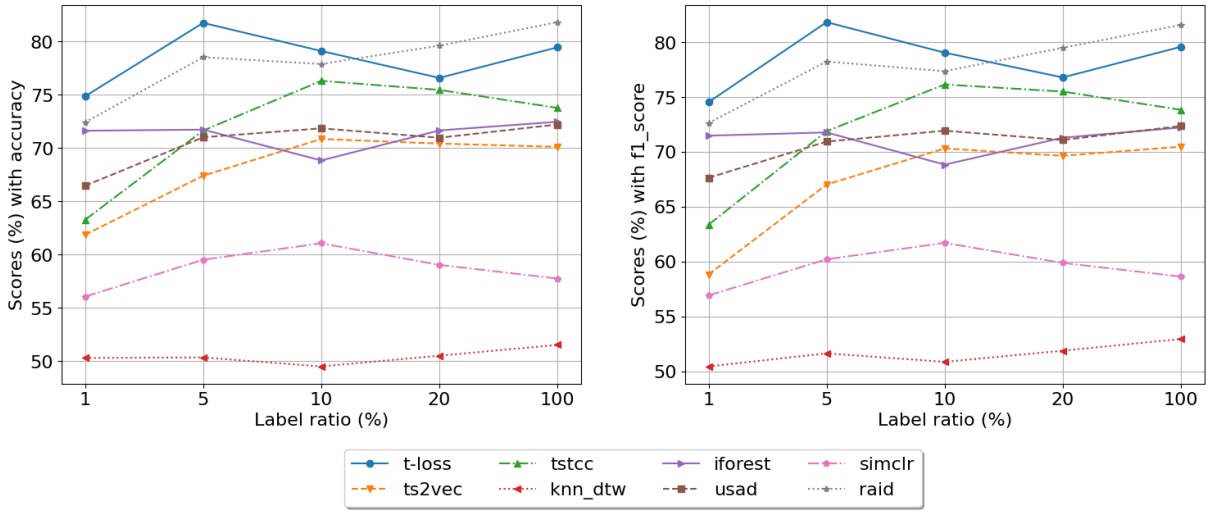


Figure 5.6: Performance variation regarding the labels ratio.

initial anomaly detection phase, unlike one-stage models that require complete retraining, including the pretraining phase. This makes RAID more efficient for scenarios requiring periodic updates or retraining, reducing overall downtime and resource consumption.

In summary, RAID strikes a practical balance between training efficiency and inference speed, making it highly effective for real-time RCD in dynamic and large-scale network environments.

Takeaways

RAID achieves a practical balance between training efficiency and inference speed, with a training time of 200 seconds and an inference time of 3.5 milliseconds. RAID's two-stage design further enhances its practicality by requiring retraining only for the supervised classifier when new anomalies are detected, unlike one-stage models that necessitate full retraining.

5.6 Conclusion

This chapter presents a Root Cause Diagnosis (RCD) approach for identifying network issues in Cloud VR sessions over Wi-Fi networks, utilizing time series KPIs collected from access points. By employing a two-stage framework, we demonstrated the effectiveness of our approach compared to traditional time series classification methods. Our proposed architecture, which integrates contrastive learning into the anomaly detection process, has shown significant improvements in both identifying anomalies and diagnosing the root causes of Cloud VR performance issues. This provides a good foundation for future research in real-time diagnostics for cloud-based VR applications. One key strength of this approach is its balance between training time and inference speed, making it ideal for real-time diagnostics in dynamic network environments. Moreover, the two-stage design enhances efficiency by isolating retraining to the supervised classifier, thus avoiding full model retraining when new causes are introduced.

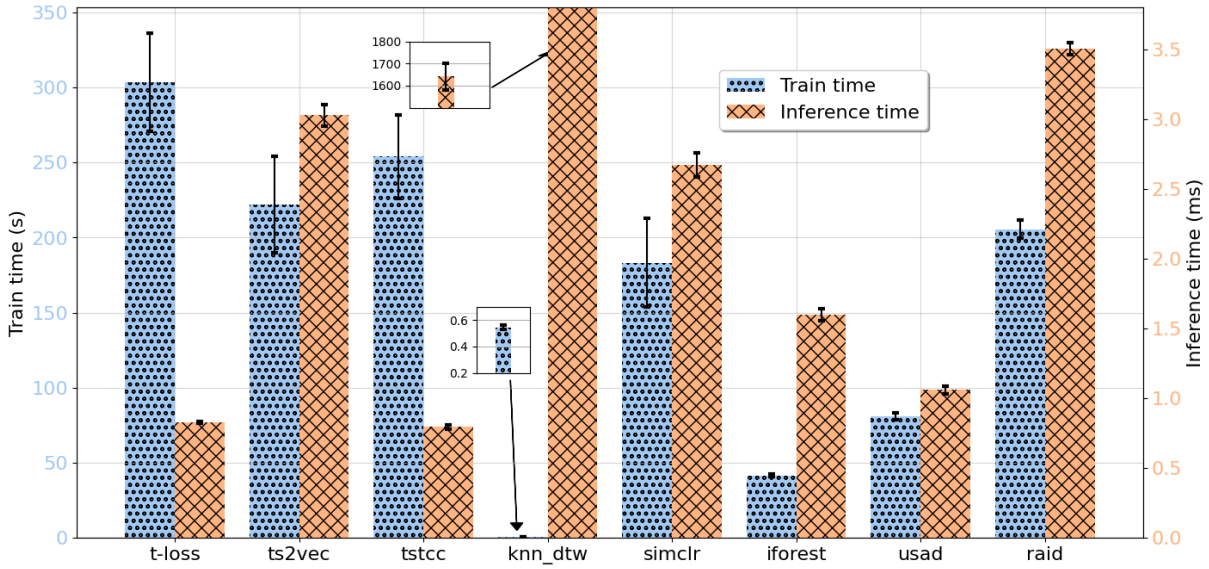


Figure 5.7: Time complexity of each model.

CONCLUSION AND PERSPECTIVES

General conclusion

Low-latency applications, such as cloud gaming, cloud XR, and the tactile internet, are poised to transform the future of communications and revolutionize industries including entertainment, healthcare, and education. These applications enable groundbreaking use cases such as real-time collaboration, lifelike simulations, and immersive virtual environments. However, their success is intrinsically tied to the performance of high-speed, low-latency networks. Variability in network conditions—such as fluctuating bandwidth, latency, jitter, or packet loss—can severely impact the QoE, leading to performance failures in environments like cellular and Wi-Fi networks. For network operators, addressing these challenges is both a technical necessity and a business imperative, as it involves detecting and diagnosing the root causes of performance issues to maintain service reliability.

This thesis addresses these challenges by contributing insights and innovative solutions for detecting anomalies and diagnosing root causes in low-latency applications. The research begins with the foundational task of data collection, described in Chapter 2. To create datasets representative of real-world LL application performance, we established multiple testbeds. First, we collected transmission opportunities (txops) from Orange's 4G network to simulate realistic network conditions. These txops were used to collect time series Key Performance Indicators (KPIs) from three commercial CG platforms. Additionally, a testbed was designed to collect Cloud VR KPIs under controlled Wi-Fi network conditions, using NVIDIA Cloud XR stack, encompassing both normal and impaired scenarios, to simulate real-world challenges.

In Chapter 3, we leverage the collected cloud gaming datasets to explore the applicability of unsupervised ML models for detecting QoE degradations in cloud gaming applications. Unsupervised ML offers the advantage of not requiring labeled data, which is often challenging to obtain in practice. This chapter examines eight unsupervised ML models, focusing on three key criteria: detection accuracy, robustness to data contamination, and computational efficiency. By introducing the Window Anomaly Decision (WAD) method and evaluating model performance using metrics such as F1-score and Matthews Correlation Coefficient (MCC), the chapter provides insights and practical recommendations on selecting the most suitable unsupervised ML models based on the application and configuration requirements. The findings are applied to a practical task in the MOSAICO project—cloud gaming traffic detection—though not covered in this thesis. This task, essential for reducing latency and improving QoE in cloud gaming, is reformulated as an unsupervised learning problem with USAD AD model, which as showed in Chapter 3, is the most effective model, demonstrating robustness to data contamination and low inference time. Addressing CG traffic detection this way also addresses the limitations of supervised methods by achieving better generalization to unseen traffic patterns. This work was published at an international conference [175].

To further improve anomaly detection (AD) in multivariate time series and address limitations such as limited robustness to data contamination, Chapter 4 proposes a novel framework called CATS (Contrastive Anomaly detection on Time Series). CATS leverages contrastive learning to enhance time series representation learning. Its core innovation lies in a novel loss function that clusters temporally similar time series windows while pushing apart dissimilar points in the latent space. The framework also incorporates negative data augmentation techniques to generate synthetic anomalies. Evaluated on public benchmark datasets and the cloud gaming

datasets collected in this thesis, CATS achieves superior performance compared to state-of-the-art AD models, while maintaining robustness to data contamination.

Finally, in Chapter 5 focuses on root cause diagnosis in Cloud VR applications operating on Wi-Fi networks. We propose a novel two-stage framework that integrates the CATS anomaly detection model from Chapter 4 with a lightweight supervised classifier. The first stage detects anomalies in Cloud VR KPIs collected from Orange's Livebox, while the second stage classifies these anomalies into specific categories. Evaluated on datasets collected under controlled Cloud VR experiments, this framework outperforms existing one-stage and two-stage approaches. Notably, it performs well even with limited labeled data, making it suitable for real-world deployment. Furthermore, its low inference and retraining times ensure practicality for dynamic network environments where new anomaly classes may emerge.

Perspectives for future research

This thesis has addressed the challenges of anomaly detection and root cause diagnosis for low-latency applications such as Cloud Gaming and Cloud VR. While the proposed contributions offer substantial advances, certain limitations remain, which provide opportunities for insightful future research. Below, we outline several promising directions for further exploration and development.

QoE models for LL applications

This work focused on diagnosing network issues impacting the QoE of LL applications. In QoE degradation detection for Cloud Gaming (CG), platform-specific recommendations were used to identify poor QoE sessions. For Cloud VR RCD, Wi-Fi impairment scenarios that impact our perceived session quality in lab were simulated.

Although existing QoE models for CG and Cloud VR are available in the literature [176, 177, 178], these models rely on features not available in our testbed and are not open-sourced. Since building QoE models was not the primary goal of this thesis, we use the aforementioned simplifications to conduct experiments. However, future work should explore the integration of QoE models for LL applications. By relabeling the CG and Cloud VR KPI datasets collected in this thesis, and by retraining the anomaly detection and RCD performance of the frameworks developed in this thesis, better evaluations could be achieved.

Improving Anomaly Detection in time series

The proposed CATS model for anomaly detection in time series demonstrated strong performance but also revealed limitations. The core innovation, a temporal contrastive loss based on Dynamic Time Warping (DTW) similarity, incurs a high computational cost due to the quadratic complexity of Soft-DTW divergence. This restricts its efficiency, particularly as only one negative pair for contrastive learning is used in this loss to limit computational costs. Future research could develop optimized loss functions that preserve the benefits of DTW-like metrics while reducing computational overhead. This would enable faster model training for efficient AD in time series.

Additionally, despite outperforming other methods in presence of data contamination, CATS performance is still impacted by the presence of anomalies in the training set. This sensitivity to data contamination in training could be mitigated by advanced negative data augmentations or incorporating contamination estimation methods and model uncertainty in AD [78, 179].

Moreover CATS being model-agnostic, exploring advanced architectures, such as GNN [180], could further enhance its ability to capture temporal dependencies in time series data.

Further data collection for Cloud VR experiments

The proposed RCD framework was validated on Cloud VR datasets collected under Wi-Fi networks with limited impairment scenarios. Future studies should expand this by including scenarios such as congestion, hidden terminals, or non-Wi-Fi interference, which are also commonly encountered in real-world Wi-Fi network environments. Emulating such impairments would improve the framework's generalizability.

Moreover, collecting data from real-world environments, such as enterprise networks, large-scale public Wi-Fi (e.g., campuses) or home Wi-Fi, would provide insights into more complex, intertwined causes of performance issues. Evaluating the RCD framework in these settings would further validate its robustness and effectiveness.

Root Cause Diagnostic over 5G networks

This thesis focused on Wi-Fi networks due to their accessibility for realistic lab-based experimentation. However, 5G networks, particularly standalone (SA) deployments, present new opportunities and challenges for RCD. With their lower latency and higher reliability, 5G networks are critical for Industry 5.0 applications that benefit from immersive technologies like Cloud VR [181].

Future work should explore RCD in 5G networks, addressing its specific impairments such as congestion, handover failures, signal attenuation or RAN misconfigurations. Collecting data through a 5G-to-Wi-Fi gateway [39]—as most current HMDs lack cellular connectivity—would enable testing the adaptability of the framework to 5G-specific challenges.

Few-shot learning for efficient labeling

A major challenge in applying machine learning to RCD tasks is the scarcity of labeled data, particularly for rare anomalies. As shown in Chapter 5, having more labeled data improves model performance. Few-shot learning approaches, such as Prototypical Networks [182] or Model-Agnostic Meta-Learning (MAML) [183], that aim at training models to achieve high performance on new tasks with only limited labeled examples (usually with $k = 5$ examples per class) could help in circumventing this challenge. Such methods are increasingly used in fault diagnosis to reduce labeling efforts while enabling quicker adaptation and improved performance in new environments [184, 185].

Leveraging multiple sources of data for RCD

In Chapter 2, data from multiple source including PCAP traffic, Livebox metrics, and client-side application metrics, was collected to provide a multi-view perspective on Cloud VR performance. However, only one Livebox metrics were utilized in Chapter 5 for RCD due to the simplier accessibility to these metrics for ISP. Integrating these data sources in future research can be insightful and a promising way to do so is Multi-View Learning (MVL) [186, 187] that combines features and structural information from diverse sources to extract shared properties that enhance learning performance. For unsupervised settings, Multi-View Clustering (MVC) [188, 189], a subdomain of MVL, could be employed to cluster samples with shared patterns

across different data sources, enabling improved RCD performance by leveraging the complementarity of diverse perspectives.

Novel Class Discovery

Current RCD approaches, including our own, assume a fixed set of degradation causes. However, in real-world scenarios, new, previously unseen causes may emerge. Novel Class Discovery (NCD) techniques [190] could address this challenge by learning to categorize unlabeled data into appropriate new classes. These methods leverage prior knowledge from known classes (e.g., existing causes of degradation) to infer new, unknown categories.

Causal Discovery

Understanding the causal relationships between KPIs could provide deeper insights into the mechanisms driving performance impairments in LL applications. Causal discovery techniques [191, 192] uncover cause-effect relationships and represent this underlying causal structure of the set of KPIs observed in causal graphs (with directed arrows from the cause to the effect). They can hence help identifying the root sources of faults in the network. Incorporating causal discovery into RCD frameworks would enhance their robustness, explainability, and utility for network operators by revealing how certain network metrics lead to impairments.

In conclusion, although the contributions presented here have their limitations and much remains to be explored, a Ph.D. marks the beginning of a research journey. It is a foundation upon which we aspire to build further, aiming to deliver more constructive, impactful, and meaningful advancements in the future.

RÉSUMÉ EN FRANÇAIS

Cette section présente un résumé détaillé en français de ce tapuscrit de thèse. Ce résumé vise à rendre le contenu de la thèse plus accessible aux lecteurs francophones tout en contribuant à la valorisation de la langue française dans le contexte académique et scientifique. Dans les sections suivantes, nous présenterons l'objectif de cette thèse et soulignerons son importance, avant de proposer un résumé du contenu et des contributions de chaque chapitre.

Introduction

Au cours des dernières décennies, les réseaux filaires et sans fil ont connu des avancées remarquables, révolutionnant la connectivité et permettant une large gamme d'applications dans divers secteurs. Les réseaux filaires ont évolué, passant des infrastructures à base de cuivre à la technologie avancée de fibre optique, offrant des vitesses de l'ordre du gigabit. Parallèlement, les réseaux sans fil ont également connu une croissance considérable, les réseaux mobiles évoluant de la 2G à la 5G, et les standards Wi-Fi progressant du 802.11b vers le Wi-Fi 6 et bientôt du Wi-Fi 7.

Ces avancées ont conduit à l'émergence des applications à faible latence, telles que les jeux en nuage ou jeux à la demande (*cloud gaming* en anglais), la réalité virtuelle et augmentée en nuage (*cloud virtual reality* en anglais), ainsi que l'Internet tactile ou la chirurgie à distance. Ces applications sont considérées comme des innovations majeures qui transformeront les secteurs du divertissement, de la santé, de l'éducation et de l'industrie en offrant des expériences collaboratives en temps réel, des simulations immersives et des environnements virtuels interactifs.

Le succès de ces applications à faible latence repose sur des réseaux capables de fournir des vitesses élevées, une latence ultra-faible et une fiabilité accrue. Cependant, ces exigences posent d'importants défis dans des environnements réseau à capacité variable, comme les réseaux Wi-Fi ou les réseaux cellulaires comme la 4G ou la 5G. Les fluctuations de bande passante, la latence, la gigue ou encore les pertes de paquets sont des problèmes fréquents pouvant dégrader considérablement la qualité d'expérience des utilisateurs.

Ces défis rendent la détection des anomalies et le diagnostic des causes racines essentiels pour maintenir les performances des réseaux et garantir une qualité d'expérience satisfaisante. Les opérateurs réseau, tels qu'Orange, doivent disposer d'outils efficaces pour identifier rapidement les problèmes et les résoudre, surtout à l'ère des technologies immersives qui deviennent de plus en plus omniprésentes.

Cette thèse, menée dans le cadre d'un contrat CIFRE entre Orange Innovation et le centre INRIA de l'Université de Lorraine, vise à répondre aux limitations actuelles dans la détection d'anomalies et le diagnostic des causes racines dans les applications à faible latence. Réaliser cet objectif nécessite de résoudre quelques défis significatifs qui sont les suivants:

- Collecter des datasets d'indicateurs de performances sur différentes applications à faible latence comme le cloud gaming ou le cloud en réalité virtuelle, dans des conditions réseaux réalistes et comportant des scénarios qui sont sujets à des dégradations de performance sur ces applications ;

- Développer des méthodes efficaces et robustes au phénomène de contamination de données. Ces méthodes permettraient de détecter les anomalies en utilisant des techniques

d'apprentissage automatique non supervisées, à partir des différents indicateurs de performance collectés ;

- Identifier les causes de dégradation de performance des applications à faible latence à l'aide de techniques de diagnostic reposant sur l'apprentissage automatique.

Pour répondre à ces problématiques, des contributions que nous détaillerons dans la suite, ont été effectuées:

- La mise en place de bancs de test pour collecter des données pertinentes sur des plateformes de cloud gaming et de cloud VR opérant sous des réseaux à capacité variable ;
- L'évaluation de modèles d'apprentissage automatique non-supervisé pour la détection d'anomalies dans les métriques collectées sur notre banc de test de cloud gaming ;
- Le développement d'un algorithme à base d'apprentissage contrastif pour améliorer la détection d'anomalies et la robustesse à la contamination de données ;
- La conception d'un modèle à deux étapes pour le diagnostic des causes racines combinant apprentissage contrastif et classification supervisée pour identifier les sources spécifiques de dégradation lors de l'utilisation des applications de Cloud VR avec des réseaux Wi-Fi.

Etat de l'art

Dans le Chapitre 1, nous explorons les fondements et l'état de l'art des thématiques essentielles à cette thèse.

Tout d'abord nous nous sommes intéressés aux applications à faible latence. Ces applications partagent un besoin commun : une latence de bout en bout minimale et des débits élevés. Dans le cas du cloud gaming et du cloud VR, qui sont au cœur de cette thèse, ces applications nécessitent une fiabilité réseau élevée pour fournir des expériences interactives et immersives. Leur fonctionnement repose sur des architectures et des protocoles complexes, où des facteurs comme le délai, la gigue, le débit et la perte de paquets jouent un rôle critique. De nombreuses études mettent en évidence l'impact de ces paramètres réseau sur la qualité d'expérience utilisateur.

Ensuite, nous avons présenté les avancées dans les réseaux à capacité variable, à savoir les réseaux cellulaires et Wi-Fi. L'évolution des réseaux cellulaires, de la 1G à la 5G, a permis des avancées significatives. La 5G se distingue par une latence ultra-faible, une bande passante accrue et des concepts innovants tels que le *network slicing* ou le calcul en périphérie du réseau (*edge computing*). Malgré ces avancées, les réseaux cellulaires continuent de faire face à des défis comme l'instabilité de la latence et l'indisponibilité de la bande passante, qui affectent les performances des applications à faible latence. Le même constat a été effectué avec les réseaux Wi-Fi où les normes, de l'IEEE 802.11 au Wi-Fi 6E, ont considérablement amélioré le débit et réduit la latence. Cependant, des limitations subsistent, notamment des interférences, des congestions, et des problématiques liées aux environnements denses qui contribuent à dégrader l'expérience utilisateur.

Les techniques d'apprentissage automatique ou profond, que nous utilisons pour répondre aux problématiques de cette thèse, et qui constituent une des thématiques clés sont présentées. Ces techniques jouent un rôle clé dans la résolution de problèmes complexes dans divers domaines, y compris les réseaux. L'apprentissage profond, basé sur les réseaux de neurones, est

particulièrement performant pour capturer des motifs complexes dans les données. Il est utilisé selon plusieurs paradigmes d'apprentissage : supervisé, non supervisé, ou auto-supervisé. La conception d'une solution d'apprentissage profond implique plusieurs étapes à savoir la préparation des données, la conception du modèle, l'entraînement, l'évaluation et le déploiement qui garantissent la performance et la généralisation nécessaires pour des scénarios réels.

Nous nous sommes ensuite intéressés à la détection d'anomalies dans les séries temporelles qui vise à identifier les déviations par rapport aux schémas normaux, et qui est ainsi cruciale pour les systèmes de surveillance, notamment les réseaux. La littérature montre que ce problème est majoritairement abordé par des approches d'apprentissage non supervisées telles que les modèles statistiques, les techniques de regroupement (*clustering*) ou des approches basées sur la reconstruction ou la prévision. Des progrès récents à base d'apprentissage auto-supervisé ou d'apprentissage contrastif, ont amélioré les capacités de détection d'anomalies. L'évaluation de ces techniques repose sur des métriques comme le F1-score, le MCC ou l'AU-PR.

La dernière thématique clé de cette thèse est le diagnostic des causes racines. Il vise à identifier les facteurs spécifiques responsables des dégradations de performance sur les réseaux à capacité variable. Traditionnellement, le diagnostic reposait sur des méthodes basées sur l'expertise humaine. Cependant, ces approches s'avèrent aujourd'hui inefficaces en raison de l'énorme quantité de données à traiter et sont désormais remplacées par des solutions automatisées et pilotées par l'apprentissage automatique telles que le clustering, les méthodes statistiques, les modèles d'apprentissage supervisé classiques ou profond.

Collecte de données sur les applications à faible latence

Dans le Chapitre 2, nous présentons en détail les méthodologies et les expérimentations mises en place pour collecter des données essentielles à l'évaluation des modèles d'apprentissage automatique destinés à la détection d'anomalies et au diagnostic des causes racines dans les applications à faible latence. Ces données constituent la base des analyses et des modèles développés dans les chapitres suivants.

Tout d'abord un banc de test a été mis en place pour collecter des fichiers d'opportunités de transmission permettant d'effectuer des expérimentations reproductibles sur les réseaux à capacité variable. Nous avons ainsi collecté des fichiers permettant de reproduire les conditions 4G du réseau opérationnel mobile d'Orange en utilisant un outil de génération appelé *Saturator*, sous différents niveaux de qualité de connectivité 4G.

A partir de ces fichiers, un banc de test a été mis en place pour collecter des données sur trois (03) plateformes commerciales de cloud gaming (*Stadia* de Google, *GeForce Now* de NVIDIA et *XCloud* de Microsoft) en conditions de réseau 4G émulées en utilisant l'outil *Mahimahi-Linkshell* et l'outil open-source *DECAF*. Nous avons ainsi collecté des séries temporelles multivariées contenant des métriques réseaux et applicatives reflétant les performances des plateformes de cloud gaming sous différents scénarios.

Le troisième et dernier banc de test développé était dédié aux applications de Cloud VR opérant sous des réseaux Wi-Fi. Ces expérimentations ont inclus à la fois des conditions normales et des scénarios de dégradation émulés en laboratoire, tels que des interférences et une atténuation du signal. Les métriques collectées comprennent des métriques applicatives issues des outils Oculus OVR Metrics et CloudXR de NVIDIA, incluant des indicateurs de performance de la session de Cloud VR, des métriques réseau capturées directement depuis le routeur Livebox, et des données réseaux brutes PCAP collectées à l'aide de sondes placées dans des cages de Faraday près des casques VR pour analyser les interactions au niveau des paquets entre les

clients VR et les serveurs.

Ces jeux de données représentent des ressources précieuses à l'étude des performances des applications à faible latence. Ils permettent de modéliser des conditions réseau réalistes, d'évaluer et de comparer les performances des modèles de détection d'anomalies et de diagnostic des causes racines et de tester la généralisabilité des solutions proposées dans cette thèse.

Evaluation de modèles non-supervisé pour la détection d'anomalies

Le Chapitre 3 s'intéresse à l'application et l'évaluation de modèles d'apprentissage non-supervisé pour la détection d'anomalies dans les séries temporelles multivariées. L'objectif principal est de comparer différentes approches pour identifier les anomalies dans les indicateurs de performance des applications à faible latence, en mettant un accent particulier sur l'efficacité de détection de ces méthodes, leur robustesse à la contamination de données et leur efficacité en temps de calcul et d'inférence.

Plusieurs modèles d'apprentissage non-supervisé existent dans la littérature, mais nous avons evalué dans ce chapitre les plus utilisés, à savoir les forêts d'isolation (*Isolation Forest*), les algorithmes de classification à une classe (*OC-SVM, Deep-SVDD*) et les approches de types reconstruction à base d'analyse en composantes principales (*PCA*) ou d'autoencodeurs (*AE, VAE, DAGMM, USAD*). Afin d'effectuer une évaluation juste et cohérente de ces algorithmes, une approche nommée *WAD* a été introduite pour améliorer l'évaluation et la performance des algorithmes a été mesurée avec des métriques comme le F1-score et le MCC.

Cette évaluation a ainsi montré que les modèles comme l'Isolation Forest, DAGMM et USAD présentent de bonnes performances pour détecter des anomalies. Aussi, nous avons noté que les modèles voient leur performance se dégrader en présence de données d'entraînement contaminées. Un autre résultat notable est que le modèle le plus rapide à l'entraînement et à l'inférence est le PCA, qui n'a malheureusement pas d'aussi bonnes performances en détection d'anomalies que DAGMM et USAD, qui eux ont une efficacité de calcul modérée.

Ce chapitre est conclu par des recommandations pratiques sur l'utilisation des modèles d'apprentissage non supervisé pour la détection d'anomalies dans des scénarios réels.

Une nouvelle approche de détection d'anomalies par apprentissage contrastif

Le Chapitre 4 présente CATS (*Contrastive Anomaly detection on Time Series*), une approche non-supervisée d'apprentissage conçu pour améliorer la détection d'anomalies dans les séries temporelles multivariées. CATS est une contribution majeure de cette thèse, apportant une approche innovante basée sur l'apprentissage contrastif pour surmonter les limitations des méthodes existantes en termes de robustesse à la contamination des données, et de performance sur les séries temporelles. CATS repose sur plusieurs innovations clés :

- L'introduction d'une nouvelle fonction de perte contrastive utilisant la similarité temporelles basée sur la métrique temporelle DTW (*Dynamic Time Warping*). Cette fonction de perte contribue à regrouper dans une portion de l'espace latent les fenêtres temporelles similaires d'un point de vue temporel tout en éloignant les autres ;
- La génération d'anomalies synthétiques dans le jeu d'entraînement à l'aide de techniques de transformation de données telles que le masquage et le bruitage, renforçant ainsi la robustesse du modèle face aux données contaminées ;

- CATS est agnostique à l'architecture neuronale utilisée pour encoder les séries temporelles. Il peut être combiné avec des architectures comme des autoencodeurs, des réseaux de neurones récurrents, ou des Transformers pour capturer des dépendances locales et globales.

Evalué en utilisant des jeux de données publics et nos données collectées précédemment pour le cloud gaming, CATS montre qu'il surpasse les modèles de l'état de l'art en terme de précision et aussi de robustesse à la contamination de données. La principale limite de CATS reste son coût computationnel élevé en raison de la métrique DTW qui limite son applicabilité et aussi l'étendue de son efficacité.

Diagnostic des causes racines dans le cloud en réalité virtuelle

Le Chapitre 5 se concentre sur le diagnostic des causes racines dans les applications de réalité virtuelle en cloud fonctionnant sur des réseaux Wi-Fi. Il propose un algorithme à deux étages pour détecter et diagnostiquer les dégradations de performance, en combinant la détection d'anomalies avec un classifieur supervisé. Cette approche vise à relever les défis liés à l'identification des causes spécifiques des dégradations dans des environnements réseau dynamiques.

Les deux étages de cette approche sont les suivantes:

- La détection d'anomalies en utilisant le modèle CATS (présenté au chapitre précédent) qui donnera un score d'anomalies qui servira à l'étape suivante.

- La classification des causes à l'aide d'un classifier supervisé tel qu'une machine à vecteurs de supports (*Support Vector Machine*) pour mapper les anomalies précédemment détectées à des classes spécifiques de causes (par exemple les interférence ou l'atténuation).

Les expérimentations ont été réalisées en utilisant les données collectées dans le Chapitre 2 sur le cloud VR, notamment les métriques réseau capturées au niveau de la Livebox. Les résultats montrent que notre approche surpasse les approches de classification de la littérature dans le diagnostic des causes de dégradation dans le cloud VR. Elle démontre aussi de bonnes performances même dans les scénarios où peu de données étiquetées sont disponibles grâce à l'efficacité du détecteur d'anomalies. Cette approche modulaire présente aussi une faible complexité computationnelle la rendant applicable à des déploiements en temps réel.

Conclusion et perspectives

Cette thèse a abordé le problème de la détection d'anomalies et le diagnostic des applications à faible latence. Cependant, de nouvelles opportunités de recherche se dessinent pour approfondir et élargir les contributions présentées. Tout d'abord, des collectes de données supplémentaires pourraient être réalisées dans des environnements Cloud VR, en explorant des scénarios de dégradation de la qualité Wi-Fi plus complexes ou en effectuant des collectes à large échelle dans des réseaux domestiques ou d'entreprise. Ensuite, l'application de notre solution de diagnostic proposée à des réseaux 5G, où les dégradations sont spécifiques, permettrait d'évaluer leur généralisation dans d'autres environnements.

Par ailleurs, l'intégration de multiples sources de données (par exemple, les données PCAP, les métriques client et celles des points d'accès) grâce à des approches d'apprentissage multimodales pourrait améliorer le diagnostic. Enfin, des techniques comme la découverte de classes

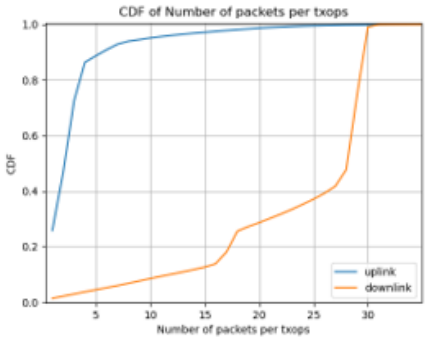
inconnues et la découverte causale pourraient être explorées pour détecter des causes de dégradations encore non identifiées et comprendre les relations causales entre les métriques réseau, renforçant ainsi l'explicabilité des solutions proposées.

Characterization of Orange 4G transmission opportunities

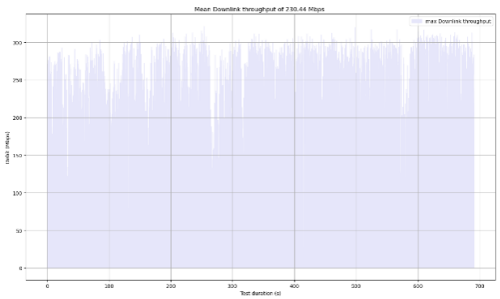
This appendix provides a comprehensive presentation of the transmission opportunity (txops) files collected during the experiments described in Chapter 2. These files capture Orange commercial 4G network characteristics and are available as Open Data [27]. For each of the six (06) txops files, two figures are included:

- **The Cumulative Distribution Function** illustrating the distribution of the number of packets per txops.
- **The throughput analysis**, which highlights the data rate behavior across the network conditions.

By visualizing these metrics, the appendix aims to offer insights into the network conditions observed during the experiments, serving as a valuable resource for understanding the basis of the datasets used throughout this thesis. Each section corresponds to one txops file, presenting both figures side by side for clarity and easy comparison.



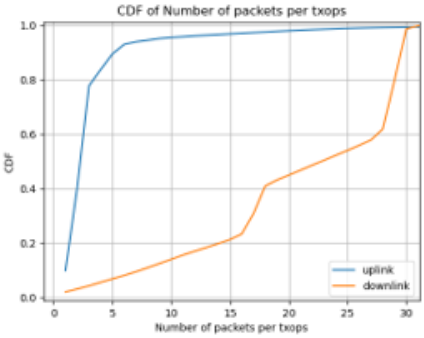
(a) CDF of the number of packets per txops



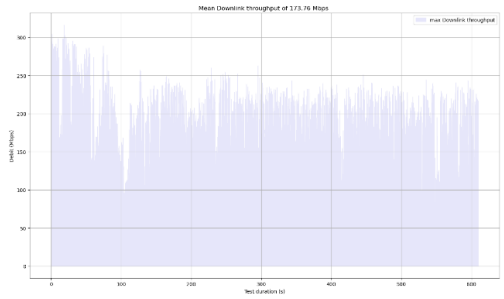
(b) Downlink throughput capacity

Figure A.1: Txops File E.

[27] https://cloud-gaming-traces.lhs.loria.fr/data.html

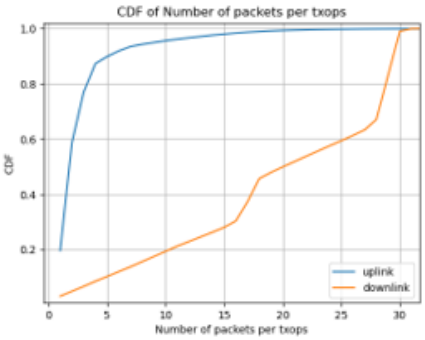


(a) CDF of the number of packets per txops

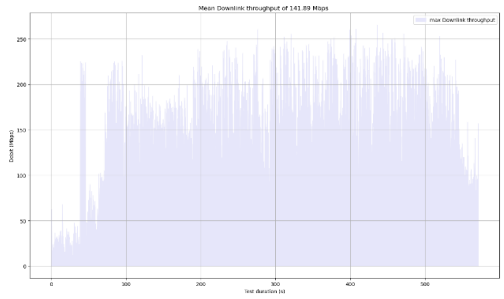


(b) Downlink throughput capacity

Figure A.2: Txops File D.

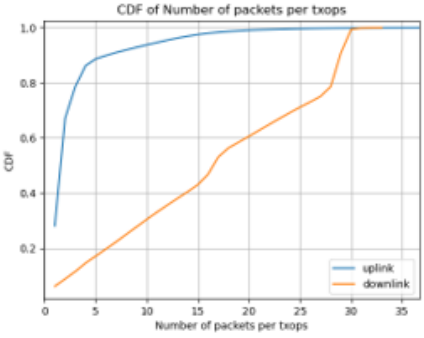


(a) CDF of the number of packets per txops

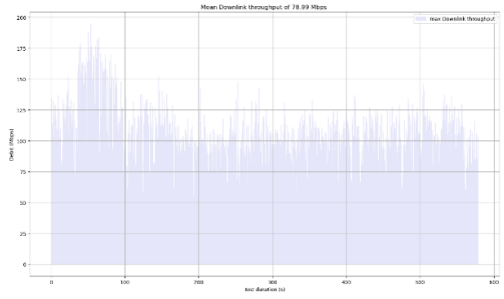


(b) Downlink throughput capacity

Figure A.3: Txops File C.

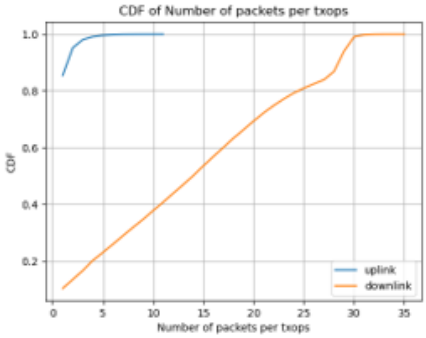


(a) CDF of the number of packets per txops

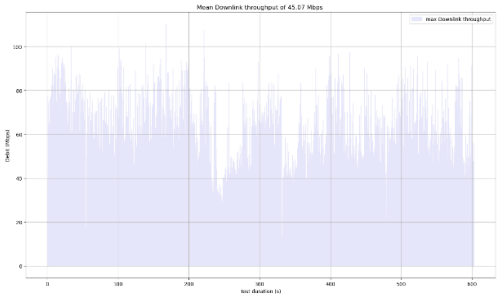


(b) Downlink throughput capacity

Figure A.4: Txops File B.

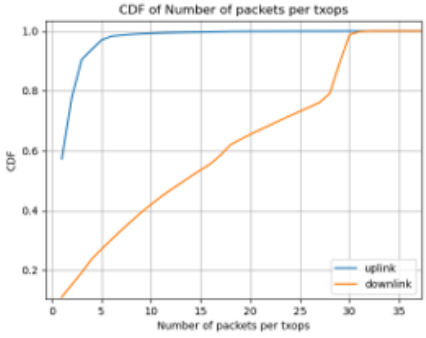


(a) CDF of the number of packets per txops

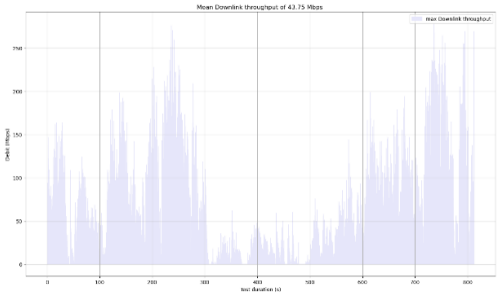


(b) Downlink throughput capacity

Figure A.5: Txops File A.



(a) CDF of the number of packets per txops



(b) Downlink throughput capacity

Figure A.6: Txops File Highway.

Supplemental results for Chapter 4

Table B.1 presents the anomaly detection performance of CATS compared to other methods, evaluated using the AUPR (Eq. 1.8), AUC (Eq. 1.7), F1 (Eq. 1.4), and MCC (Eq. 1.5) metrics on the cloud gaming datasets collected in Chapter 2. We applied the WAD metric approach proposed in Chapter 3. These results demonstrate that CATS outperforms other methods, consistent with the findings presented in Chapter 4.

Table B.1: Performance comparison on the datasets using WAD metric approach. Mean and standard deviation computed over five (5) runs. Bold values indicate best results.

	Models	IForest	Deep-SVDD	AE	USAD	SimCLR	SimSiam	TS2Vec	CATS
STD	**AUPR**	69.58$_{(\pm 1.25)}$	97.77$_{(\pm 0.46)}$	98.52$_{(\pm 0.12)}$	98.62$_{(\pm 0.03)}$	98.52$_{(\pm 0.25)}$	74.51$_{(\pm 8.48)}$	99.10$_{(\pm 0.41)}$	**99.59**$_{(\pm 0.04)}$
	AUC	74.38$_{(\pm 1.60)}$	96.66$_{(\pm 0.78)}$	98.09$_{(\pm 0.18)}$	98.27$_{(\pm 0.04)}$	98.04$_{(\pm 0.37)}$	74.61$_{(\pm 10.7)}$	98.76$_{(\pm 0.72)}$	**99.60**$_{(\pm 0.05)}$
	F1	70.82$_{(\pm 1.74)}$	93.34$_{(\pm 0.60)}$	94.07$_{(\pm 0.23)}$	94.00$_{(\pm 0.05)}$	94.93$_{(\pm 0.29)}$	69.45$_{(\pm 9.42)}$	96.02$_{(\pm 0.22)}$	**96.65**$_{(\pm 0.10)}$
	MCC	37.71$_{(\pm 3.72)}$	87.33$_{(\pm 0.99)}$	88.53$_{(\pm 0.38)}$	88.79$_{(\pm 0.08)}$	90.31$_{(\pm 0.47)}$	37.82$_{(\pm 18.1)}$	92.04$_{(\pm 0.32)}$	**93.15**$_{(\pm 0.18)}$
GFN	**AUPR**	64.37$_{(\pm 1.36)}$	83.89$_{(\pm 1.30)}$	84.05$_{(\pm 0.66)}$	85.10$_{(\pm 0.55)}$	87.68$_{(\pm 0.72)}$	78.09$_{(\pm 6.20)}$	86.57$_{(\pm 2.93)}$	**92.96**$_{(\pm \mathbf{0.54})}$
	AUC	65.50$_{(\pm 1.29)}$	78.18$_{(\pm 2.24)}$	78.57$_{(\pm 0.86)}$	80.80$_{(\pm 0.87)}$	83.82$_{(\pm 1.12)}$	72.05$_{(\pm 4.36)}$	80.82$_{(\pm 4.17)}$	**90.87**$_{(\pm \mathbf{0.82})}$
	F1	63.89$_{(\pm 1.61)}$	70.50$_{(\pm 1.81)}$	70.35$_{(\pm 1.02)}$	72.48$_{(\pm 1.46)}$	75.35$_{(\pm 0.89)}$	66.02$_{(\pm 2.21)}$	74.17$_{(\pm 4.53)}$	**83.56**$_{(\pm \mathbf{0.83})}$
	MCC	25.16$_{(\pm 2.71)}$	41.91$_{(\pm 3.74)}$	41.70$_{(\pm 1.92)}$	45.06$_{(\pm 1.87)}$	51.61$_{(\pm 1.75)}$	31.53$_{(\pm 6.86)}$	49.22$_{(\pm 8.86)}$	**67.73**$_{(\pm \mathbf{1.35})}$
XC	**AUPR**	62.40$_{(\pm 1.62)}$	62.36$_{(\pm 7.88)}$	77.72$_{(\pm 4.53)}$	78.71$_{(\pm 0.50)}$	84.20$_{(\pm 3.18)}$	71.34$_{(\pm 13.9)}$	87.43$_{(\pm 2.41)}$	**92.42**$_{(\pm 0.76)}$
	AUC	82.30$_{(\pm 0.94)}$	79.25$_{(\pm 8.07)}$	91.93$_{(\pm 1.96)}$	91.91$_{(\pm 0.31)}$	92.93$_{(\pm 2.02)}$	85.87$_{(\pm 10.3)}$	96.49$_{(\pm 0.70)}$	**97.92**$_{(\pm 0.21)}$
	F1	58.04$_{(\pm 1.99)}$	58.63$_{(\pm 7.72)}$	73.73$_{(\pm 3.11)}$	72.51$_{(\pm 0.59)}$	76.22$_{(\pm 3.45)}$	63.77$_{(\pm 13.4)}$	81.84$_{(\pm 1.99)}$	**86.62**$_{(\pm 0.91)}$
	MCC	45.61$_{(\pm 2.03)}$	46.73$_{(\pm 8.19)}$	65.31$_{(\pm 4.13)}$	63.77$_{(\pm 0.93)}$	69.13$_{(\pm 4.24)}$	53.36$_{(\pm 16.2)}$	76.17$_{(\pm 2.66)}$	**82.48**$_{(\pm 1.19)}$

BIBLIOGRAPHY

[1] Marché des communications électroniques en France - Les chiffres au 2ème trimestre 2024 | Arcep — arcep.fr. https://www.arcep.fr/cartes-et-donnees/nos-publications-chiffrees/observatoire-des-marches-des-communications-electroniques-en-france/t2-2024.html. [Accessed 08-12-2024].

[2] Cloud Gaming Market Size, Share & Industry Statistics Report, 2032 — fortunebusinessinsights.com. https://www.fortunebusinessinsights.com/cloud-gaming-market-102495. [Accessed 10-12-2024].

[3] Virtual Reality (VR) Market Analysis | 2024-2030 — nextmsc.com. https://www.nextmsc.com/report/virtual-reality-market. [Accessed 10-12-2024].

[4] Flavien Ronteix et al. *Réduction de la latence et de la gigue dans le réseau d'accès radio 5G*. PhD thesis, Ecole nationale supérieure Mines-Télécom Atlantique Bretagne Pays de la Loire, 2022.

[5] Google wants to reduce Stadia lag with 'negative latency' — engadget.com. https://www.engadget.com/2019-10-10-google-stadia-negative-latency.html. [Accessed 10-12-2024].

[6] Xavier Marchal, Philippe Graff, Joël Roman Ky, Thibault Cholez, Stéphane Tuffin, Bertrand Mathieu, and Olivier Festor. An analysis of cloud gaming platforms behaviour under synthetic network constraints and real cellular networks conditions. *Journal of Network and Systems Management*, 31(2):39, 2023.

[7] Joël Roman Ky, Bertrand Mathieu, Abdelkader Lahmadi, and Raouf Boutaba. Assessing unsupervised machine learning solutions for anomaly detection in cloud gaming sessions. *2022 18th International Conference on Network and Service Management (CNSM)*, pages 367–373, 2022.

[8] Joël Roman Ky, Bertrand Mathieu, Abdelkader Lahmadi, and Raouf Boutaba. Ml models for detecting qoe degradation in low-latency applications: A cloud-gaming case study. *IEEE Transactions on Network and Service Management*, 20:2295–2308, 2023.

[9] Joël Roman Ky, Bertrand Mathieu, Abdelkader Lahmadi, and Raouf Boutaba. Cats: Contrastive learning for anomaly detection in time series. In *2024 IEEE International Conference on Big Data (BigData)*, pages 1352–1359. IEEE, 2024.

[10] Wikipedia contributors. Onlive — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=OnLive&oldid=1249751721, 2024. [Online; accessed 29-November-2024].

[11] Wikipedia contributors. Gaikai — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Gaikai&oldid=1255763675, 2024. [Online; accessed 29-November-2024].

[12] Google Stadia. https://stadia.google.com/.

[13] Xbox Cloud Gaming (Beta). `https://www.xbox.com/en-US/cloud-gaming`.

[14] NVIDIA GeForce NOW. `https://www.nvidia.com/en-us/geforce-now/`.

[15] Amazon Luna Cloud Gaming. `https://luna.amazon.com/`.

[16] Hassan Iqbal, Ayesha Khalid, and Muhammad Shahzad. Dissecting cloud gaming performance with decaf. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 5:1 – 27, 2021.

[17] Antonio Di Domenico, Gianluca Perna, Martino Trevisan, Luca Vassio, and Danilo Giordano. A network analysis on cloud gaming: Stadia, geforce now and psnow. *ArXiv*, abs/2012.06774, 2021.

[18] Patrick Le Callet, Sebastian Möller, Andrew Perkis, Kjell Brunnström, Sergio Beker, Katrien De Moor, Ann Dooms, Sebastian Egger, Marie-Neige Garcia, Tobias Hoßfeld, et al. *Qualinet white paper on definitions of quality of experience*. PhD thesis, Qualinet (www. qualinet. eu), 2013.

[19] G. 1032 Itu-T Recommendation. *Influence Factors on Gaming Quality of Experience*. International Telecommunication Union, Geneva, 2017.

[20] Teemu Kämäräinen, Matti Siekkinen, Antti Ylä-Jääski, Wenxiao Zhang, and Pan Hui. A measurement study on achieving imperceptible latency in mobile cloud gaming. *Proceedings of the 8th ACM on Multimedia Systems Conference*, 2017.

[21] Mark Claypool and David Finkel. The effects of latency on player performance in cloud-based games. *2014 13th Annual Workshop on Network and Systems Support for Games*, pages 1–6, 2014.

[22] Oswaldo Sebastian Peñaherrera-Pulla, Carlos Baena, Sergio Fortes, Eduardo Baena, and Raquel Barco. Measuring key quality indicators in cloud gaming: Framework and assessment over wireless networks. *Sensors*, 21(4):1387, 2021.

[23] Xiaokun Xu and Mark Claypool. Measurement of the responses of cloud-based game streaming to network congestion. *Proceedings of the 32nd Workshop on Network and Operating Systems Support for Digital Audio and Video*, 2022.

[24] Franck Aumont, Frédérique Humbert, Christoph Neumann, Charles Salmon-Legagneur, and Charline Taibi. Dissecting cloud game streaming platforms regarding the impacts of video encoding and networking constraints on qoe. *Proceedings of the Workshop on Game Systems* (*GameSys '21*), 2021.

[25] Michael Oliveira, Thomas Flanagan, and Carter Nakagawa. Effects of jitter on quality of experience in cloud gaming. Technical report, Worcester Polytechnic Institute, March 2023.

[26] Mirko Suznjevic, Ivan Slivar, and Lea Skorin-Kapov. Analysis and qoe evaluation of cloud gaming service adaptation under different network conditions: The case of nvidia geforce now. In *2016 Eighth International Conference on Quality of Multimedia Experience* (*QoMEX*), pages 1–6. IEEE, 2016.

[27] Michael Jarschel, Daniel Schlosser, Sven Scheuring, and Tobias Hoßfeld. Gaming in the clouds: Qoe and the users' perspective. *Mathematical and computer modelling*, 57(11-12):2883–2894, 2013.

[28] Kuan-Ta Chen, Yu-Chun Chang, Hwai-Jung Hsu, De-Yu Chen, Chun-Ying Huang, and Cheng-Hsin Hsu. On the quality of service of cloud gaming systems. *IEEE Transactions on Multimedia*, 16(2):480–495, 2013.

[29] Huawei iLab. Cloud vr network solution white paper. `https://www-file.huawei.com/-/media/corporate/pdf/ilab/2018/cloud_vr_network_solution_white_paper_2018_en_v1.pdf`. [Accessed 27-09-2024].

[30] G. 1035 Itu-T Recommendation. *Influencing factors on quality of experience for virtual reality services*. International Telecommunication Union, Geneva, 2021.

[31] Jiarun Song, Xionghui Mao, and Fuzheng Yang. The impact of black edge artifact on qoe of the fov-based cloud vr services. *IEEE Transactions on Multimedia*, 25:8020–8035, 2022.

[32] Kuan-Yu Lee, Ashutosh Singla, Pablo Cesar, and Cheng-Hsin Hsu. Adaptive cloud vr gaming optimized by gamer qoe models. *ACM Transactions on Multimedia Computing, Communications and Applications*, 2024.

[33] Maximilian Warsinke, Tanja Kojić, Maurizio Vergari, Jan-Niklas Voigt-Antons, and Sebastian Möller. Vr cloud gaming ux: Exploring the impact of network quality on emotion, presence, game experience and cybersickness. *arXiv preprint arXiv:2408.12238*, 2024.

[34] Yen-Chun Li, Chia-Hsin Hsu, Yu-Chun Lin, and Cheng-Hsin Hsu. Performance measurements on a cloud vr gaming platform. In *Proceedings of the 1st Workshop on Quality of Experience (QoE) in Visual Multimedia Applications*, pages 37–45, 2020.

[35] Henrique Souza Rossi, Karan Mitra, Samuel Larsson, Christer Åhlund, and Irina Cotanis. Subjective qoe assessment for virtual reality cloud-based first-person shooter game. In *ICC 2024-IEEE International Conference on Communications*, pages 4698–4703. IEEE, 2024.

[36] Minimum technical performance requirements for imt-2020 radio interface(s). `https://itu.int/en/ITU-R/study-groups/rsg5/rwp5d/imt-2020/Documents/S01-1_Requirements%20for%20IMT-2020_Rev.pdf`.

[37] Zhaowei Tan, Yuanjie Li, Qianru Li, Zhehui Zhang, Zhehan Li, and Songwu Lu. Supporting mobile vr in lte networks: How close are we? *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 2(1):1–31, 2018.

[38] Sandeepa Bhuyan, Shulin Zhao, Ziyu Ying, Mahmut T Kandemir, and Chita R Das. End-to-end characterization of game streaming applications on mobile platforms. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 6(1):1–25, 2022.

[39] OS Peñaherrera-Pulla, Sebastian Bro Damsgaard, Boyan Yanakiev, Preben Mogensen, Sergio Fortes, and Raquel Barco. Cloud vr on 5g: A performance validation in industrial scenarios. *IEEE Open Journal of the Communications Society*, 2024.

[40] Gabriele Maiorano, Gabriele Proietti Mattia, and Roberto Beraldi. Local and remote fog based trade-offs for qoe in vr applications by using cloudxr and oculu air link. In *2022*

international conference on edge computing and applications (*ICECAA*), pages 95–101. IEEE, 2022.

[41] Huanle Zhang, Ahmed Elmokashfi, and Prasant Mohapatra. Wifi and multiple interfaces: Adequate for virtual reality? In *2018 IEEE 24th International Conference on Parallel and Distributed Systems* (*ICPADS*), pages 220–227. IEEE, 2018.

[42] Matthijs Jansen, Jesse Donkervliet, Animesh Trivedi, and Alexandru Iosup. Can my wifi handle the metaverse? a performance evaluation of meta's flagship virtual reality hardware. In *Companion of the 2023 ACM/SPEC International Conference on Performance Engineering*, pages 297–303, 2023.

[43] Jaime Burbano, Jorge Pillco, Sebastian Genovez, and Patricia Ortega. Impact of network factors on qoe-sensitive virtual reality environments: An experimental analysis perspective using vr-based mindfulness. In *2024 IEEE International Conference on Artificial Intelligence and eXtended and Virtual Reality* (*AIxVR*), pages 367–374. IEEE, 2024.

[44] Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE computer society conference on computer vision and pattern recognition* (*CVPR'05*), volume 1, pages 539–546. IEEE, 2005.

[45] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[46] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[47] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[48] Lukas Ruff, Jacob R Kauffmann, Robert A Vandermeulen, Grégoire Montavon, Wojciech Samek, Marius Kloft, Thomas G Dietterich, and Klaus-Robert Müller. A unifying review of deep and shallow anomaly detection. *Proceedings of the IEEE*, 109(5):756–795, 2021.

[49] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys* (*CSUR*), 41(3):1–58, 2009.

[50] Hui Luo and Shisheng Zhong. Gas turbine engine gas path anomaly detection using deep learning with gaussian distribution. In *2017 Prognostics and System Health Management Conference* (*PHM-Harbin*), pages 1–6. IEEE, 2017.

[51] Asrul H Yaacob, Ian KT Tan, Su Fong Chien, and Hon Khi Tan. Arima based network anomaly detection. In *2010 Second International Conference on Communication Software and Networks*, pages 205–209. IEEE, 2010.

[52] Van Loi Cao, Miguel Nicolau, and James McDermott. One-class classification for anomaly detection with kernel density estimation and genetic programming. In *Genetic Programming: 19th European Conference, EuroGP 2016, Porto, Portugal, March 30-April 1, 2016, Proceedings 19*, pages 3–18. Springer, 2016.

[53] Florian Knorn and Douglas J Leith. Adaptive kalman filtering for anomaly detection in software appliances. In *IEEE INFOCOM Workshops 2008*, pages 1–6. IEEE, 2008.

[54] Wanpracha Art Chaovalitwongse, Ya-Ju Fan, and Rajesh C Sachdeo. On the time series k-nearest neighbor classification of abnormal brain activity. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 37(6):1005–1016, 2007.

[55] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 93–104, 2000.

[56] Istvan Kiss, Béla Genge, Piroska Haller, and Gheorghe Sebestyén. Data clustering-based anomaly detection in industrial control systems. In *2014 IEEE 10th International Conference on Intelligent Computer Communication and Processing (ICCP)*, pages 275–281. IEEE, 2014.

[57] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, page 226–231. AAAI Press, 1996.

[58] Randy Paffenroth, Kathleen Kay, and Les Servi. Robust pca for anomaly detection in cyber networks. *arXiv preprint arXiv:1801.01571*, 2018.

[59] Li Zonglin, Hu Guangmin, and Yao Xingmiao. Multi-dimensional traffic anomaly detection based on ica. In *2009 IEEE Symposium on Computers and Communications*, pages 333–336. IEEE, 2009.

[60] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 eighth ieee international conference on data mining*, pages 413–422. IEEE, 2008.

[61] Bernhard Schölkopf, Robert C Williamson, Alex Smola, John Shawe-Taylor, and John Platt. Support vector method for novelty detection. *Advances in neural information processing systems*, 12, 1999.

[62] Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeki Cho, and Haifeng Chen. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International conference on learning representations*, 2018.

[63] Chuxu Zhang, Dongjin Song, Yuncong Chen, Xinyang Feng, Cristian Lumezanu, Wei Cheng, Jingchao Ni, Bo Zong, Haifeng Chen, and Nitesh V. Chawla. A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):1409–1416, Jul. 2019.

[64] Julien Audibert, Pietro Michiardi, Frédéric Guyard, Sébastien Marti, and Maria A Zuluaga. Usad: Unsupervised anomaly detection on multivariate time series. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 3395–3404, 2020.

[65] Haowen Xu, Wenxiao Chen, Nengwen Zhao, Zeyan Li, Jiahao Bu, Zhihan Li, Ying Liu, Youjian Zhao, Dan Pei, Yang Feng, et al. Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In *Proceedings of the 2018 world wide web conference*, pages 187–196, 2018.

[66] Daehyung Park, Yuuna Hoshi, and Charles C Kemp. A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder. *IEEE Robotics and Automation Letters*, 3(3):1544–1551, 2018.

[67] Ya Su, Youjian Zhao, Chenhao Niu, Rong Liu, Wei Sun, and Dan Pei. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2828–2837, 2019.

[68] Jiehui Xu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Anomaly transformer: Time series anomaly detection with association discrepancy. In *International Conference on Learning Representations*, 2022.

[69] Shreshth Tuli, Giuliano Casale, and Nicholas R Jennings. Tranad: Deep transformer networks for anomaly detection in multivariate time series data. *arXiv preprint arXiv:2201.07284*, 2022.

[70] Dan Li, Dacheng Chen, Baihong Jin, Lei Shi, Jonathan Goh, and See-Kiong Ng. Mad-gan: Multivariate anomaly detection for time series data with generative adversarial networks. In *International conference on artificial neural networks*, pages 703–716. Springer, 2019.

[71] Bin Zhou, Shenghua Liu, Bryan Hooi, Xueqi Cheng, and Jing Ye. Beatgan: Anomalous rhythm detection using adversarially generated time series. In *IJCAI*, volume 2019, pages 4433–4439, 2019.

[72] Jonathan Goh, Sridhar Adepu, Marcus Tan, and Zi Shan Lee. Anomaly detection in cyber physical systems using recurrent neural networks. In *2017 IEEE 18th International Symposium on High Assurance Systems Engineering (HASE)*, pages 140–145. IEEE, 2017.

[73] Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Soderstrom. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 387–395, 2018.

[74] Wentai Wu, Ligang He, Weiwei Lin, Yi Su, Yuhua Cui, Carsten Maple, and Stephen Jarvis. Developing an unsupervised real-time anomaly detection scheme for time series with multi-seasonality. *IEEE Transactions on Knowledge and Data Engineering*, 34(9):4147–4160, 2020.

[75] Mohsin Munir, Shoaib Ahmed Siddiqui, Andreas Dengel, and Sheraz Ahmed. Deepant: A deep learning approach for unsupervised anomaly detection in time series. *Ieee Access*, 7:1991–2005, 2018.

[76] Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4393–4402. PMLR, 10–15 Jul 2018.

[77] Hongzuo Xu, Guansong Pang, Yijie Wang, and Yongjun Wang. Deep isolation forest for anomaly detection. *IEEE Transactions on Knowledge and Data Engineering*, 35(12):12591–12604, 2023.

[78] Hongzuo Xu, Yijie Wang, Songlei Jian, Qing Liao, Yongjun Wang, and Guansong Pang. Calibrated one-class classification for unsupervised time series anomaly detection. *IEEE Transactions on Knowledge and Data Engineering*, 2024.

[79] Rui Wang, Chongwei Liu, Xudong Mou, Kai Gao, Xiaohui Guo, Pin Liu, Tianyu Wo, and Xudong Liu. Deep contrastive one-class time series anomaly detection. In *Proceedings of the 2023 SIAM International Conference on Data Mining (SDM)*, pages 694–702. SIAM, 2023.

[80] Bin Li and Emmanuel Müller. Contrastive time series anomaly detection by temporal transformations. In *2023 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2023.

[81] HyunGi Kim, Siwon Kim, Seonwoo Min, and Byunghan Lee. Contrastive time-series anomaly detection. *IEEE Transactions on Knowledge and Data Engineering*, 2023.

[82] Yang Jiao, Kai Yang, Dongjing Song, and Dacheng Tao. Timeautoad: Autonomous anomaly detection with self-supervised contrastive loss for multivariate time series. *IEEE Transactions on Network Science and Engineering*, 9(3):1604–1619, 2022.

[83] Zahra Zamanzadeh Darban, Geoffrey I. Webb, Shirui Pan, Charu C. Aggarwal, and Mahsa Salehi. Carla: Self-supervised contrastive representation learning for time series anomaly detection. *Pattern Recognition*, 157:110874, 2025.

[84] Yiyuan Yang, Chaoli Zhang, Tian Zhou, Qingsong Wen, and Liang Sun. Dcdetector: Dual attention contrastive representation learning for time series anomaly detection. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3033–3045, 2023.

[85] Brian W. Matthews. Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et biophysica acta*, 405 2:442–51, 1975.

[86] Davide Chicco and Giuseppe Jurman. The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation. *BMC Genomics*, 21, 2020.

[87] Zhehui Zhang, Shu Shi, Varun Gupta, and Rittwik Jana. Analysis of cellular network latency for edge-based remote rendering streaming applications. In *Proceedings of the ACM SIGCOMM 2019 Workshop on Networking for Emerging Applications and Technologies*, pages 8–14, 2019.

[88] Yueyang Pan, Ruihan Li, and Chenren Xu. The first 5g-lte comparative study in extreme mobility. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 6(1):1–22, 2022.

[89] Dongzhu Xu, Anfu Zhou, Xinyu Zhang, Guixian Wang, Xi Liu, Congkai An, Yiming Shi, Liang Liu, and Huadong Ma. Understanding operational 5g: A first measurement study on its coverage, performance and energy consumption. In *Proceedings of the Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication*, pages 479–494, 2020.

[90] Yiqing Zhou, Ling Liu, Hongyan Du, Lin Tian, Xiaodong Wang, and Jinglin Shi. An overview on intercell interference management in mobile cellular networks: From 2g to

5g. In *2014 IEEE International Conference on Communication Systems*, pages 217–221. IEEE, 2014.

[91] Bob Briscoe, Anna Brunstrom, Andreas Petlund, David Hayes, David Ros, Jyh Tsang, Stein Gjessing, Gorry Fairhurst, Carsten Griwodz, and Michael Welzl. Reducing internet latency: A survey of techniques and their merits. *IEEE Communications Surveys & Tutorials*, 18(3):2149–2196, 2014.

[92] Changhua Pei, Youjian Zhao, Guo Chen, Ruming Tang, Yuan Meng, Minghua Ma, Ken Ling, and Dan Pei. Wifi can be the weakest link of round trip network latency in the wild. In *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*, pages 1–9. IEEE, 2016.

[93] Ioannis Pefkianakis, Henrik Lundgren, Augustin Soule, Jaideep Chandrashekar, Pascal Le Guyadec, Christophe Diot, Martin May, Karel Van Doorselaer, and Koen Van Oost. Characterizing home wireless performance: The gateway view. In *2015 IEEE Conference on Computer Communications (INFOCOM)*, pages 2713–2731. IEEE, 2015.

[94] Ilias Syrigos, Nikos Sakellariou, Stratos Keranidis, and Thanasis Korakis. On the employment of machine learning techniques for troubleshooting wifi networks. In *2019 16th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, pages 1–6. IEEE, 2019.

[95] Kyung-Hwa Kim, Hyunwoo Nam, and Henning Schulzrinne. Wislow: A wi-fi network performance troubleshooting tool for end users. In *IEEE INFOCOM 2014-IEEE Conference on Computer Communications*, pages 862–870. IEEE, 2014.

[96] Elif Dilek Salık, Gökhan Görbilek, Berkcan Okur, and Aysun Gurur Önalan. Non-wifi interference detection and throughput estimation at the wifi edge for 2.4 and 5 ghz bands with machine learning. In *2023 IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom)*, pages 371–378. IEEE, 2023.

[97] Partha Kanuparthy, Constantine Dovrolis, Konstantina Papagiannaki, Srinivasan Seshan, and Peter Steenkiste. Can user-level probing detect and diagnose common home-wlan pathologies. *ACM SIGCOMM Computer Communication Review*, 42(1):7–15, 2012.

[98] Márquez-Israel Salinas. *Home Wi-Fi Impairments*. PhD thesis, UPMC (Paris 6)-Sorbonne Université, 2018.

[99] Shravan Rayanchu, Arunesh Mishra, Dheeraj Agrawal, Sharad Saha, and Suman Banerjee. Diagnosing wireless packet losses in 802.11: Separating collision from weak signal. In *IEEE INFOCOM 2008-The 27th Conference on Computer Communications*, pages 735–743. IEEE, 2008.

[100] Yoshinori Watanabe, Yasuhiko Matsunaga, Kosei Kobayashi, Toshio Tonouchi, Tomohiro Igakura, Shinji Nakadai, and Kenichiro Kamachi. Utran o&m support system with statistical fault identification and customizable rule sets. In *NOMS 2008-2008 IEEE Network Operations and Management Symposium*, pages 560–573. IEEE, 2008.

[101] Cameron Kane. System and method for identifying problems on a network, October 27 2015. US Patent 9,172,593.

[102] Giorgos Dimopoulos, Ilias Leontiadis, Pere Barlet-Ros, Konstantina Papagiannaki, and Peter Steenkiste. Identifying the root cause of video streaming issues on mobile devices. In *Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies*, pages 1–13, 2015.

[103] Kuo-Ming Chen, Tsung-Hui Chang, Kai-Cheng Wang, and Ta-Sung Lee. Machine learning based automatic diagnosis in mobile communication networks. *IEEE Transactions on Vehicular Technology*, 68(10):10081–10093, 2019.

[104] Xuewen Liu, Gang Chuai, Weidong Gao, Kaisa Zhang, and Xiangyu Chen. Kqis-driven qoe anomaly detection and root cause analysis in cellular networks. In *2019 IEEE Globecom Workshops (GC Wkshps)*, pages 1–6. IEEE, 2019.

[105] Xiaofeng Shi, Matthew Osinski, Chen Qian, and Jia Wang. Towards automatic troubleshooting for user-level performance degradation in cellular services. In *Proceedings of the 28th Annual International Conference on Mobile Computing And Networking*, pages 716–728, 2022.

[106] Mah-Rukh Fida, Azza H Ahmed, Thomas Dreibholz, Andrés F Ocampo, Ahmed Elmokashfi, and Foivos I Michelinakis. Bottleneck identification in cloudified mobile networks based on distributed telemetry. *IEEE Transactions on Mobile Computing*, 2023.

[107] Antor Hasan, Conrado Boeira, Khaleda Papry, Yue Ju, Zhongwen Zhu, and Israat Haque. Root cause analysis of anomalies in 5g ran using graph neural network and transformer. *arXiv preprint arXiv:2406.15638*, 2024.

[108] Diego Da Hora, Karel Van Doorselaer, Koen Van Oost, and Renata Teixeira. Predicting the effect of home wi-fi quality on qoe. In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*, pages 944–952. IEEE, 2018.

[109] ARCEP. Qualité des services mobiles. `https://www.arcep.fr/actualites/actualites-et-communiques/detail/n/qualite-des-services-mobiles-191121.html`, 2021. [Accessed 30-09-2024].

[110] Ravi Netravali, Anirudh Sivaraman, Somak Das, Ameesh Goyal, Keith Winstein, James Mickens, and Hari Balakrishnan. Mahimahi: accurate {Record-and-Replay} for {HTTP}. In *2015 USENIX Annual Technical Conference (USENIX ATC 15)*, pages 417–429, 2015.

[111] Bertrand Mathieu and Stéphane Tuffin. Evaluating the l4s architecture in cellular networks with a programmable switch. In *2021 IEEE Symposium on Computers and Communications (ISCC)*, pages 1–6. IEEE, 2021.

[112] James Cannady. Artificial neural networks for misuse detection. In *National information systems security conference*, volume 26, pages 443–456. Baltimore, 1998.

[113] Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. Efficient algorithms for mining outliers from large data sets. In *SIGMOD '00*, 2000.

[114] Goodfellow Ian, Pouget-AbadieJean, MirzaMehdi, Xubing, Warde-FarleyDavid, OzairSherjil, CourvilleAaron, and BengioYoshua. Generative adversarial networks. *Communications of The ACM*, 2020.

[115] Alexander Lavin and Subutai Ahmad. Evaluating real-time anomaly detection algorithms – the numenta anomaly benchmark. *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pages 38–44, 2015.

[116] Wei Cai, Ryan Shea, Chun-Ying Huang, Kuan-Ta Chen, Jiangchuan Liu, Victor C. M. Leung, and Cheng-Hsin Hsu. A survey on cloud gaming: Future of computer games. *IEEE Access*, 4:7605–7620, 2016.

[117] Kjetil Raaen, Ragnhild Eg, and Carsten Griwodz. Can gamers detect cloud delay? *2014 13th Annual Workshop on Network and Systems Support for Games*, pages 1–3, 2014.

[118] Ragnhild Eg, Kjetil Raaen, and Mark Claypool. Playing with delay: With poor timing comes poor performance, and experience follows suit. *2018 Tenth International Conference on Quality of Multimedia Experience (QoMEX)*, pages 1–6, 2018.

[119] Sara Vlahovic, Mirko Suznjevic, and Lea Skorin-Kapov. The impact of network latency on gaming qoe for an fps vr game. *2019 Eleventh International Conference on Quality of Multimedia Experience (QoMEX)*, pages 1–3, 2019.

[120] Sebastian Schmidl, Phillip Wenig, and Thorsten Papenbrock. Anomaly detection in time series: A comprehensive evaluation. *Proc. VLDB Endow.*, 15:1779–1797, 2022.

[121] Jing Ren, Feng Xia, Ye Liu, and Ivan Lee. Deep video anomaly detection: Opportunities and challenges. *2021 International Conference on Data Mining Workshops (ICDMW)*, pages 959–966, 2021.

[122] Andrew A. Cook, Goksel Misirli, and Zhong Fan. Anomaly detection for iot time-series data: A survey. *IEEE Internet of Things Journal*, 7:6481–6494, 2020.

[123] Guansong Pang, Chunhua Shen, Longbing Cao, and Anton van den Hengel. Deep learning for anomaly detection. *ACM Computing Surveys (CSUR)*, 54:1 – 38, 2020.

[124] Kukjin Choi, Jihun Yi, Changhwa Park, and Sungroh Yoon. Deep learning for anomaly detection in time-series data: Review, analysis, and guidelines. *IEEE Access*, 9:120043–120065, 2021.

[125] Ya Liu, Yingjie Zhou, Kai Yang, and Xin Wang. Unsupervised deep learning for iot time series. *IEEE Internet of Things Journal*, pages 1–1, 2023.

[126] D'Jeff Kanda Nkashama, Ariana Soltani, Jean-Charles Verdier, Marc Frappier, Pierre-Marting Tardif, and Froduald Kabanza. Robustness evaluation of deep unsupervised learning algorithms for intrusion detection systems. *ArXiv*, abs/2207.03576, 2022.

[127] Astha Garg, Wenyu Zhang, Jules Samaran, Ramasamy Savitha, and Chuan-Sheng Foo. An evaluation of anomaly detection and diagnosis in multivariate time series. *IEEE Transactions on Neural Networks and Learning Systems*, 33:2508–2517, 2022.

[128] Siwon Kim, Kukjin Choi, Hyun-Soo Choi, Byunghan Lee, and Sungroh Yoon. Towards a rigorous evaluation of time-series anomaly detection. In *AAAI*, 2022.

[129] Won-Seok Hwang, Jeong-Han Yun, Jonguk Kim, and Byung gil Min. "do you know existing accuracy metrics overrate time-series anomaly detections?". *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing*, 2022.

[130] Nesime Tatbul, Tae Jun Lee, Stanley B. Zdonik, Mejbah Alam, and Justin Emile Gottschlich. Precision and recall for time series. In *NeurIPS*, 2018.

[131] Kaoru Amano, Naokazu Goda, Shin'ya Nishida, Yoshimichi Ejima, Tsunehiro Takeda, and Yoshio Ohtani. Estimation of the timing of human visual perception from magnetoencephalography. *The Journal of Neuroscience*, 26:3981 – 3991, 2006.

[132] Takaya Saito and Marc Rehmsmeier. The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PLoS ONE*, 10, 2015.

[133] Songqiao Han, Xiyang Hu, Hailiang Huang, Minqi Jiang, and Yue Zhao. Adbench: Anomaly detection benchmark. *Advances in Neural Information Processing Systems*, 35:32142–32159, 2022.

[134] Na Zhao, Biao Han, Ruidong Li, Jinshu Su, and Cong Zhou. A multivariate kpis anomaly detection framework with dynamic balancing loss training. *IEEE Transactions on Network and Service Management*, pages 1–1, 2022.

[135] John Paparrizos, Yuhao Kang, Paul Boniol, Ruey Tsay, Themis Palpanas, and Michael J. Franklin. Tsb-uad: An end-to-end benchmark suite for univariate time-series anomaly detection. *Proc. VLDB Endow.*, 15:1697–1711, 2022.

[136] Seyed Soheil Johari, Nashid Shahriar, Massimo Tornatore, Raouf Boutaba, and Aladdin Saleh. Anomaly detection and localization in nfv systems: an unsupervised learning approach. *NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium*, pages 1–9, 2022.

[137] Laurens van der Maaten and Geoffrey E. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[138] Chang Cao, Davide Chicco, and Michael M. Hoffman. The mcc-f1 curve: a performance evaluation technique for binary classification. *ArXiv*, abs/2006.11278, 2020.

[139] Damien Fourure, Muhammad Usama Javaid, Nicolas Posocco, and Simon Tihon. Anomaly detection: How to artificially increase your f1-score with a biased evaluation protocol. In *ECML/PKDD*, 2021.

[140] Houssam Zenati, Manon Romain, Chuan-Sheng Foo, Bruno Lecouat, and Vijay Ramaseshan Chandrasekhar. Adversarially learned anomaly detection. *2018 IEEE International Conference on Data Mining (ICDM)*, pages 727–736, 2018.

[141] Emadeldeen Eldele, Mohamed Ragab, Zhenghua Chen, Min Wu, C. Kwoh, Xiaoli Li, and Cuntai Guan. Time-series representation learning via temporal and contextual contrasting. In *International Joint Conference on Artificial Intelligence*, 2021.

[142] Jean-Yves Franceschi, Aymeric Dieuleveut, and Martin Jaggi. Unsupervised scalable representation learning for multivariate time series. *Advances in neural information processing systems*, 32, 2019.

[143] Zhihan Yue, Yujing Wang, Juanyong Duan, Tianmeng Yang, Congrui Huang, Yu Tong, and Bixiong Xu. Ts2vec: Towards universal representation of time series. In *AAAI Conference on Artificial Intelligence*, 2021.

[144] Xiaochen Zheng, Xing-Yu Chen, Manuel Schurch, Amina Mollaysa, Ahmed Allam, and M. Krauthammer. Simts: Rethinking contrastive representation learning for time series forecasting. *ArXiv*, abs/2303.18205, 2023.

[145] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 9912–9924. Curran Associates, Inc., 2020.

[146] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9726–9735, 2019.

[147] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15745–15753, 2020.

[148] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020.

[149] Sana Tonekaboni, Danny Eytan, and Anna Goldenberg. Unsupervised representation learning for time series with temporal neighborhood coding. In *International Conference on Learning Representations*, 2021.

[150] Dongsheng Luo, Wei Cheng, Yingheng Wang, Dongkuan Xu, Jingchao Ni, Wenchao Yu, Xuchao Zhang, Yanchi Liu, Yuncong Chen, Haifeng Chen, and Xiang Zhang. Time series contrastive learning with information-aware augmentations. In *AAAI Conference on Artificial Intelligence*, 2023.

[151] Gerald Woo, Chenghao Liu, Doyen Sahoo, Akshat Kumar, and Steven Hoi. CoST: Contrastive learning of disentangled seasonal-trend representations for time series forecasting. In *International Conference on Learning Representations*, 2022.

[152] Chen Qiu, Timo Pfrommer, Marius Kloft, Stephan Mandt, and Maja Rudolph. Neural transformation learning for deep anomaly detection beyond images. In *International conference on machine learning*, pages 8703–8714. PMLR, 2021.

[153] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. In *Neural Information Processing Systems*, 2016.

[154] Brian Kenji Iwana and Seiichi Uchida. An empirical survey of data augmentation for time series classification with neural networks. *PLoS ONE*, 16, 2020.

[155] Abhishek Sinha, Kumar Ayush, Jiaming Song, Burak Uzkent, Hongxia Jin, and Stefano Ermon. Negative data augmentation. In *International Conference on Learning Representations*, 2021.

[156] Hiroaki Sakoe. Dynamic-programming approach to continuous speech recognition. In *1971 Proc. the International Congress of Acoustics, Budapest*, 1971.

[157] Marco Cuturi and Mathieu Blondel. Soft-dtw: a differentiable loss function for time-series. In *International Conference on Machine Learning*, 2017.

[158] Mathieu Blondel, Arthur Mensch, and Jean-Philippe Vert. Differentiable divergences between time series. In *International Conference on Artificial Intelligence and Statistics*, pages 3853–3861. PMLR, 2021.

[159] Jordan Hochenbaum, Owen S Vallis, and Arun Kejariwal. Automatic anomaly detection in the cloud via statistical learning. *arXiv preprint arXiv:1704.07706*, 2017.

[160] Jing Chen, Ming Chen, Xianglin Wei, and Bing Chen. Matrix differential decomposition-based anomaly detection and localization in nfv networks. *IEEE Access*, 7:29320–29331, 2019.

[161] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data mining and knowledge discovery*, 33(4):917–963, 2019.

[162] Johann Faouzi. Time series classification: A review of algorithms and implementations. *Machine Learning (Emerging Trends and Applications)*, 2022.

[163] Navid Mohammadi Foumani, Lynn Miller, Chang Wei Tan, Geoffrey I Webb, Germain Forestier, and Mahsa Salehi. Deep learning for time series classification and extrinsic regression: A current survey. *ACM Computing Surveys*, 56(9):1–45, 2024.

[164] Junichi Kawasaki, Genichi Mouri, and Yusuke Suzuki. Comparative analysis of network fault classification using machine learning. In *NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium*, pages 1–6. IEEE, 2020.

[165] Jeremias Dötterl and Zahra Hemmati Fard. Classification of home network problems with transformers. In *Proceedings of the 39th ACM/SIGAPP Symposium on Applied Computing*, pages 1081–1087, 2024.

[166] Anthony Bagnall, Jason Lines, Aaron Bostrom, James Large, and Eamonn Keogh. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data mining and knowledge discovery*, 31:606–660, 2017.

[167] Houtao Deng, George Runger, Eugene Tuv, and Martyanov Vladimir. A time series forest for classification and feature extraction. *Information Sciences*, 239:142–153, 2013.

[168] Aaron Bostrom and Anthony Bagnall. Binary shapelet transform for multiclass time series classification. In *Big Data Analytics and Knowledge Discovery: 17th International Conference, DaWaK 2015, Valencia, Spain, September 1-4, 2015, Proceedings 17*, pages 257–269. Springer, 2015.

[169] Anthony Bagnall, Jason Lines, Jon Hills, and Aaron Bostrom. Time-series classification with cote: the collective of transformation-based ensembles. *IEEE Transactions on Knowledge and Data Engineering*, 27(9):2522–2535, 2015.

[170] Yi Zheng, Qi Liu, Enhong Chen, Yong Ge, and J Leon Zhao. Exploiting multi-channels deep convolutional neural networks for multivariate time series classification. *Frontiers of Computer Science*, 10:96–112, 2016.

[171] Zhiguang Wang, Weizhong Yan, and Tim Oates. Time series classification from scratch with deep neural networks: A strong baseline. In *2017 International joint conference on neural networks (IJCNN)*, pages 1578–1585. IEEE, 2017.

[172] Bendong Zhao, Huanzhang Lu, Shangfeng Chen, Junliang Liu, and Dongya Wu. Convolutional neural networks for time series classification. *Journal of systems engineering and electronics*, 28(1):162–169, 2017.

[173] Dani Kiyasseh, Tingting Zhu, and David A Clifton. Clocs: Contrastive learning of cardiac signals across space, time, and patients. In *International Conference on Machine Learning*, pages 5606–5615. PMLR, 2021.

[174] Xiang Zhang, Ziyuan Zhao, Theodoros Tsiligkaridis, and Marinka Zitnik. Self-supervised contrastive pre-training for time series via time-frequency consistency. *Advances in Neural Information Processing Systems*, 35:3988–4003, 2022.

[175] Joël Roman Ky, Philippe Graff, Bertrand Mathieu, and Thibault Cholez. A hybrid p4/nfv architecture for cloud gaming traffic detection with unsupervised ml. In *2023 IEEE Symposium on Computers and Communications (ISCC)*, pages 733–738. IEEE, 2023.

[176] Kuan-Yu Lee, Jia-Wei Fang, Yuan-Chun Sun, and Cheng-Hsin Hsu. Modeling gamer quality-of-experience using a real cloud vr gaming testbed. In *Proceedings of the 15th International Workshop on Immersive Mixed and Virtual Environment Systems*, pages 12–17, 2023.

[177] Huai-Sheng Huang and Yen-Shuo Su. A practical study of qoe on cloud gaming in 5g networks. In *2023 International Wireless Communications and Mobile Computing (IWCMC)*, pages 638–643. IEEE, 2023.

[178] Md Tariqul Islam, Christian Esteve Rothenberg, and Pedro Henrique Gomes. Predicting xr services qoe with ml: Insights from in-band encrypted qos features in 360-vr. In *2023 IEEE 9th International Conference on Network Softwarization (NetSoft)*, pages 80–88. IEEE, 2023.

[179] Lorenzo Perini, Paul-Christian Bürkner, and Arto Klami. Estimating the contamination factor's distribution in unsupervised anomaly detection. In *International Conference on Machine Learning*, pages 27668–27679. PMLR, 2023.

[180] Ming Jin, Huan Yee Koh, Qingsong Wen, Daniele Zambon, Cesare Alippi, Geoffrey I Webb, Irwin King, and Shirui Pan. A survey on graph neural networks for time series: Forecasting, classification, imputation, and anomaly detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[181] Tiago M Fernández-Caramés and Paula Fraga-Lamas. Forging the industrial metaverse-where industry 5.0, augmented and mixed reality, iiot, opportunistic edge computing and digital twins meet. *arXiv preprint arXiv:2403.11312*, 2024.

[182] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30, 2017.

[183] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

[184] Han Wang, Jingwei Wang, Yukai Zhao, Qing Liu, Min Liu, and Weiming Shen. Few-shot learning for fault diagnosis with a dual graph neural network. *IEEE Transactions on Industrial Informatics*, 19(2):1559–1568, 2022.

[185] Chaoying Yang, Jie Liu, Qi Xu, and Kaibo Zhou. A generalized graph contrastive learning framework for few-shot machine fault diagnosis. *IEEE Transactions on Industrial Informatics*, 2023.

[186] Chang Xu, Dacheng Tao, and Chao Xu. A survey on multi-view learning. *arXiv preprint arXiv:1304.5634*, 2013.

[187] Yingming Li, Ming Yang, and Zhongfei Zhang. A survey of multi-view representation learning. *IEEE transactions on knowledge and data engineering*, 31(10):1863–1883, 2018.

[188] Guoqing Chao, Shiliang Sun, and Jinbo Bi. A survey on multi-view clustering. *arXiv preprint arXiv:1712.06246*, 2017.

[189] Lihua Zhou, Guowang Du, Kevin Lü, Lizheng Wang, and Jingwei Du. A survey and an empirical evaluation of multi-view clustering approaches. *ACM Computing Surveys*, 56(7):1–38, 2024.

[190] Colin Troisemaine, Vincent Lemaire, Stéphane Gosselin, Alexandre Reiffers-Masson, Joachim Flocon-Cholet, and Sandrine Vaton. Novel class discovery: an introduction and key concepts. *arXiv preprint arXiv:2302.12028*, 2023.

[191] Uzma Hasan, Emam Hossain, and Md Osman Gani. A survey on causal discovery methods for temporal and non-temporal data. *arXiv preprint arXiv:2303.15027*, 2023.

[192] Chang Gong, Chuzhe Zhang, Di Yao, Jingping Bi, Wenbin Li, and YongJun Xu. Causal discovery from temporal data: An overview and new perspectives. *ACM Computing Surveys*, 57(4):1–38, 2024.

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

3GPP ..... 3rd Generation Partnership Project

5G NR ..... 5G New Radio

5GC ..... 5G Core

AD ..... Anomaly Detection

Adam ..... Adaptative Momentum Estimation

AE ..... Autoencoder

AI ..... Artificial Intelligence

ANRT ..... *Agence Nationale de la Recherche et de la Technologie* [*fr*]

AP ..... Access Point

ARCEP ..... *Autorité de régulation des communications électroniques, des postes et de la distribution de la presse* [*fr*]

ARIMA ..... AutoRegressive Integrated Moving Average

AU-PR ..... Area Under Precision-Recall Curve

AU-ROC ..... Area Under Receiver Operating Characteristic Curve

BS ..... Base Station

BSS ..... Basic Service Set

CG ..... cloud gaming

CIFRE ..... *Convention Industrielle de Formation par la REcherche* [*fr*]

CL ..... Contrastive Learning

CNN ..... Convolutional Neural Network

COCA ..... Contrastive One-Class Anomaly

COUTA ..... Calibrated One-class classification for Unsupervised Time series Anomaly detection

CTAD ..... Contrastive Time series Anomaly Detection

CV ..... Computer Vision

DAGMM ..... Deep Autoencoding Gaussian Mixture Model

DBSCAN ..... Density-Based Spatial Clustering of Applications with Noise

Deep-SVDD ..... Deep Support Vector Data Description

DHCP ..... Dynamic Host Configuration Protocol

DIF ..... Deep Isolation Forest

DL ..... Deep Learning

DTLS ..... Datagram Transport Layer Security

DTW ..... Dynamic Time Warping

ELBO ..... Evidence Lower Bound

eMBB ..... Enhanced Mobile Broadband

eNB ..... evolved NodeB

EPC ..... Evolved Packet Core

E-UTRA ..... Evolved Universal Terrestrial Radio

FEC ..... Forward Error Correction

FPR ..... False Positive Rate

FTTH ..... Fiber-To-The-Home

GAN ..... Generative Adversarial Network

GFN ..... GeForce Now

GMM ..... Gaussian Mixture Model

gNB ..... Next Generation NodeB

GNN ..... Graph Neural Network

GPU ..... Graphic Processing Units

GRU ..... Gated Recurrent Unit

HD ..... High Definition

HMDs ..... Head-Mounted Displays

HSPA+ ..... Evolved High Speed Packet Access

ICA ..... Independent Component Analysis

IF ..... Isolation Forest

IoT ..... Internet of Things

IP ..... Internet Protocol

ISP ..... Internet Service Provider

ITU ..... International Telecommunication Union

KDE ..... Kernel Density Estimation

KNN ..... k-th Nearest Neighbor

KPIs ..... Key Performance Indicators

KQIs ..... Key Quality Indicators

LAN ..... Local Access Network

LL ..... Low-latency

LLM ..... Large Language Model

LOF ..... Local Outlier Factor

LSTM ..... Long Short-Term Memory

LTE ..... Long-Term Evolution

MAML ..... Model-Agnostic Meta-Learning

MCC ..... Matthews Correlation Coefficient

MCMC ..... Markov Chain Monte Carlo

MCS ..... Modulation and Coding Scheme

MEC ..... Multi-Access Edge Computing

MIMO ..... Multiple Input Multiple Output

ML ..... Machine Learning

MLP ..... Multilayer Perceptron

MME ..... Mobility Management Entity

mMTC ..... Massive Machine Type Communication

MOS ..... Mean Opinion Score

MSCRED ..... Mult-Scale Convolutional Recurrent Encoder-Decoder

MTU ..... Maximum Transmission Unit

MVC ..... Multi-View Clustering

MVL ..... Multi-View Learning

NCD ..... Novel Class Discovery

NFV ..... Nerwork Function Virtualization

NG-RAN ..... Next Generation RAN

NLP ..... Natural Language Processing

NSA ..... Non-Standalone

NT-Xent ..... Normalized Temperature-scaled Cross Entropy

OC-SVM ..... One-Class Support Vector Machine

OFDMA ..... Orthogonal Frequency Division Multiple Access

PA ..... point-adjust

PCA ..... Principal Component Analysis

PCAP ..... Packet Capture

P-GW ..... Packet Data Network Gateway

PSN ..... PlayStation Now

PW ..... point-wise

QoE ..... Quality of Experience

QoS ..... Quality of Service

RAID ..... Root cause Anomaly Identification and Diagnosis

RAN ..... Radio Access Network

RBF ..... Radial Basis Function

RCD ..... Root-Cause Diagnosis

RF ..... Radio Frequency

RNN ..... Recurrent Neural Network

RPA ..... revised point-adjust

RSSI ..... Received Signal Strength Indicator

RTP ..... Real Time Protocol

RTT ..... Round Trip Time

SA ..... Standalone

SBA ..... Service-Based Architecture

SGD ..... Stochastic Gradient Descent

S-GW ..... Serving Gateway

SL ..... Supervised Learning

SMS ..... Short Message Service

SSID ..... Service Set Identifier

SSL ..... Self-supervised Learning

STD ..... Stadia

SVM ..... Support Vector Machines

TCP ..... Transmission Control Protocol

TPR ..... True Positive Rate

TSC ..... Time Series Classification

UDP ..... User Datagram Protocol

UE ..... User Equipment

UL ..... Unsupervised Learning

URLLC ..... Ultra-Reliable Low Latency Communication

USAD ..... UnSupervised Anomaly Detection

VAE ..... Variational Autoencoder

VAR ..... Vector AutoRegression

ViT ..... Vision Transformers

VR ..... virtual reality

WAD ..... Window Anomaly Detection

WAN ..... Wide Access Network

WiMAX ..... Worldwide Interoperability for Microwave Access

WPA ..... Wi-Fi Protected Access

XC ..... Xbox Cloud

XR ..... extended reality

Résumé

Titre: Détection d'Anomalies et Diagnostic des Causes Racines des Applications à Faible-Latence sur les Réseaux à Capacité Variable.

L'évolution des réseaux a conduit à l'émergence d'applications à faible latence (FL) telles que le cloud gaming (CG) et la réalité virtuelle basée sur le cloud (Cloud VR), qui exigent des conditions réseau strictes, notamment une faible latence et une bande passante élevée. Cependant, les réseaux à capacité variable introduisent des dégradations, telles que du délai, des fluctuations de bande passante et des pertes de paquets, qui peuvent significativement altérer l'expérience utilisateur sur les applications FL. Cette thèse vise à concevoir des méthodologies pour détecter et diagnostiquer les anomalies de performance des applications FL fonctionnant sur des réseaux cellulaires et Wi-Fi.

Pour atteindre cet objectif, des bancs d'essai expérimentaux réalistes ont été mis en place pour collecter des bases de données caractérisant les performances du réseau et capturant les indicateurs clés de performance (KPI) des applications CG et Cloud VR dans des environnements 4G et Wi-Fi. Ces données constituent la base de l'évaluation et du développement d'algorithmes de détection d'anomalies et de diagnostic basés sur l'apprentissage automatique.

Les principales contributions de cette thèse incluent le développement de CATS, une solution de détection d'anomalies basé sur l'apprentissage contrastif, capable d'identifier efficacement les dégradations de l'expérience utilisateur dans les applications CG tout en restant robuste face à la contamination des données. De plus, cette thèse introduit RAID, un système de diagnostic en deux étapes conçu pour identifier les causes racines des problèmes de performance dans le Cloud VR. RAID a démontré une grande efficacité dans le diagnostic des dégradations Wi-Fi, même avec un nombre limité de données annotées.

Les résultats de ce travail font progresser les domaines de la détection d'anomalies et du diagnostic des causes racines, offrant des perspectives concrètes aux opérateurs de réseaux pour optimiser les performances de leurs réseaux et améliorer la fiabilité des services et mieux supporter les applications FL, qui sont appelées à révolutionner les technologies de communication et à stimuler l'innovation dans de nombreuses industries.

Mots-clés: Applications à faible latence, Réseaux à capacité variable, Détection d'anomalies, Diagnostic des causes, Apprentissage machine

Abstract

Title: Anomaly Detection and Root Cause Diagnosis for Low-Latency Applications in Time-Varying Capacity Networks.

The evolution of networks has driven the emergence of low-latency (LL) applications such as cloud gaming (CG) and cloud virtual reality (Cloud VR), which demand stringent network conditions, including low latency and high bandwidth. However, time-varying capacity networks introduce impairments such as delays, bandwidth fluctuations, and packet loss, which can significantly degrade user experience on LL applications. This research aims to design methodologies for detecting and diagnosing performance anomalies in LL applications operating over cellular and Wi-Fi networks.

To achieve this, realistic experimental testbeds were established to collect datasets that characterize network performance and capture key performance indicators (KPIs) of CG and Cloud VR applications over 4G and Wi-Fi environments. These datasets serve as the foundation for evaluating and developing machine learning-based anomaly detection and diagnostic frameworks.

The key contributions of this thesis include the development of CATS, a contrastive learning-based anomaly detection framework capable of efficiently identifying user experience degradation in CG applications while remaining robust to data contamination. Additionally, this research introduces RAID, a two-stage root causes diagnosis framework designed to pinpoint the root causes of performance issues in Cloud VR. RAID demonstrated high efficiency in diagnosing Wi-Fi impairments, even with limited labeled data.

The findings of this work advance the fields of anomaly detection and root cause diagnosis, offering actionable insights for network operators to optimize network performance and enhance service reliability to support LL applications, which are set to revolutionize communication technologies and drive innovation across various industries.

Keywords: Low-latency applications, Time-varying capacity networks, Anomaly Detection, Root Cause Diagnosis, Machine Learning