UNIVERSITY OF TURKU

EDGE COMPUTING AND BLOCKCHAIN FOR PRIVACY-CRITICAL AND DATA-SENSITIVE HEALTH APPLICATIONS

Anum Nawaz

EDGE COMPUTING AND BLOCKCHAIN FOR PRIVACY-CRITICAL AND DATA-SENSITIVE HEALTH APPLICATIONS

University of Turku

Faculty of Technology
Department of Computing
Information and Communication Technology
Doctoral Programme in Technology

Supervised by

Professor, Tomi Westerlund
University of Turku, Finland

Professor, Zhuo Zou
Fudan University

Reviewed by

Professor, Muhammad Ali Imran
University of Glasgow

Associate Professor, Haseeb Hassan
Shenzhen Technology University

Opponent

Associate Professor, Bo Tan
University College London

ISBN ISBN-978-02-0082-4 (PRINT)
ISBN ISBN-978-02-0082-4 (PDF)
ISSN (PRINT)
ISSN (ONLINE)

To my parents, the architects of my dreams and the champions of my aspirations

UNIVERSITY OF TURKU
Faculty of Technology
Department of Computing
Information and Communication Technology
NAWAZ, ANUM: Edge Computing and Blockchain for Privacy-Critical and Data-Sensitive Health Applications
Doctoral dissertation, 129
Doctoral Programme in Technology
February 2025

ABSTRACT

The widespread adoption of ubiquitous healthcare enhances the accessibility, quality, and efficiency of healthcare; however, the risk of data privacy breaches is also increasing due to third party service providers. According to Cisco's annual report (2018–2023), 94% data processing on cloud servers and continuous tracing of personal data create privacy vulnerabilities. To deal data privacy challenges, recent studies integrate edge computing with distributed ledger-based technologies (DLTs) such as blockchain, but optimization of privacy requirements is still needed. Therefore, this study is conducted to minimize the existing gaps related to data privacy by proposing edge-intelligence (lightweight machine learning/deep learning) based distributed system *EdgeBot*. Specifically, our framework aims to optimize data privacy, fine-grained access control, and data ownership rights of real-time healthcare systems in time sensitive scenarios.

Initially, we proposed computing model of *EdgeBot*, formulate on/off chain secure communication and fine-grained data access scheme for processed health monitoring data. Thereafter, we constructed resource-efficient one directional convolutional neural network (1D-CNN) for multiclass arrhythmia detection in real-time health monitoring system. Bayesian optimization algorithm is embedded within the network to adaptively select the optimal combination of hyperparameters. We employed 2-channel ECG system using AD8232 along with STM32F427 and raspberry pi boards as edge gateways. To ascertain the precision and credibility of AD8232 based ECG system, a comparison of HR and RR interval measurement was obtained from Polysomnography (PSG) device. Comparative analysis of our proposed 1D-CNN shows average of 97.4% accuracy while utilizing significantly fewer resources. Average ECG processing time along with data sharing on private ethereum requires only 150ms and 143ms respectively.

Secondly, P2P trustless data trade system is formulated utilizing edge gateways (pi 3, model B+), and STM32F427 (M3,M4 and M7) as lightweight nodes, leveraging ethereum. Results of employing ECDSA and ECIES on M3,M4 and M7 shows an average of 17.253s, 1.462s and 1.156s execution time respectively, while average power consumption by M3 and M4 was 200mW, whereas M7 uses an average of 290mW. Results indicate the superior performance of M4 cortex microcontrollers while consuming less resources. Resource and performance analysis shows less than 40% of average computing resources were utilized during transaction handling; it is remarkable that TRD requires only 34.6ms, VTR 36ms, and TCT 73.6ms on aver-

age. FobSim simulator was utilized to check the scalability of the proposed scheme and results shows that gossip protocol decreases 35% latency during parallel transactions ranging from 5 to 500. *EdgeBot* post-quantum resistant extension was implemented, and results were compared with other lightweight KEMs. Latency and memory efficiency of Kyber512 KEM with other lightweight KEMs was compared on STM32F427 and it required 22-25 bytes of memory. Moreover, Kyber512 was found to be the best performer, balancing energy consumption and memory usage. Results of performance analysis shows *EdgeBot*, a viable option for fortifying data trade, data ownership, and exchange through edge gateways.

Thirdly, we presented the implementation and evaluation of real-time data sharing and tracing among trusted stakeholders while preserving transparency using sawtooth on a linux platform, leveraging AWS EC2 instances for server communication. We propose service optimization in sawtooth, a mathematical foundation for determining the most efficient combination of databatch size, transactions per second (tps), resource utilization and network resources while ensuring the reliability of transaction commitment. Furthermore, performance evaluations were conducted on both AWS and local PCs, utilized cAdvisor for docker containers and cloudWatch for AWS metrics. Results indicated significant spikes in CPU and network usage during transaction processing, with the system successfully managing 82% of 1,000 parallel statements. Overall, results analysis shows high scalability and reliability of *EdgeBot* while utilizing less resources as compared to recent studies, particularly for large-scale operations and seamless integration with diverse underlying DLTs.

KEYWORDS: Ubiquitous Healthcare, Artificial Intelligence, Blockchain, Edge Computing, Data privacy, Data Trade

TURUN YLIOPISTO
Teknillinen tiedekunta
Tietojenkäsittelytieteen laitos
Tieto- ja viestintätekniikka
NAWAZ, ANUM: Edge Computing and Blockchain for Privacy-Critical and Data-Sensitive Health Applications
Väitöskirja, 129
Tekniikan tohtoriohjelma
Helmikuu 2025

TIIVISTELMÄ

Ubiikin terveydenhuollon laajamittainen käyttöönotto parantaa terveydenhuollon saavutettavuutta, laatua ja tehokkuutta; kuitenkin, tietosuojaloukkausten riski kasvaa myös kolmannen osapuolen palveluntarjoajien vuoksi. Ciscon vuosiraportin (2018–2023) mukaan 94% tietojenkäsittelystä tapahtuu pilvipalvelimilla, ja henkilötietojen jatkuva jäljitys luo tietosuojahaavoittuvuuksia. Tietosuojahaasteiden ratkaisemiseksi uusissa tutkimuksissa yhdistetään reunalaskenta ja hajautettuun tilikirjatekniikkaan (DLTs), kuten lohkoketjuun, mutta tietosuojavaatimusten optimointia tarvitaan edelleen. Siksi tämä tutkimus on tehty minimoimaan nykyisiä tietosuojaan liittyviä puutteita ehdottamalla edge intelligenceen (kevyt koneoppiminen/syvällinen opovpiminen) perustuvaa hajautettua järjestelmää EdgeBot. Erityisesti kehyksemme pyrkii optimoimaan tietosuojaa, hienojakoista pääsynhallintaa ja tietojen omistusoikeuksia reaaliaikaisissa terveydenhuoltojärjestelmissä kiireellisissä tilanteissa.

Aluksi ehdotimme EdgeBotin laskentamallia, muotoilimme on/off-chain suojatun viestinnän ja hienojakoisen tietojen pääsynhallinnan terveysseurantatiedolle. Sen jälkeen rakensimme resurssitehokkaan yhden suunnan konvoluutiohermoverkon (1D-CNN) moniluokkaiseen rytmihäiriöiden tunnistamiseen reaaliaikaisessa terveysseuranta järjestelmässä. Bayesin optimointialgoritmi on upotettu verkkoon mukautuvasti valitsemaan hyperparametrien optimaalisen yhdistelmän. Käytimme 2-kanavaista ECG-järjestelmää käyttäen AD8232 yhdessä STM32F427 ja raspberry pi korttien kanssa reunaportteina. AD8232-pohjaisen EKG-järjestelmän tarkkuuden ja uskottavuuden varmistamiseksi verrattiin HR ja RR välimittausta polysomnografialaitteesta (PSG). Ehdottamamme 1D-CNN vertailuanalyysi osoittaa keskimäärin 97.4% tarkkuuden samalla kun käytetään huomattavasti vähemmän resursseja. Keskimääräinen EKG käsittelyaika sekä tietojen jakaminen yksityisessä Ethereumissa vaativat vain 150ms ja 143ms vastaavasti. Toiseksi, P2P-luottamukseton tietokauppajärjestelmä muotoiltiin hyödyntämällä reunaportteja (pi 3, mallia B+) ja STM32F427 (M3, M4 ja M7) kevyinä solmuina, hyödyntäen Ethereumia. ECDSA ja ECIES käyttämisen tulokset M3ä, M4ä ja M7ä osoittavat keskimäärin 17.253s, 1.462s ja 1.156s suoritusaikaa vastaavasti, kun taas M3 ja M4 keskimääräinen virrankulutus oli 200mW, kun taas M7 käytti keskimäärin 290mW. Tulokset osoittavat M4 Cortex -mikrokontrollereiden ylivoimaisen suorituskyvyn samalla kun resursseja kulutetaan vähemmän. Resurssi- ja suorituskykyanalyysi osoittaa, että alle 40% keskimääräisistä laskentaresursseista käytettiin tapahtumien käsittelyn aikana; on huomionarvoista, että TRD vaatii keskimäärin vain 34.6ms, VTR 36ms ja TCT 73.6ms. FobSim simulaattoria käytettiin ehdote-

tun järjestelmän skaalautuvuuden tarkistamiseen, ja tulokset osoittavat, että gossip-protokolla vähentää 35% viivettä rinnakkaisten tapahtumien aikana, jotka vaihtelevat 5ä 500. EdgeBotin kvanttivastustuskykyinen laajennus toteutettiin, ja tuloksia verrattiin muihin kevyisiin KEM. Kyber512 KEM ja muiden kevyiden KEM viive ja muistitehokkuus vertailtiin STM32F427, ja se vaati 22-25 tavua muistia. Lisäksi Kyber512 todettiin parhaaksi suorittajaksi, tasapainottaen energiankulutuksen ja muistinkäytön. Suorituskykyanalyysin tulokset osoittavat EdgeBotin olevan varteenotettava vaihtoehto tietokaupan, tietojen omistusoikeuden ja vaihdon vahvistamiseen reunaporttien kautta.

Kolmanneksi esitimme reaaliaikaisen tietojen jakamisen ja jäljityksen toteutuksen ja arvioinnin luotettujen sidosryhmien kesken säilyttäen avoimuuden, käyttäen Sawtoothia Linux-alustalla ja hyödyntämällä AWS EC2 -instansseja palvelinviestintään. Ehdotamme palveluoptimointia Sawtoothissa, matemaattista perustaa tehokkaimman yhdistelmän määrittämiseksi tietopakettikoolle, tapahtumia sekunnissa (tps), resurssien käyttöä ja verkkovaroja samalla kun varmistetaan tapahtumien sitoumuksen luotettavuus. Lisäksi suorituskykyarvioinnit tehtiin sekä AWS ä että paikallisilla tietokoneilla, käyttäen cAdvisoria Docker-konteille ja CloudWatchia AWS-mittareille. Tulokset osoittivat merkittäviä piikkejä suorittimen ja verkkokäytön aikana tapahtumien käsittelyssä, ja järjestelmä hallitsi onnistuneesti 82% 1000 rinnakkaisesta lausunnosta. Kokonaisuutena tulosanalyysi osoittaa EdgeBotin korkean skaalautuvuuden ja luotettavuuden samalla kun se käyttää vähemmän resursseja verrattuna viimeaikaisiin tutkimuksiin, erityisesti suurissa operaatioissa ja saumattomassa yhteensopivuudessa moninaisten hajautettujen tilikirjojen (DLTs) kanssa.

ASIASANAT: Ubiikki terveydenhuolto, tekoäly, lohkoketju, reunalaskenta, tietosuoja, tietokauppa

Acknowledgements

The completion of this doctoral research has been a lengthy and challenging journey, particularly during the period of confinement at home in Pakistan due to the COVID-19 pandemic. As I write these words, I am filled with profound gratitude, and I would like to take this opportunity to express my sincere appreciation to all those who have supported me throughout this journey.

First and foremost, I am deeply indebted to my advisor, Dr. Tomi Westerlund, for his invaluable guidance and feedback throughout my thesis process. It has been an enriching experience to work on problems together, bringing ideas into reality, always with passion and kindness. The long days in the laboratory to put the finishing touches on submissions have been truly inspiring. Thank you, Tomi Westerlund, for pushing me beyond my comfort zone. Your support as an academic advisor has been invaluable. I also extend my sincere gratitude to my dissertation committee member, Dr. Zhuo Zuo, for his insightful contributions and support.

Additionally, I would like to thank Prof. Dr. Haibik Kan from Fudan University for his invaluable support and resources. Prof. Kan was always available whenever I encountered challenges or had questions about my research.

My experience at UTU would not have been the same without my lab colleagues, whose continuous collaboration enabled my progress. I want to thank my friends for their love, trust, and motivation, which helped me stay focused on my goals. Special mentions go to Aisha, Faiza, and Fatima for their positivity that boosted my morale. I am deeply thankful, as this accomplishment would not have been possible without you. Moreover, I wish to express my heartfelt gratitude to my irreplaceable friend, Anam, who inspired me to embark on this doctoral journey and provided unwavering support. Her intelligence, creativity, and boundless energy have been a constant source of inspiration.

Finally, I am profoundly grateful to my family for their unwavering support. I could not have begun this doctorate program without the encouragement and constant support of my parents, Muhammad Nawaz Kahloon and Shahnaz Nawaz. I hope this achievement brings pride to my siblings, Farrukh and Nida. Moreover, I would like to mention my husband, Ahmad Virk, a recent and joyous addition to my life. Lastly, I feel compelled to express my gratitude for the joy that Rayyan, Ibrahim, Hamza, and Muhammad have brought into my life.

I can truly say that I have enjoyed my doctoral journey, and I am grateful to ev-

eryone who made it an exciting and fulfilling experience. I am also thankful to all those who have contributed to this achievement in their own ways. Their support has been integral to my success. A doctoral degree represents the pinnacle of academic achievement. However, the pursuit of knowledge and research is a continuous journey, akin to an unending challenge; it does not end at this milestone. Let us continue to encourage and support one another in our future endeavours. Thank you all.

17 February 2025
Anum Nawaz

ANUM NAWAZ

Anum Nawaz pursuing her Joint doctorate degree with Shanghai Key Laboratory of Intelligent Information Processing Lab, Fudan University, China and at Turku Intelligent Embedded and Robotic Systems (TIERS) Group, University of Turku, Finland. She received her double Masters (Tech.) degree in Information and Communication Science and Technology from the University of Turku, Finland and Fudan University, Shanghai. She got a fully-funded scholarship grant from the Shanghai Government Scholarship (SGS) and Chinese Government Scholarship (CSC) for her Ph.D degree and Masters Degree respectively. Her research interests include information security, the privacy of edge devices, blockchain, ubiquitous healthcare and autonomous systems.

Table of Contents

Abbreviations

uHealth	Ubiquitous healthcare
dHealth	Digital health monitoring
DLTs	Distributed ledger technologies
PHI	Personal health information
IoMT	Internet of Medical Things
IoT	Internet of Things
EGs	Edge Gateways
1D-CNN	One directional Convolutional Neural Network
AI	Artificial Intelligence
ML	Machine Learning
P2P	Peer-to-Peer
SBCs	Single board computers
ECIES	Elliptic curve integrated encryption scheme
CKD	Child key derivation
ECDSA	Elliptic curve digital signature algorithm
IPFS	InterPlanetary File System
EHRs	Electronic health records
IaaS	Infrastructure as a Service
PaaS	Platform as a Service
SaaS	Software as a Service
EC2	Amazon's Elastic Compute Cloud

SOA	Service-oriented architectures
EMRs	Electronic medical records
EVM	Ethereum Virtual Machine
ANN	Artificial neural network
DApps	Decentralized applications
CFT	Crash fault tolerant
PKI	Public-key infrastructure
PBFT	Practical Byzantine Fault Tolerance
PoW	Proof-of-work
PoS	Proof-of-stake
DHKE	Diffie-Hellman Key-Exchange
M2M	Machine-to-Machine
LSTM	Long short-term memory
MEC	Mobile Edge Computing
SDN	Software Defined Networking

List of Original Publications

This dissertation is based on the following original publications, which are referred to in the text by their Roman numerals:

I. **Nawaz Anum**, Jorge Peña Queralta, Jixin Guan, Muhammad Awais, Tuan Nguyen Gia, Ali Kashif Bashir, Haibin Kan, and Tomi Westerlund. "Edge Computing to Secure IoT Data Ownership and Trade with the Ethereum Blockchain". Sensors 20, no. 14 (2020): 3965. DOI: 10.3390/s20143965

II. **Nawaz Anum**, Liguan Wang, Muhammad Irfan and Tomi Westerlund. "Hyperledger sawtooth based Supplychain Traceability System for Counterfeit Drugs." Computers and Industrial Engineering 190 (2024): 110021. DOI: 10.1016/j.cie.2024.110021

III. **Nawaz Anum**, Tuan Nguyen Gia, J. Peña Queralta, and Tomi Westerlund. "Edge AI and blockchain for privacy-critical and data-sensitive applications." In 2019 Twelfth International Conference on Mobile Computing and Ubiquitous Network (ICMU), pp. 1-2. IEEE, 2019. DOI:10.23919/ICMU48249.2019.9006635

IV. Gia Tuan Nguyen, **Nawaz, Anum**, Jorge Peña Querata, Hannu Tenhunen, and Tomi Westerlund. "Artificial Intelligence at the Edge in the Blockchain of Things." In International Conference on Wireless Mobile Communication and Healthcare, pp. 267-280. Springer, Cham, 2019.

V. **Nawaz Anum**, Xianjia Yu, Zho Zhou and Tomi Westerlund."Blockchain assisted privacy-preserved federated edge learning in smart healthcare systems" (unpublished).

The following related publications are not directly included in this thesis:

I. **Nawaz, Anum**, Muhammad Irfan, Hafza Ayesha and Tomi Westerlund. "Edge Based Skin Cancer Decision Support System Using Machine Learning Algorithms". In IEEE Intl Conf on Pervasive Intelligence and Computing, (PiCom) 2023.

II. **Nawaz, Anum**, Muhammad Irfan and Tomi Westerlund. "Optical character recognition using optimized convolutional networks". In Eighth International Conference on Fog and Mobile Edge Computing (FMEC 2023).

III. Muhammad Irfan, H. Jawad, Li Chen, S.F. Abbasi, **Nawaz, Anum**, Saeed Akbarzadeh, Muhammad Awais, Tomi Westerlund and Wei Chen. "Non-wearable IoT-based smart ambient behavior observation system." IEEE Sensors Journal 21, no. 18 (2021): 20857-20869.

IV. Awais Muhammad, Chen Chen, Xi Long, Bin Yin, **Nawaz, Anum**, Saadullah F. Abbasi and all. "Novel Framework: Face Feature Selection Algorithm for Neonatal Facial and Related Attributes Recognition." IEEE Access 8 (2020): 59100-59113. DOI: 10.1109/ACCESS.2020.2982865

V. M. Irfan, H. Ayesha, A. Nahli, C. Chen, Y. Xu, L. Wang, **Nawaz, Anum**, A. Subasi, Tomi Westerlund, Wei Chen. "An Ensemble Voting Approach with Innovative Multi-Domain Feature Fusion for Neonatal Sleep Stratification". IEEE Access 2023.

VI. Ayesha Siddiqa, Z. Tang, Y. Xu, L. Wang, M. Irfan, S. Farooq Abbasi, **Nawaz, Anum**, C. Chen, Wei Chen. "Single-Channel EEG Data Analysis Using a Multi-Branch CNN for Neonatal Sleep Staging". IEEE Access 2024.

VII. Subasi, Abdulhamit, Hafza Ayeshaa Siddiqa, Abdelwahed Nahliis, Chen Chen, Yan Xu, Laishuan Wang, **Nawaz, Anum**, Tomi Westerlund, and Wei Chen. "An Ensemble Voting Approach With Innovative Multi-Domain Feature Fusion for Neonatal Sleep Stratification." IEEE Access (2023).

The list of original publications have been reproduced with the permission of the copyright holders.

1 Introduction

The widespread adoption of digital health monitoring (dHealth) and ubiquitous healthcare (uHealth) systems empowers users with handheld devices and embedded sensors, enabling a diverse array of healthcare services. Ubiquitous healthcare refers to a system where healthcare is easily accessible and available to everyone, regardless of their location or situation [1]. It revolves around the continuous collection, transmission, and analysis of health-related data through digital devices. These devices, which could range from wearable fitness trackers to sophisticated medical sensors, monitor vital signs, physiological parameters, and various health metrics [2]. The collected data is then transmitted to healthcare professionals or centralized systems for analysis and interpretation. Real-time monitoring empowers medical practitioners to detect anomalies or changes in a patient's health status promptly, allowing for timely interventions and personalized treatment plans [3].

Ubiquitous healthcare enhances patient engagement and involvement in their healthcare and enables healthcare providers to deliver proactive and personalized care, leading to improved health outcomes and more efficient healthcare services. Patients need to engage in self-management activities to understand their disease better, enhance their communication with their doctors, and increase their self-confidence [4]. This approach has significantly reduced geographical barriers, enabling individuals to access healthcare services regardless of physical location [5]. It leverages communication technologies, such as video calls and remote monitoring devices, to bridge the gap between patients and healthcare providers, thereby enhancing the accessibility and availability of medical assistance [6]. Health parameters monitoring plays a crucial role in remote health services, particularly in detecting health deterioration among individuals with acute or chronic diseases [7]. This monitoring is in high demand for remote locations and the elderly population, given the increased vulnerability and health challenges associated with ageing [8].

Nonetheless, ubiquitous healthcare systems encounter significant challenges related to data security and privacy that must be effectively addressed to enhance the prevalence of these technologies. The security and privacy of data collected, processed, and stored by ubiquitous healthcare applications and systems have become increasingly concerning. Breaches in the security and privacy of healthcare information systems can have severe negative consequences for the individuals involved, ranging from embarrassment and reputational harm to various forms of discrimina-

tion. It can adversely impact the rights, freedoms, and physical and mental well-being of individuals. Given that security and privacy concerns have emerged as the most critical challenges for healthcare information systems, it is imperative to thoroughly understand and address these issues with urgency. Figure 1 shows the year-wise personal health information (PHI) data breaches in different domains.

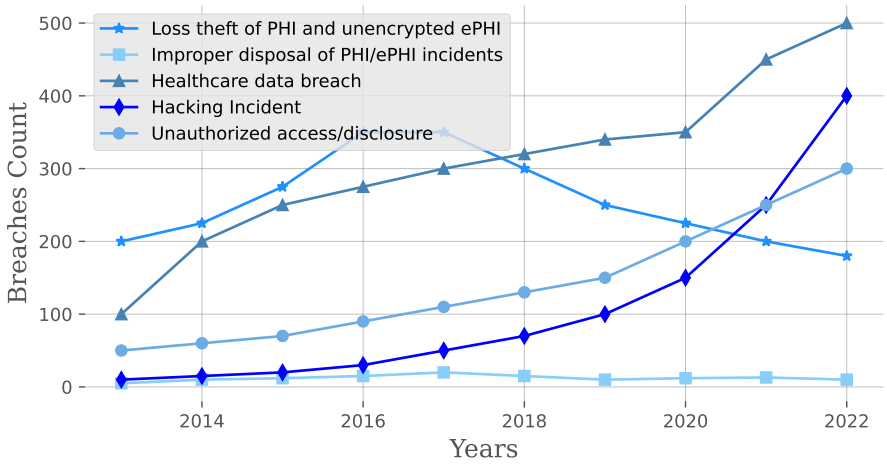


Figure 1: Breaches in Data Privacy and Security of Personal Health information (PHI) [9]

1.1 Motivation

According to the department of health and human services office for civil rights (OCR), in the United States, 4,419 health data breaches of 500 or more records were reported between 2009 and 2021, which led to the loss, theft, exposure, or impermissible disclosure of 314,063,186 healthcare records [10]. That equates to more than 94.63% of the 2021 population of the United States [11]. A general upward trend has been seen with increasingly ubiquitous IoMT devices. Ubiquitous healthcare devices are scarce computing devices highly dependent on third-party services for their computing and resource management requirements. In traditional structures, cloud services are utilized to provide several services to these scarce computing devices, such as Infrastructure as a Service (IaaS) [12], Software as a Service (SaaS) [13], and Platform as a Service (PaaS) [14]. A few popular names are Amazon's Elastic Compute Cloud (EC2) [15], Microsoft Azure [16], and IBM [17]. Cloud services have traditionally been used to analyze, aggregate, and store data acquired by end devices. Because of the firm reliance of the digital world on third-party cloud services, a wide range of concerns have been raised regarding the security of data and the protection

of privacy [18].

Table 1: Data Breaches in Healthcare domain

Year	Organization	Effected Individuals
2023	Florida Health Sciences Cente	2.4m individuals affected
2022	Broward Health Data Breach	Around 1.3m individuals affected
2022	Shields Healthcare Group	More than 2m patients affected
2021	Universal Health Services	4.5m patients affected
2020	US med collection Agency	More than 20m patients affected
2020	Teladoc	Nearly 20m patients affected
2019	LabCorp	More than 7m individuals affected
2019	Quest Diagnostics	Nearly 12m patients affected
2018	Aetna	Nearly 20m individuals affected
2017	Equifax	143m individuals affected
2016	Community Health Systems	More than 4.5m individuals affected
2015	Anthem Inc.	More than 80m individuals affected

A sustainable ubiquitous healthcare system is required to consider personal data privacy and security implications. However, current systems are vulnerable to data privacy and security [19]. In Table 1, we summarize a few examples of the health data breaches that have occurred in the healthcare industry over the past few years. Even with the significant impact and the efficient services third-party applications have provided, there are still security and privacy concerns relating to how the providers handle users' data. Problems associated with insecure cloud computing platforms such as web-based outsourcing, mobile cloud computing, and service-oriented architectures (SOA) are security and privacy issues of personalized data [20; 21; 22]. Significant shifts and optimization in edge computing lead to better network usability and significant reductions in the network load [23]. However, these changes in the machine-to-machine topologies come with new sets of vulnerabilities related to the privacy of data producers [24]. Due to server failure and privacy-related vulnerabilities during transmission, the edge computing paradigm shifts the data processing closer to the point where it is being generated [25; 26].

Recent developments in distributed ledger technologies (DLTs), particularly blockchain have revolutionized ubiquitous healthcare and digital health monitoring systems. Blockchain's decentralized and secure architecture offers significant benefits, enhancing the effectiveness and efficiency of these systems. One of the primary advantages of blockchain in healthcare is its ability to store data securely, including patient records and diagnostic results, on a decentralized ledger. This ensures data integrity, privacy, and accessibility by eliminating the need for a central authority. Moreover, blockchain facilitates secure interoperability and data sharing among healthcare stakeholders, empowering patients with fine-grained access control and

promoting innovation through incentivized data sharing. Blockchain integration with fog and edge computing opens new opportunities for P2P security and authorization [27; 28]. Permissioned DLTs are particularly appealing due to their customization using smart contracts, which are scripts that are validated as part of a transaction on the blockchain [29]. As a result, it secures and gives control access to health data by utilizing the inherent security provided by smart contracts.

1.2 Edge computing in ubiquitous healthcare systems

Edge computing is a process in which multiple sources of information are combined efficiently and makes it easier to perform operations at the edge devices (Internet of Medical Things (IoMT)) or send data using network technologies [30]. Edge computing consists of sensors, communication protocols, and software that connects technology-based systems utilizing online computer networks [31]. This approach can reduce latency and improve the responsiveness of systems, as well as reduce the amount of data transmitted over the network and reduce the load on centralized data centres. Additionally, it also increases privacy and security by allowing data to be processed locally, rather than being transmitted to centralized locations where it may be vulnerable to hacking or other security breaches. In other words, edge computing is a distributed computing paradigm in which data is processed and analyzed at edge gateways, closer to the data generating devices [2]. The goal of edge computing is to bring computing power closer to the devices and sensors that are generating the data, rather than relying on data being transmitted over a network to centralized data centres or cloud computing environments for processing and storage [32]. In [33], authors present an IoMT-based data processing and analysis framework through a series of network layers, extending a cloud-centric architecture. In addition to reducing network latency, moving computation to the edge of the network also reduces network load. Moreover, it is not always necessary to store raw data in many applications, as the results of the analysis are stored [25]. This means that all data need not be transmitted to the cloud servers. It is possible to use edge computing and fog computing to increase the security of personal data in certain scenarios, such as healthcare IoT [34]. Furthermore, the addition of intermediate layers between IoMT devices, cloud servers, and end-user applications means that there are more risks of security vulnerabilities, malicious code injection, and cyberattacks. A combination of edge computing and embedded intelligence offer artificial intelligence mechanisms at the edge using machine learning models to process and analyze data locally on edge gateways with limited resources [35].

With the increasing penetration of third-party services, security vulnerabilities, and cyber attacks become more prevalent in health monitoring applications [36]. In light of this, it is evident that a more secure method of exchanging and trading personal information and ensuring privacy is needed [37]. Recently, in [38], Peng *et*

al. presented a secure and efficient mobile healthcare system called Edgecare, which combines edge computing with a stackelberg optimization algorithm to achieve fair data trading by providing a hierarchical distributed architecture to manage and ensure healthcare data privacy. In another study [39], Lin *et al.* investigate how edge-AI can be used to convert sensory data into commercially valuable information. Toward the development of a knowledge market, the authors propose a proof-of-trade consensus mechanism and a non-cooperative game-based optimal knowledge approach. A recent review by Krittanawong *et al.* [40] gives a brief overview of the opportunities and challenges associated with integrating blockchain and AI to develop personalized medicine for cardiovascular disease. According to the authors in [41], the AI blockchain combination could provide a boost to the availability of reliable data for personalized medicines; nevertheless, the privacy of patients and data producers must be considered. Edge computing is employed for remote monitoring of patients at home through the process of ubiquitous healthcare. With continuous health monitoring, patients are not required to visit a hospital or a doctor's office every time they are unsure of their health status or if their condition changes. However, secure communication, the cost of service, handling, and many other types of costs should be addressed. This can be particularly valuable for people with chronic conditions, as it enables healthcare providers to track their progress and adjust treatment plans as needed [42]. For example, a patient with diabetes may wear a continuous glucose monitor that wirelessly transmits data to their healthcare provider, who can then adjust the patient's insulin dosage accordingly.

Healthcare providers are becoming increasingly concerned about the security of sensitive data that flows through the IoMT, such as confidential health information [43]. However, continuously generated data also raises concerns about privacy and security and the need for new regulations and standards to ensure data is used ethically and responsibly [44]. Moreover, users and governmental bodies are highly concerned about securing sensitive data, such as confidential health information, that flows through IoMT. [43]. Individuals' physiological data contains a great deal of personally sensitive information. Privacy is a major concern for healthcare applications, especially if IoT sensors and body area networks are integrated into the solutions [41].

Edge computing can improve privacy and security by allowing healthcare data to be processed locally, rather than being transmitted to centralized data centres where it may be vulnerable to hacking or other security breaches [45]. This concept introduces edge-intelligent systems by integrating lightweight machine and deep learning models at the edge layer to process data locally. This is particularly important in the healthcare sector, where patient data is highly sensitive and confidential. Because of the sensitivity and confidentiality of healthcare data, technologies handling healthcare data are subject to additional limitations and networks are appealing targets for cyber attacks due to their sensitive nature[46].

1.3 Core components of Edge Computing

There are three primary components of the edge computing ecosystem, including data acquisition, communication gateways, and server layer [47]. An overview is provided of the functional and structural characteristics of each of the layers in an edge-computing architecture [48; 49].

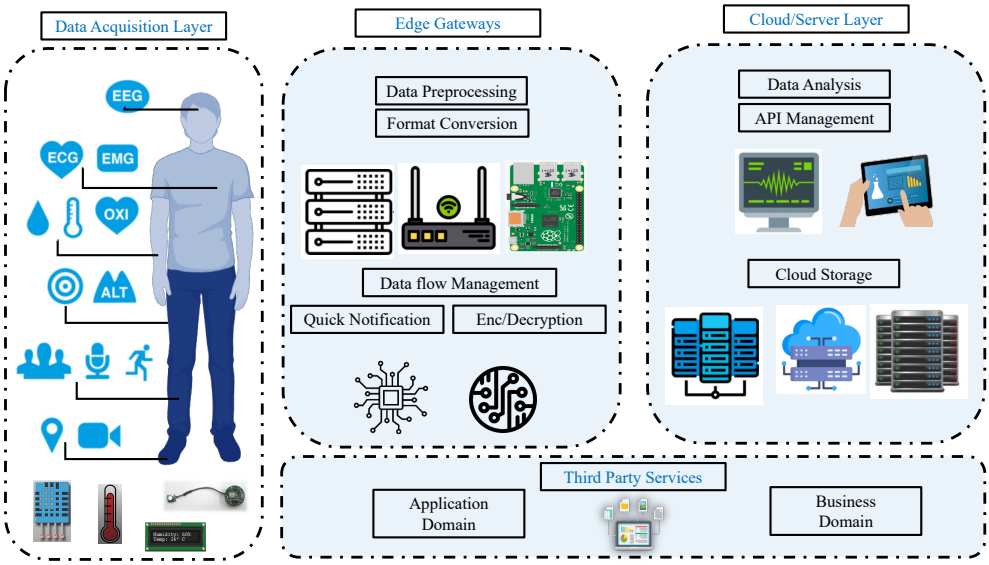


Figure 2: Core components of Internet of Medical Things

1.3.1 Data Acquisition Layer:

The data acquisition layer consists of biosensors, which consist of electro-mechanical devices that are used to acquire biological signals. These signals tend to be low amplitude contaminated with noise and require pre-processing and digitization. Preprocessing operations include amplification and filtration. If information is not well preserved and not lost during the data collection process, incorrect decisions may be made during diagnosis. By connecting people with healthcare systems, biosensors can transform traditional healthcare systems significantly and are capable of generating and transmitting a tremendous amount of data. Moreover, biosensors can speed up diagnostic procedures by automating data processing, and users can monitor their health. The data acquisition layer performs two major tasks before sending this generated data to the gateway layer[50].

1. **Pre-processing:** Biological signals are commonly weak and noisy, making processing challenging, therefore, amplifiers are used to amplify. The typical amplifier consists of an electronic circuit that produces a high-amplitude output signal when provided with a weak signal. The amplifier increases the signal frequency without changing other parameters, such as its morphology or frequency. A typical amplifier circuit uses a resistor, a conductor, and a transistor. The term ”single-stage amplifier” refers to an amplifier that includes a single transistor, and the term ”multi-stage amplifier” refers to one that includes multiple transistors. A multi-stage amplifier is widely employed in practical applications. An amplifier's gain can be used to determine how much amplification it provides [51]. In the post-amplification process, filtering removes unwanted parts from the signal. The filtering process employs a variety of different types of filters. High-pass, low-pass, band-block, and band-pass filters are all distinct types of filters. When a signal is subjected to a high-pass filter, the low-pass filter is employed to attenuate the signal's low-frequency components.

2. **A/D conversion:** Analog to digital conversion is achieved through A/D converters. Numerical values comprise the digital version of a recorded signal. Computers of today can store and process numerical values easily. Sampling and quantization are the main components of A/D conversion. Continuous time is converted to discrete-time by sampling. During sampling, the signal's value is measured at certain intervals of time. This procedure is called sampling [52].

1.3.2 Edge gateway:

Edge gateways connect sensors, smart devices, cloud systems and transport data from a local system to a cloud server or from a local computer to the cloud. Consequently, edge gateways serve as a communication link between connected intelligent objects in the field. Firstly, it does data normalization as a variety of sensors collect data in different formats. This stage is designed to minimize and filter the quantity of data that must be uploaded to the cloud. Pre-processing data minimizes the quantity of information that must be delivered, processed, and stored.

A sensor has the bare minimum capability of being interconnected to the internet. Unlike more expansive networks such as wide area networks (WAN), they can't be directly connected to them. As a result, the edge gateway provides connectivity to external networks over mobile data, BLE, wi-fi, or some other means of communication [53].

1.3.3 Server/cloud Layer:

As part of the edge ecosystem, servers are responsible for detecting abnormal activities and performing analyses. Following the acquisition and preliminary analysis of signals from the sensor nodes, the data is sent to the servers or cloud for analysis. Data mining is used in the IoMT field to classify bio-signals according to their important characteristics. For cloud servers, data processing involves signal identifying, enhancing, and separating features, followed by analysis of results. Typically, the data collected from sensors is time-domain data, which accumulates over a period of time. The time-domain signal contains data on the amplitude over a certain period. Signal-to-noise ratio measurements and event detection can be performed based on time and amplitude information [54].

A frequency-domain representation of time-domain signals is also possible. A frequency-domain representation of time-domain signals contains details about their power at various frequencies. For determining the amplitude of each frequency, a fourier transform is commonly used. The frequency-domain signal may be analyzed for several features, including central frequency, sub-band energy, and spectral bandwidth. Cloud servers work on data mining after analyzing and creating results based on that analysis. It plays an essential role in early diagnosis and ongoing monitoring of patients. Recommendation systems involve systematically extracting information from large sets of biodata and identifying patterns within the data. To construct a model that may aid in diagnosing some conditions, algorithms are applied to acquire available information [55].

Two broad classifications of data mining are supervised and unsupervised models. A supervised model is created by training an algorithm on labelled data and then using it to categorize unlabeled data. Labelled data is data in which each input variable has a corresponding output variable. On the other hand, no labelled data are available in unsupervised models. To identify relationships between the data, an unsupervised classification algorithm groups them into subgroups [56].

1.4 Ubiquitous Healthcare Systems and Blockchain

In recent years, distributed ledger-based technology (DLT), Blockchain, has led to significant advancements in the development and implementation of ubiquitous healthcare and digital health monitoring systems [57; 58]. In addition to its decentralized and secure nature, blockchain technology offers several benefits to ubiquitous healthcare systems that can improve their effectiveness and efficiency. Loopholes in centralized cloud systems turned the interest towards decentralized and distributed blockchain systems [59]. The tracing and transparent capabilities of blockchain represent a novel manifestation of privacy and security solutions. It constitutes a temporally sequenced series of blocks systematically arranged by the computational enti-

ties of diverse participants [60]. Within the blockchain framework, the immutability of blocks containing the transactional history among nodes in a P2P network is ensured by applying hash functions [61; 62]. Blockchain technology can be integrated as an embedded web layer to facilitate various functionalities, including but not limited to payment processing, currency exchange, token reception and distribution, digital asset transfers, and the execution of smart contracts [63]. In particular, distributed ledger technologies provide improved control of data by providing increased data transparency using blockchain [64], privacy by utilising distributed technologies [65], ownership incorporation by enabling blockchain-based solutions [66], and leveraging blockchain as a security measure [67].

Thus, Blockchain technologies can counter falsified transactional activities through increased traceability of goods [68]. However, distributed-based solutions encounter challenges related to high variability in structural requirements [69]. DLTs based scalability challenges were address with pegged sidechains [70], hierarchical trees [71], cross chains [72], and DAG [73].

For ubiquitous healthcare systems, blockchain technology offers the advantage of securely and immutably storing data. With the help of blockchain technology, healthcare data such as patient records, medical histories, and diagnostic results can be stored in a decentralized ledger. By encrypting and distributing the data across a network of computers, it ensures the data's integrity, privacy, and accessibility. Data breaches and unauthorized access are reduced by eliminating the need for a central authority in decentralized storage. Furthermore, the immutability of blockchain records ensures that healthcare data is accurate and tamper-proof, enhancing trust and reliability. In ubiquitous healthcare systems, blockchain technology facilitates interoperability and data sharing, another essential aspect. The blockchain allows healthcare data to be securely shared among stakeholders, including healthcare providers, researchers, and patients while maintaining data integrity and privacy. By utilizing smart contracts based on blockchain technology, patients can enable fine-grained access control, allowing authorized parties to access specific data [74]. As a result of this interoperability and data-sharing capability, collaboration can be enhanced, comprehensive patient care can be provided, and medical research and innovation can be supported.

In ubiquitous healthcare systems, blockchain technology also addresses issues of data provenance and auditability. A blockchain-based audit trail is created by timestamped, digitally signed, and linked transactions, creating an immutable record of all transactions recorded on the blockchain. With this feature, stakeholders can track healthcare data origins, modifications, and access histories, ensuring data exchange transparency and traceability. An auditable blockchain-based system enhances accountability, facilitates regulatory compliance, and enables quality assurance in ubiquitous healthcare. Further, blockchain technology can be utilized in ubiquitous healthcare systems to incentivize and monetize healthcare data. Patients

can receive rewards, tokens, or customized healthcare services by sharing anonymous health data with researchers, pharmaceutical companies, or other entities in exchange for sharing their anonymized health data [75]. A blockchain-based token or cryptocurrency can be used to encourage data sharing, promote medical research, and foster innovation in healthcare. As a result of this economic model, patients are more likely to engage in treatment, data sharing is encouraged, and new treatments are developed more quickly.

1.5 Privacy Concerns

Ensuring the security and privacy of personal data is crucial for maintaining trust and safeguarding sensitive information. However, traditional systems often struggle to adequately protect patient data, resulting in potential breaches and compromises in security and privacy. The NIST computer security handbook defines privacy as ”a term associated with confidentiality, ensuring that individuals maintain control over, or possess influence over, the collection, storage, disclosure, and access to information concerning themselves and their affairs” [76].

In the realm of ubiquitous healthcare systems, DLTs bring promising solutions to the security and privacy of patient data to maintain trust and safeguard sensitive information. Blockchain’s inherent tamper-resistant nature and its ability to facilitate transactions without intermediaries present it as a viable solution to various challenges in IoMT ecosystem to provide ubiquitous healthcare. Blockchain is increasingly utilized in healthcare applications to ensure the security, transparency, and immutability of data records through autonomous contracts. However, when dealing with private and confidential data, the issue of transparency and privacy requires further consideration. This is a critical aspect of integrating blockchain into our research. DLTs inherit the capabilities of security and privacy. However, some of the areas highlighted in table 2 show the need for improvement in terms of privacy.

Table 2: Security and Privacy Requirements and Techniques for Healthcare Blockchain

Security and Privacy Reqs	Default DLT support	Techniques	Need to be Enhanced
Consistency of txns	Yes	Consensus algorithms, Merkle tree, SPV	Yes (Security analysis of smart contracts)
Integrity of txns	Yes	Hash chained storage, Sig verification	No
Authenticity of txns	Yes	Signature	No
Tracking and auditing	Yes	Hash chained storage	No
Availability	Partial	Consensus algorithms	No
Anonymity of users	Partial	Public key as pseudonyms	Yes (Anonymous signature)
Patient-control	Partial	Public key as pseudonyms	Yes (encryption (by data owner)
Confidentiality	No		Yes (ABE, HE, SMPC, NIZK)
Access control	No		Yes (ABE, key management, HMAC)
Authentication of users	No		Yes (anonymous authentication)
Scalability	No		Yes

Table 3: Privacy-Concerned surveys from 2013 to 2019

Reference	Data	Usage	Location
[80]	×	×	✓
[87; 88; 89]	✓	×	✓
[90]	✓	×	✓
[91]	✓	✓	×
[92]	✓	×	✓
[93]	✓	×	×
[94]	✓	✓	×
[95]	✓	✓	×
[96]	✓	✓	×
[97]	✓	✓	×
[98],[99]	×	×	×

According to the findings of a comprehensive global survey on blockchain technologies [77], a notable percentage of Chinese data owners (62%) expressed apprehension regarding privacy issues when considering the adoption of blockchain technologies. Similarly, both Malaysia and the USA exhibited a similar level of concern, with 51% of business owners sharing these apprehensions. Moreover, among the surveyed companies across twelve different countries, 50% believed that blockchain should be exclusively utilized for private or internal purposes. These insights highlight the necessity of integrating the concept of data protection by design, which has recently emerged as a means of enhancing privacy [77], into blockchain systems. Previous research, such as the studies conducted in [78; 79], has explored the potential of blockchain technology in ubiquitous healthcare, demonstrating its capacity to enhance data security and privacy. Despite the predominantly human-centric nature of edge computing-enabled solutions [80], the concept of privacy within such systems remains insufficiently defined. Privacy provision poses a significant challenge for edge/fog developers [81; 82] due to various major barriers, including heterogeneity, mobility concerns, limited control over open-air transmission mediums, and complex business relationships within multi-owned scenarios. Based on authors in [83], privacy is typically categorized into three principal types: data privacy, usage privacy, and location privacy. Some studies consider identity privacy as a fourth type, while others include it as part of data privacy. Table 3 shows privacy-concerned survey studies in the last few years and their focused areas. Drawing from several references focused on privacy and edge/Fog Computing [84; 85; 86], we can define a private edge/fog system as ”a system capable of maintaining and preserving edge/fog properties while ensuring data privacy, usage privacy, and location privacy”.

1.6 Research Objective

Privacy and confidentiality is an ancient and well-warranted social value. Kay Redfield Jamison

Based on the above discussion the core objective is to optimize privacy, fine-grained access control, and data ownership rights of healthcare data producers in real-time health monitoring. The main design ideas pursued are edge computing and distributed ledger technologies for ubiquitous healthcare systems, where timely data sharing is crucial. The following are the key objectives of this study. Figure 3 includes the main contributions of this thesis to achieve our objectives.

- **Develop and implement a privacy-preserving healthcare framework:** To design and implement the *EdgeBot* framework, focusing on creating a ubiquitous healthcare system which integrates machine learning and deep learning algorithms at the edge device level, ensuring patient privacy and data security in time-critical scenarios.
- **Formulate and optimize a secure communication and fine-grained data access scheme:** To formulate and evaluate a hybrid on-chain and off-chain secure communication and data access scheme, utilizing private distributed ledger technologies and quantum-based security protocols, with a focus on scalability, interoperability, and resource efficiency.
- **Deploy and evaluate machine learning algorithms for real-time healthcare applications:** Develop and test resource-efficient models, formulate and implement resource-optimized 1D-CNN for multiclass arrhythmia classification on edge devices with two-channel ECG systems for rapid anomaly detection and real-time health monitoring. The proposed method should be efficient, fast (real-time classification), non-complex and simple to use (combined feature extraction and selection, and classification in one stage).
- **Design trustless peer-to-peer data trade system:** To design and implement a peer-to-peer trustless data trade flow system, utilizing lightweight communication and fine-grained exchange schemes on edge gateways leveraging private ethereum network.
- **Optimize scalable real-time data sharing and tracing system:** To develop and optimize a resilient and scalable real-time healthcare data sharing and tracing system among trusted stakeholders. Propose service optimization in hyperledger sawtooth, a mathematical foundation for determining the most efficient combination of databatch size, transactions per second (tps), resource utilization and network resources while ensuring the reliability of transaction commitment.

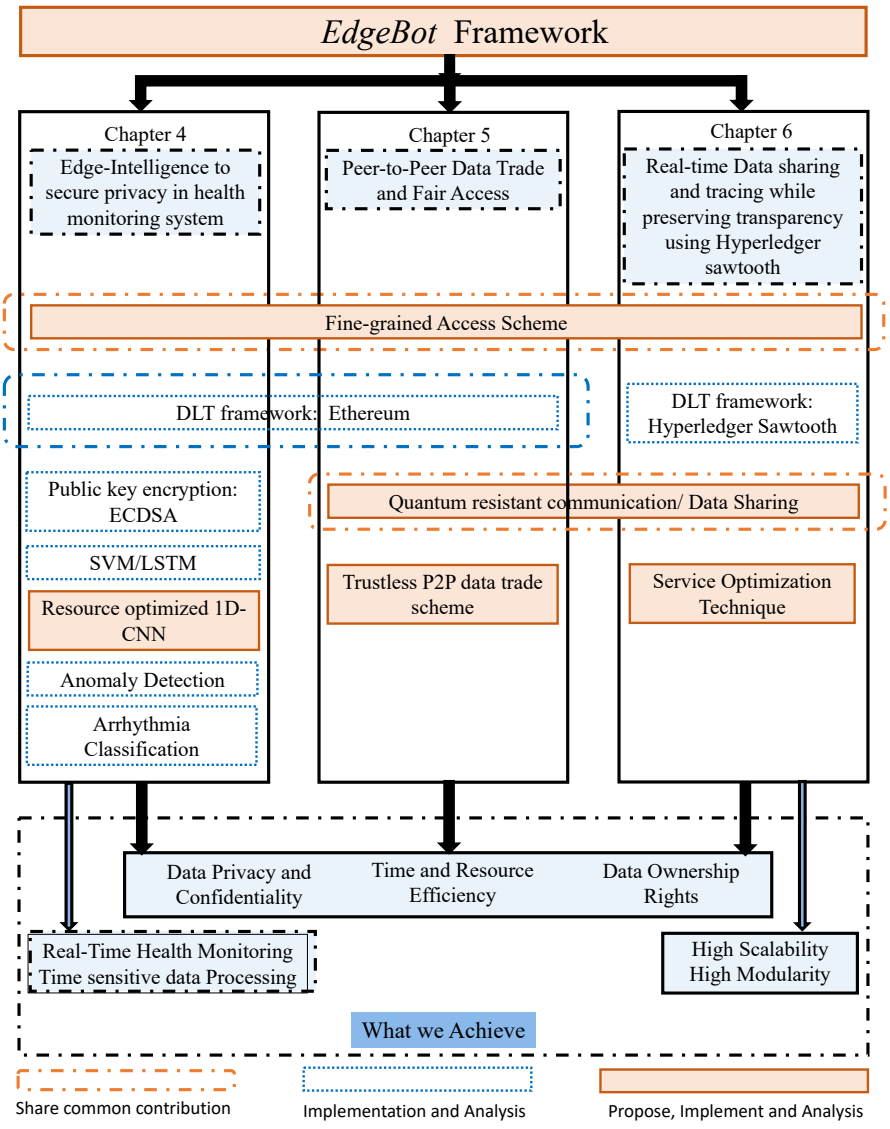


Figure 3: Technical Route map of contributions and objectives of this thesis

2 Preliminaries

2.1 Distributed ledger technologies for ubiquitous healthcare systems

Distributed ledger technologies (DLTs), such as blockchain, have the potential to revolutionize ubiquitous healthcare by providing secure, decentralized, and tamper-proof storage of health data. The first successful DLT application was bitcoin, which manages digital assets to solve problems of double spending and anonymity issues [100]. One of the key benefits of DLTs in ubiquitous healthcare is the ability to create a shared, interoperable health record that can be accessed by authorized parties in real-time [101]. This can enable a more efficient and effective healthcare system by reducing data silos and ensuring that all stakeholders have access to the same information. Another key benefit of DLT is the ability to improve data security and privacy. It can provide end-to-end encryption of health data and enable secure sharing of data with authorized parties, such as healthcare providers and owners. This can help to prevent data breaches and protect patient privacy [102]. In addition, DLT can enable new business models and revenue streams. For example, DLT-based platforms can be used to enable secure sharing of health data among owners, researchers and drug developers, which can lead to faster and more effective health monitoring and drug development. Table 4 provides a perspective mapping of blockchain as an entity of solution for security, privacy and tracking of operations.

Finally, DLT can enable ”smart contracts/chain codes” which are self-executing codes that can automate the process of sharing health data and other healthcare-related processes such as reimbursement, traceability and clinical trial management. Debe *et al.* [43] explores a decentralized IoT-based trust model that consists of fog nodes that possess a reputation based on the public availability of the fog nodes. The trust level is maintained by using feedback from existing users and interactions, which allows for a more transparent and reliable reputation management system compared to existing third-party-based systems. In another recent research work [103], Mazzei *et al.* developed a trustless industrial solution that enables industrial IoT to be integrated into the virtual world of digital twins. An interoperable tracking system for industrial applications is provided by this blockchain-based solution. To the best of our knowledge, our proposed architecture is generic compared to this and other solutions available, and we have therefore focused on a system-level view rather than on specific integrations, which will be the focus of our future work.

Cha *et al.* [104] developed a robust method of securely securing IoT device privacy preferences on a blockchain. As part of a comprehensive survey, Yang *et al.* [97] addressed several important challenges related to the integrated implementation of blockchain-based edge computers, including security and privacy, function integration, scaling, and resource management. Also, the authors provided use-case-based scenarios to describe state-of-the-art solutions and explore the technologies applicable to the blockchain-based edge computing domain. Bergquist *et al.* [105] introduced the concept of blockchain technology combined with smart contracts for use in building privacy-sensitive application

Edge computing based on blockchain presents another significant challenge such as scaling. Teutsch *et al.* [106] introduces a scalable decentralized autonomous blockchain solution based on edge computing, which can update billions of state updates in a second.

In addition to efficiently matching hierarchical ledger topology, it provides fixed withdrawal delays to ensure scalability by minimizing damage. Transaction handling is performed in an asynchronous manner. It was suggested that a similar approach be used, namely ziliqa [107], sharding to provide scalability. They divided the load into smaller shards so that transactions could be processed in parallel. Furthermore, a new scripting language for smart contracts and an execution environment were proposed as an underpinning architecture, along with sharding. In the context of high-scale parallel transactions, this makes a significant difference in the computation platform used. The tree-based distributed ledger has been proposed in [108], as a substitute for the chain framework in a similar research work. These researchers investigated the effects of network delay on double-spending. The GHOST protocol enhances the security of a transaction by reducing the confirmation time of transactions by modifying the Bitcoin protocol.

Table 4: Perspective mapping of Blockchain as an Entity of solution for security, privacy and tracking of operations

Sr. No	Entity	Relation	Entity	Perspective
1	Blockchain	*consists of*	Blocks	Distributed Ledger
2	Blocks	*contain*	Header	Chain Records
3	Blocks	*contain*	Body	Transactional Records
4	Blocks	*contain*	Reference	Distributed Trust
5	Blockchain	*maintains*	Historical Record	Timestamped Blocks
6	Historical Records	*preserves*	Record Irreversibility	Provenance
7	Record Irreversibility	*maintains*	Transparency	Reliability
8	Transparency	*uses*	Pseudonimityy	Anonimity
9	Blockchain	*includes*	Shared Database	Distributed Trust
10	Shared Database	*secures*	P2P Transmission	Validity
11	P2P Transmission	*includes*	Encrypted Transmission	Integrity

However, despite these potential benefits, some challenges need to be addressed

before DLT can be widely adopted in d/u healthcare. These include issues related to scalability, regulatory compliance, and data governance. Nevertheless, DLT has the potential to revolutionize d/u healthcare and bring about a new era of patient-centred, data-driven healthcare. Key components of blockchain modelling include consensus mechanisms, distributed ledger-based technologies and applications [109].

2.1.1 Permissionless versus Permissioned Networks

Blockchain-enabled solutions leverage different DLT frameworks, and some of them are designed for specific domains. Some frameworks are good for applications requiring permissionless architectures, and a few are specifically designed for permissioned networks. Permissionless blockchain solutions are publicly available for everyone to join, which creates scalability and performance issues [110]. For example, the transaction rate of bitcoin is limited to 7 transactions per second, which makes it incompetent to handle high-frequency trades. Furthermore, in permissionless networks, the transaction confirmation rate increases exponentially as the network expands. Along with the performance, it creates transaction cost issues. Due to the above-mentioned issues, many observers believe permissionless networks are unsuitable for large-scale non-financial applications [111].

In contrast to permissionless networks, permissioned blockchain networks identify each node, and administrator nodes are capable of removing malicious nodes [112]. These network models improve performance throughput by using more adaptive consensus protocols like practical byzantine fault tolerance [113], side chains [114], and blockchain-based edge-computing solutions [115]. In recent years, researchers proposed various permissioned blockchain frameworks such as ethereum [116], EOS [117], hyperledger [118], and ripple [119]. Permissioned blockchain networks are highly suitable for domain specific solutions and can be customized according to business needs and requirements. They support transactional-level privacy and network-level transparency.

However, it's worth noting that each framework may have its strengths and weaknesses according to specific domain requirements. For instance, ethereum is focusing on Layer 2 rollups, a technique that can support many transactions per second. In [120], ethereum blockchain is utilised along with off-chain storage to make tracking products easier in the pharmaceutical supply chain. The proposed method ensures the origin of data, removes the need for middlemen, and provides a secure, unchangeable history of transactions for everyone involved. Dwivedi *et. al* [121] proposed and developed a smart contract algorithm using directed graphs with six states and six actions. In addition to performing strong key management in smart contracts, it also achieves reasonable performance in terms of computation and communication overheads. The proposed protocol was robust and achieved reasonable performance regarding smart contract performance. Authors in [122] proposed a framework and

smart contracts to ensure data accuracy and authenticity within supply chains requiring highly accurate and authentic data. The proposed framework has yet to be tested in an industrial setting. Transaction and maintenance costs were not analyzed on the blockchain network. In [123], authors optimize data sharing among participating stakeholders using restricted blockchain using edge computing concept. They proposed a blockchain-based architecture and enabled a flexible configuration to securely share and access medical data between healthcare organizations, enabling the detection of probable epidemics, remote monitoring of patients and quick response times.

The tracing capabilities of blockchain represent a novel manifestation of DLTs. It constitutes a temporally sequenced series of blocks systematically arranged by the computational entities of diverse participants [60]. Within the blockchain framework, the immutability of blocks containing the transactional history among nodes in a P2P network is ensured by applying hash functions [61; 62].

Blockchain technology can be integrated as an embedded web layer to facilitate various functionalities, including but not limited to payment processing, currency exchange, token reception and distribution, digital asset transfers, and the execution of smart contracts [63]. In particular, DLTs provide improved control of data by providing increased transparency [64], privacy by utilising distributed technologies [65], ownership incorporation by enabling blockchain-based solutions [66], and leveraging blockchain as a security measure [67].

However, distributed-based solutions encounter challenges related to high variability in structural requirements for each involved entity [69]. Scalability and reliability issues due to complex requirements and large-scale implementations [124], integration with existing conventional solutions based on dispersed technologies and interoperability challenges.

2.1.2 Modular Distributed Ledge Technologies

Hyperledger frameworks, on the other hand, offer a modular design, allowing for extensive customization. Authors in [125] present hyperledger fabric-based application for drug discovery. The proposed application allows organizations to upload, update, view, and verify their contributions. Each contribution is given a unique identifier using a secure hash algorithm, and the design also allows regulatory authorities to issue certificates confirming the ownership of contributions. Uddin *et.al* [126] investigated tracking related challenges and proposed hyperledger fabric-based solution, but their proposed solution lacks any kind of implementation test-beds which leads to the feasibility of the proposed framework. Work of [127] proposed hyperledger fabric-based agricultural supply chain to enhance the trust between the end consumer and the product they are purchasing, proposed solution *"grainchain"* enables the tracking of the grain from the farmer to the retailer. Authors in [128] present

an IoT-based seafood supply chain by leveraging hyperledger sawtooth framework. Consumers can track seafood and notify drivers through a seafood supply chain when the container temperature exceeds a specified level. The authors recorded the shipment log as a blockchain transaction by connecting the application, the devices, and the blockchain database. Another study is proposed [129] as generic ownership and traceability of products using hyperledger sawtooth. The proposed system can prevent counterfeit goods from entering the supply chain.

A comparison of ethereum with hyperledger fabric and sawtooth is presented in table 5. Hyperledger is seen as one of the most advanced technologies for creating consortium chains. Unlike other blockchain technologies, hyperledger is designed to help build applications that meet the standards of businesses. It stands out because of its unique way of managing permissions, its ability to control access very precisely, its flexibility with different consensus algorithms, and its speed in processing transactions. This makes hyperledger particularly useful for systems in healthcare facilities, where its features can enhance the sharing and tracking of information. Hyperledger sawtooth's event system can potentially lead to more efficient and effective operations in permissioned domain specific business logic while ethereum platforms perform well in private permissionless domains. Hyperledger sawtooth's broadcasting and relaying of events across the network enables real-time updates and actions. Due to its modularity, each participating entity can define its business logic and interact through transaction families. Transaction families work similarly, such as smart contracts in ethereum.

Table 5: Comparison of Hyperledger Sawtooth, Hyperledger Fabric and Ethereum

Feature	Hyperledger Sawtooth	Hyperledger Fabric	Ethereum
Ledger Type	Permissioned	Permissioned	Permissionless
Smart Contract Functionality	Transaction Processors	Chaincode	Ethereum Virtual Machine
Consensus Algorithm	Pluggable Framework	Pluggable Framework	Proof of Stake
Governance	Linux Foundation	Linux Foundation	Public

2.1.3 Symmetric and Asymmetric Key cryptography

Cryptography is the science of securing communication and data through the use of mathematical techniques, ensuring confidentiality, integrity, authenticity, and non-repudiation of information.

Symmetric key encryption techniques, or shared key cryptography, are encryption schemes based on a single shared secret between communication parties. A symmetric key algorithm allows a single party (the encryptor) to encrypt a message x with a key k, resulting in a ciphertext $y = ek;(x)$. By applying the inverse op-

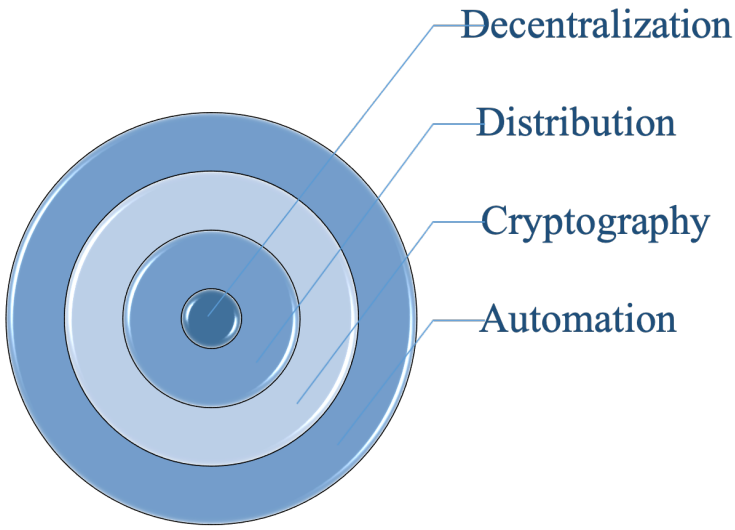


Figure 4: Building blocks of Blockchain.

eration to the ciphertext y, a second party (the decryptor) can decrypt it and recover the message $x = dk; (y)$. Without access to the key k in a network controlled by an adversary, recovering the original message x is computationally impossible.

Stream algorithms and *block algorithms* are two types of symmetric encryption algorithms. In streams, each bit of plaintext is encrypted individually using a key stream. Even though block algorithms encrypt entire blocks at a time with the same key, block sizes range from 64 to 256 *bits* (for example, DES, AES, and Blowfish). According to their mode of operation, block algorithms can provide different levels of security. The message can only be guaranteed to be confidential when a block cypher algorithm is employed in a cypher block chaining mode, for example. They reduce the complexity of encryption and are very secure and highly effective. However, they have many disadvantages:

- **Key distribution:** In symmetric encryption, the security of the key determines the degree of security. It is considered best practice for key exchange to take place over a secure channel or through a secure key exchange protocol. The use of a secure channel or a secure key exchange protocol for key exchange is considered best practice. Through key-distribution protocols, it is possible to create a safe channel between two users who do not ordinarily share a secret, but the protocol needs the parties to rely on a secure channel to a central provider.

- **Key management:** Managing just a few keys can significantly reduce management overhead. It becomes extremely challenging to manage and distribute

keys on a large scale when there are many users.

- **No protection against cheating:** Two parties get the same level of capability when sharing a key. Therefore, both can identify messages between each other. However, neither of them could prove to a third party that the other had delivered a particular message because they might both generate it. Asymmetric cryptography can be used to achieve this attack against non-repudiation.

Asymmetric key cryptography differs from symmetric-key cryptography in that it employs a separate set of principles for encryption and decryption. In asymmetric key algorithms, there are two sorts of keys: the encryption key, also known as the public key, which is used just for encryption and does not need to be kept secret, and the decryption key, sometimes known as the private key, which is needed for decryption. Asymmetric key cryptography, in comparison to symmetric key cryptography, is a completely new concept that was only developed in the 1970s to address some of the flaws of symmetric cyphers.

The cryptography provided by asymmetric algorithms can offer several capabilities, including the establishment of shared keys, non-repudiation, integrity, and identification of messages. Asymmetric cyphers are secure due to the difficulty of solving a mathematical problem, such as solving *one-way functions*, for example. One-way functions are simple to compute for any input but complex to reverse for any image corresponding to a random input. The function is generally used in this context to produce the key pair and related security settings. The *discrete logarithm problem* and *integer factorisation problem* are the two most common one-way functions in asymmetric cryptography. According to the problem of integer factorisation, the composition of two large prime products is reasonably easy to calculate. However, decomposing that product is incredibly difficult. A cryptographic scheme based on *Rivest-Shamir-Adleman* is RSA. When the parameters are very large, discrete logarithms become more difficult to compute modulo a prime.

Diffie-Hellman Key-Exchange (DHKE) was the first asymmetric-key scheme to be published, and it was based on a discrete logarithmic issue. DHKE, an algorithm extensively used in encryption protocols such as Transport Layer Security (TLS) and Secure Shell, solves a key distribution problem (SSH). Its goal is to use the Diffie-Hellman algorithm to allow two parties, Alice and Bob, to share secret keys for a symmetric cypher via an insecure communication channel. The premise behind DHKE is to compute the value k and use it as the joint secret value.

$$k = g^{ab}(mod\ p) = (g^a)^b(mod\ p) = (g^b)^a(mod\ p) \tag{1}$$

The following steps are included in the protocol.

1. In order for *Alice* and *Bob* to cooperate, they must determine the domain parameters p and g, where p is a huge prime and g is a primitive root modulo p,

also known as common parameters.

2. *Alice* calculates A = g^a*(mod p)* and sends it to *Bob* as a secret large random number g.

3. *Bob* decides a secret random number b, and then computes B = gb*(mod p)*, which he used to sends to *Alice*.

4. According to *Alice*, k = B^a*(mod p)* = $(g^b)^a$*(mod p)* is her shared secret.

5. Using the formula k = B^b*(mod p)* = $(g^a)^b$*(mod p)*, *Bob* plots the shared secret k.

6. A secure communication channel can be established between *Alice* and *Bob* using k now.

2.1.4 Post Quantum Cryptography

Public-key cryptography is a cryptographic system that relies on mathematical problems that are conjectured to be hard to solve. The most widely used examples of public-key cryptography are RSA, which is based on the integer factorization problem, and elliptic curve cryptography (ECC), which is based on the discrete logarithm problem. However, both RSA and ECC are vulnerable to polynomial-time attacks using a quantum computer.

To defend against this threat, research is focusing on post-quantum cryptography (PQC), which aims to develop cryptographic algorithms that are resistant to attacks by quantum computers. The Kyber protocol is an IND-CCA2 secure key encapsulation method (KEM) designed to resist cryptanalytic attacks with future powerful quantum computers. It allows two communicating parties to establish a shared secret without an IND-CCA2 attacker in the transmission system being able to decrypt it. The protocol is constructed by applying a Fujisaki-Okamato style transformation on InnerPKE, which is the underlying IND-CPA secure Public Key Encryption scheme. The $InnerPKE$ private key is a vector s over R of length k, which is small in a particular way. The public key in the Kyber protocol consists of two values: a k-by-k matrix x, which is sampled uniformly at random t, which is calculated as $t = x * s + e$, where e is a suitably small masking vector.

To ensure the security of its operations, Kyber uses cryptographic primitives such as pseudorandom functions (PRF), extendable output functions (XOF), key derivation functions (KDF), hash functions (H), and generators (G). These cryptographic primitives play a crucial role in maintaining the integrity and confidentiality of the shared secret.

- **Modular Lattice Error Learning (MLWE):** The security of Kyber hinges on the Module Learning With Errors (MLWE) problem. The special structure of

the matrix x contributes to the security of the key generation process in a significant way. It provides three variants of kyber512,Kyber768 and Kyber1024 and their matrix structures are explained below:

$$\begin{pmatrix} \alpha_{1,1} & \alpha_{1,2} \\ \alpha_{2,1} & \alpha_{2,2} \end{pmatrix} \tag{2}$$

$$\begin{pmatrix} \alpha_{1,1} & \alpha_{1,2} & \alpha_{1,3} \\ \alpha_{2,1} & \alpha_{2,2} & \alpha_{2,3} \\ \alpha_{3,1} & \alpha_{3,2} & \alpha_{3,3} \end{pmatrix} \tag{3}$$

$$\begin{pmatrix} \alpha_{1,1} & \alpha_{1,2} & \alpha_{1,3} & \alpha_{1,4} \\ \alpha_{2,1} & \alpha_{2,2} & \alpha_{2,3} & \alpha_{2,4} \\ \alpha_{3,1} & \alpha_{3,2} & \alpha_{3,3} & \alpha_{3,4} \\ \alpha_{4,1} & \alpha_{4,2} & \alpha_{4,3} & \alpha_{4,4} \end{pmatrix} \tag{4}$$

In Kyber512, the matrix x is a negacirculant matrix. This means that each row of the matrix is a cyclic shift of the previous row, and the first element of each row is the negative of the last element of the previous row. The special form of this matrix means that the matrix multiplication is consistent with the multiplication when performed modulo $X256 + 1$. This allows the MLWE problem to be represented as a matrix equation over the ring $Z[X]/<q, X256 + 1>$, which effectively increases the complexity of the problem, making it harder for an attacker to solve it where each $\alpha_{i,j}$ is a negacirculant matrix of the form.

- An element $x \in \mathbb{Z}_q$ is converted to an d-bit integer by $\mathrm{Compress}_q(x, d)$. An d-bit integer x is converted to a $\mathbb{Z}_q$ element by Decompress $q(x, d)$. They are defined as follows:

$$\mathrm{Compress}_q(x, d) = \left\lceil \left(2^d/q\right) \cdot x \right\rfloor \bmod 2^d,$$
$$\mathrm{Decompress}_q(x, d) = \left\lceil \left(q/2^d\right) \cdot x \right\rfloor$$

where $\lceil a \rfloor$ is the closest integer to a. When each function is applied to a polynomial (or a vector/matrix of polynomials), it is applied to each coefficient individually.

Moreover, a polynomial (or a vector/matrix of polynomials) is serialized to byte arrays by using $\mathrm{Encode}_\ell()$ function, where ℓ is the bit-length of each coefficient. On the other hand, $\mathrm{Decode}_\ell()$ is the inverse of $\mathrm{Encode}_\ell()$, and it deserializes the byte arrays to polynomials. Lastly, Parse ( ) converts a byte stream to the NTT representation of a polynomial in R_q.

- The noise is sampled from a centered binomial distribution B_η for $\eta = 2$ or $\eta = 3$. For a sample $(a_1, \ldots, a_\eta, b_1, \ldots, b_\eta) \leftarrow \{0,1\}^{2\eta}$, the output is computed as

$$\sum_{i=1}^{\eta} (a_i - b_i)$$

The possible outputs are $\{-2, -1, 0, 1, 2\}$ when $\eta = 2$, and $\{-3, -2, -1, 0, 1, 2, 3\}$ when $\eta = 3$, respectively.

Using B_η, a polynomial $f = \sum_{i=0}^{255} f_i X^i$ in R_q can be sampled by sampling each coefficient f_i deterministically from 512η-bit output $(\beta_0, \ldots, \beta_{512\eta-1})$ of a pseudo-random function:

$$f_i = \sum_{j=0}^{\eta-1} (\beta_{2i\eta+j} - \beta_{2i\eta+j+\eta}) \quad i = 0, 1, \ldots, 255.$$

For this purpose, Kyber uses a function namely CBD_η, which takes 512η-bit input and outputs the corresponding polynomial.

2.1.5 Digital Signatures and Hash Functions

Digital signature's goal is to guarantee message integrity and authentication, also known as data origin authentication, as well as non-repudiation. To accomplish this, public key cryptography is used. Public key cryptography is used to create digital signatures, which are created by utilizing a secret key sk to sign the message and a corresponding public key pk to validate the signature. In essence, the signature is made up of a big integer number using two operations $sign(-)_{sk}$ and $ver(-)_{pk}$ that can only be generated by the owner of the private secret key sk.

The following are the properties of the keys:

- It is impossible to compute the private signing key sk given the public signature verification key pk.
- There is a digital signature function $sign(-)_{sk}$ that provides a signature $sign(x)_{sk}$ from a message x and a private signing key sk.
- The signature verification function $ver(-)_{pk}$ takes the signature $sign(x)_{sk}$ and the public verification key pk and returns TRUE if the signature was successfully computed with sk and FALSE otherwise.

Data integrity is protected by hashing, or simply by utilising the hash function. Hash functions are linear-time methods that can be used to generate a fixed-size digest or hash value from a collection of variables or data. A secure hash function satisfies the following requirements: *Deterministic*—the same input always produces

the same output—and efficient—the output is computed promptly.
Distributed- uniformly dispersed across the output range, implying that similar data should not be associated with similar hashes.
Ore-image-resistance- it must be physically impossible to locate the input sk using the hash value *h(x)* and collision resistance, no two distinct inputs x and y must produce the same hash.

$$h(x) = h(y) \Rightarrow x = y \tag{5}$$

Implementation of a digital signature requires the use of a hash function. The message needs to be signed by the hash, as it is considered a unique representation of the message. Security and performance will be enhanced as a result, as:

- Digital signatures perform cryptographic operations very slowly when compared to symmetric cryptography.
- A digital signature is a way to encrypt a document without using the public key. Because the signature is the same length as the message, it is referred to as a digital signature.

In general, the hash value computed over a document is unique; another document can't have the same hash value, so a hash that is signed is equivalent to signing the document in its entirety.

2.2 Deep Learning Techniques in ubiquitous Healthcare

Machine learning algorithms encounter challenges when dealing with unstructured and high-dimensional inputs, such as raw spatial or temporal information. To deal with such problems deep learning algorithms have emerged as a powerful tool. [130; 131]. At the core of deep learning is the concept of neural networks. Neural networks are mathematical models that are designed to mimic the behaviour of neurons in the brain. They consist of interconnected layers of artificial neurons, known as nodes or units, which process and transmit information. The behaviour of neural networks can be described using mathematical equations that capture the relationships between the inputs, outputs, and parameters of the network [132].

The training process involves feeding the neural network with labelled examples and updating its parameters based on the errors it makes. This iterative process continues until the network achieves a satisfactory level of accuracy. These equations allow us to understand and analyze the behavior of the network, and to make predictions about its performance. In the limit, even a single *fully-connected* (FC) hidden layer of such neurons is capable of learning any input function [133]. Consequently, such models are frequently regarded as universal function approximators

[132]. Each node receives input from the previous layer b using learned weights wi applies an activation function to the input, and passes the output to the next layer:

$$f_{\text{neuron}} = \phi\left(\sum_i w_i x_i + b\right)$$

One commonly used mathematical model in deep learning is the backpropagation algorithm. This algorithm uses the chain rule of calculus to compute the gradients of the network's parameters concerning a given loss function. These gradients are then used to update the parameters in a way that minimizes the loss function, thereby improving the performance of the network. Another important mathematical concept in deep learning is optimization. Optimization algorithms, such as stochastic gradient descent, are used to find the values of the network's parameters that minimize the loss function. These algorithms iteratively adjust the parameters in the direction of steepest descent, gradually reducing the error until convergence is achieved.

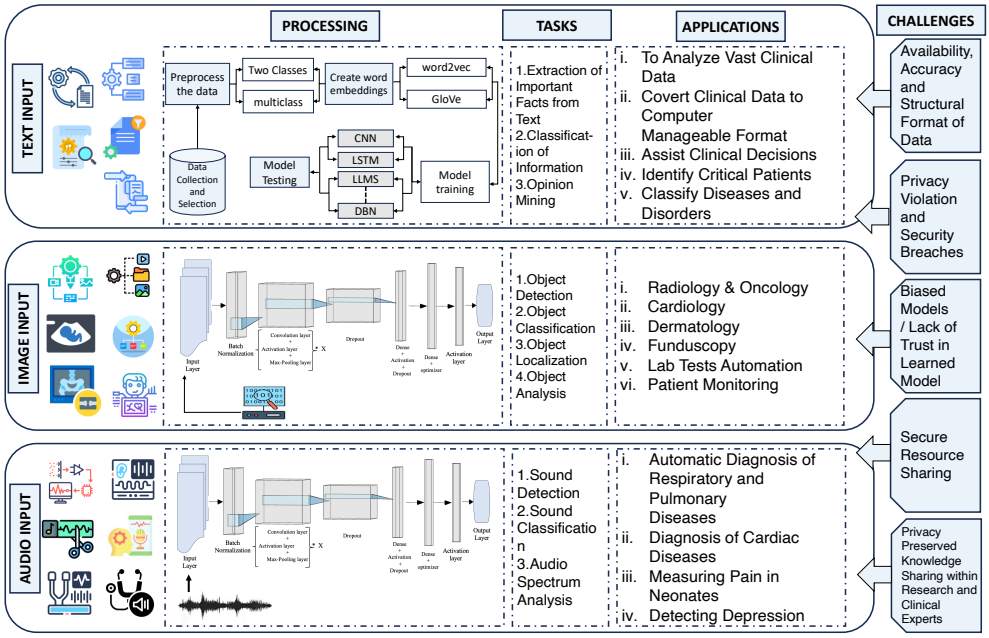


Figure 5: Artificial Intelligence in Healthcare

Deep learning techniques have emerged as a powerful tool in the field of ubiquitous healthcare systems. Deep learning techniques, a subset of machine learning, have shown great potential in analyzing large amounts of healthcare data and extracting meaningful insights. One of the key applications of deep learning in ubiquitous healthcare systems is in the analysis of medical images. Deep learning algorithms can be trained to detect and diagnose diseases from medical images such as X-rays,

MRI scans, and CT scans. By analyzing patterns and features in these images, deep learning models can provide accurate and timely diagnoses, helping healthcare professionals make informed treatment decisions. Another area where deep learning techniques are being applied in ubiquitous healthcare systems is in the analysis of physiological signals. Wearable devices such as heart rate monitors, electrocardiograms (ECGs), and electroencephalograms (EEGs) generate vast amounts of data that can be used to monitor a person's health condition. Deep learning algorithms can be used to analyze these signals and identify patterns that may indicate the presence of certain health conditions or abnormalities. This can enable early detection and intervention, leading to improved patient outcomes.

Table 6: Recent ubiquitous healthcare systems Artificial intelligence and blockchain technologies

Ref	Year	Direction	Technique	Framework	Contribution
[134]	2021	Secure Healthcare Framework	Optimal deep learning-based secure blockchain	Intelligent IoT and healthcare diagnosis model	Ensures secure transactions, hash value encryption, and facilitates medical diagnosis
[135]	2022	Medical Data Transmission and Diagnosis	ResNet-v2, SVM and MFO with ECC	Securely transmits medical images and disease diagnosis framework	Healthare data transmission and diagnostic model.
[136]	2022	Data Sharing in Industrial Healthcare	ZKP with smart contract-based consensus and self-attention-based Bi-LSTM	Permissioned blockchain and smart contracts for secure and efficient data-sharing framework	Attack detection process and secures sensitive data.
[137]	2023	Secure Data Transmission in IoT Healthcare	ZKP with off-chain storage via IPFS, Deep Sparse AutoEncoder with Bi-LSTM	Novel and scalable blockchain architecture	Robust intrusion detection system.
[138]	2023	Smart Healthcare System Using Metaheuristics	ABE combined with KMC, FNN, ANN, and KMNN	Healthcare monitoring framework with secure data storage	Enables real-time health monitoring while maintaining data security and confidentiality.
[139]	2023	Multimedia Medical Data Processing	ECC, ECDH, ECDS and AI	Image processing architecture comprising edge, fog, cloud, and private blockchain	Reduces computational costs and offers robust security, ensuring higher efficiency in enc and dec processes.
[140]	2023	Healthcare for Smart Cities	Blockchain, IoMT, and IoV concepts	Smart vehicles for real-time medical data transmission to healthcare professionals	Report real-time accident locations to nearby healthcare services.

Table 7: Recent ubiquitous healthcare systems Artificial intelligence and blockchain technologies (Continue)

Ref	Year	Direction	Technique	Framework	Contribution
[141]	2023	Intelligent blockchain for smart healthcare systems	Distributed Random Forest and SVM, with optimal feature selection	Private blockchain-based selective sharing mechanism with mutual authentication	Improves transparency, integrity, and traceability, while ensuring low latency and high throughput
[142]	2023	Privacy-preserving diagnostic platform	ECC, RSA, and AES for sensitive medical information.	Access control system for privacy-sensitive medical data	Enhanced efficiency in computing time and blockchain's capability to fend off various security threats.
[143]	2023	AI-blockchain based healthcare framework	LSTM models to classify malicious or non-malicious	Blockchain and AI within a secure 6G network framework	Improved scalability and latency of AI-driven framework for trusted healthcare management
[144]	2024	Real-time pandemic (COVID-19 and pneumonia) screening using tomographic analysis	Improved ResNeXt-50 and segmentation decoders from multi-domain input data	Edge computing for multi-site data fusion to improve accuracy and scalability	Introduces practical tool for real-time COVID-19 segmentation in smart cities for enhanced clinical decision-making.
[145]	2024	Secure healthcare data using DL with blockchain	LSTM models with reinforcement learning to detect and counter malware attacks	Framework focuses on efficient malware detection and validation using blockchain	Prevent familiar and novel attacks in industrial cyber-physical systems

In addition to medical image analysis and physiological signal analysis, deep learning techniques are also being used in other aspects of ubiquitous healthcare systems. For example, natural language processing techniques, which are a branch of deep learning, can be used to analyze patient records and extract relevant information for diagnosis and treatment. Deep learning models can also be used to predict patient outcomes, such as the likelihood of readmission or the risk of developing complications, based on various patient factors [146].

Despite the promising applications of deep learning in ubiquitous healthcare systems, there are still challenges that need to be addressed. One of the main challenges is the need for large amounts of labelled data to train deep learning models effectively. Healthcare data is often sensitive and confidential, making it difficult to collect and share. Additionally, the inter operability of deep learning models is a concern, as they are often considered black boxes that make predictions without providing explanations. Deep learning techniques offer great potential for enhancing ubiquitous healthcare systems. From medical image analysis to physiological signal analysis and beyond, deep learning algorithms can provide valuable insights and improve patient care. Table 7 describes some recent integrated studies of deep learning and blockchain for ubiquitous healthcare systems. However, further research is needed to address challenges related to data privacy, availability and model interpretability. With continued advancements in deep learning, we can expect more innovative applications in the field of ubiquitous healthcare systems in the future.

2.2.1 Convolutional Neural Networks

Convolutional neural networks (CNNs) are a class of deep learning algorithms that are specifically designed for processing structured grid data, such as images or time series data. CNNs have been widely used in computer vision tasks, such as image classification, object detection, and image segmentation. The key idea behind CNNs is to leverage the local spatial correlations present in the input data. Unlike traditional neural networks, which treat the input data as a vector and process it using fully connected layers, CNNs use convolutional layers to extract local features from the data. This allows CNNs to capture the hierarchical structure of the input data and learn meaningful representations. In a CNN, the input data is typically represented as a multi-dimensional grid, where each element corresponds to a pixel or a time step [147].

The first layer of a CNN is a convolutional layer, which applies a set of learnable filters to the input data. Each filter is a small matrix that is convolved with the input data, producing a feature map. The filters are learned through a process called backpropagation, where the network adjusts the filter weights to minimize a given loss function. After the convolutional layer, CNNs often include pooling layers, which downsample the feature maps by taking the maximum or average value in a

neighbourhood. This helps reduce the dimensionality of the data while preserving the important features. The pooled feature maps are then passed through additional convolutional and pooling layers to extract more abstract features.

$$f_{\text{feature}}(t) = \sum_{a=a_{\min}}^{a_{\max}} x(a)w(t-a)$$

The final layers of a CNN are typically fully connected layers, which take the flattened feature maps as input and output the desired predictions. These layers are responsible for making the final decision based on the learned features. The output layer is usually a softmax layer, which produces a probability distribution over the different classes in a classification task. One of the main advantages of CNNs is their ability to automatically learn hierarchical representations from the data [148]. By stacking multiple convolutional layers, CNNs can capture increasingly complex features, which allows them to achieve state-of-the-art performance on various computer vision tasks. Additionally, CNNs can handle input data of different sizes, thanks to the use of convolutional operations that are translation invariant. CNNs are a powerful class of deep-learning algorithms that have revolutionized computer vision tasks. By leveraging the local spatial correlations in the input data, CNNs can learn hierarchical representations and achieve impressive performance on image classification, object detection, and other related tasks [131].

The key layers of a typical CNN structure are:

- **Convolutional layer:** This layer performs a convolution operation on the input image to extract features. It uses a set of learnable filters (also known as kernels or weights) to detect patterns in the input data. The filters are moved across the input image and perform element-wise multiplication with the values in the input image and sum them up. The output of the convolution operation is a feature map that represents the detected features.

- **Activation layer:** After the convolution operation, the activation layer is used to introduce non-linearity into the model. The activation function, such as ReLU, applies an element-wise activation function to the output of the convolution operation.

- **Pooling layer:** The pooling layer performs a down-sampling operation on the feature maps to reduce the spatial dimensions and computational complexity. It also helps to make the features invariant to small translations. Commonly used pooling operations are max pooling and average pooling.

- **Fully connected layer:** The fully connected layer takes the output from the previous layers and performs a final classification or regression. The output from the previous layers is flattened and connected to one or more neurons, which perform the final prediction.

1D Convolutional Neural Networks (1D CNNs) and 2D Convolutional Neural Networks (2D CNNs) are types of CNNs that operate on 1-dimensional and 2-dimensional data, respectively. 1D CNNs are used for processing sequential data such as audio signals, text data, and time-series data. They operate on 1-dimensional signals and convolve over time to extract features. The filters used in 1D CNNs have a width of 1 and a height equal to the number of time steps.

2D CNNs, on the other hand, are used for image classification and object recognition tasks. They operate on 2-dimensional images and convolve over both height and width dimensions to extract features. The filters used in 2D CNNs have a width and height greater than 1, and can detect patterns in both spatial dimensions. Bidirectional Convolutional Neural Networks (2D-CNNs) are a type of 2D CNNs that process input data in both forward and backward directions. They are used for sequential data processing tasks where the context from both past and future is important for making a prediction. In a bidirectional 2D CNN, the input data is processed in two separate branches, one processing the input data in the forward direction and the other processing it in the reverse direction. The outputs from both branches are then concatenated and used as the input for the next layer. In table ??, characteristic comparison of 1D, 2D and Multi-dimensional CNNs is presented.

Table 8: Comparison of 1D, 2D, and multi-dimensional networks

Feature	**1D CNN**	**2D CNN**	**3D CNN (Multi-dimensional)**
Input Data	time-series, signals	images, spectrograms	video, volumetric data
Conv Filters	along 1 axis	along 2 axes	along 3 axes
Comp Complexity	Low	Medium	High
Spatial Context	local dependencies in 1 dims	local dependencies in 2 dims	spatio-temporal dependencies across 3 dims
Data Structure	1D arrays	2D matrices	3D tensors
Memory Requirements	Low	Moderate	High
Training Time	Fast	Moderate	Slow
Common Applications	Signal processing (e.g., ECG, speech), NLP	Image classification, face recognition	Video action recognition, 3D object classification
Advantages	Simple, efficient for 1D data	Effective for image-related tasks	modeling complex data structures
Disadvantages	Limited to 1D relationships	Cannot handle temporal or volumetric data	High computational cost, complex to implement

2.2.2 Long Short Term Memory Networks

Long short term memory (LSTM) networks are a type of recurrent neural network (RNN) that are designed to address the vanishing gradient problem in traditional RNNs. LSTM networks are widely used in the field of natural language processing, speech recognition, and various other sequence learning tasks. The vanishing gradient problem occurs when the gradients in a traditional RNN become extremely small, making it difficult for the network to learn long-term dependencies [149; 150]. This is especially problematic when dealing with sequences that are several steps long, as the information from earlier steps can quickly diminish and become irrelevant. LSTM networks overcome this issue by introducing memory cells and gating mechanisms. The memory cells allow the network to store and access information over long periods, while the gating mechanisms control the flow of information within the network.

The key components of an LSTM network are the input gate, forget gate, memory cell, and output gate. The input gate determines which information from the current input should be stored in the memory cell. The forget gate decides which information from the previous memory cell should be discarded. The memory cell stores information over time and is responsible for maintaining long-term dependencies. Finally, the output gate determines which information from the memory cell should be outputted as the final prediction. The gating mechanisms in LSTM networks are implemented using $sigmoid$ and $tanh$ activation functions. Sigmoid functions are used to control the flow of information, with values ranging from 0 to 1. A value of 0 indicates that the gate is closed, while a value of 1 indicates that the gate is open. $Tanh$ functions are used to regulate the values in the memory cell, allowing for the storage and retrieval of information. During training, LSTM networks learn to adjust the weights and biases of their gates and memory cells to optimize their performance on a given task. This is done using backpropagation through time, where the gradients are propagated backwards through the network to update the parameters. By introducing memory cells and gating mechanisms, LSTM networks address the vanishing gradient problem and have become a popular choice for various sequence learning tasks in the field of machine learning [151].

2.2.3 Scaled-Dot-Product Attention and the Transformer Architecture

In the previous section, it was mentioned that RNNs, including LSTMs, can face the vanishing/exploding gradient problem which can hinder learning. This problem becomes even more pronounced when processing long sequences with long-term dependencies. However, this problem has been addressed in the field of NLP, particularly in machine translation, through the use of encoder-decoder architecture. This

architecture consists of two LSTMs, the encoder and decoder, which turn the input sentence into a feature vector and then use it to produce the output sentence, respectively. In 2014, Bahdanau et al. [152] introduced an attention mechanism that allowed the decoder to focus on the most relevant parts of the input. Authors in [153] proposed a new architecture, the transformer, that relied solely on the attention mechanism and eliminated the limitations of RNNs. The transformer has since been adapted and forms the basis of modern machine translation models like BERT and roBERTa and language-based general intelligence models like GPT-2 and GPT-3.

Transformer is powered by the multi-head attention layer that uses scaled-dot-product attention and takes three matrices as input and projects them into a subspace through learned matrices to obtain the matrices Q, K, and V. The scaled-dot-product attention mechanism performs a differentiable dictionary lookup using the dot product as a similarity measure between the query and key. In Scaled-Dot-Product Attention, the relevance or similarity between two tokens is computed by taking the dot product of their respective feature vectors. However, to ensure numerical stability during training, the dot product is scaled by the square root of the dimension of the feature vectors. The attention weights are then obtained by applying a softmax function to the scaled dot products. These weights indicate how much attention should be given to each token in the sequence. The tokens with higher attention weights are considered more relevant. The Transformer architecture is built upon the concept of self-attention, where each token in the sequence attends to all other tokens. This allows the model to capture long-range dependencies between tokens, which is crucial for tasks such as machine translation and text summarization. In the Transformer architecture, multiple attention heads are used to compute attention independently. Each attention head focuses on different aspects of the input sequence, allowing the model to capture different types of relationships. After computing the attention weights, a weighted sum of the input tokens is computed, taking into account their respective attention weights. This weighted sum, along with the original input token, is then passed through a feed-forward neural network to generate the final output. The use of self-attention in the Transformer architecture has several advantages. It allows the model to capture both local and global dependencies, making it effective for tasks that require an understanding of long-range relationships. Additionally, the parallel nature of self-attention makes it highly efficient for training on modern hardware accelerators. The Transformer architecture, built upon self-attention, has achieved state-of-the-art performance in various natural language processing tasks and has become a cornerstone in the field.

2.3 Summary

This chapter briefly discussed some preliminary technologies, key characteristics, and working mechanisms. Additionally, it incorporated some latest state-of-the-art

studies related to the proposed techniques by summarizing their main contributions and limitations. With the emergence of ubiquitous healthcare systems based on edge computing using distributed ledger technologies refers to healthcare infrastructure and real-time healthcare services while protecting the reliability and integrity of available information. In recent years, the realm of edge intelligence provides promising solution to privacy and security risks associated to ubiquitous healthcare systems. This chapter covers blockchain technologies and their cryptographic primitives. It also provides initial studies on machine learning and deep learning protocols which we are going to use in our proposed framework.

3 Development of Framework as Research Environment

3.1 Overview

This chapter presents a detailed ubiquitous healthcare framework *"EdgeBot"*. This framework is a topology based on edge intelligence (implementation of machine learning and deep learning algorithms on edge devices) leveraging private distributed ledger technologies. To enhance the experience of ubiquitous healthcare, the framework emphasizes early warning systems in critical situations, continuous monitoring, and personal data storage to build patient history, enabling privacy and peer-to-peer data sharing without using a third party. This hybrid ubiquitous healthcare system based on edge gateways allows the extension of a private ethereum network to resource-constrained edge devices. Edge gateways work as a bridge to provide interaction between different layers of the network. Moreover, edge gateways can define access control rules for their data, such as data trades and sharing with third parties. Its versatility, scalability, and capacity to accommodate a wide range of healthcare applications make *"EdgeBot"* a competitive solution in the line of ubiquitous healthcare. It is a real-time ubiquitous healthcare system that encompasses healthcare providers, healthcare professionals (doctors), and patients. Specifically, in-home continuous monitoring is explored from a user-centric perspective, focusing on the growing tendency of online and quick medical care and edge-based intelligence. Additionally, it focuses on data producers' ownership rights and data owners' privacy. The main contributions of this chapter include the following:

- We propose a ubiquitous healthcare framework, *EdgeBot* while preserving patients' privacy and data security. It is a topology of edge device-level intelligence (implementation of machine learning and deep learning algorithms on edge devices).
- We construct a hierarchical architecture and computing model based on the MAPE-K framework.
- Formulation of privacy-preserved hybrid on-chain and off-chain secure communication and data access scheme.
- Utilize private distributed ledger technologies as service framework

3.2 Proposed Framework of *EdgeBot*

3.2.1 Enhanced MAPE-K Model

Our proposed computing model leverages MAPE-K (Monitor-Analyse-Plan-Execute over a shared Knowledge) [154] as an established framework to facilitate automated management and self-adaptive behavior originally introduced by IBM [155]. The base model consists of four distinct computing components, all with access to a shared knowledge base. Our proposed model is an extension of the base model with an addition to edge intelligence component. The base model is illustrated in figure 6, and the enhanced MAPE-K model is depicted in figure 7. Components of the proposed model are described in Table 9.

Table 9: Computing Components

Component	**Explanation**
Monitor	This component, located closest to the sensing tier, is responsible for acquiring and aggregating data from various sources.
Analyse	The Analyse component processes and models the acquired data, extracting meaningful insights and patterns.
Plan	Based on the analysis performed by the previous component, the Plan component constructs a procedure or strategy for the system to follow.
Execute	The Execute component is responsible for implementing the planned procedure and executing necessary changes in the system to achieve the desired outcome.
Edge Intelligence	The Edge intelligence component takes real-time input from monitor component, implements trained AI algorithms to plan and execute commands according to user requirements. It helps to automate the whole process of action by analyzing the components. is responsible for implementing the planned procedure and executing necessary changes in the system to achieve the desired outcome.

The MAPE-K model facilitates efficient automated management and adaptive behaviour within distributed systems by incorporating these components and enabling access to shared knowledge. The hierarchical IoT architecture uses the four computing components to enable its functionality. We propose an additional component called edge intelligence to the computing model using EGs-based intelligence to implement a closed-loop technique. The role of the edge-intelligence component is to dynamically reconfigure the system's settings based on feedback received from the user's condition. The hierarchical structure of the enhanced computing model is explained in section 3.3. The perception layer encompasses the Monitor component,

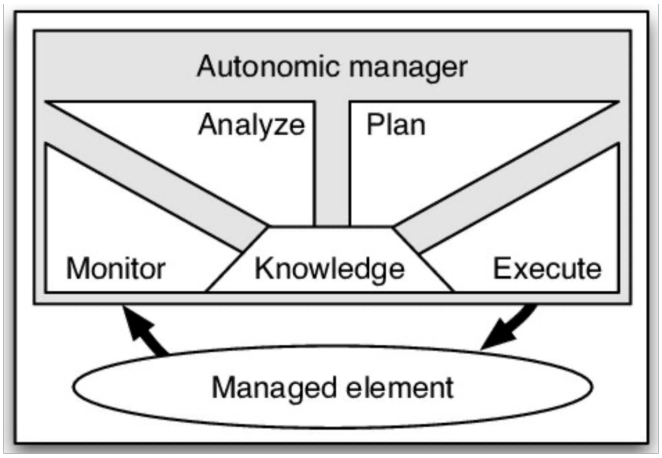


Figure 6: MAPE-K base model initially proposed by IBM [157]

which operates as an intermediary bridging the sensors and other computational entities.

The gateway layer comprises three distinct components: Plan, designated for localized decision-making; Execute, aimed at configuring system behaviour according to pre-decided situations; and System Management, dedicated to refining system configurations. This entity is responsible for training an inference model, referred to as a hypothesis function, derived from user-generated data.

The Plan component implements local decisions and establishes the system's procedures. The initial step involves processing the streaming data received from the system management module, including feature extraction. The Execute component, on the other hand, is in charge of actuating the system and providing feedback to other units. Upon detection of any abnormalities, users are promptly notified. These notifications are also dispatched to patients and healthcare providers as part of this process. Subsequently, the execute command proceeds to update the system management component. This allows the system's configuration to be adjusted based on the user's current state. In conclusion, the computing component provides feedback to the analysis component, such as a report on local decisions. This feedback is used to retrain the classifier.

Distributed ledger scheme ethereum [156] is utilized at EGs level along with its turing complete language to incorporate user instructions at EG level. Smart contracts convey terms and conditions based on situation and threshold values. Side chains [114] are utilized along with a private ethereum network for data storage to enhance data reliability. Side chains work at EGs and save only hashes of data blocks and metadata at a local level. In contrast, complete blocks are stored only in distributed data storage locations in an encrypted form using public key encryption.

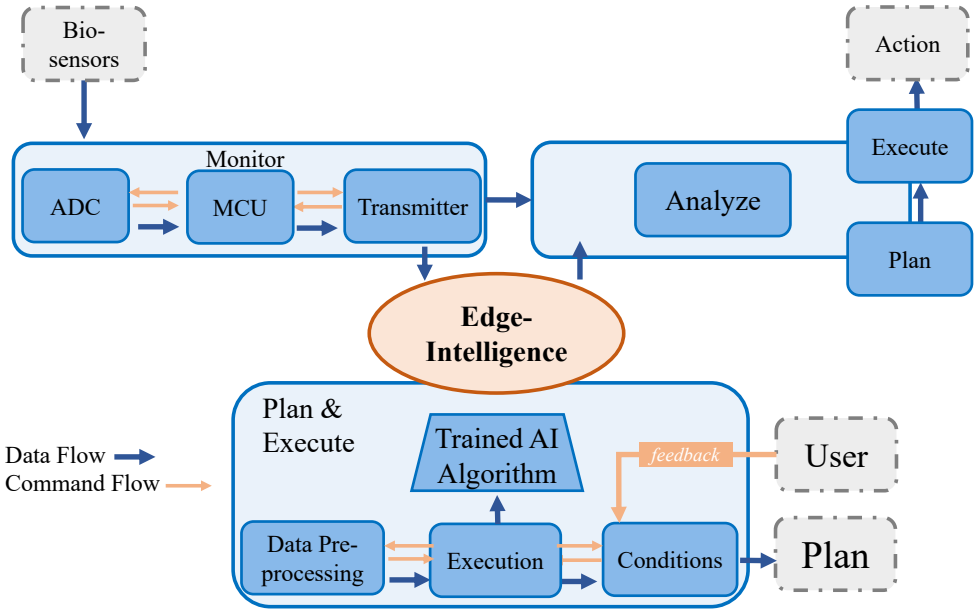


Figure 7: Enhanced MAPE-K model by utilizing edge intelligence unit

3.3 Architecture of *EdgeBot*

Our proposed framework *EdgeBot* extends the blockchain paradigm by incorporating side chains at edge gateways (EGs), thereby obviating the necessity for multiple network layers. This approach empowers resource-constrained computing devices by enabling them to make autonomous decisions and process data at the edge, consequently eliminating the requirement for intermediaries in decision-making processes and data processing. In addition, along with the continuous monitoring system, it incorporates an early warning system of patients' health; it has a process for storing information about medical history efficiently by storing raw and processed information to use as patient history. In practice, the data generated by sensors is filtered before being written into the blockchain. Thus, we can reduce the blockchain size and be able to run on gateway devices. System decision-making, threshold limits, and agreements are incorporated by utilizing turing complete ethereum smart contracts. Using smart contracts, after a specified time interval T data is sent to EGs, processed, and predicted using the edge-intelligence unit of our computing model previously explained in Figure 7. In case of an emergency or unusual patterns, trigger an alert to specified devices through alert notification, hospital, or any pre-defined entity to ensure prompt treatment.

The layered architecture of *EdgeBot* comprises the Perception (bio-sensors), edge gateways (single board computers, SBCs), and Application layers. The discrete functions and roles of these computational components embedded within three layers are

elucidated in figure 8.

As *EdgeBot* utilizes a P2P interaction topology, the devices are fully autonomous, enabling them to interact directly with third parties and exchange data directly. Smart contracts are used to implement policies governing data exchange, access policies and ownership rights. It is possible to scale a system using side chains for parallel transactions. By doing so, they also limit the bandwidth and power edge gateways used when updating the blockchain. To secure data, the devices use an elliptic curve integrated encryption scheme (ECIES), which generates a unique key for every saved batch of data by utilizing a child key derivation function (CKD). Our solution also provides double authentication for devices sending requests frequently, which makes our solution resistant to such attacks. To share these unique keys, we have implemented an elliptic curve digital signature algorithm (ECDSA) for communication. The proposed *EdgeBot* architecture serves as a generic model to secure IoT data ownership rights while preserving users' privacy. It can be implemented in smart home devices, the industrial internet of things (IIoT), smart health applications (including precision medicine, clinical trials, or accuracy diagnosis), knowledge-based systems, or developing different IoT products. The components and functionalities of each layer are explained in the next sections.

3.3.1 Perception Layer

Perception layer includes individual sensors, cameras, actuators, or other light nodes that possess too little computational capabilities to participate in a network for data processing, trade, and participation in the blockchain network directly. While the term "perception layer" is not commonly used in IoT architecture literature, it likely refers to the specific devices within the perception layer that are responsible for capturing and interpreting data from the environment. For instance, in uHealth, sensors might measure vital signs like heart rate, blood pressure, or oxygen levels. They are designed to interact with the physical world and convert those interactions into digital signals that can be processed by the system. This layer is essential because it provides the initial input that informs higher levels within the framework. As a result, sensor nodes rely on the EGs they are connected to, which act as intermediaries between the perception and application layers.

3.3.2 Edge Gateways

EGs consist of a local network (BLE, radio access points, wifi) and single-board computers such as Raspberry Pis or Intel UPs with sufficient computing power to run algorithms designed for devices with limited resources. EGs incorporate different roles, such as miner nodes and manager nodes (in terms of blockchain network), which work as arbitrators, regulatory authorities, and transaction and data handlers

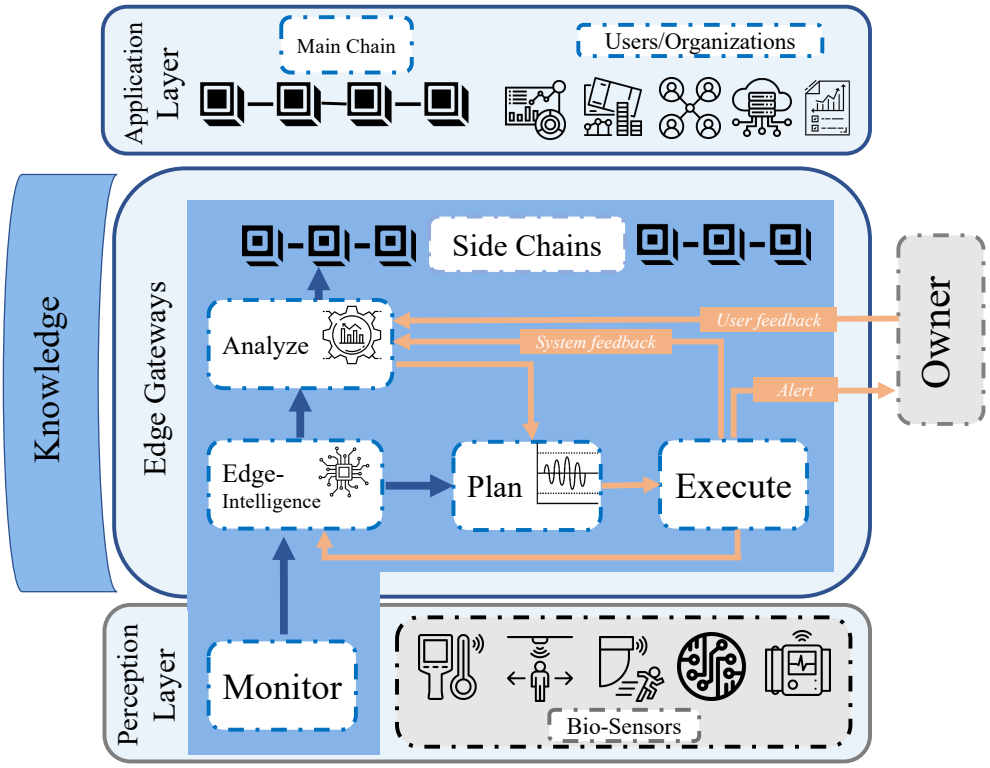


Figure 8: Proposed Computing Model of *EdgeBot*

for the ethereum blockchain. The manager nodes represent most of the network nodes that are autonomously operated (pre-defined smart contracts or scripts), as well as cloud services and applications running on other high-performance computing platforms. Due to their large number of parallel operations, they require higher computational capabilities. EGs provide the final link between the perception layer and the application layer. EGs can connect to local networks and are located at the edge of every sidechain. A private ethereum network, TESTRPC, has been used to verify the proposed smart contract. These edge gateways in sidechains store data hash as records of previous data batches and validation records. The Analyze component of the computing model describes them in detail. To reduce the vulnerability of data compromise, sensor nodes are connected to only one edge gateway. An adversary may only be able to gain access to one particular edge gateway if one sensor node has been compromised. In this way, single encrypted connections provide increased levels of data security, ensuring the right to access and own data and adding sensor nodes to the backbone chain of *EdgeBot's* network. On the other hand, the manager nodes can communicate without any external supervision with any other

manager nodes in the network in an entirely autonomous manner. The nodes can also be configured to act as customized access control points.

3.3.3 Blockchain Layer

The blockchain layer works as a service layer at EGs and the application layer. Ethereum private network is created at the application level, and side chains are incorporated at EGs level. Sidechains are separate blockchains that run parallel to the main network, connect to it through a two-way bridge, and are designed for efficient transaction processing. Sidechains work independently, having different histories, development roadmaps, and design considerations. Validator nodes on sidechains verify and process transactions, produce data blocks, broadcast block hashes, and store the network state. They are also responsible for maintaining consensus across the network and securing it against malicious attacks. Sidechains are an effective way to increase the capacity of the ethereum network, offering efficient and cheap transaction processing. However, they do involve trade-offs. While they help scale the network, they sacrifice some of the decentralization and security inherent to ethereum.

The process of adding new transactions to the blockchain is governed by the consensus process. Various consensus algorithms have been utilized by researchers and explained previously in Chapter 2. We employ 'proof of stake' (PoS) in our proposed architecture to update the blockchain. The process begins with network participants generating transactions with the collected data to form a block. Next, nodes sign the transaction using a private key and broadcast it to the network. When the network receives a block, trusted nodes locate the source public key to verify the signature. Following successful signature verification, the trusted nodes assess the media access control (MAC) address and compare it to the received one for a second round of evaluation. After successful authentication, the trusted nodes broadcast the validated blocks with PoS identification. Then, individual network users verify the PoS information to add blocks to the chain. Upon accepting a valid block, the user calculates a hash value to link the next block and retrieves the previous block's hash value to store in the current block. The PoS approach adheres to a traditional blockchain working model with lightweight block verification. It eliminates the reverse hash function used in the PoW approach, making the transaction process lighter. As a result, the ethereum network can be efficiently integrated with resource-constrained devices in *EdgeBot*.

3.3.4 Application Layer

A top-level layer of the architecture is the application layer. Data owners, buyers, and health professionals are provided with different interfaces. It refers to the data access layer for third parties and the data storage infrastructure employing the blockchain

and an IPFS network. Our method suggests a fully distributed architecture that eliminates centralized third parties needing to handle or store patient data. However, due to the high cost, storing all health data on the blockchain is not feasible. As a result, we have opted to use a distributed storage system like IPFS for most health data storage. The data stored in IPFS is identified by its hash, which is permanently stored in the blockchain's smart contract. Users who need the data can refer to the smart contract, find the hash, and request the corresponding health record from IPFS, ensuring secure and immutable data storage. Blockchain is a database that is shared, append-only, fault-tolerant, and distributed. It keeps a collection of records or blocks that are transparent and accessible to all nodes in the blockchain. However, these blocks are immutable and cannot be deleted.

In the system proposed, the blockchain network serves four purposes: (1) it verifies the authenticity of health professionals through a smart contract, (2) it records patients' health records metadata and ensures they are only accessible to authorized healthcare professionals, (3) it provides resilience against data breaches and availability failures, and (4) it provides peer-to-peer data exchange and trade topologies and gives a complete right of transaction policies to the owner of data. The InterPlanetary File System (IPFS) is a protocol for storing distributed data in a peer-to-peer network. Files stored in an IPFS-based network are referenced by a unique hash calculated from their content. These files are immutable, meaning if a file is altered, IPFS treats the altered file as a new object and calculates a new hash. The IPFS network is utilized here due to the high financial cost of storing large files on the blockchain. In this scenario, the IPFS network stores the health records, while the blockchain only stores the data hash and metadata.

The data users are those who have been granted appropriate authorization by the patient, such as doctors, dentists, nutritionists, specialized clinics, and others. Data buyers refer to the entities that buy some portion of data for research and development purposes without knowing the identities of data owners. In terms of communication, processing, and data dealing, the ethereum network is used as a service platform for running smart contracts. Generating a genesis file enables the creation of a private ethereum blockchain network. As part of the device addition process, a pair of public and private keys are generated, which are then used to identify each edge gateway in the network.

3.3.5 Proposed Access Scheme

To implement secure communication based on TLS 1.3, we incorporate the Kyber512 [158] key encapsulation method and dilithium3 [159] as the digital signature algorithm for a robust and secure approach to sharing private keys of encrypted data blocks. This scheme leverages the strengths of cryptographic techniques to ensure the confidentiality, integrity, and authenticity of private keys while allowing

authorized nodes to access them when needed. Kyber is known for its post-quantum security and plays a pivotal role in this access scheme by encapsulating private keys. When a node wants to access a private key, Kyber generates a secure encapsulation, which is transmitted over the network. The encapsulation process ensures that even if an eavesdropper intercepts the communication, it cannot derive any meaningful information about the private key without the appropriate decryption key, which remains securely stored on the target node.

Dilithium3, on the other hand, is employed as the digital signature algorithm to authenticate the access request. When a node seeks access to a private key, it sends a request accompanied by a digital signature generated using dilithium3. This signature serves as proof of the authenticity and authorization of the request, making it highly resistant to forgery or tampering. Upon receiving the request and the associated dilithium signature, the recipient node can verify the signature's validity using the corresponding public key. If the signature is valid, the requestor is authorized to access the private key encapsulated by Kyber. Only then is the encapsulated private key decrypted and made available by the requesting node. This combined use of Kyber and dilithium ensures a multi-layered security approach. Kyber safeguards the confidentiality of the private key during transmission, while dilithium guarantees the authenticity and authorization of the access request. The scheme provides a robust solution for securely sharing private keys within a networked system, even in the face of potential threats, thereby bolstering the overall security and integrity of the network.

Data is encrypted using AES-128 [160] along with its child derivation keys (CKD) [161] before storing in some storage place. This randomly generated master key from AES128 serves as the foundation of the encryption scheme. Once the master key is established, the system proceeds to derive a child key, which is used specifically to encrypt the data batch. To do this, we employ HMAC-based Key Derivation Function (HKDF) [162] to incorporate additional contextual information, such as a unique identifier for the data batch or other relevant details. This combined input is hashed using a secure hashing algorithm, creating a new, distinct child key unique to the data batch. This child key derivation process ensures that each data batch is encrypted with its key, enhancing security.

With the derived child key, the system encrypts the data batch symmetrically using the AES encryption algorithm after a defined time interval T. This involves cypher Block Chaining and a randomly generated Initialization Vector (IV). The data is also padded to match the AES block size and then encrypted with the child key and IV. The resulting ciphertext, along with the IV, is stored together. This approach ensures that even if one child key is compromised, the security of other data batches remains intact, as each batch is encrypted with a unique and derived child key. This encrypted data can now be safely stored on a distributed storage device. We use python libraries $crypto$, $hmac$ and $hashlib$ to implement our proposed data access

scheme.

3.4 Summary

This chapter introduces *EdgeBot*, a comprehensive framework for ubiquitous healthcare, focusing on edge intelligence, which involves deploying machine learning and deep learning algorithms on edge devices. The framework leverages private distributed ledger technologies to enhance early warning systems, continuous monitoring, and personal data storage, enabling patient history creation and secure peer-to-peer data sharing without intermediaries. *EdgeBot* extends a private ethereum network to resource-limited edge devices, using edge gateways to facilitate interaction between network layers and define data access controls. The system supports real-time healthcare, particularly in-home continuous monitoring, aligning with the increasing demand for online medical care and edge-based intelligence. The chapter's key contributions include proposing the *EdgeBot* framework, establishing a hierarchical architecture based on the MAPE-K model, and developing a privacy-preserved hybrid communication scheme using private distributed ledger technologies. This framework emphasizes patient privacy, data security, and ownership rights, offering a scalable and versatile solution for diverse healthcare applications.

4 Edge-Intelligence to secure privacy in health monitoring system

This chapter presents the implementation of our proposed framework *EdgeBot* as a continuous monitoring system of individuals along with edge-level intelligence for quick medical aid. It uses methods with minimal computational demands suitable for patient self-monitoring and preventive healthcare utilizing IoT devices. Moreover, this implementation includes the storage of processed medical information of individuals utilizing a private blockchain network, ethereum, for patient history. This medical information can be shared with legal guardians of patients, personal doctors, and for research purposes in an anonymous way while protecting individuals' privacy.

The system utilizes a range of bio-sensors, including a heart rate sensor, temperature & humidity sensor, and an ECG monitoring system for detecting abnormalities and identifying arrhythmias. Contextual sensors are strategically placed throughout the home to observe the patient's surrounding environment and physiological signals. The integration of advanced technologies in smart monitoring systems and the continuous generation of data is crucial for promoting health and independence, particularly in the context of the growing elderly population and remote inhabitants. It is widely acknowledged that chronic diseases become more prevalent with age, and a significant proportion of individuals with chronic ailments are aged over 65 years. Initially, a linear Support Vector Machine (SVM) approach is employed to execute abnormality detection via simple binary classification on the signal. In a separate scenario, resource-efficient 1-directional convolutional neural network is proposed and employed as a deep learning algorithm to identify various arrhythmias, constituting a multi-class classification task. The main contributions of this chapter include the following:

- Machine learning algorithms deployed at the edge device level for rapid anomaly detection.
- We propose a resource-efficient 1-directional convolutional neural network (1D-CNN) for multiclass classification of arrhythmia.
- Implementation of real-time anomaly detection and arrhythmia classification using 2-channel ECG systems.

- Ethereum based private blockchain network integration at edge gateways level.
- Extensive set of experiments are carried out to authenticate the efficiency of our proposed system.

4.1 Proposed Optimized 1DCNN

We proposed resource-optimized one directional convolutional neural network (1D-CNN) architecture for the ECG Arrhythmia classification at the edge device level. This architecture comprises two main components: the extraction phase and the classification phase. The extraction phase encompasses batch normalization, convolution, activation, and max-pooling layers, while the classification phase is characterized by flattened, fully-connected, and softmax layers.

The 1D-CNN architecture accepts an input matrix of dimensions $M * N$, where M represents the length of the time window under consideration, and N denotes the number of ECG channels. The initial step involves applying a batch normalization layer, which aims to standardize the input data by minimizing internal covariate shifts. Each 1D convolutional layer employs a kernel of variable dimensions $Q * N$, where Q signifies the temporal window that the filter covers. These kernels move exclusively along the elements of a single dimension of the input pattern. In the one-dimensional convolution layer, each neuron is connected to a local window from the preceding layer, referred to as the receptive field, which shifts along the time axis and shares synaptic weights. The mathematical representation of a 1D convolutional layer is as follows:

$$y_r = f\left(\sum_{q=1}^{Q}\sum_{n=1}^{N} w_{qn}x_{r+q,r+n} + b\right) \tag{6}$$

where y_r is the output of unit r of the filter feature map of size R (R equals to M in the case where stride=1), x is the two-dimensional input portion overlapping with the filter, w is the connection weight of the convolutional filter, b is the bias term, and f is the activation function of the filter, which in this case is $reLu$.

This model facilitates the reduction of the number of weights and aids in the generalization process. The neurons oriented vertically represent the evolution of the input data over time, which is dependent on the receptive field and delay values. The number of neurons along the horizontal axis can be manually defined, enabling the transformation of input features into a higher-order sequence. For each neuron, the rectified linear unit function (reLu) is applied to return the weighted sum of the input data if it is positive and zero otherwise.

To calculate the dimension of the filter feature map after the convolution operation (R), the following formula can be utilized:

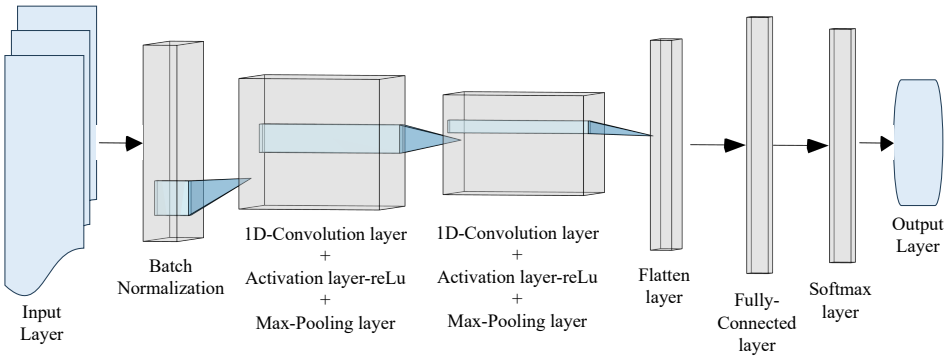


Figure 9: Resource efficient 1-D Convolutional neural network

$$R = \left[\frac{M - (K-1) + 2}{S}\right] \tag{7}$$

S is the stride (the number of positions skipped by each shift of the filter during convolution).

The overall features of the CNN architecture are depicted in figure 9. Batch normalization (BN) is applied, which involves normalizing the input to the next layer, typically leading to a significantly increased learning speed and notable regularization effects that enhance the network's generalization. BN operates differently during training and testing. During training, BN normalizes and zero-centers the input based on the entire batch, allowing the model to learn the optimal scaling of the input.

To normalize and zero center, the input BN estimates the parameter dependent mean mu and variance $sigma^2$ computed over the batch. The zero-centered normalized value $hatX(i)$ for each instance is computed as $xi = 10^{-5}$ to avoid zero divisions. BN adds a further step during training, using trained parameters, to further scale and offset the values as needed.

During testing, the mean μ and variance σ^2 parameters cannot be computed based on the batch, so the algorithm uses the values computed with a moving average during training.

$$\begin{aligned} \boldsymbol{\mu} &= \frac{1}{b}\sum_{i=1}^{b}\mathbf{X}^{(i)} \\ \boldsymbol{\sigma}^2 &= \frac{1}{b}\sum_{i=1}^{b}\left(\mathbf{X}^{(i)} - \boldsymbol{\mu}\right)^2 \end{aligned} \tag{8}$$

In the context of batch normalization (BN), the number of instances in the batch is denoted by b, and each instance is represented by $X(i)$. Following the computation of the zero-centered normalized value $\hat{X}(i)$ for each instance, BN introduces an

additional step during training. This step utilizes trained parameters to further scale and offset the values as required.

$$\hat{\boldsymbol{X}}^{(i)} = \frac{\boldsymbol{X}^{(i)} - \boldsymbol{\mu}}{\sqrt{\boldsymbol{\sigma}^2 + \xi}}. \tag{9}$$

The element-wise multiplication, denoted by $\otimes$, involves multiplying each input value by the corresponding scaling parameter, denoted by γ. The offset parameters, denoted by β, are also learned during training. This process allows for the optimal scaling and shifting of the normalized values, enhancing the network's performance by adjusting the distribution of inputs to fall within a specific range, thereby facilitating more effective training.

$$\mathbf{z}^i = \boldsymbol{\gamma} \otimes \hat{\boldsymbol{X}}^{(i)} + \boldsymbol{\beta} \tag{10}$$

The second convolutional layer is with the same parameters as the first convolutional layer. Pooling is crucial for CNNs to reduce the input size, and decrease the required computation, and the number of network parameters.

Furthermore, this size reduction tends to make the representation space invariant concerning small translations of the input, allowing the network to recognize specific patterns at different locations within the feature map. A one-dimensional max-pooling layer is applied to preserve, for each activation map, the neuron with the higher value. The classification part is analogous to a multi-layer perception.

$$y_j^{(l)} = f\left(\sum_{i=1}^{I} w_{ji}^{(l)} \cdot x_i^{(l-1)} + b_j^{(l)}\right) \tag{11}$$

This is followed by a flattened layer that reshapes the matrix input into a vector to support the processing of the subsequent non-spatial layers. The flattening layer consists of converting the data of the extraction part into a 1D-vector format. One hidden layer with the dropout function is implemented, and the neurons of the output layer correspond to the classes of heartbeats disease. Each unit activation $y_j(l)$ is computed as follows:

$$f(x) = \begin{cases} x, & \text{if } x > 0 \\ 0.01 \cdot x, & \text{otherwise} \end{cases}. \tag{12}$$

$$\hat{y}_i = \operatorname{argmax}\left(\frac{e^{y_i}}{\sum_{i=1}^{5} e^{y_i}}\right). \tag{13}$$

In the simplest case, each unit is retained with a fixed probability p independent of the other units. The output layer nodes of the proposed model represent 5 different heartbeat groups as specified by the Association for the Advancement of Medical Instrumentation (AAMI) standard.

4.1.1 Hyperparameters Optimization

The optimization of hyperparameters is crucial in determining both the efficiency and effectiveness of a model. In the proposed classification framework, a one-dimensional convolutional neural network (1D-CNN) architecture is utilized after its hyperparameters optimization. This architecture, a variant of the CNN, operates with neurons represented as queues, facilitating the exchange of positive and negative "signals" or "particles." Unlike other neural network models, CNNs are grounded in a rigorous mathematical framework, supported by extensive theoretical results that describe their behavior. To optimize the performance of the CNN model, several key hyperparameters must be carefully fine-tuned. We utilise bayesian optimization (BO), a probabilistic technique designed to globally minimize a black-box objective function for hyperparameters optimization. The BO framework consists of three essential components: a surrogate model update, a Bayesian update process, and an acquisition function. The surrogate model approximates the objective function based on all evaluated points and is iteratively updated using the Bayesian process after each new evaluation. The acquisition function then guides the next evaluation.

Let $f : \mathcal{X} \to \mathbb{R}$ be the objective function we aim to minimize (or maximize), where $\mathcal{X} \subset \mathbb{R}^d$ is the space of hyperparameters.

The surrogate model $\hat{f}$ approximates the true objective function f based on a set of observed data points $\mathcal{D}_t = \{(\mathbf{x}_i, y_i)\}_{i=1}^{t}$, where $\mathbf{x}_i$ are hyperparameter settings and $y_i = f(\mathbf{x}_i) + \epsilon_i$ are the corresponding noisy observations (with $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$).

BO can employ various surrogate models, such as Gaussian Processes (GP) or sequential optimization using decision trees, alongside different acquisition functions, including Expected Improvement (EI), Lower Confidence Bound (LCB), and Probability of Improvement (PI). In this study, GP is used as the surrogate model and EI is the acquisition function.

- **Gaussian Processes (GP):** GP is a widely recognized and powerful probabilistic model, valued for its descriptive power and analytical tractability. GP assumes that any finite set of random variables follows a multivariate normal distribution. Specifically, GP posits a priority that the probability distribution $P(f(x_1), f(x_2), \ldots, f(x_n))$ for a finite collection of points $x_1, x_2, \ldots, x_n \in \mathbb{R}^d$ follows a multivariate normal distribution with a mean function $\mu(x)$ and a covariance function $k(x, x')$, where k is a positive definite kernel, such as the squared exponential, rational quadratic, or Matern kernel.

- **Expected Improvement (EI):** The El acquisition function balances the trade-off between exploration and exploitation, determining the next evaluation point for optimization. El is defined as $EI(x) = \mathbb{E}[\max(0, f(x) - f(x^+))]$, where $f(x^+)$is the best-observed value and $\mathbb{E}[f(x)]$ represents the expected function

values at x. The improvement is $f(x) - f\,(x^{+})$if it is positive; otherwise, it is zero.

4.1.2 Performance evaluation metrics

The performance evaluation of the proposed multi-class classification model is undertaken using a comprehensive set of assessment metrics, including accuracy, precision, recall, F1 score, computational capacity, resource utilization, and latency. By employing this diverse array of evaluation criteria, the study achieves a detailed understanding of the model's effectiveness.

- Accuracy (AC): Accuracy measures the proportion of correctly classified instances across all five arrhythmia classes within the dataset. It represents the percentage of instances where the model correctly identifies the type of arrhythmia. Accuracy is mathematically represented as:

$$\mathrm{AC} = \frac{TP + TN}{TP + TN + FP + FN}$$

- Precision (PR): Precision calculates the proportion of correctly identified instances of a particular arrhythmia class relative to the total number of instances classified as that class. For each class, precision provides insight into the model's ability to avoid false positives. Precision is mathematically represented as:

$$\mathrm{PR} = \frac{TP}{TP + FP}$$

- Recall (RC): Recall determines the ratio of correctly identified instances of a specific arrhythmia class to the total number of actual instances of that class (true positives plus false negatives). Recall provides a measure of the model's ability to capture all relevant instances for each class. Recall is mathematically represented as:

$$\mathrm{RC} = \frac{TP}{TP + FN}$$

- F1 Score: The F1 score computes the harmonic mean of precision and recall for each arrhythmia class, offering a balanced measure that considers both false positives and false negatives. This metric is particularly useful when dealing with imbalanced class distributions. The F1 score is mathematically represented as:

$$\mathrm{F1} = 2 \times \frac{\mathrm{PR} \times \mathrm{RC}}{\mathrm{PR} + \mathrm{RC}}$$

- Macro Average: Macro averaging calculates the average performance metric independently for each class and then takes the arithmetic mean of these individual metrics. In this approach, each class is treated equally, regardless of its size. - Macro Precision (Macro PR): The arithmetic mean of precision values across all classes.

$$\text{Macro PR } = \frac{1}{N} \sum_{i=1}^{N} \text{PR}_i$$

- Macro Recall (Macro RC): The arithmetic mean of recall values across all classes.

$$\text{Macro RC } = \frac{1}{N} \sum_{i=1}^{N} \text{RC}_i$$

- Macro F1 Score (Macro F1): The arithmetic mean of F1 scores across all classes.

$$\text{Macro F1} = \frac{1}{N} \sum_{i=1}^{N} \text{F}_i$$

Where N is the number of classes, and PR_i, RC_i, and F 1 are the precision, recall, and F 1 scores for class i, respectively.

- Weighted averaging accounts for the different class sizes by computing a weighted mean of the performance metrics, where the weight of each class is proportional to the number of true instances in that class. Weighted Precision: The weighted mean of precision values across all classes, with weights proportional to the number of instances in each class.

$$\text{Weighted PR} = \sum_{i=1}^{N} w_i \times \text{PR}_i$$

Weighted Recall: The weighted mean of recall values across all classes, with weights proportional to the number of instances in each class.

$$\text{Weighted RC} = \sum_{i=1}^{N} w_i \times \text{RC}_i$$

Weighted F1 Score: The weighted mean of F1 scores across all classes, with weights proportional to the number of instances in each class.

$$\text{Weighted F1} = \sum_{i=1}^{N} w_i \times \text{F}_i$$

Where w_i is the proportion of instances belonging to class i in the entire dataset, calculated as:

$$w_i = \frac{\text{Number of instances in class } i}{\text{Total number of instances in the dataset}}$$

4.2 Experimental Setup

To monitor vital signals like heart rate and pulse, we use the Photoplethysmography sensor MAX30100 along with ESP8266-12E WiFi module. It employs a pair of LEDs (red and infrared) and a photodetector, complemented by optimized optics and low-noise analogue signal processing techniques, to detect pulse oximetry and heart rate signals. It uses 1.8V or 3.3V power supplies and features a programmable power-down mode, which allows for negligible standby current. This feature offers the convenience of maintaining a constant power supply connection. The MAX30100 is commonly found in fitness assistant devices, medical monitoring devices, and other wearable technology. Its integrated LEDs, photo sensor, and high-performance analogue front end make it straightforward and user-friendly. Its compact form factor (5.6mm x 2.8mm x 1.2mm 14-Pin optically enhanced system-in-package) renders it suitable for wearable devices.

To optimize power management and facilitate energy savings, this system incorporates programmable settings for sample rate, LED current, and an ultra-low shutdown current, typically resting at 0.7μA. Additional standout attributes encompass a high signal-to-noise ratio (SNR), integrated ambient light cancellation, and a capability for high sample rates. The experiments are conducted on 18 individuals aged 20 to 52 years in a room with multiple readings. RR and HR values from a random sample of 25 minutes recording with 500 Hz are depicted in figure 10. The infrared module and DHT11 sensor are used for environment monitoring. Figure 11 illustrates the pin configuration of the MAX30100, infrared sensor and DHT11 sensor.

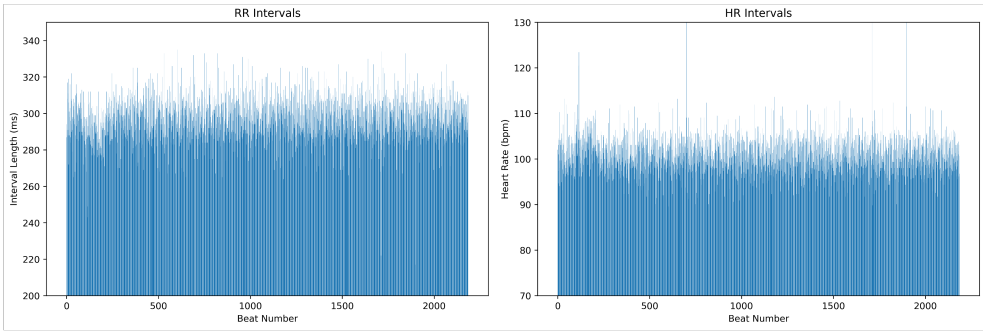


Figure 10: RR and HR values from random sample of 25 minutes ECG recording with 500 Hz sampling rate and 12 bits resolution

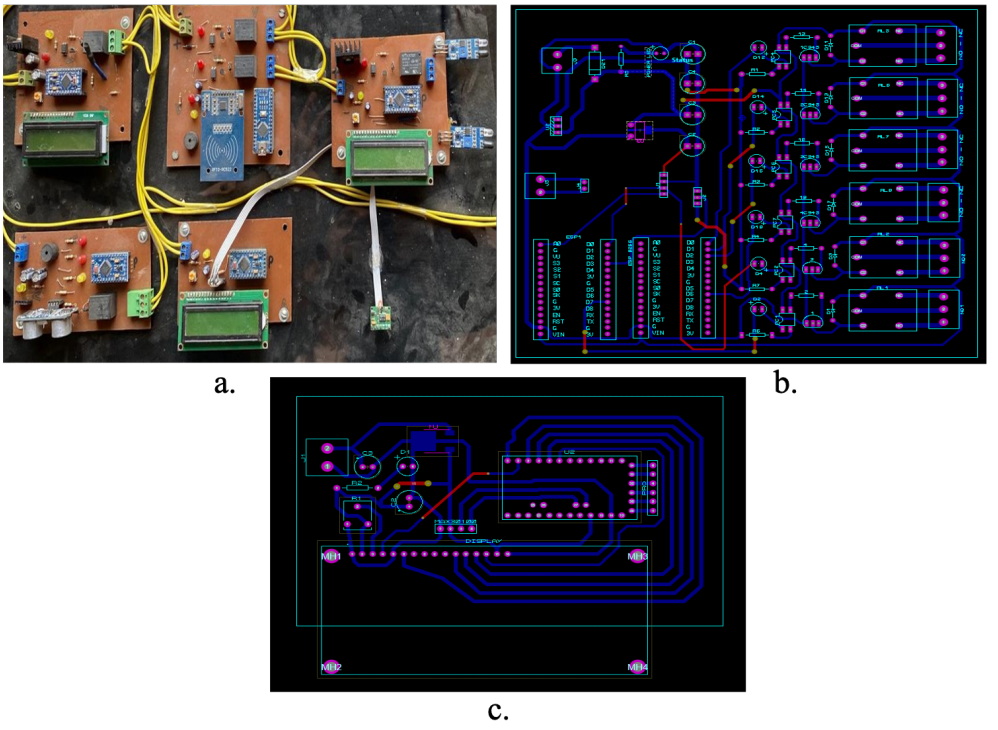


Figure 11: Sensors Experimental Setup: PPG, Infrared Sensor, and DHT11 Sensor

4.2.1 ECG monitoring

Cardiac signal surveillance plays a critical role in the early diagnosis and prevention of cardiovascular diseases, with electrocardiography (ECG) being a key tool for such identification. This technique records the heart's electrical activities during a cardiac cycle, the resultant bioelectrical activity leads to variations in the skin's electrical potential, detectable via specialized instruments called electrocardiographs. Electrodes are thoughtfully positioned on the skin to capture this phenomenon. The output, termed an ECG, is a visualization of these potential discrepancies, a cornerstone of cardiological evaluations conducted by medical professionals. ECG monitoring systems are categorized based on electrode count, yielding designations like 3-lead, 5-lead, and 12-lead ECG systems. Widely employed in healthcare centres, these non-invasive devices capture vital cardiac information. Pre-hospital settings commonly use 3-lead systems, while hospitals prefer systems with additional leads for more comprehensive data.

Figure 12 section (c) depicts a representative ECG cycle comprising distinct waves – Q, R, S, P, T, and U. Each wave signifies specific cardiac events, with Q, R, and S being consistently prominent, while the U wave's appearance may vary.

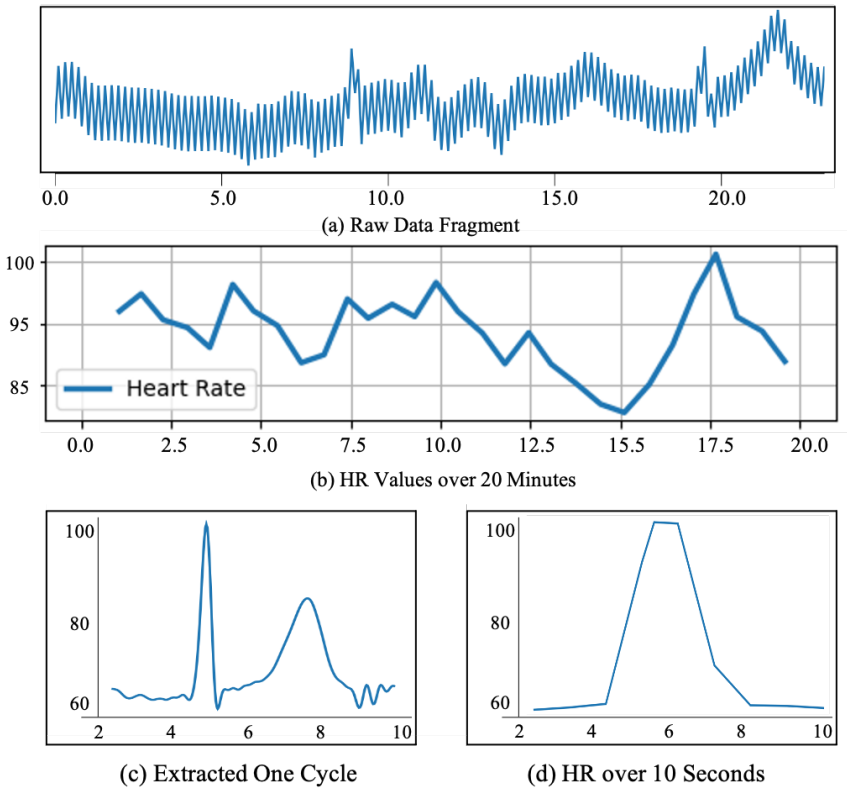


Figure 12: Raw and Processed HR fragments

To assess our proposed architecture, we examined real-time arrhythmia detection via continuous monitoring of cardiovascular patients. The framework includes an early warning system and an arrhythmia classification system. We compared two distinct testbeds against the baseline ECG system using PSG and its electrodes for reference.

4.2.2 First Testbed

In our first testbed, a 2-channel ECG system is employed, with the perception layer anchored by the AD8232, a specialized signal conditioning block tailored for electrocardiogram (ECG) and heart rate monitoring. The EGs consist of Arduino UNO, STM32F427, and Raspberry Pi 3. The sensor node AD8232 captures data and transmits it to the edge gateway node. An ECG feature extraction service is implemented within the edge gateway to derive crucial parameters like heart rate, P wave, and T wave from the ECG signal. This service follows the structure outlined in Figure 13, progressing through several stages: movement artifact removal, wavelet transforma-

tion, threshold estimation, and P wave and T wave detection.

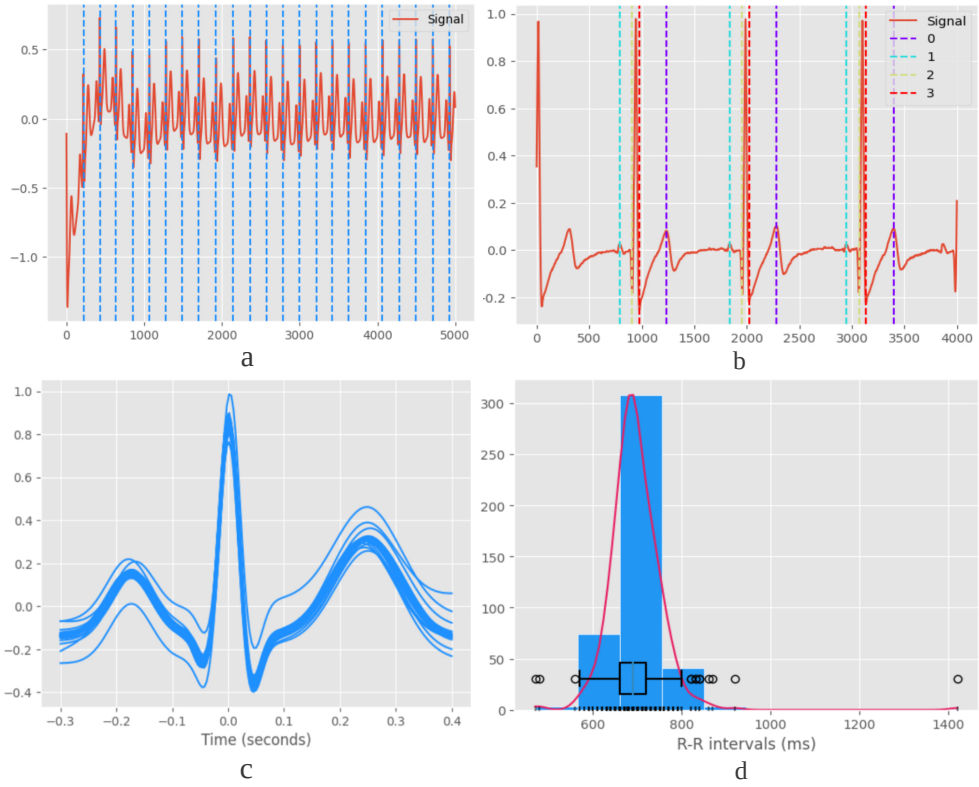


Figure 13: ECG Extraction Cycle

The movement artifact removal phase employs band-pass and moving average filters to counter environmental noise (e.g., 50 Hz power-line noise). The filtered data then enters the wavelet transformation, where the Daubechies-4 wavelet is chosen due to its efficiency in extracting P-wave and T-wave components without excessive computational delay. Thresholds for identifying R, P, and T waves are determined based on the wavelet transformation results, with R wave thresholds generally higher in millivolts compared to P and T waves. For instance, 1 mV might be set as the R peak threshold in the lead I, while thresholds of 0.08 mV and 0.1 mV are used for P and T waves in the lead II, respectively. These values can vary depending on the specific ECG leads utilized. The heart rate is computed using the R-R interval information derived from these thresholds, applying the formula:

$$Heartrate = 60/R - Rinterval$$

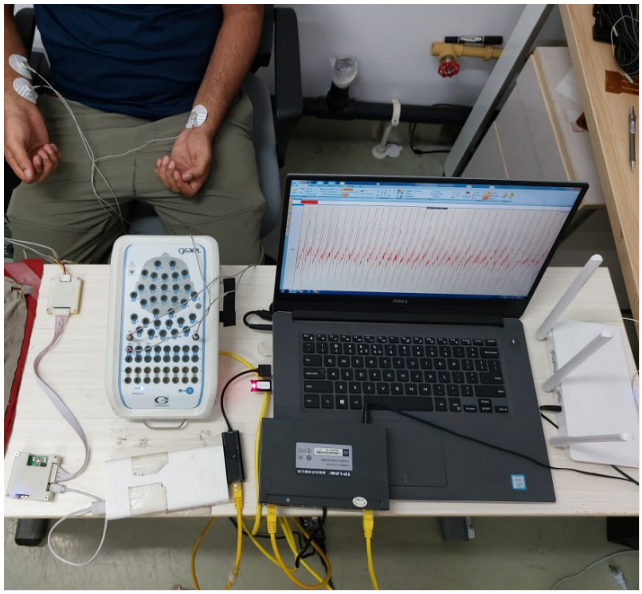


Figure 14: ECG sample collection

The proposed ECG feature extraction, driven by wavelet transformation, efficiently employs network bandwidth. The complete ECG feature extraction cycle is shown in figure 13. At each discrete wavelet transform level, the data sample count is halved. Instead of storing the raw ECG data, emphasis is placed on preserving data following the wavelet transformation and the associated coefficient values. This approach significantly conserves network bandwidth between 40% and 80%, contingent on the wavelet transformation types and levels, while potentially introducing a minor uptick in system latency. The selection of wavelet transform types and levels should be carefully calibrated based on specific application needs to minimize potential errors in the inverse transformation process.

In essence, deploying the ECG feature extraction service within EGs, employing the outlined template and wavelet transformation methods, facilitates accurate real-time monitoring of vital parameters while resourcefully optimizing network assets.

Second Testbed Figure 14 illustrates our proposed testbed system configuration, outlining the component arrangement and person's positioning during ECG data collection. The setup encompasses an ECG front end, an embedded software module, and additional elements on the right side. Three electrodes, the positive electrode, negative electrode, and driven right leg (DRL) to detect electro potential changes due to cardiac electrical activity.

The ADS1292r serves as the electric signal converter for capturing the ECG signal in this system. The hardware is equipped with two 32-bit microcontrollers (MCU), namely the STM32F401CCU6 and STM32F103C8T6 from ST Electron-

ics. Additionally, it incorporates digital-to-analogue converters (DAC) MCP4921, general-purpose operational amplifiers (OPAMP) LF353, and a wye resistor network. The STM32F401CCU6, belonging to the ARM Cortex-M4 Cores family, operates at 84 MHz, offering a balance of cost-effectiveness and high performance. It features standard communication peripherals such as SPI, I2C, USB, USART, and CAN. The MCU includes a single-precision floating-point unit for rapid calculations, digital signal processing instructions, and two analog-to-digital converters. The DACs are configured to operate with an external voltage reference, receiving clock signals up to 20 MHz from the MCU. A stable DC supply powers the DACs, providing the necessary external voltage reference of +5 V.

To minimize noise impact on signal integrity, a bypass capacitor is introduced. ECG waveforms with amplitudes tenfold larger than real ECG amplitudes, including offset levels simulating baselines, are generated. The converter outputs undergo a non-inverting amplifier stage to ensure physiologically consistent amplitudes. This amplifier, comprising general-purpose operational amplifiers and resistors (R1, R2, Rg, and Rf), produces differential output signals with low offset voltage and minimal noise. The design generates ECG waveforms within the 0.5 mV to 4 mV range, exhibiting low noise and limited offset effects. Data transmission from the module occurs via the serial peripheral interface (SPI) and is sent to EGs. Three electrodes are strategically positioned: the positive electrode on the left arm, the negative electrode on the right arm, and the DRL on the right arm, with a horizontal separation of about 5 cm. The RLD electrode enhances the common mode rejection ratio (CMRR) by transmitting the common mode signal of the two sensing electrodes back to the user's body. The analogue front end integrates the ADS1292, a programmable gain amplifier (PGA), an analogue-digital converter (ADC), and an RLD circuit.

Post-acquisition, data is transmitted to data transmission modules via SPI. The MCU and Wi-Fi module then transmits signals via Wi-Fi to the user interface. A xiaomi power bank supplies power to the hardware system, utilizing the ESP32's built-in WiFi module for communication. During measurements, some subjects were supine and instructed to relax muscles, minimizing muscular artefacts and evaluating pure ECG quality. High-pass and low-pass filters with cutoff frequencies of 0.1 Hz and 200 Hz were implemented in the measurement electronics. After digitization at an 800 Hz sampling frequency, notch filters at multiples of 50 Hz were used to eliminate power line interference. Expectedly, movement introduces diverse movement artefacts in the ECG signal, particularly muscular artefacts and baseline shifts. However, these can be mitigated effectively through digital signal-processing techniques.

Data collection duration varied based on participant age groups, dividing data into chunks of 10 minutes for young individuals and 5 minutes for elderly participants. Collection was conducted in shorter segments of 2, 5, and 10 minutes each. Figure 15 shows an example of the ECG cycle collected through the second testbed

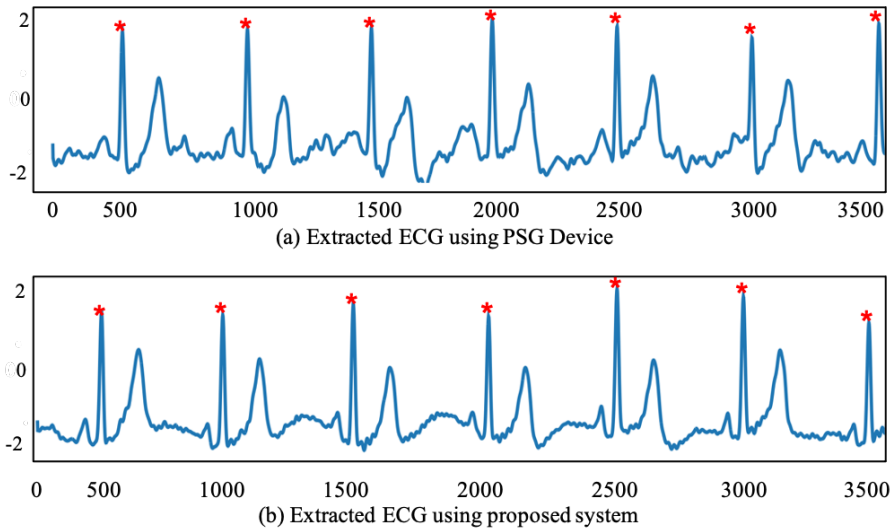


Figure 15: ECG extracted cycles (a) Extracted ECG using PSG Device (b) Extracted ECG using proposed system

ECG collection process.

4.2.3 Performance Analysis

Analyzing ECG signals for cardiovascular health assessment relies on crucial parameters like heart rate (HR) and RR intervals. HR, quantified in beats per minute (bpm), is deduced by measuring the time between consecutive R peaks in an ECG signal. On the other hand, the RR interval signifies the time span between successive R peaks. It verifies the regularity of the heart's rhythm, highlighting irregularities such as arrhythmia. Figure 16 shows raw signals, processed signals, ECG wave and RR interval of the extracted ECG cycle.

To ascertain the precision and credibility of the proposed ECG device, a comparison of HR and RR interval measurements against those obtained from a standard Polysomnography (PSG) device [163] is crucial. PSG monitors a spectrum of physiological aspects, including ECG, to present a comprehensive health snapshot. The robust alignment between measurements from the cost-effective home-based ECG collection system and those from the PSG device validates the accuracy and dependability of the proposed system in HR and RR interval monitoring. The accuracy and precision of derived measurements can be influenced by the duration of ECG data collection. The optimal data collection duration hinges on the specific application or context in which the information is being acquired. Figure 17 shows HR non-linear analysis of non-linear dynamics of heart rate intervals over 25 minutes of sampling.

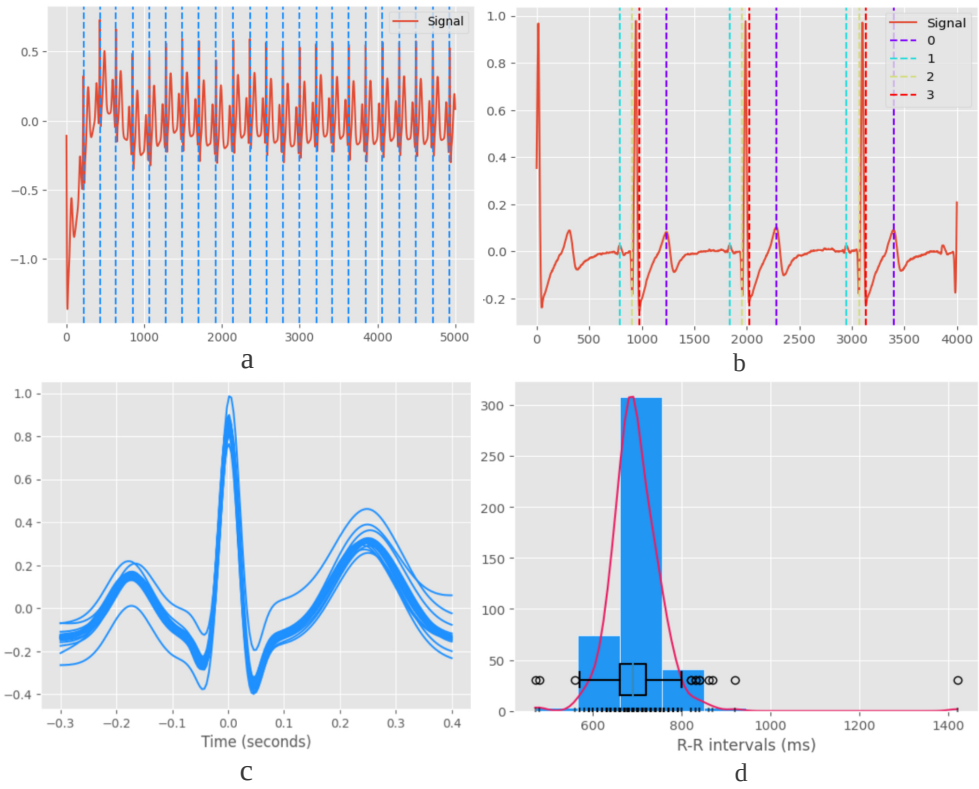


Figure 16: (a) Raw signals, (b) Processed signals, (c) ECG wave and (d) RR interval of the extracted ecg cycle.

It is used to detect abnormalities over abnormal autonomic regulations and makes it easier to spot deviations from normal heart behaviour. It provides a statistical summary of heart rate variability, helping clinicians assess short- and long-term dynamics, detect abnormalities, and understand the complex interactions of the autonomic nervous system with heart function.

A comparative evaluation was conducted between the proposed system and a baseline configuration employing a conventional observe decide act (ODA) control strategy [164]. Data was gathered within the perception layer in the baseline setup and subsequently transmitted to a cloud server for analysis and decision formulation. The resultant decision vector and notifications were then relayed to the end-user. The computational load in this baseline architecture centred on the cloud server, with the gateway device primarily serving as a communication conduit between the two distinct layers. Moreover, we extended the proposed model as edge intelligence by implementing lightweight machine algorithms for efficient and secure information management and privacy protection in healthcare data.

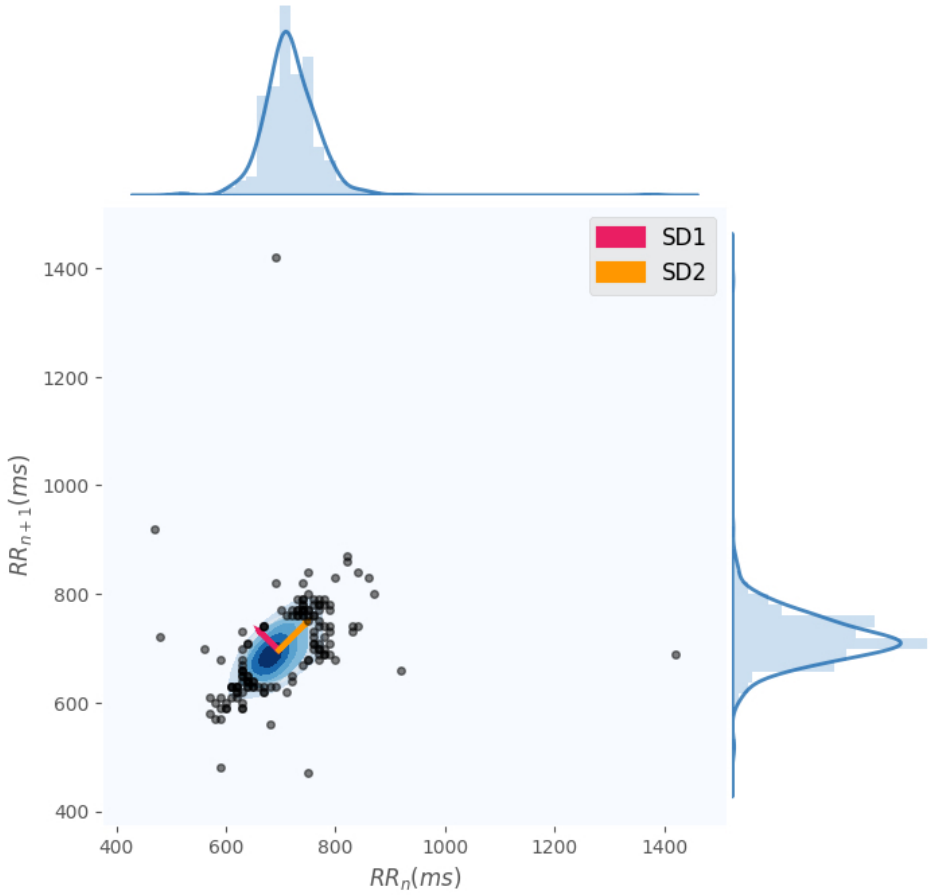


Figure 17: Poincaré plot to represent HR Non linear analysis

4.3 Early warning system

The real-time push notification service serves to promptly notify designated individuals (e.g., guardians) of detected abnormalities, ensuring swift responses such as immediate first-aid interventions. This service triggers notifications upon detecting abnormal heart rates or ECG signals (e.g., prolonged P waves or elevated T wave amplitude). Additionally, notifications are dispatched if the internal temperature of a smart gateway surpasses a predefined threshold or if the gateway ceases to receive incoming data from sensor nodes over a specified period. The content and priority level of push messages vary based on specific events. For instance, a heart rate exceeding 80 bpm triggers a priority level 1 message, while a heart rate above 120 bpm prompts a priority level 3 message. Depending on the application, push notifications can be executed and activated at the gateway level.

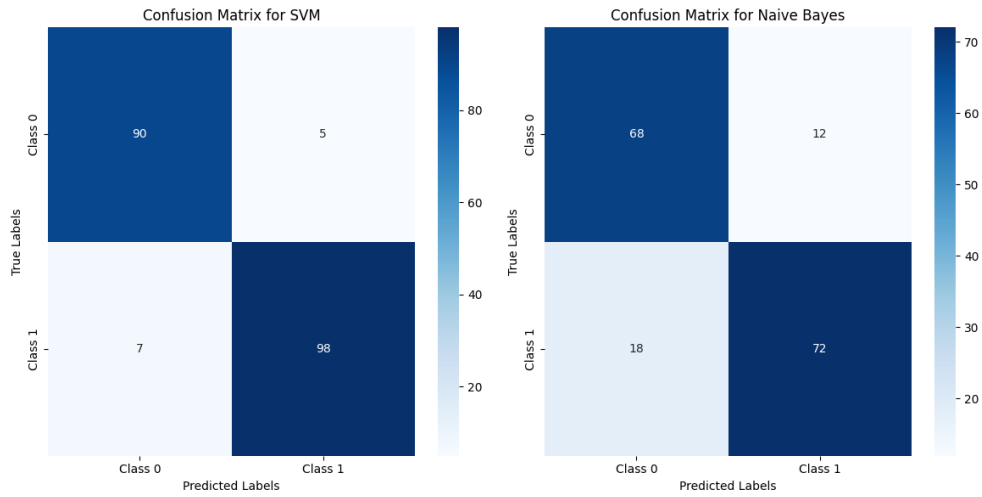


Figure 18: Confusion Matrix for Abnormality Detection using ECG signal processing (a) Abnormality Detection Accuracy using SVM (b) Abnormality Detection Accuracy using Naive Bayes

In the proposed systems, the push notification service harnesses binary classification, distinguishing normal and abnormal beats in the initial stage to prompt first aid actions.

To validate the system's effectiveness, the model was trained using physiobank databases [165], and real-time testing using collected data samples utilising python libraries such as scikitlearn [166] and biosppy [167]. Real-time decision-making regarding a user's health condition is enabled through the utilization of the linear support vector machine (SVM) method and naive bayes due to their less complexity and fast response rate. Figure 18 depicted that SVM shows better results in terms of accuracy as compared to naive bayes. This proposed method classifies incoming signals as either normal or abnormal. SVM has a higher precision of 95.15% compared to naive bayes 85.71%. This shows SVM makes fewer false positives, meaning it's better at predicting abnormal heart conditions when they truly exist.

We collected 2 hours of ECG signals from healthy individuals and cardiovascular issues. The perception layer divides the chunks of 10 seconds of signals and sends them to EGs where data pre-processing is handled and labels the signal as normal or abnormal. These features encompassed QRS complex duration, T wave duration, RR interval, PR interval, and ST segment (refer to figure 16). During runtime, incoming test data were locally classified, with the decision vector sent to the Execute component for actuation. Test data included ECG signals with random arrhythmia points added to normal ECG data to simulate emergency scenarios. These scenarios were tested on data from four new users.

Table 10: Comparison of 1 Minute HR values of Male(Y) and Female(X) individuals using single channel proposed(Prpsd) HR monitoring vs PSG device.

Ind	**X Prpsd**	**X PSG**	**Y Prpsd**	**Y PSG**
Ind1	72	71	70	71
Ind2	73	70	67	66
Ind3	69	70	71	70
Ind4	72	72	70	70
Ind5	73	71	68	69
Ind6	71	69	70	69
Ind7	70	70	70	69
Ind8	71	69	71	71
Ind9	72	71	70	69

Table 10 shows the evaluation results of HR values of Male(Y) and Female(X) individuals using a single channel proposed HR monitoring system versus baseline PSG device. In table 11, we measure the RR interval difference using 1 Minute RR intervals of Male(Y) and Female(X) individuals using a single channel proposed(Prpsd) ECG Monitoring system versus baseline PSG device.

4.4 Arrhythmia Classification

Abnormal cardiac rhythms, referred to as arrhythmias, can be detected and classified by analyzing their deviations from a normal heartbeat pattern. Each type of arrhythmic beat follows a distinctive structure, making it feasible to develop machine learning algorithms that automate the detection and classification of electrocardiogram (ECG) signals. Most contemporary approaches to this task employ either convolutional neural networks (CNNs) or long short-term memory networks (LSTMs).

Although existing research has demonstrated the effectiveness of CNNs and LSTMs in the automated detection and classification of arrhythmias, the primary focus has been on achieving high accuracy, often at the expense of considerations related to the computational resources and time required for prediction. In contrast, this thesis investigates the feasibility of deploying these networks in real-time, resource-constrained environments, such as single-board computers. By enabling real-time classification, physicians could obtain long-term electrocardiograms that are pre-annotated, which typically necessitates the patient remaining in the hospital to use a bulky and expensive ECG machine. Additionally, this approach offers the advantage of alerting the wearer immediately if an abnormal heart rhythm is detected. In this study, we worked on five-class arrhythmias classification, depicted in

Table 11: Comparison of 1 Minute RR intervals of Male(Y) and Female(X) individuals using single channel proposed(Prpsd) ECG Monitoring system vs PSG device.

Ind	X Prpsd	X PSG	Y Prpsd	Y PSG
Ind1	0.8225	0.8200	0.9565	0.9465
Ind2	0.8371	0.82345	0.8840	0.8654
Ind3	0.8526	0.85102	0.8393	0.8346
Ind4	0.9414	0.93400	0.8469	0.8297
Ind5	0.8428	0.84012	0.8205	0.8567
Ind6	0.9617	0.9514	0.8740	0.86634
Ind7	0.8790	0.8507	0.8511	0.8391
Ind8	0.8504	0.8497	0.9391	0.9240
Ind9	0.8445	0.8363	0.8494	0.8329

figure 19.

To tackle time and resource utilisation challenges, we proposed resource-efficient 1D-CNN 4.1. This proposed model is initially trained using MIT-BIH [165] ECG fragments and later tested on our real-time raw data. Table 12 depicted the number of samples against each heartbeat class during model classification.

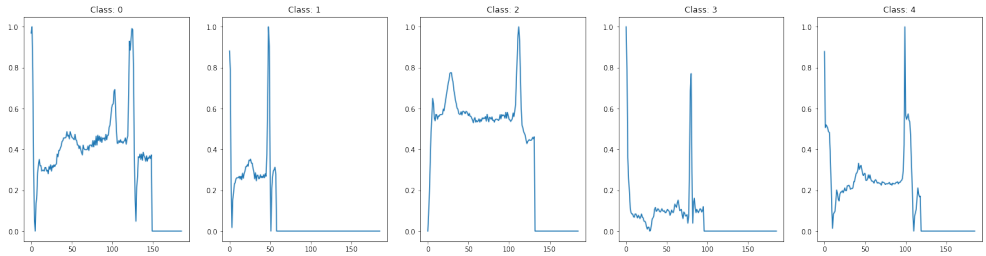


Figure 19: MIT-BIH Arrhythmia heartbeats with 5 classes. 0=Normal, 1=Fusion of paced and normal, 2=Premature ventricular contraction, 3=Artial Premature , 4=Fusion of ventri and normal

4.4.1 Hyperparameters Tunning

This approach first establishes a general network architecture integrating a 1D-CNN feature extraction module with a multi-layer perceptron decision module. Through extensive testing, the range of hyperparameters is determined. During model optimization, the Bayesian optimization algorithm is embedded within the network ar-

Table 12: Number of samples against each class of heartbeat

Class	**Assigned number**	**Training Samples**	**Testing Samples**
Normal	N	72471	18118
Fusion of paced and normal	FPNs	2223	556
Premature ventricular contraction	PVCs	5788	1447
Artial Premature	AP	641	160
Fusion of ventri and normal	FVNs	6431	1608

Table 13: Hyper-parameters Tuning

Hyper-parameter	**Range**	**Best value**
Batch size	$[16, 32, 64, 128]$	64
Learning rate	$[0.01, 0.001, 0.0001]$	0.001
Optimizer	[Adam, Adagrad, Nadam]	Adam
Activation function	[seLu, reLu]	reLu

chitecture to adaptively select the optimal combination of hyperparameters.

Based on the fundamental structure of the convolutional neural network, a set of n hyperparameter elements, including both structural and training hyperparameters, is defined to form the set x. Structural hyperparameters include the number of convolutional layers, the number of filters per layer, kernel size, stride, the number of pooling layers, pooling kernel size, padding, the number of fully connected layers, the number of neurons, batch normalization layers, dropout parameters, and the activation function. Training hyperparameters include the optimizer, learning rate, batch size, and the ratio of training to test data. The search range for all hyperparameters is determined based on related research and empirical testing.

The initialized parameter set D is obtained through model training, assuming it satisfies the surrogate model M. Using the prior distribution and known parameter set D, the posterior distribution $p(y \mid x, D)$ is computed, and the acquisition function S is constructed. This acquisition function is then used to select the next hyperparameter set x_i for network training, and the corresponding evaluation output y_i is obtained. After T iterations, an approximately optimal hyperparameter combination is identified.

By utilizing gaussian processes alongside acquisition functions, bayesian optimization effectively navigates the hyperparameter space, striking a balance between exploration and exploitation to identify the optimal configuration for neural network training. The hyperparameter configuration for the proposed model is outlined in table 13.

4.4.2 Model Training and Testing

Edge-gateways receive raw data from the sensor nodes. Raw ECG signal data is pre-processed and normalized. The extraction includes batch normalization, convolution, activation, and max-pooling layers. The flattening, fully connected, and softMax layers constitute the classification part. Input is standardized using a batch normalization layer to reduce the internal co-variate shifts. Each neuron is connected to the local window from the previous layer, known as the receptive field, which shifts according to timestamps and shares synaptic weights. By using this approach, we can reduce the number of weights, which facilitates the generalization process. A Rectified linear unit function (ReLu) is applied to return the weighted sum of the input data. After this, a one-dimensional maxpooling layer is applied to preserve the neurons of each activation layer. However, the classification part is the same as with multi-layer perceptron. Table 14 presents the CNN network configuration used to train and test arrhythmia classification.

Table 14: 1D-CNN Network configuration

Network Part	Description	
	319 Neurons	
Extraction part Block 1,2	Conv-1D	Kernels: 64 Receptive field: 2 Stride: 1
Input Layer	Activation	reLu
	Dropout	Probability: 0.4
	Max-pooling	Pool size: 2
Classification part	FC layer	512 neurons
	Dropout:	Probability: 0.2
	Output layer:	5 neurons

The baseline LSTM model [168] is also used to train and predict the raw data at the edge gateways to compare its computational efficiency with our proposed 1D-CNN. Usually, LSTM is preferred for edge-based systems due to its lower resource consumption and better fit for sequential data. LSTM enables the system to forget about unnecessary information from the previous outputs, which makes it suitable for scarce computing devices. After this, new input $X(t)$ is decided, and apply the sigmoid function to decide the updation of the next value. A $tanh$ layer creates the vector of all possible values from the upcoming input. The sigmoid layer decides the part of the information that will go to the final layer. The trained hypothesis function was subsequently transmitted to the edge gateways devices, enabling the classification of incoming ECG signals into five classes: 0=Normal, 1=Fusion of paced and normal, 2=Premature ventricular contraction, 3=Artial Premature, 4=Fusion of ventri and normal.

Table 15: Average Performance measures of proposed 1D-CNN

Class Category	Accuracy	Precision	Recall	f1 score
N	0.996	0.991	0.996	0.994
FPNs	0.981	0.992	0.997	0.991
PVCs	0.990	0.989	0.991	0.994
AP	0.989	0.997	0.996	0.990
FVNs	0.998	0.986	1.000	0.999
macro avg	0.9908	0.991	0.996	0.9938
weighted avg	0.9904	0.9852	0.9911	0.9895

4.4.3 Performance Analysis

This section presents and discusses the results of the experiments as well as the performance of the proposed model. In particular, the classification results were assessed using standard evaluation metrics, including accuracy (Acc), sensitivity (SEN), recall, and F1-score for each model.

The average performance measures of 1D-CNN are described in Table 15 and an average performance matrix of LSTM is described in Table 16. From the comparison results in table 15 and table 16, it can be concluded that the proposed 1D-CNN performs better in comparison to the baseline LSTM model while utilising the same resources and time. A comparative analysis of our proposed 1D-CNN with similar studies demonstrates an average accuracy of 97.4%, while utilizing significantly fewer resources than other studies.

Results of the confusion matrix in figure 20 demonstrate high accuracy of 1D-CNN, with most predictions correctly placed on the diagonal, indicating strong performance and minimal misclassifications using a balanced dataset. The limited off-diagonal values suggest that the model rarely confuses one class with another. In contrast, the LSTM model, while still performing well, has slightly more off-diagonal values, indicating a higher number of misclassifications. This is reflected in its slightly lower precision and recall compared to the 1-D CNN, with the confusion matrix showing a bit more spread, indicating that the LSTM is less certain in its predictions.

The results demonstrate that the 1D-CNN significantly outperforms the LSTM model in the classification of arrhythmias across all evaluated performance metrics. The 1D-CNN achieves notably higher accuracy, with category-specific accuracies nearing 1.000, and superior precision and recall values, particularly excelling in the identification of false positives and ventricular non-sustained arrhythmias (FPNs and FVNs). The F1 score, a measure of the balance between precision and recall, further underscores the 1D-CNN's robustness, with values consistently above 0.990. In contrast, the LSTM model, while still performing adequately, exhibits lower accu-

Table 16: Average Performance measures of baseline LSTM [168]

Class Category	Accuracy	Precision	Recall	f1 score
N	0.94	0.96	0.94	0.94
FPNs	0.96	0.95	0.97	0.96
PVCs	0.96	0.98	0.97	0.96
AP	0.97	0.98	0.96	0.96
FVNs	0.94	0.94	0.95	0.93
macro avg	0.9908	0.9910	0.9960	0.9923
weighted avg	0.9953	0.9906	0.9960	0.9209

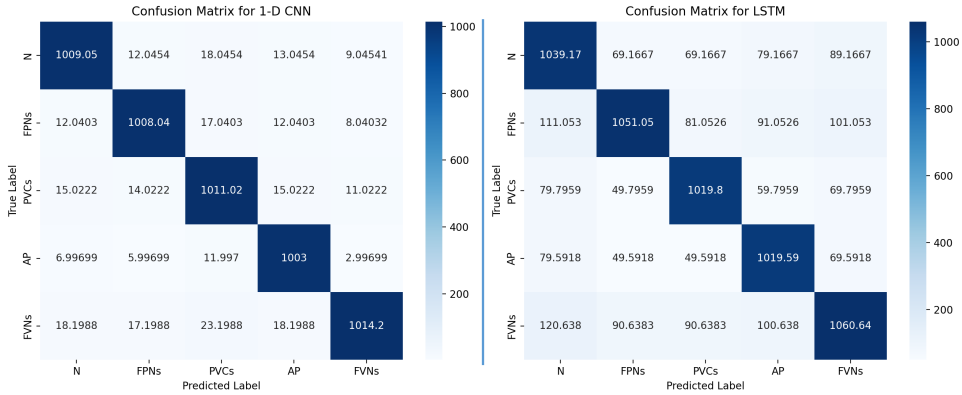


Figure 20: Confusion Matrix of proposed 1D-CNN and LSTM baseline model

racy, precision, recall, and F1 scores, particularly in the detection of normal (N) and ventricular non-sustained arrhythmias (FVNs), suggesting that it may struggle with maintaining the balance between precision and recall.

These findings indicate that the 1D-CNN is a more effective and reliable model for real-time arrhythmia classification, particularly in scenarios where the accurate and timely detection of irregular heart rhythms is critical. CNN operations can be applied across the entire sequence simultaneously, leading to faster processing times. This efficiency is crucial for real-time applications where quick decision-making is essential. While, sequential nature makes LSTMs computationally more intensive and slower compared to CNNs, particularly when dealing with long sequences, which can be a limitation in real-time scenarios.

To assess the performance of the proposed network, we compared it to some state-of-the-art methods in the literature. We record the performance of the proposed network model (in bold) and some recent arrhythmia classification using the MIT-BIH arrhythmia database in table 17 .From table 17, it is evident that our proposed 1D-CNN achieved good performance while utilising less resources.

Table 17: Comparison between the related work and the proposed 1D-CNN model

Ref.	Year	Classification Technique	Resource Cnsmptn	No. of Layers	No. of Classes	Accuracy
[169]	2021	Neural network	High	15	5	99.31%
[170]	2021	Transfer Learning	High	18	2	90.42%
[171]	2022	STFT-CNN	Moderate	.....	5	99.0%
[172]	2023	H-PSOCNN	Moderate	.....	5	98.0%
[173]	2023	CNN,DAE + Transformer	Moderate	5	5	97.66%
[174]	2024	CNN,Attention + Transformer	High	8	5	99.58%
[175]	2024	1D-CNN+LSTM	Moderate	11	9	98.24%
[176]	2024	CAD-Net(1D-CNN)	Moderate	...	5	99.54%
[177]	2024	1DCNN-BiLSTM	Low	7	5	93.7%
This Study	**2024**	**Proposed 1D-CNN**	**Low**	**2**	**5**	**97.4%**

4.4.4 1D-CNN Comparison against different class configurations

The performance metrics of the resource-optimized 1D-CNN model for arrhythmia classification across unbalanced, oversampled, and undersampled datasets reveal distinct trade-offs associated with each data sampling strategy. Table 18 depicts the performance metrics of 1D-CNN for different diagnostic classes across various model configurations.

Using an unbalanced dataset, the model demonstrates strong overall performance, with high accuracy (ranging from 0.975 to 0.998), precision, and recall across all diagnostic classes. The F1 scores are also consistently high, indicating a well-balanced performance. For example, the recall for FPNs (0.995 ± 0.0140) and FVNs (0.997 ± 0.0080) is particularly impressive, suggesting that the model is adept at correctly identifying positive instances even with an unbalanced dataset.

In an oversampled dataset, the model generally maintains high performance, but there are some notable changes. Precision tends to increase slightly, especially in the N class (from 0.983 to 0.995), suggesting that oversampling helps the model to reduce false positives. However, there is a drop in recall for some classes, such as FPNs (from 0.995 ± 0.0140 to 0.9455 ± 0.0080), indicating that while the model becomes more precise, it may miss more positive instances when trained on oversampled data. The F1 scores reflect this trade-off, with minor increases in classes where precision improves and decreases in those where recall drops.

Using an undersampled dataset, the model shows a mixed performance when trained on undersampled data. While accuracy remains consistent, precision gener-

Table 18: 1D-CNN Performance metrics for different diagnostic classes across various model configurations

Diagnostic Class	Metric	Unbalanced	Oversampled	Undersampled
N	Accuracy	0.975 ± 0.0030	0.973 ± 0.0030	0.951 ± 0.0035
	Precision	0.983 ± 0.0030	0.995 ± 0.0030	0.975 ± 0.0030
	Recall	0.9915 ± 0.0030	0.960 ± 0.0030	0.983 ± 0.0030
	F_1 score	0.982 ± 0.0030	0.995 ± 0.0020	0.973 ± 0.0020
FPNs	Accuracy	0.992 ± 0.0030	0.990 ± 0.0025	0.984 ± 0.0025
	Precision	0.992 ± 0.0105	0.990 ± 0.0085	0.962 ± 0.0085
	Recall	0.995 ± 0.0140	0.9455 ± 0.0080	0.975 ± 0.0080
	F_1	0.980 ± 0.0085	0.990 ± 0.0060	0.980 ± 0.0060
PVCs	Accuracy	0.998 ± 0.0025	0.979 ± 0.0025	0.962 ± 0.0025
	Precision	0.990 ± 0.0115	0.990 ± 0.0075	0.980 ± 0.0075
	Recall	0.97 ± 0.0095	0.995 ± 0.0090	0.985 ± 0.0090
	F_1	0.9840 ± 0.0075	0.998 ± 0.0060	0.973 ± 0.0060
AP	Accuracy	0.998 ± 0.0015	0.991 ± 0.0025	0.962 ± 0.0015
	Precision	0.988 ± 0.0040	0.985 ± 0.0025	0.978 ± 0.0035
	Recall	0.995 ± 0.0015	0.991 ± 0.0030	0.985 ± 0.0030
	F_1	0.990 ± 0.0020	0.980 ± 0.0020	0.980 ± 0.0020
FVNs	Accuracy	0.997 ± 0.0020	0.997 ± 0.0020	0.968 ± 0.0020
	Precision	0.989 ± 0.0050	0.991 ± 0.0055	0.974 ± 0.0055
	Recall	0.997 ± 0.0080	0.992 ± 0.0075	0.984 ± 0.0075
	F_1	0.997 ± 0.0045	0.998 ± 0.0050	0.988 ± 0.0050

ally decreases, as seen in the PVCs class (from 0.990 ± 0.0115 to 0.939 ± 0.0075), indicating a higher rate of false positives. Conversely, recall remains relatively stable or even improves in some cases, such as in the N class (from 0.9915 ± 0.0030 to 0.993 ± 0.0030). This results in F1 scores that are generally stable or slightly lower compared to the unbalanced scenario, indicating that undersampling may help the model focus more on detecting positives but at the cost of precision.

The results demonstrate that while the 1D-CNN model performs robustly across all sampling strategies, trade-offs are depending on the data configuration. The unbalanced dataset provides the best overall performance with a good balance between precision and recall. Oversampling improves precision at the expense of recall, while undersampling can enhance recall but may reduce precision. The choice of sampling strategy should therefore align with the specific goals of the arrhythmia classification task, whether prioritizing the reduction of false positives or the capture of true positives.

4.4.5 Resource, Time and Cost Analysis

The performance of a proposed system in terms of resource utilization and time is investigated through raspberry pis and STM32 M4-cortex microcontrollers. The raspberry pi devices run raspbian, based on linux kernel version 4.14.52-v7+, with 1GB of RAM and 4-core ARM processors (BCM2837 @ 1.4GHz). The STM32 family relies on a linux system. Communication between different edge-gateways and their perception layer is facilitated through a local Wi-Fi network. Raspberry Pi 3 Model B+ and STM32 M4-cortex microcontrollers are utilized at edge-gateway layer as manager nodes. Lightweight nodes are implemented using ardiuno and less-computational powered bio-sensors. For simulating the application layer, desktop computers with an intel core i7 processor and 16GB of RAM are employed. The proposed approach is trained and investigated using the MIT-BIH dataset and real-time ECG signals collected from 18 individuals aged 18- 52 years.

Ethereum, a private network is leveraged as a service layer. To implement various components of the system, we utilize the Go programming language (golang), solidity for smart contracts, and a suite of web technologies (Node.js®, HTML5, CSS3, jQuery) for the front-end application. Smart contract deployment is accomplished using the Remix IDE. Metamask, a browser extension facilitating parallel transaction flows during experiments, generates data requests and transactions to and from third-party cloud services. Proof-of-stake consensus is used for block confirmation and gossip protocol to ensure fast, and attack-resilient message propagation for transaction handling, keeping nodes synchronized and avoiding forks. Complete implementation of *EdgeBot* system is described in section 5.1.

The proposed model demonstrates significantly lower resource consumption and implementation costs compared to recent studies. As shown in Table 17, the model effectively classifies multiclass arrhythmia using only two convolutional layers, optimized for deployment on single-board computers. Its implementation with biosensors, integrated with raspberry pi and STM-based boards, offers a cost-effective solution for home monitoring applications. This makes it a practical complement to large-scale ubiquitous healthcare systems, contributing to medical history documentation and enabling real-time responses in time-sensitive scenarios.

Table 19: Average Loading time of the different Arrhythmia classification requirements

	Execution Time
Loading Numerical Libraries	960 ms
Loading Tensorflow and Keras	1478 ms
Loading Trained Model	6683 ms

In table 19 we illustrates the average loading time corresponding to the numerical libraries, deep learning libraries (Tensorflow and Keras), and the model at edge gateways. Figure 21 shows the execution times of the different processes. The following steps are listed in order of importance: the extracting of data (ECG), arrhythmias classification (ARR), time to retrieve data (TRD), transaction confirmation time (TCT), and response time (RT). Complete data sharing and access scheme is described in section 3.3.5.

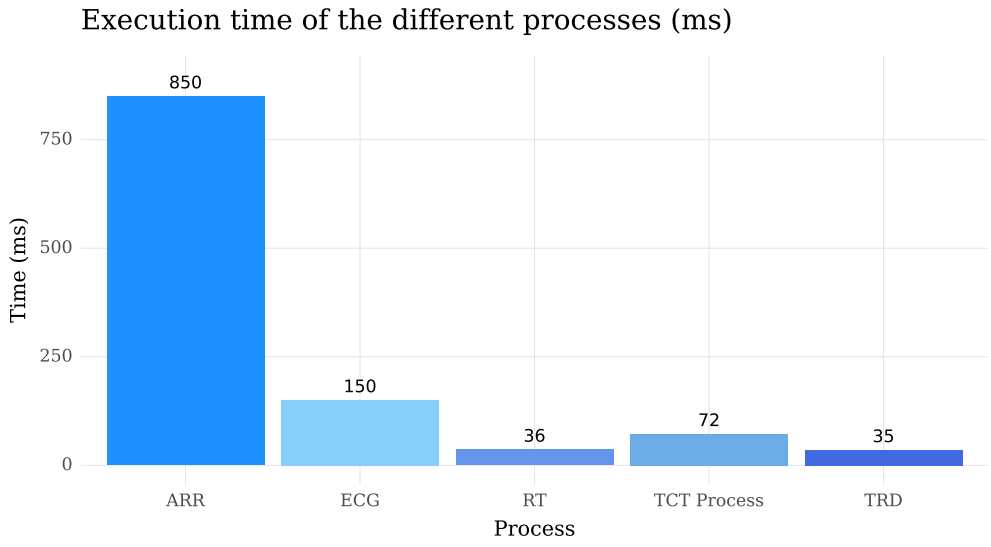


Figure 21: Execution time of the different processes in miliseconds

4.5 Summary

This chapter details the implementation of the proposed framework, EdgeBot, which functions as a continuous monitoring system for individuals, incorporating edge-level intelligence for rapid medical intervention. The system is designed to facilitate patient self-monitoring and preventive healthcare through the use of IoMT devices, employing computationally efficient methods. Processed medical data is securely stored using a private ethereum network, allowing for the sharing of anonymized information with legal guardians, healthcare providers, and researchers, while safeguarding patient privacy. The edge layer based on a single board computer is capable of processing privacy-critical sensitive information at the edge node. It ensures the user's privacy by discarding the raw data and only saving the processed information.

Firstly, SVM is employed for simple binary classification of abnormal signals. Additionally, a resource-efficient 1D-CNN is proposed for the multi-class classifi-

cation of arrhythmias. The chapter's key contributions include the deployment of machine learning algorithms at the edge device level for rapid anomaly detection and real-time detection and classification of arrhythmias using a two-channel ECG system. The system integrates various biosensors, including heart rate, temperature and humidity sensors, and an ECG monitoring system, to detect abnormalities and identify arrhythmias. This technology is particularly beneficial for promoting health and independence among the elderly and remote populations, where chronic diseases are increasingly prevalent. An extensive set of experiments and their detailed comparative analysis shows the viability of our proposed resource-optimized 1D-CNN in time-critical scenarios.

5 Peer-to-Peer Trustless Data Trade and Fair Access using Ethereum Platform

This chapter proposes and validates methods of peer-to-peer (P2P) trustless data trade using a private ethereum network. Transaction and network topologies are implemented through smart contracts, which operate as self-executing contracts and facilitate seamless and conflict-free transactions, trade agreements, and access policies. In the realm of personal medical data, *Edgebot* presents the opportunity to grant data owners (data producers) comprehensive control over their electronic health records (EHRs) by providing complete authority over access permissions and mitigates the risk of unauthorized individuals and organizations accessing personal health information. Furthermore, the private ethereum network acts as a secure repository for a patient's complete record, safeguarded by encryption through the patient's private key, thereby enhancing security beyond prevailing systems. The main contributions of this chapter include the following:

- We formulate and construct trustless peer-to-peer data trade flow system utilizing a private etherem network.
- We proposed a lightweight secure communication and data exchange scheme using ECDSA and ECIES along with it's implementation on single-board computers (raspberry pi and STM32F427 boards).
- We conduct a security, performance, and scalability analysis to demonstrate the efficiency and reliability of our proposed fine-grained access scheme.
- We proposed extension of *EdgeBot*, as a post-quantum-resistant solution for edge gateways. We integrate a resource-optimized, quantum-based secure data access scheme.

5.1 System Implementation

We implement P2P data trade policies of our proposed framework using the real-time health monitoring data at edge gateways (from chapter 4) as sample data as well as accelerometer, temperature and humidity data. Raspberry Pi 3 model B+ minicomputers are used as edge gateways (manager nodes of ethereum network) and

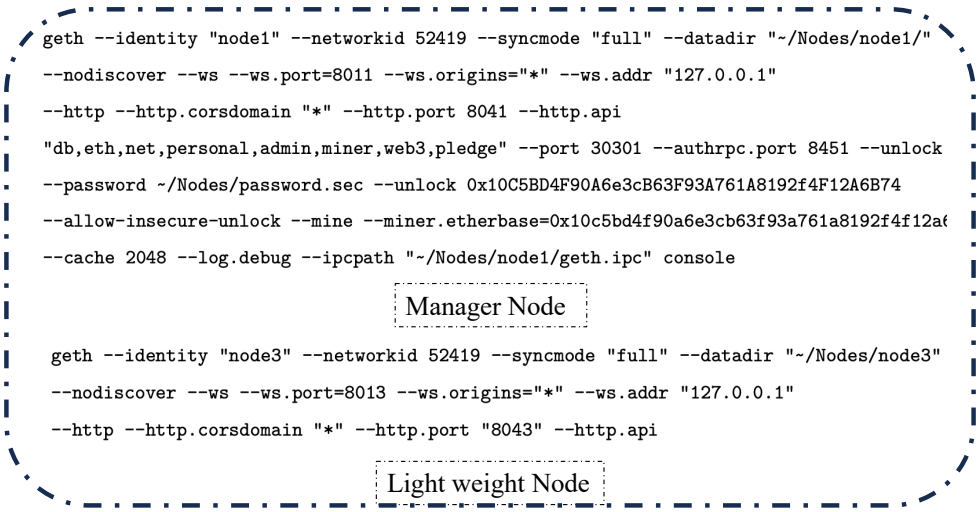


Figure 22: Manager nodes and lightweight nodes after initialization

lightweight nodes are implemented using STM32F427 development boards (low-power ARM Cortex M4 processors), which are used for high-speed implementation of asymmetric cryptographic algorithms. The elliptic curve digital signature algorithm (ECDSA) is used to generate public and private keys, and device authentication mechanisms. The STMicroelectronics X-CUBE-CRYPTOLIB library is utilized to implement several standard cryptographic algorithms with the ARM Cortex-M series processors.

For the implementation of the different parts of the system, the Go programming language (*golang*), Solidity for the smart contracts, and a suite of web technologies (Node.js®, HTML5, CSS3, jQuery) for the front-end application are used. We run raspbian based on linux kernel version 4.14.52-v7+ in the raspberry pi, which has 1GB of RAM and 4-core ARM processors (BCM2837 @ 1.4GHz). A local wi-fi network has been used to enable communication between different edge gateways and its data acquisition layer. We simulate the application layer with desktop computers that are equipped with an intel core i7 processor and 16GB of RAM and smart contracts are deployed using the Remix IDE. Metamask, a browser extension is used to enable parallel flow of transactions during the experiments, generates data requests and transactions to and from third-party cloud services. Proof-of-stake consensus is used for block confirmation and gossip protocol to ensure fast, attack-resilient message propagation for transaction handling, synchronizing nodes, and avoiding forks. Figure 22 shows Manager nodes and lightweight nodes after initialization.

The experimental setup is divided into six processes, as described below. Each process is accompanied by a short pseudo-code algorithm, which describes the different steps. SC is used to refer to smart contracts. The complete implementation of

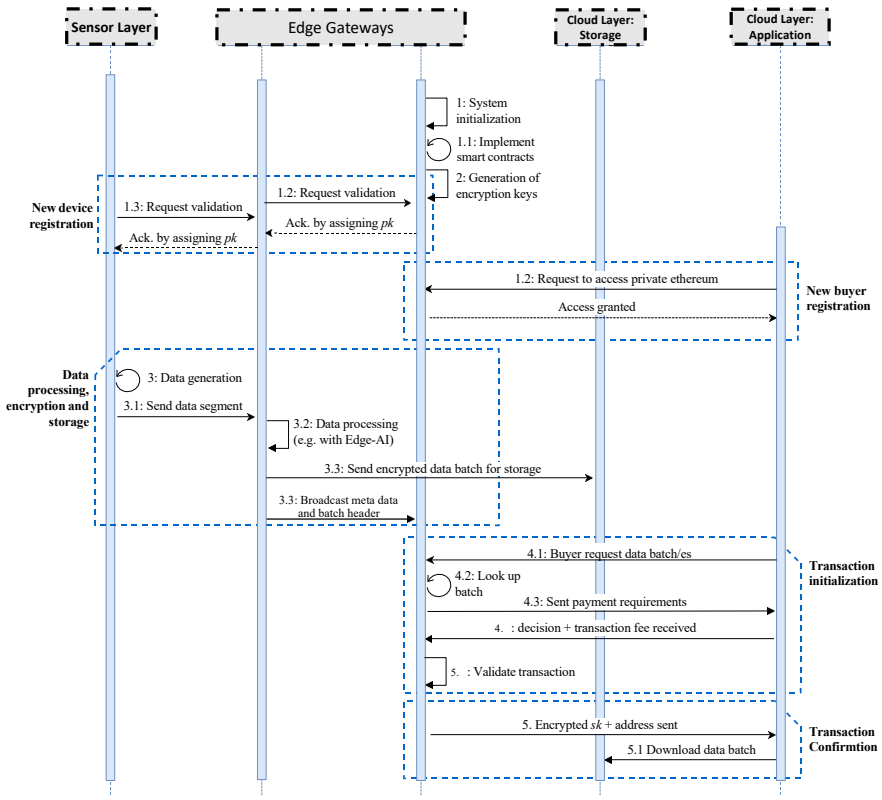


Figure 23: System sequence diagram depicting a sequential picture of complete network

system sequences is depicted in figure 23.

1. **System initialization:** The process of setting up a private ethereum network involves several steps. Encryption parameters and genesis file is generated 25, and the network is initialized. During this phase, clients and their devices are registered to the network, a necessary step for re-authentication. The registration process is split into two parts: user registration and device registration. In the user registration phase, a system administrator creates a unique ID for a client. This ID is then sent to the blockchain node as a transaction proposal. The node checks the network for the existence of this ID using a smart contract. If the ID is already in use, the transaction is rejected, and the admin is notified. The smart contract approves the transaction if the ID is not in use. Following the execution of the proof of stake (PoS) mechanism, a new block is created and shared with all the blockchain nodes. If the user registration is successful, the blockchain generates a user ID certificate with its private key and sends it to the admin. Admins can then extract the certificate information using their private keys.

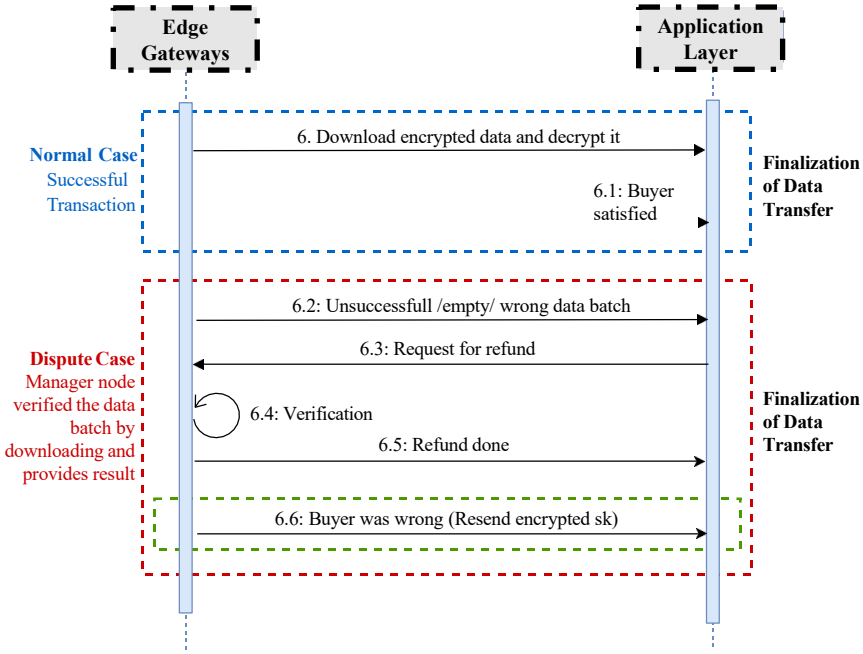


Figure 24: Sequential diagram to elaborate on possible scenarios to settle down transaction requests

The device registration phase is handled by edge gateways and the device registration request as a new transaction proposal. A smart contract checks the device ID and executes the transaction on the deployed network. Once the PoS process is finished, a new block is created and shared with all the nodes present in the network. All sensors and actuators are registered to the network with a unique ID, following the same process.

The generation of encryption parameters, creation of the genesis file, and initialization of the ethereum blockchain are performed through smart contracts. Through the implementation of smart contracts, terms of usage, certificates, and policies are defined. A new device/buyer registration process is also implemented.

2. **Generation of encryption keys:** Each connected device generates its key pair of secret and public keys (s_k, p_k). The secret key is randomly generated, and then the private key and child secret keys are derived from it. Each child's secret key is used to encrypt one data batch. For example, each child's secret key is used to encrypt a specific batch of data. The complete encryption process, along with key encapsulation and digital signature algorithm, is described in the section proposed access scheme 3.3.5.

Algorithm 1:

Result: System set up
ECDSA parameters:
$T = (P, a, b, G, n, H)$
Encryption keys length: κ;
Blockchain genesis block, Hash function F_h;
Smart contract codes SC;

Algorithm 2:

Result: Generation of encryption keys
Unique secret/public key pair (sk, pk)

Secret key: $sk = random(\kappa)$;
$pk = [sk]G$;
Child secret keys $csk_0, \ldots, csk_n$;

3. **Data processing, encryption, and storage:** Edge gateways divide the acquired data segments into data batches after every time T and implement embedded edge AI algorithms. If edge gateways do not have enough resources to process huge block of data, this specific data block is saved as raw data. After encryption, devices call the smart contract function. *NewDataBatch* function to add a transaction to the blockchain and includes the data hash, the encrypted AES encryption key, the timestamp of the storage operation, the type and size of stored data, and the price. Each batch is sent to storage after encryption and broadcasting the hash and metadata.

4. **Transaction Initialization:** On each data request, a smart contract function called *LookUpBatch* is used to query available records for a data batch. The buyer can initiate trade requests for its selected data batches after pre-defined deposit agreement. To perform this action, the smart contract must call the *Deposit* function, through which the secured channel builds and sends a secret key after the key encapsulation method and digital signature algorithm. It consists of its address or identifier and the batch identifier.

Algorithm 3:

Result: Data processing, encryption, and storage
while *True* **do**
    Data batch db;
    $data\_features = processEdgeAI(db)$;
    $hash = \mathbf{F}_h(data\_features)$;
    $db_E =$ **AES_encrypt**$(dfcsk)$;
    $db_{json} =$ **json**(db_E);
    $uid =$ **cloud_store**(db_{json});
    $db\_info = \{$
        uid,
        $hash$,
        $timestamp$,
        $data\_type$,
        $data\_size$,
        $price$,
    $\}$;
    SC.**NewDataBatch**(db_info);
    delay T_1;

Algorithm 4:

Result: Transaction Initialization
Data type: $data\_type$;
Maximum buying price: max_price;
Data batches for sale: $dbs = []$;
Time window lower limit: $start\_time$;
Time window upper limit: end_time;
Buyer's public encryption key: bpk;
$result = SC$.**LookUpBatch**$(data\_type,$
    $start\_time, end\_time)$;
foreach $(addr, db\_info)$ ***in*** $result$ **do**
    if $db\_info.selling\_price < max\_price$ **then**
        dbs.**append**$(\{dev\_addr, db\_info\})$;

foreach $(addr, db\_info)$ ***in*** dbs ***if*** ***meetsBuyingConditions****(db_info)* **do**
    Buying price: $price < max\_price$;
    Timestamp: ts;
    $sign = price \oplus db\_info.uid \oplus addr \oplus ts$;
    SC.**Deposit**$(price, uid, addr, bpk, sign)$;

5. **Transaction Confirmation:** This process runs periodically, requesting any *Data-Batch* requests that are present and depositing the requested data. Edge gateways encode the *csk* with the buyer's public key using the key encapsulation method and confirm the *Deal* if any deposit meets the pre-defined conditions for the type of data in the requested batch. A combination of AES and ECDSA is used and the process is automated using turing complete capability of smart contracts. Figure 26 depicts the real-time transaction confirmation.

6. **Finalization of data transfer:** When a receiver receives an encrypted data batch

AES key, it uses the ECIES decryption algorithm to obtain the AES batch key. Then, it queries the storage provider with the batch address and decrypts it to obtain the information. The data transfer, or purchase, is done if a buyer is satisfied. If the receiver is unsatisfied with the received data batch, it will ask for a refund. The edge gateways will resolve a dispute by cross-checking the batch details. After getting the results, gateways will respond accordingly. The complete process of transaction confirmation and finalization of data transfer is depicted in figure 24.

Algorithm 5:

Result: Transaction Confirmation
while *True* **do**
    $results = SC.$**LookUpDeal**$(device\_address)$;
    foreach $result$ ***in*** $results$ ***if*** ***meetsSellingConditions****(result)* **do**
        $csk_E =$ **ECIES_encrypt**$(csk, result.bsk)$;
        $addr =$ **cloud_address**$(result.uid)$;
        $SC.$**Deal**$(result.bsk, csk_E, uid, addr)$;
    delay T_2;

Algorithm 6:

Result: Finalization of Data Transfer
$result =$
SC.**LookUpDeposit**(bsk, uid)
if *result.done* ***is*** *True* **then**
    $csk =$ **ECIES_encrypt**$(result.csk_E)$;
    $data_E =$ **cloud_request**$(result.addr)$;
    $data\_batch =$ **AES_decrypt**$(data_E, csk)$;
T revert;

5.2 Data and communication security

Our proposed access scheme is used along with the widely used Elliptic Curve Integrated Encryption Scheme (ECIES), Child Key Derivation function (CKD), and the Elliptic Curve Digital Signature Algorithm (ECDSA) are cryptographic techniques to ensure secure data storage and communication. ECIES works by independently deriving a bulk encryption key and a MAC key from a common secret. The data is first encrypted under a symmetric cipher, and then the ciphertext is authenticated under an authentication scheme. Finally, the common secret is encrypted under the public part of a public/private key pair. The CKD function is used for managing data in batches; child key derivation functions are used in hierarchical deterministic wallets (HD wallets). It helps in generating a tree of keys from a single master key, which can be very useful for managing multiple keys in a secure and systematic way. ECDSA is a digital signature algorithm that is used for secure key sharing

for both data and communication. The ECDSA ensures that the data and communication are coming from the stated sender (authenticity), have not been altered in transit (integrity), and repudiation by the sender can be disputed (non-repudiation). Implementing ECIES, CKD, and ECDSA in this proposed system provides a robust framework that ensures secure data storage and communication. The ECIES offers a strong encryption scheme for data protection, the CKD provides an efficient way to manage data in batches, and the ECDSA guarantees secure key sharing and data authenticity.

5.2.1 Results and Analysis

We have conducted a performance evaluation of three different types of STM32F427 M series processors within the framework of asymmetric cryptography. We utilized the X-CUBE-CRYPTOLIB library to implement the ECDSA. To ascertain the statistical error of the results obtained over the number of executions, we calculated the mean, standard deviation, and standard error using the appropriate equations.

The execution time was determined, which encompasses the total time required for key generation, encryption, and decryption using ECDSA. To identify the optimal execution time of ECDSA, we examined the records of different numbers of executions for each processor. The execution time for each processor for ECDSA is visually represented in figure 27. For a comprehensive analysis, the execution time was calculated in terms of mean, standard deviation, and standard error for each processor.

The estimated execution time of ECDSA for processor M3 is 26.352 s $\pm$ 0.002s, and the execution time for M4 processor is 1.451s $\pm$ 0.007s and 1.167s $\pm$ 0.002s for M7. Based on the results, the average execution times of M3 processors are 17.253 seconds, while the execution times for processors M4 and M7 are 1.462 seconds and 1.156 seconds, respectively. The data suggests that the M4 and M7 processors exhibit superior performance in executing ECDSA. These time measurements facilitate easy planning and adjustments to determine the delay tolerance in the network.

Power consumption is an important parameter of microcontrollers, we determine power utilization during the execution of cryptographic algorithms. The power consumed was determined by measuring the voltage across a shunt resistance R. The current consumption of the processors was calculated using ohm's law. To determine the average power consumption, ECDSA was executed for 15 runs and a comparison of power consumption is presented in figure 28. The average power consumption by M3 and M4 was $\pm$ 200mW, whereas M7 used an average of $\pm$ 290mW. Results indicate the superior performance of M4 cortex microcontrollers are best fit while consuming fewer resources.

Table 20 illustrates how different security measures have been taken into account utilizing *Edgebot*, such as service attacks, denial of service (DDoS), and man-in-the-

Table 20: Security Measures

Parameter	Implementation
Authorization	Public key cryptography to encrypt CSK
Confidentiality	Public key cryptography and proof of authority
Integrity	Broadcast hash of each data_batch
Availability	Achieved by limiting number of requests
Anonymity	Discard raw data and store only processed information

middle (MITM). By limiting requests from the edge gateways to bio-sensors, registered users, and vice versa. Our proposed system tackles DoS and time-delay attacks. *Edgebot* can combat attacks on its security by using different encryption schemes and system elements. The elliptic curve integrated encryption scheme (ECIES) [178] is used in the project to ensure that data is saved securely. ECIES is used in the key encapsulation mechanism combined with the data encapsulation mechanism, which is used to secure an edge gateway's public key and send encrypted data to a central management node. ECC requires fewer keys to provide the same degree of security as Rivest-Shamir-Adleman (RSA), the more widely used cryptosystem. However, ECC usually does not require as many computational and memory resources, which makes it ideal for computing devices with limited resources. A public key is established in practice when data transfers occur in a unidirectional fashion or when data is encrypted and stored using the public key.

A deterministic wallet is used for CKD functions to determine a child's key from a parent's key. Using this technique, each batch of data to be encrypted in the device is given a unique secret key [179]. The 512-bit hash is calculated according to the parent's public key (public and private keys are 256 bits) and the desired child index. It is impossible to deduce the original parent key from the nth-child key because of the one-way hashing used in the process. This process appears to generate random numbers due to the additions *modulo n*.

While the elliptic curve digital signature algorithm (ECDSA) [180] is used to share the unique key for both the data and communication. After the same buyer makes three requests per minute, double authentication is required. Buyers obtain encrypted data from manager nodes (edge gateways) that are maintained on a third-party storage service, e.g., the cloud. Instead of sending the requested data batch directly to the buyer, the manager node transmits the encryption key of the requested data batch, along with its address on the cloud. In addition to reducing bandwidth requirements and computational workloads for edge gateways, this scheme also avoids storage limitations. The *Edgebot* uses a sophisticated encryption scheme (AES) for storing private data, cloud services, and other third-party data storage solutions. The ability to store data securely in an encrypted is particularly important for minor nodes in the network with insufficient resources for local storage after data processing. We use asymmetric encryption to transfer the child secret key securely, *csk*, to the buyer.

If csk_N is the key to the data batch N, then it is the key for the data batch. This will be followed by the node encrypting csk_N with the buyer's public key, which will decrypt at the receiving end by the buyer's secret key. The encrypted payload contains information regarding the location of the stored information, as well as csk_N, if the buyer did not already know this information.

5.3 Resource and performance Analysis

Resource consumption analysis is conducted based on its performance during transaction initialization, handling and committing on ethereum ethereum. A comparison of the percentage consumption of RAM and CPU is shown in Figure 29. At idle, RAM usage is approximately 17%, and during transactions, the percentage of RAM usage increases by 10% to 15% with an average of 24.31%, which does not significantly impact overall system memory usage during a transaction. Nevertheless, there are sharp peaks in CPU utilization; in idle mode, only 8% of the CPU resources are consumed on average, while during a transaction, it can consume up to 45% with an average of 16.47%.

The edge gateway measures and divides the time needed to complete a transaction into three sections: Time to retrieve data (TRD), time for checking the transaction (VTR), and time for confirming the transaction (TCT). Figure 30 presents the results of the measurements. Based on the available resources, it is impressive that TRD requires only 34.6 milliseconds on average, VTR 36 milliseconds on average, and TCT 73.6 milliseconds on average. Additionally, it is essential to note that TCT also relies on the network, which in this experiment was adversely affected by our shared Wi-Fi's slow response time, causing the time to be extended overall.

end-to-end delay = request initialization by interested buyer + time to retrieve metadata
+ response time by manager nodes + time to confirm one transaction

Figure 31 illustrates that increasing concurrent transaction requests leads to increased end-to-end delays. This study's results demonstrate the proposed model's effectiveness for autonomously implementing data trade in information-critical systems. Using this trustless structure, data trade becomes more reliable and transparent. We have concluded that single-board computers can act as data and transaction managers, with no need for third-party cloud services, as the necessary computation makes space for other edge services and data processing processes to run simultaneously. However, a parallel number of transactions will cause a significant delay.

5.3.1 Scalability Analysis

Edge gateways based DLTs offer several benefits but also face inherent limitations in terms of scalability, which limit their application range. Nevertheless, *EdgeBot*

handles this challenge using the side chains concept, gossip protocol, and PoS consensus. FoBSim simulation tool[1] is utilized to check the scalability of the proposed model. Manager nodes ranging from 5 to 500 were used to check the performance matrix of the proposed model and measure the total time required to complete the request procedure at edge gateways versus at the cloud layer. Figure 32 shows that cloud layer utilises around double the time as compared to edge gateways to complete transaction requests during concurrent transactions starting from 5 to 100 transactions at a time.

The impact of the gossip protocol on the total elapsed time at edge gateways was evaluated to demonstrate its scalability enhancement. Figure 33 illustrates the improved scalability of the proposed access scheme when utilizing the gossip protocol, as compared to transactions conducted without it. Block confirmation time was also measured using different numbers of manager nodes, and the average time is illustrated in the figure for proof of stake versus two famous consensus algorithms PoW and PoA 34. *EdgeBot* does not present a scalability issue due to the fact that it works in a private P2P network that can be segmented into side chains. It does not require edge gateways to process many requests per minute. The manager nodes must check whether new requests for data have been received after every time T. If it has a queue of requests, the manager nodes will respond to each request one at a time.

5.4 Post-Quantum Cryptographic Communication Protocol

We extend our proposed model *EdgeBot* as a post-quantum resistant framework. With the advent of quantum computers, coupled with Shor's algorithm, resource constrained devices can potentially compromise their communications and data exchange process that depend on traditional cryptosystems. We demonstrate the feasibility of integrating post-quantum key encapsulation and key exchange methods along with digital signatures to ensure the integrity of data for ubiquitous healthcare systems. We consider algorithms currently participating in the NIST competition for post-quantum encryption standards and integrate them in *EdgeBot* to assess the impact on device resource consumption in real-time scenarios.

One of the key candidates for post-quantum cryptography is the Kyber protocol for resource-constrained devices, which is a lattice-based key algorithm. The Kyber protocol has several advantages in the selection of post-quantum algorithms and is likely to become the standard algorithm for public key encryption.

[1] https://github.com/sed-szeged/FobSim

5.4.1 System Seutp

Initially, we configured Mosquitto MQTT broker by setting up server certificates and adjusting the $mosquitto.conf$ file to incorporate security certificates, and specifying both TLS version and port number. For handling multiple parallel ports, a linux system is employed, with static IP addresses assigned to different ports. This is achieved through the use of multiple containers in docker. In the real-time implementation, an LM75A temperature sensor is interfaced with an Arduino board to read sensor data and its transmission to the Mosquitto MQTT broker over a TLS 1.3-secured connection. Arduio Uno is used to get real-time data samples from client nodes.

To create an MQTT client on the Arduino, the $ArduinoMqttClient$ library is utilized for publishing and subscribing to MQTT topics. The LM75A sensor operates via the I2C communication protocol, which uses the SDA and SCL pins on the Arduino to measure ambient temperature. The Arduino code's loop function is modified to read the temperature data from the sensor, convert the reading into a string format, and publish this string to a topic on the MQTT broker.

5.4.2 key encapsulation mechanism (KEM)

KEM combines the kyber key encapsulation algorithm with the kyber public-key encryption algorithm. The process of encryption and decryption of messages using the kyber encryption algorithm involves several steps which are explained in algorithms 7,8 and 9. In these pseudo algorithms, 32-octet cpaseed is used as input for the key generation process. $cpaPublicKey$, $cpaPrivateKey$, h, and z are variables representing the components needed to derive the private key. The publicKey is derived directly from $cpaPublicKey$, while the privateKey is derived from a combination of $cpaPrivateKey$, $cpaPublicKey$, h, and z. To ensure the security of its operations, kyber uses cryptographic primitives such as pseudorandom functions (PRF), extendable output functions (XOF), key derivation functions (KDF), hash functions (H), and generators (G). These cryptographic primitives play a crucial role in maintaining the integrity and confidentiality of the shared secret.

Algorithm 7: Key Generation

Input : 32-octet cpaseed
Output: Public Key, Private Key

Function *Kyber512KeyGeneration(*Seed*)* **:**
    PublicKey ← cpaPublicKey;
    PrivateKey ←
     cpaPrivateKey,cpaPublicKey,h[32],z[32];
    return *PublicKey, PrivateKey*;

Call the Key Generation function:
Seed ← 32-byte seed;
PublicKey, PrivateKey ←
`Kyber512KeyGeneration(`*cpaSeed*`)`;

Output the generated keys:
Public Key: *public key*;
Private Key: *private key*;

Algorithm 8: Kyber512 Encapsulation

Input : Public Key
Output: Ciphertext, Shared Secret

Function *Kyber512Encapsulation(*PublicKey*)* **:**
    SharedSecret ← GenerateRandomValue();
    Ciphertext ← Encrypt(SharedSecret, PublicKey);
    return *Ciphertext, SharedSecret*;

Call the Encapsulation function:
PublicKey ← your public key here;
Ciphertext, SharedSecret ←
`Kyber512Encapsulation(`*PublicKey*`)`;

Output gen ciphertext and shared secret:
Ciphertext: *ciphertext*;
Shared Secret: *shared secret*;

Algorithm 9: Kyber512 Decapsulation

Input : Private Key, Ciphertext
Output: Shared Secret

Function *Kyber512Decapsulation(*PrivateKey, Ciphertext*)* **:**
    SharedSecret ← Decrypt(Ciphertext, PrivateKey);
    return *SharedSecret*;

Call the Decapsulation function:
PrivateKey ← *private key*;
Ciphertext ← ¡*ciphertext*;
SharedSecret ← `Kyber512Decapsulation(`*PrivateKey, Ciphertext*`)`;

Output the generated shared secret:
Shared Secret: *shared secret*;

5.4.3 Dilithium3

Dilithium3 is a lattice-based learning with errors (LWE) problem, which is believed to be resistant to attacks by quantum computers. It offers a high level of security by relying on the hardness of the LWE problem. It has been extensively analyzed by cryptographers and proven to be resistant to various cryptographic attacks. Despite its strong security guarantees, it is designed to be efficient in terms of computation, memory usage, and bandwidth requirements. This makes it suitable for a wide range of applications, including resource-constrained devices and high-performance systems. It is one of the leading candidates for post-quantum secure digital signatures, and it has been submitted to the National Institute of Standards and Technology (NIST) for standardization. Its inclusion in the NIST post-quantum cryptography competition demonstrates its credibility and potential for widespread adoption. The process of signature verification using dilithium3 involves several steps which are explained in algorithms 10,11 and 12.

Algorithm 10: Key Generation

Data: Predefined parameters
Result: Public key (pk), Secret key (sk)
Parameter Setup:;
parameters ← predefinedParameters();
Generate Secret Key:;
$s1, s2$ ← generateSecretPolynomial(parameters);
seed ← generateRandomSeed();
$sk \leftarrow (s1, s2, \text{seed})$;
Generate Public Key:;
A ← generateRandomPolynomial(parameters);
$(T1, T0)$ ← generateCommitment(A, sk);
z ← generateNoise(parameters);
$T \leftarrow T1 + T0 + z$;
spk ← generateRandomSeed();
hpk ← hashPublicSeed(spk);
$pk \leftarrow (T, \text{spk}, \text{hpk})$;
return (pk, sk);

Algorithm 11: Sign and Verify

function *sign_message*:
  sk, msg
Input : Secret key sk, message msg
Output: Signature
$signature$ ← sign_message_with_secret_key
(hash_message(msg), sk)
return $signature$;
$secret\_key$ ← generate_secret_key();
$message$ ← generate_random_message();
$signature$ ← sign_message($secret\_key, message$);
if *verify_signature*(*public_key*, $message, signature$)
 then
  print("Signature is valid");
else
  print("Signature is invalid");

5.4.4 Performance Analysis:

In the design process, memory is a crucial factor to consider, especially for resource-constrained devices. Efficient memory management is crucial to balance the trade-offs between computational speed, energy consumption, and security. Moreover, the optimization of memory usage is essential to mitigate the risks of side-channel attacks and ensure the overall performance and security of cryptographic operations in such devices. To check the memory efficiency of our implemented KEM, kyber512, we consider other lightweight KEMs lightsaber, NTRU, and frodoKEM640 for comparison analysis. In figure 36, maximum amount of memory (bytes) used during encapsulation method is depicted. kyber512 requires around 22-25 bytes of memory while it is implemented on STM32F427 (M4 cortex).

Based on the study conducted, the best-performing KEMs for resource-constrained devices are Kyber512 and LightSaber. These mechanisms strike a balance between energy consumption and memory storage. However, it is important to note that there is a trade-off between the security level and computational resources required. While choosing KEMs with lower security levels may result in smaller key sizes and reduced computational resources, it is crucial to ensure that the chosen mechanisms still offer sufficient security.

Another important consideration in resource-constrained devices is latency during key generation, encryption, and decryption process of KEMs and digital signatures. Figure 37, shows the latencies (ms) we got using kyber512 and dilithium3 using TLS-based secure communication over wi-fi. Results ensure that resource-constrained devices can leverage post-quantum cryptosystems effectively without compromising security or consuming excessive computational resources and time.

5.5 Summary

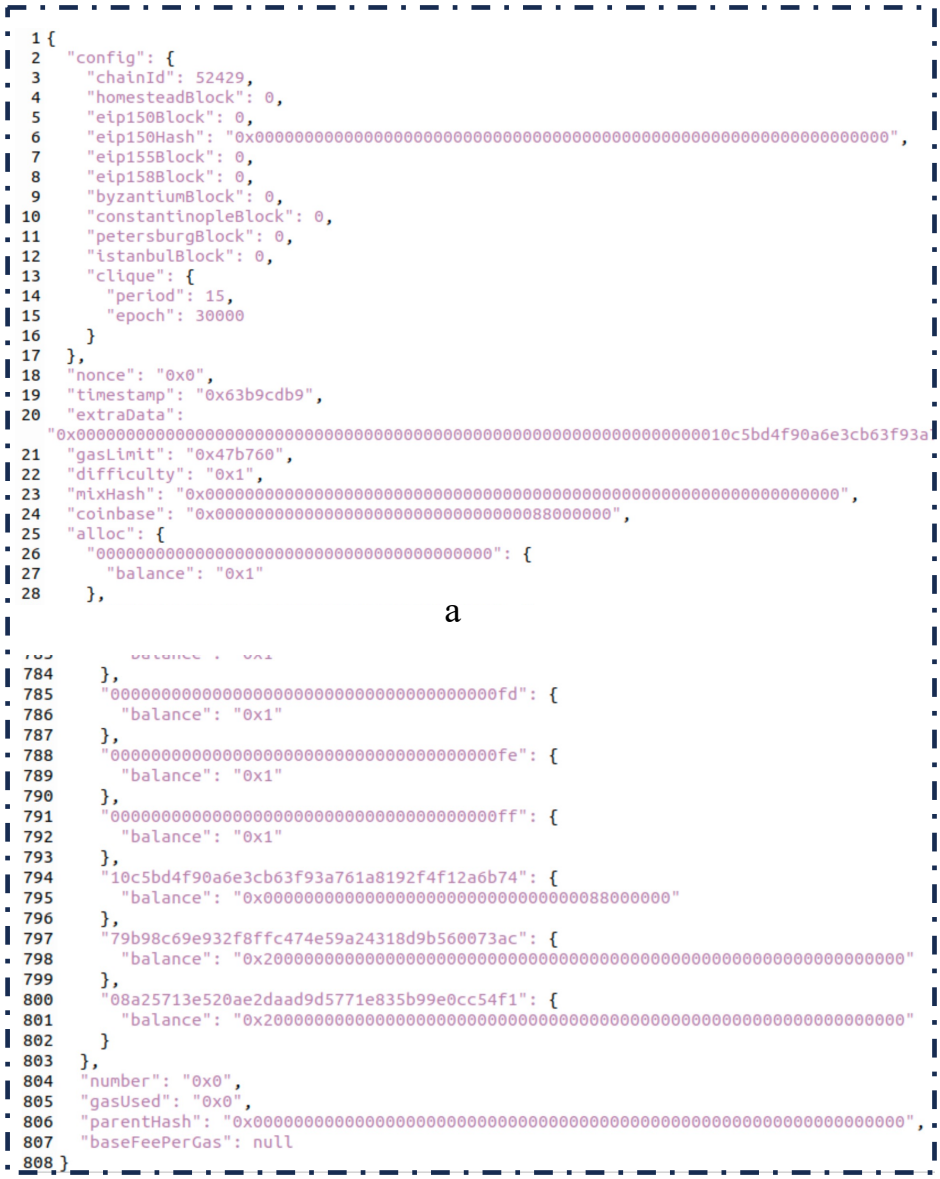
Ethereum based P2P trustless data trade is proposed and implemented using Pi as edge gateways, while STM32F427 development boards act as lightweight nodes for the high-speed execution of asymmetric cryptographic algorithms. ECDSA is employed for key generation and device authentication using X-CUBE-CRYPTOLIB library. Implementation results on M3,M4 and M7 show various execution times for ECDSA, with the M3 taking an average of 17.253 seconds, while the M4 and M7 perform significantly faster, at 1.462 and 1.156 seconds, respectively. Power consumption is also measured, with M3 and M4 consuming around 200 mW and M7 using approximately 290 mW. These results highlight the superior performance of M4 processors, which offer a balance between efficiency and lower resource consumption, making them the best fit for this implementation.

Resource consumption analysis shows only a maximum of 26% and 45% of RAM and CPU usage during ethereum transaction handling, respectively. The time

required to complete transaction modules shows an average of 34.6ms, 36ms and 73.6ms for TRD, VTR, and TCT respectively. The FoBSim simulator[2] is used to assess the scalability of the proposed model, with manager nodes ranging from 5 to 500. Results show that cloud processing takes roughly double the time compared to edge gateways during concurrent transactions. The use of the gossip protocol reduces latency by about 35% and block confirmation using POS exhibits minimal latency even with 500 concurrent transactions, outperforming POW and POA.

EdgeBot is extended as a post-quantum resistant framework and the system was configured using a mosquitto MQTT broker. The memory efficiency of Kyber512 KEM was tested against other lightweight KEMs like LightSaber, NTRU, and FrodoKEM640 on an STM32F427 microcontroller. Kyber512 required 22-25 bytes of memory. The study found Kyber512 and LightSaber to be the best-performing KEMs, balancing energy consumption and memory usage. However, there is a trade-off between security and computational resources, where lower security levels reduce resource demands but must still ensure adequate protection.

[2] https://github.com/sed-szeged/FobSim



```
{
  "config": {
    "chainId": 52429,
    "homesteadBlock": 0,
    "eip150Block": 0,
    "eip150Hash": "0x0000000000000000000000000000000000000000000000000000000000000000",
    "eip155Block": 0,
    "eip158Block": 0,
    "byzantiumBlock": 0,
    "constantinopleBlock": 0,
    "petersburgBlock": 0,
    "istanbulBlock": 0,
    "clique": {
      "period": 15,
      "epoch": 30000
    }
  },
  "nonce": "0x0",
  "timestamp": "0x63b9cdb9",
  "extraData":
"0x0000000000000000000000000000000000000000000000000000000000000000010c5bd4f90a6e3cb63f93a
  "gasLimit": "0x47b760",
  "difficulty": "0x1",
  "mixHash": "0x0000000000000000000000000000000000000000000000000000000000000000",
  "coinbase": "0x0000000000000000000000000088000000",
  "alloc": {
    "0000000000000000000000000000000000000000": {
      "balance": "0x1"
    },
```

a

```
    },
    "00000000000000000000000000000000000000fd": {
      "balance": "0x1"
    },
    "00000000000000000000000000000000000000fe": {
      "balance": "0x1"
    },
    "00000000000000000000000000000000000000ff": {
      "balance": "0x1"
    },
    "10c5bd4f90a6e3cb63f93a761a8192f4f12a6b74": {
      "balance": "0x0000000000000000000000000088000000"
    },
    "79b98c69e932f8ffc474e59a24318d9b560073ac": {
      "balance": "0x2000000000000000000000000000000000000000000000000000000000000000"
    },
    "08a25713e520ae2daad9d5771e835b99e0cc54f1": {
      "balance": "0x2000000000000000000000000000000000000000000000000000000000000000"
    }
  },
  "number": "0x0",
  "gasUsed": "0x0",
  "parentHash": "0x0000000000000000000000000000000000000000000000000000000000000000",
  "baseFeePerGas": null
}
```

b

Figure 25: (a)Genesis file creation at the start of the network (b) Genesis file after initialization of few nodes

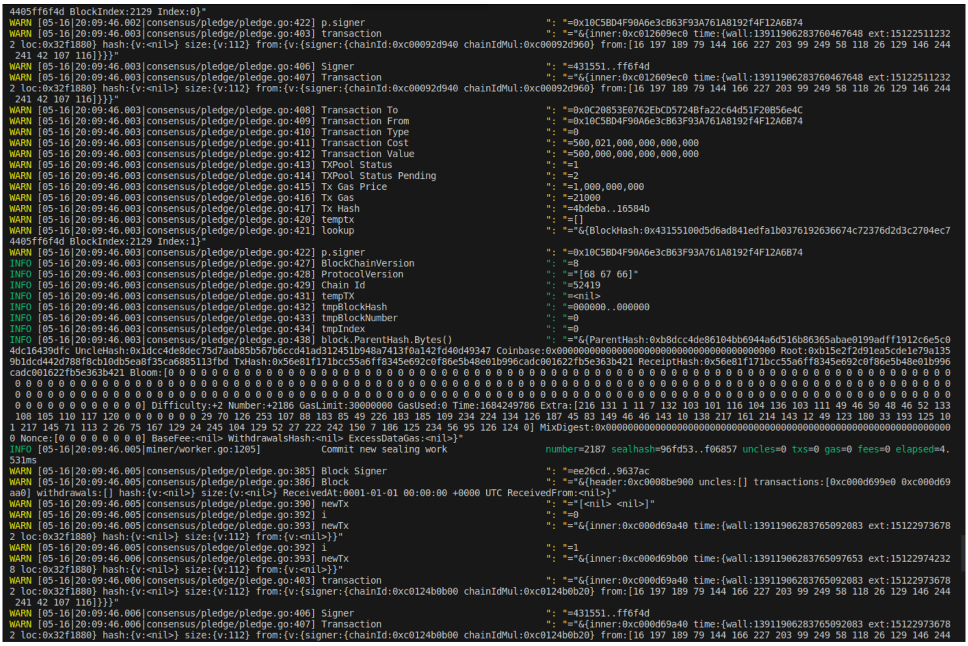


Figure 26: Transaction confirmation using etherem

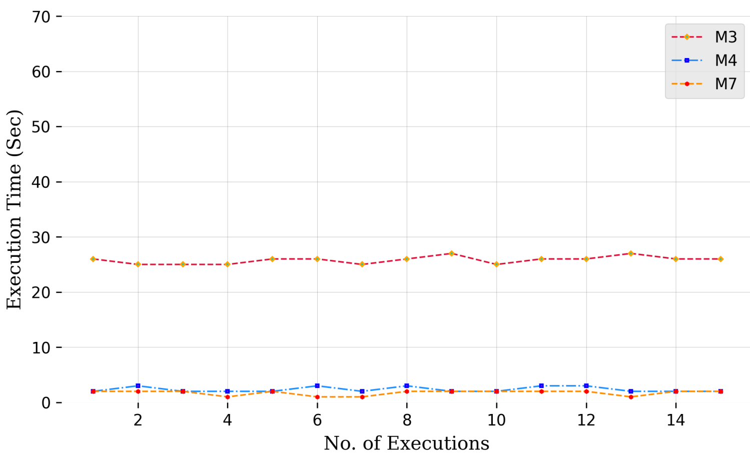


Figure 27: Execution time during ECDSA implementation using STM32F427 M series processors

Figure 28: Average power consumption during ECDSA implementation using STM32F427 M series processors

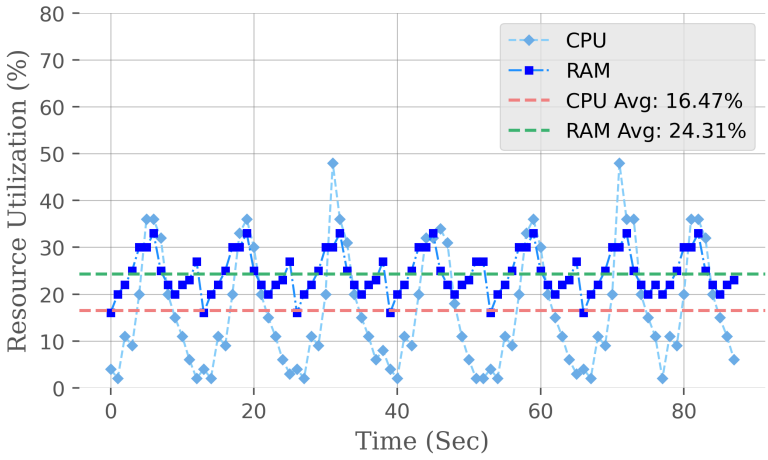


Figure 29: Consumption of CPU and RAM during randomly selected transactions and idle time in between.

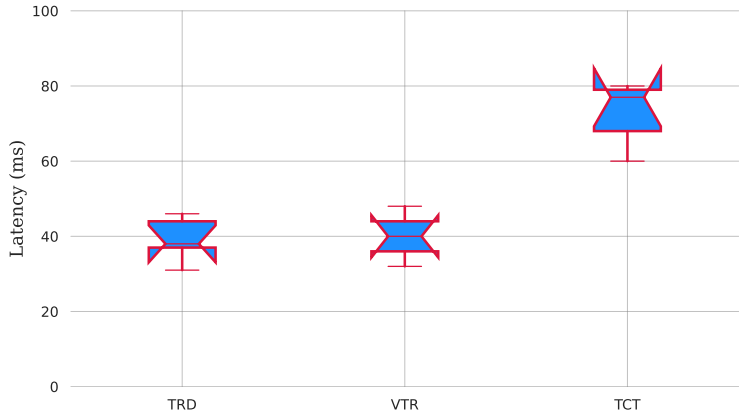


Figure 30: Latencies to retrieve mata data (TRD), transaction validation time (VTR) needed for single transaction request, and time required to confirm one transaction (TCT).

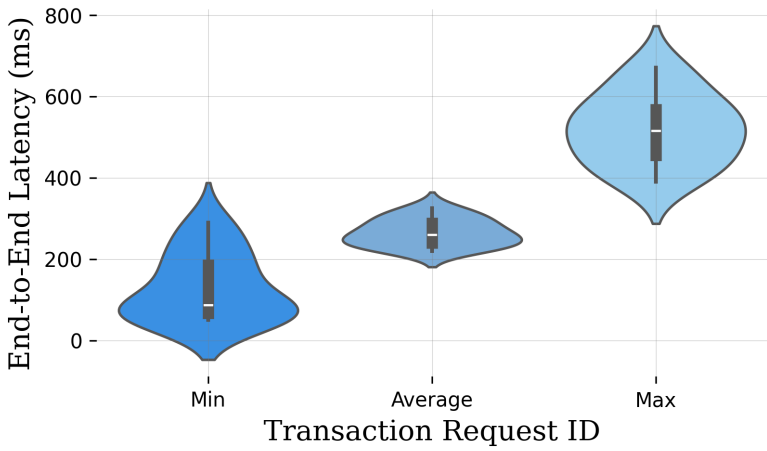


Figure 31: Impact of concurrent transactions on end-to-end transaction latency.

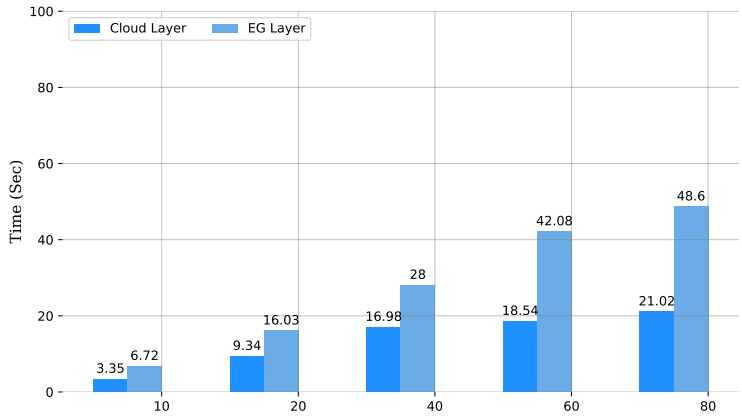


Figure 32: Impact of concurrent transactions on end-to-end transaction latency on *EdgeBot* versus cloud services

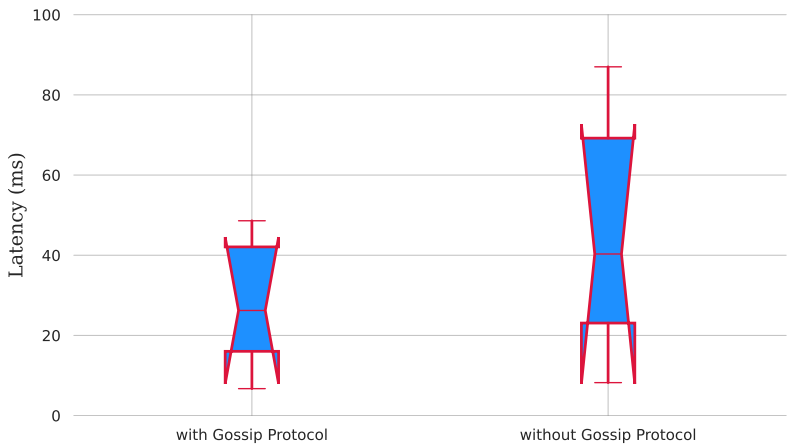


Figure 33: Impact of concurrent transactions on end-to-end transaction latency using gossip protocol versus without gossip protocol

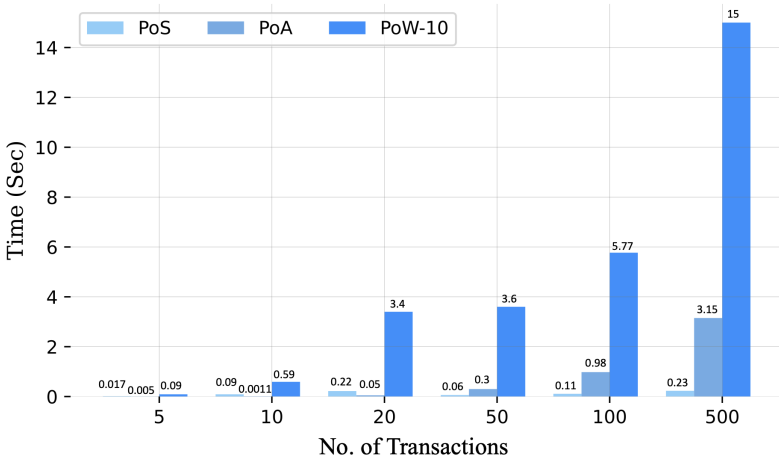


Figure 34: Average Block Confirmation Time using PoS, PoW-10, and PoA

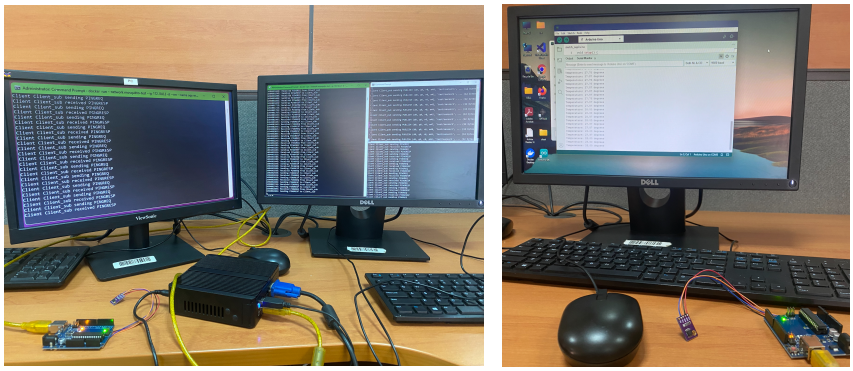


Figure 35: System Setup using Mosquitto MQTT broker and LM75A temperature sensor

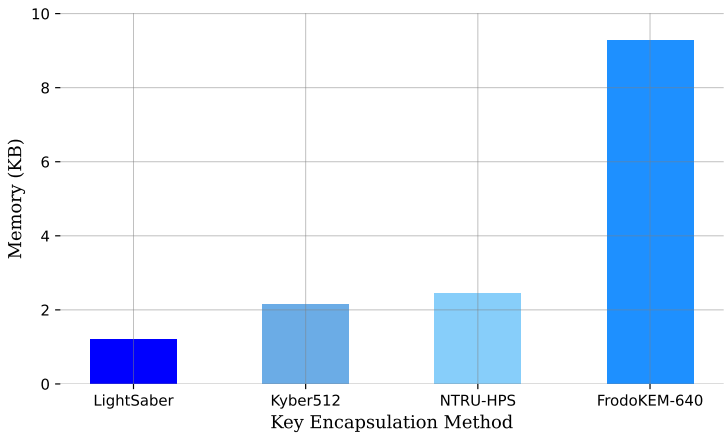


Figure 36: Maximum amount of memory, in bytes, that each KEM uses

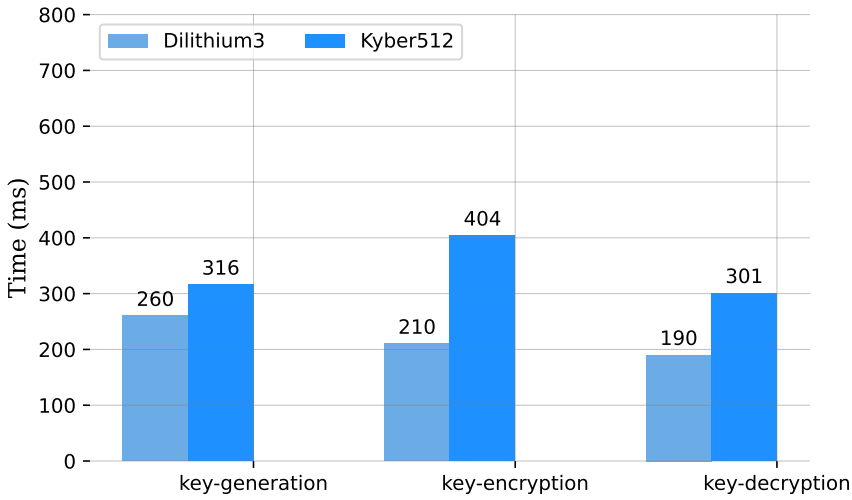


Figure 37: Latency in (ms) for kyber512 and dilithium3 proccess

6 Real-time Data sharing and tracing while preserving transparency using Hyperledger sawtooth

This chapter proposed sawtooth based real-time data sharing and tracing among trusted stakeholders (hospitals, healthcare service providers, and regulatory authorities) while offering robust solution to large-scale applications. A highly modular framework addresses scalability and simplifies the integration process with existing technologies, thereby facilitating a smoother transition as well as an adaptive design to accommodate various forms of healthcare scenarios and their compliance requirements. The main contribution of this chapter includes:

- We construct a sawtooth-assisted real-time data sharing and tracing system among trusted stakeholders while preserving transparency.

- We propose service optimization in sawtooth, a mathematical foundation for determining the most efficient combination of databatch size, transactions per second (tps), resource utilization and network resources while ensuring the reliability of transaction commitment.

- The Proposed scheme resolves scalability issues by providing a robust foundation capable of supporting large-scale operations by utilizing a highly modular structure of hyperledger sawtooth. Moreover, it addresses interoperability challenges by fostering a cohesive and collaborative environment among the diverse stakeholders in healthcare ecosystem.

- We conduct security and performance analysis, which demonstrates the efficiency and reliability of our scheme.

6.1 Hyperledger Sawtooth Framework

Under the broad spectrum of open-source hyperldeger frameworks, hyperledger sawtooth is a private permissioned network proposed and built by the linux foundation [181]. Intel designed this distributed ledger specifically for highly modular business logic where governance bodies must customize rules and regulations in a

run-time environment while maintaining immutability and privacy. Its modularity separates its core system and the application domain. Therefore, each participating entity can define a set of rules according to its requirements without knowing the underlying business logic of the core system.

In sawtooth, business decisions take place on the transaction processing layer, where transaction families work as models to handle low-level functions like sets of permissions, policies, and storing block states and logs. Transactions are explained by the ”transaction family” through which a transaction state changes. Corresponding transaction processors are bonded to the transaction's execution by each entity. This modular structure can handle several consensus protocols such as Practical Byzantine Fault Tolerance (PBFT) [113], RAFT [182], Proof of Elapsed Time (PoET) and PoET simulator [183].

Core Modules of hyperledger sawtooth are described in detail:

6.1.1 Consensus Protocol

A procedure in which all participating nodes of a network endorse or reject some transactional state is known as a consensus protocol. Several options with different attributes like throughput, finality, size, latency, threat model, censorship resistance, and failure model are available. Distributed technologies build trust among untrusted members through these algorithms. After facing many drawbacks in proof-of-X protocols, PBFT, RAFT and PoET consensus protocols are proposed with hyperledger sawtooth. The PBFT consensus engine is preferable to maintain fault tolerance and resolve issues in the original chain. Raft is a leader-based consensus protocol that is preferable for a small number of participating nodes. Intel has presented proof of elapsed time (PoET) [183], a promising, novel consensus protocol, and their SGX hardware as a trusted environment.

Nevertheless, PoET can also be used without Intel hardware support by using the PoET simulator with hyperledger sawtooth. This protocol stands out for many reasons. It is considered one of the most robust implementations of the proof-of-X protocol and comes as a part of the sawtooth project. Furthermore, it is highly parameterizable, unlike the bitcoin protocol. Moreover, it is a currency-independent protocol, which makes it best suited for use cases without financial transactions. As it comes as a suit with a sawtooth, its working environment follows the sawtooth structure:

- Validator node requests for a waiting time from an enclave (trusted module).
- Enclave assigns waiting time randomly to each validator.
- Leader is elected by checking the validator with the shortest wait time ”Create-Timer” function creates a timer which guarantees the creation of transaction

blocks by an enclave.

- Validator can claim leadership after finishing the allocated waiting time.

6.1.2 Data Model

The architectural design of the data model incorporates a sequentially arranged collection of transactions, logs for transactional activities, and a distributed framework for data storage to maintain the resultant states. The management of transaction serialization is facilitated by deploying a radix merkle tree structure [184]. Each participating node is equipped with a transaction processor and is allocated a discrete namespace for implementing its proprietary business logic. Within a sawtooth ecosystem, a transaction processor functions analogously to smart contracts within the ethereum platform. The serialization schema offers considerable adaptability and can leverage decentralized storage mechanisms, both on-chain and off-chain, to preserve batches of committed transactions chronologically. Each data batch comprises multiple transactions, a timestamp, the index hash of the previous batch, and a merkle root to authenticate the integrity of the batch data. The aggregation of transactions into a single data batch occurs at arbitrary intervals. Upon the amalgamation of several data batches, a data block is constructed, with the timing parameters being determined by the business logic. Each participating entity is free to select any decentralized storage system, such as Firebase [185], filecoin [186], or the Interplanetary File System (IPFS), based on their specific business requirements.

Every stakeholder has the option to integrate their individual off-chain storage solution. Sawtooth's utilization of off-chain storage systems offers distinct benefits and provides cost-effective scalability and efficient data processing for less sensitive information. This bifurcation allows for the secure submission and validation of requests, maintaining the confidentiality of transactional data while catering to the specific needs of various stakeholders. Moreover, the approach aids in adhering to sawtooth's business rules functionality, ensuring a robust, scalable and flexible solution.

6.1.3 Execution Model

The highly flexible structure of hyperledger sawtooth supports the running and compilation of code as docker images. Docker images are simply chain codes which can interact hyperledger's back-end via pre-defined interfaces. It supports parallel executions of chain codes and transactions in each node. The transaction processor must verify each state and transaction before adding it to the data batch. Moreover, each participating node can connect to other miners, validators, and light nodes via some RPC-like mechanisms. Third-party interfaces and applications are built on top of

the chain and can connect the application domain to the core-level module and vice versa. The REST API allows the client to communicate with the core system module "Validator" using HTTP/JSON standards. This pragmatic RESTful API provides a simple, language-neutral interface to submit transactions and read/write requests. It works as a lightweight layer on top of sawtooth's internal ZMQ communication, so it doesn't require any authentication and passes the message requests to the validator for signature verification. It uses the validator component as a black box, which can only send requests and get the required result without knowing the internal system logic.

6.2 Proposed System

In this section, we describe the extended proposed system, *EdgeBot*, in detail by highlighting its system flow, and deployment to enable a secure and reliable way to share and track real-time healthcare data by leveraging hyperledger sawtooth. We extended our proposed model to use with trusted groups like hospitals, healthcare services, regulatory authorities, and governmental bodies. Furthermore, it provides a flexible model to connect with other underlying technologies and provide on-chain governance to the participating stakeholders. It provides central authoritative controls and offers better scalability, transaction efficiency, immutability, interoperability, highly modular, fine-grained traceability, and run-time upgradation of the consensus protocol. It provides on-chain governance by using dynamic consensus protocols and provides permissioning features. Each stakeholder possesses its transaction processor and can run its business logic via chain codes, side-chain decentralized storage, and consensus protocol. Our proposed framework uses PoET consensus protocol which elects the individual nodes to execute requests at a prescribed target rate. It is similar to proof of work but replaces huge computation with a cheap random wait.

To create a trusted network, we use a P2P network of manager nodes (validator nodes in terms of sawtooth), which provides journal block management and identity management system services. It manages *User_ID* and authenticates all participating nodes (modules) by issuing *registration_certificate*. The high-level architectural diagram of the extended proposed framework *EdgeBot* is depicted in Figure 38 highlighting all the major system modules involved. Each node represents one complete module (stakeholder) of a system. The number of compulsory and optional components involved in the constitution of each node. Compulsory items are P2P sawtooth environment, transaction processor, consensus engine, validator, and REST API services to interact with the system. Optional items include side chains, storage systems, client applications and their inter-communication protocols. A highly modular structure consists of stakeholders, a sawtooth core system module, an application domain, a distributed storage system, and clients. The REST API develops commu-

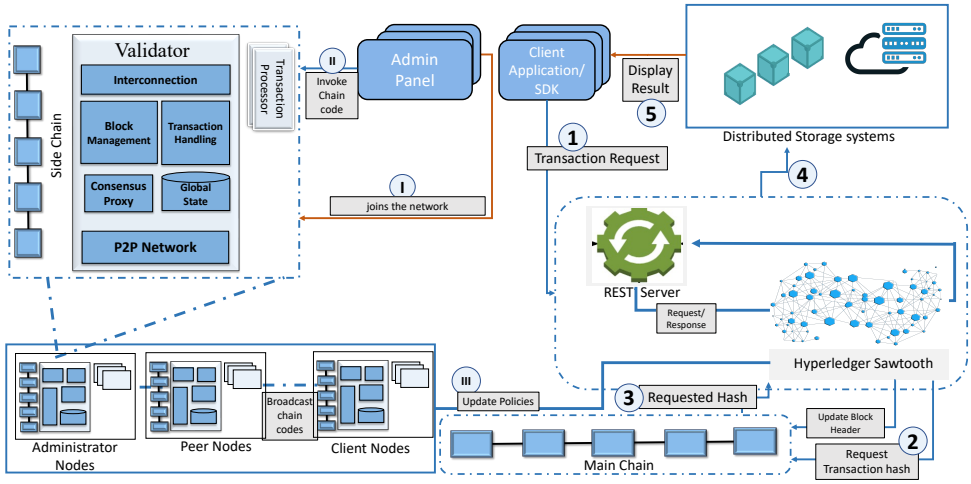


Figure 38: System flow diagram of Hyperledger sawtooth

nication between core system modules and application domains. Each stakeholder node represents some node category:

- Validator node: Each stakeholder is represented as a validator node. It is responsible for authorizing transaction requests related to it after getting approval from the transaction processor. It doesn't need to participate in other network transactions (other stakeholders' transactions).

- Leader node: The leader node is responsible for committing a batch of authorized transactions after random time interval *t*.

- Client node: Clients can submit read requests. They can track all sorts of information from the origin to the point.

Chain codes handle and deploy regulations, business logic, and transactions. These are the central and essential components for handling P2P networks and managing the complete system flow. Moreover, accessibility roles are different for each stakeholder, defined and executed via chain codes.

6.2.1 Proposed service Optimization in Sawtooth

In this section, we introduce our research problem in the context of service optimization. *EdgeBot*, represents a data exchange system which comprises α^x individual participating nodes. The number of parallel transactions generated by each participating node will be blocked together in databatches (db), and the size of databatch is denoted as $S_{txn}\,(db)$.

Each transaction within a blockchain databatch encompasses multiple attributes that collectively determine its size and can be expressed as follows:

$$S_{\text{txn}}\ (db) = \gamma_{\text{flag}}\ + \Sigma_{t=t_{\text{initial}}}^{t_{\text{elapsed}}}\ (cnt_{\text{in}}\ + cnt_{\text{out}}\ )$$

$$+t_{\text{elapsed}}\ + \alpha$$

where α is network id of registered entity node, γ_{flag} is flags data, t_{initial} is initial time, cnt $_{\text{in}}$ is initial count of input values, cnt $_{\text{out}}$ is output count list, and t_{elapsed} is total elapsed time. If the number of transactions in a databatch db are $n_{txn}\,(db)$, then the total size of transaction data in a databatch can be given as:

$$S_{txn}\,(db) = n_{txn}\,(db) \cdot S_{txn}\,(db)$$

Each databatch, in addition to the transaction data, also carries some metadata. This metadata includes elements like a databatch header, the network id of the registered entity node, a random wait time, elapsed time, merkle tree, and a time stamp. As such, the size of a databatch can be determined based on these factors.

$$S_{\text{hdr}}\ (db) = h\,(previousblock) + h\,(\alpha) + h\,(merkletree)$$

$$+h\,(totalelapsedtime) + h\,(randomwaittime) + h\,(timestamp)$$

In this context, h symbolizes the hash function employed in the hyperledger sawtooth framework. The term 'previous databatch' pertains to the hash value of the databatch preceding the current one in the network. 'Merkle tree' signifies the hash value encapsulating all the transactions in the databatch. The 'target formula' is applied to compute the target value required for mining a fresh databatch. 'Nonce' is an acronym for 'Number Only Used Once', while 'timestamp' marks the precise time the databatch was generated. Consequently, the comprehensive size of the metadata for databatch db can be defined as:

$$S_{md}\,(db) = S_{hdr}\,(db) + S_{ctr}\,(db)\,,$$

where $S_{cnt}\,(db)$ signifies the size of transaction counts. Every databatch in a network encompasses both transaction data and its corresponding metadata. Hence, by utilizing the previous four equations, the cumulative size of the databatch db for the network β_i can be expressed as:

$$S\,(db, \beta_i) = S_{txn}\,(db) + S_{md}\,(db)\,,$$

As previously explained, a databatch is composed of numerous transactions. The duration required for a network to achieve consensus on a specific databatch is contingent on the complexity of the consensus along with network latencies. Where a

databatch db is published, it undergoes a propagation period within the β_i. Consequently, the propagation delay for transmitting a databatch db on network β_i can be characterized by:

$$t_p\left(db, \beta_{\mathrm{i}}\right) = t_c\left(db, \beta_{\mathrm{i}}\right) + t_{pr}\left(db, \beta_{\mathrm{i}}\right) + t_q\left(db, \beta_{\mathrm{i}}\right)$$

Where, $t_c\left(db, \beta_i\right) =$ network delay, $t_{pr}\left(db, \beta_i\right) =$ execution delay, and $t_q\left(db, \beta_i\right) =$ wait delay. In addition to the turnaround delay, the total elapsed time taken for a databatch to complete includes propagation delay time, contributing to the overall delay before confirmation. This turnaround delay, $t_s\left(db, \beta_{\mathrm{i}}\right)$, can be represented by:

$$t_s\left(db, \beta_{\mathrm{i}}\right) = t_{cs}\left(db, \beta_{\mathrm{i}}\right) + t_{ot}\left(db, \beta_{\mathrm{i}}\right).$$

Where $t_{cs}\left(db, \beta_i\right)$ denotes the encryption plus consensus processing time, and $t_{ot}\left(db, \beta_i\right)$ represents other potential delays, such as synchronization delay or network delay. Consequently, employing the equations mentioned above, the comprehensive network latency $T\left(db, \beta_i\right)$ required for the generation of a data batch can be formulated as follows:

Where, $t_{cs}\left(db, \beta_{\mathrm{i}}\right)$ denotes the encryption plus consensus processing time, and $t_{ot}\left(db, B_{\mathrm{i}}\right)$ represents other potential delays, such as synchronization delay or network delay. Consequently, employing the equations mentioned above, the comprehensive network latency $T\left(db, \beta_i\right)$, for a databatch to be generated can be given by:

$$T\left(db, \beta_{\mathrm{i}}\right) = t_p\left(db, \beta_{\mathrm{i}}\right) + t_s\left(db, \beta_{\mathrm{i}}\right).$$

Consequently, to attain optimal performance within a network, the mathematical representation of the optimization problem can be expressed as follows:

$$\min T\left(db, \beta_i\right) \,\&\, \max S\left(db, \beta_i\right)$$

Thus, this optimization technique provides a mathematical foundation for determining the most efficient combination of databatch size, transactions per second (tps), resource utilisation and network resources while ensuring the reliability of transaction commitment.

6.2.2 Core System module

The core system module consists of a validator node, which includes a single validator, REST API service, consensus engine, state database, P2P network, and one or more transaction processors. The transaction handling module is inside the validator node, which handles messages, transaction batches and blocks, validation, and publishing after getting approval from the appointed transaction processor.

One data batch consists of a random number of transactions, and one datablock consists of databatches created at a specified time T. Multiple transactions work as an

atomic unit in a batch, hence, the validator rejects the complete transaction batch by rolling it back to the previous state, even if a single transaction failed to complete the validation process. Transaction batches are wrapped in batch lists and allows parallel transaction execution and validation. Furthermore, the validator is also responsible for P2P communication with other validator nodes (stakeholders) on the network. Moreover, REST API services are used to communicate with transaction processors and clients.

6.2.3 Application domain

The client is responsible for creating and signing transactions, combining them into batches and blocks, and submitting them to the validator through the REST-API module. The client application uses node.js based application, which provides a web user interface for the user to interact with. Furthermore, transaction families are responsible to (i) write business logic and policies, (ii) separate transaction rules and content of core functionalities, and (iii) updating state of a system using chain codes. The transaction processor and client use the same data model, serialization, and addressing scheme. Setting transaction families enables participants to agree on prescribed network policies. While transaction families authorize the transaction requests from validator and client nodes autonomously, a client can creates a transaction and submit it to the validator node for authorization. The validator then authorizes it after getting approval from the transaction processor. Each node is assigned one authorized key, which is used by the validator to authorize each requested transaction. After getting consensus approval, any validator node can act as a leader to approve a transaction batch.

6.2.4 System Flow

To track and trace patient histories and continuously generated data via bio-sensors, our proposed system flow is described in two ways: track forward and backward. Each stakeholder works as a validator node, which defines its working policies independently without leaning on the core system. To join a network, every stakeholder must implement online system requirements and write transaction policies based on their business logic. Furthermore, a stakeholder can start their validator node and associated components after getting approval from the health regulatory bodies (Leader).

However, to achieve traceability, it is essential to have two separate and asynchronous transaction flows. Transaction flow describes temporal traceability and the associated information flow going from one stakeholder to another. Using Track backward flow, participants can access the transaction record at any time, ensuring the process's accuracy, integrity, and transparency. To track backwards, rules and

regulations are defined for each stakeholder, depending on the proportionality of information revealed. Access control transaction rules are implemented and regularize throughout the system through transaction families. Access control methods define the attributes of transparency for each stakeholder in a system. While the transaction families are used to implement access control methods via smart contracts managed by leader nodes. Transaction rules are implemented in a modular approach based on the nature of tasks, such as registration, transactions, tracing, and tracking.

6.3 Implementation and Performance analysis

To evaluate the performance, two separate testnets are deployed and specifications are elaborated in table 21. A Permissionless PoET consensus engine in dev modes is used along with a client application to implement the application layer. Nodejs is used as the client-side execution engine and docker containers for the server side.

Table 21: Hardware and Software Specifications

Component	**Description**
Framework	Hyperledger Sawtooth
Operating System	Ubuntu 18.04.6 LTS x64
Server Specification PC	Core i5-6300U @ 2.5GHz
AWS Server specs	EC2 instance: t2.Large
Client Application	Nodejs Server
Runtime Environment	Docker-compose v.1.29.2

In our testnets, the complete hyperledger sawtooth system is installed and initiated on a linux system as a local host communication. EC2 instances of AWS are used for separate server communications. Each component of sawtooth, including transaction processors, events, chain codes, and nodes are launched as docker containers. Sawtooth is implemented using python embedded in docker images. Each node is only responsible for validating transactions related to it and committing them in batches after validation by the validator. In addition to the REST API, transaction processor, validator, and data from blocks and states, every validator node has several containers.

The application interface for each stakeholder depends on the rights it owns, however, leader nodes can access every module and update the entire system. Each validator node can prescribe its own set of rules and define roles for its sub-system modules and off-chain settings. Interfaces can add transactions, accept inputs, retrieve transactions, and log information. Transaction processors are server-side programs that store operations and process transaction submissions according to business logic. Validators run all the validation processes that happen in a node. They

manage everything from business logic validation by the transaction processor to consensus validation by the consensus manager.

A genesis batch, the first block of the chain, was created by the first validator node. Upcoming data batches append this chain by adding new batches. Any validator node can start the genesis node if it doesn't acquire any extra authority. All authoritative protocols, terms, and conditions can be added later at any time during the running system via chain codes. To determine the business logic for the transparent system, two transaction processors (TPs) are used, namely $stakeholder - registration - TP$ and $transaction - request - TP$, and the PoET development mode consensus is used. Successful transaction requests are combined into data batches as shown in Figure 39. Multiple transaction batches are combined to make one block.

{
- data: [
-{
- header: {
signer_public_key: "0382hed971ws109h87h0bh491dgfe0578fd08ews97c489s34k841deaj1gr480ae",
-transaction_ids: [
"83hf0735ow74926cbsj736scg3920al109deh261dbw0ok32bc103cjsoebb863mdn284bhs39fhe0osm283clenwoek30dhr57sbza0917sx0olqwn17
]
},
header_signature:"4oeh3764bdkw483gmaq0183hfeoncow901k183uqw3bem65041e5abcfad29306462c1f8d2bc2bb980b0ab1ae5bc04d39457e95abfb6
09d4d592f97f6ee538b1".
trace: false,
- transaction: [
- {
- header: {
batcher_public_key: "0382hed971ws109h87h0bh491dgfe0578fd08ews97c489s34k841deaj1gr480ae",
dependencies: [ ],
family_name: "transaction_request"
input: [
"6a0cd88919ee9e9305c5fc7ef277bf805195cf8fc583deb428a32f7f3d6883b76ea4b7255f7dd998cef79518d016eb",
nonce: "0x1.73dowu09niq12d478op23+30",
outputs: [
"6a0cd88919ee9e9305c5fc7ef277bf805195cf8fc583deb428a32f7f3d6883b76ea4b7255f7dd998cef79518d016eb"
],
payload_sha512:"2208940ab214e0d4543fa1c087a46015081d067af87eb5be86f359da92a6e608798812582c2606a4853ac6a63ccb6c24a91161
1d3e4b89f78b78b8f50fdb7efe53e",
signer_public_key: "0382hed971ws109h87h0bh491dgfe0578fd08ews97c489s34k841deaj1gr480ae"
},

Figure 39: Data batches created after transaction

6.3.1 Impact of TPS and Blocksize on Latency

Firstly, we evaluated the impact of blocksize by varying the transaction rate (transaction per second, tps) in terms of transaction latency. Transaction rate refers to the number of transactions processed by the sawtooth network within a specific time period T. Latency is defined as the time gap between the transaction submission and completion. The latency L_t is calculated as follows.

$$Lt = \frac{1}{n} \sum_{i=1}^{n} (t_{xi} - t_{yi}) \frac{1}{T_i}$$

Here, t_{xi} and t_{yi} represent the transaction completion and submission times, respectively, of the i^{th} experiment and Ti denotes the total number of transactions submit-

ted for the i^{th} experiment. It measures the rate at which transactions are submitted to and executed by the network. Transaction latency decreased linearly by decreasing the parallel number of transactions, which can be seen clearly in Figure 40.

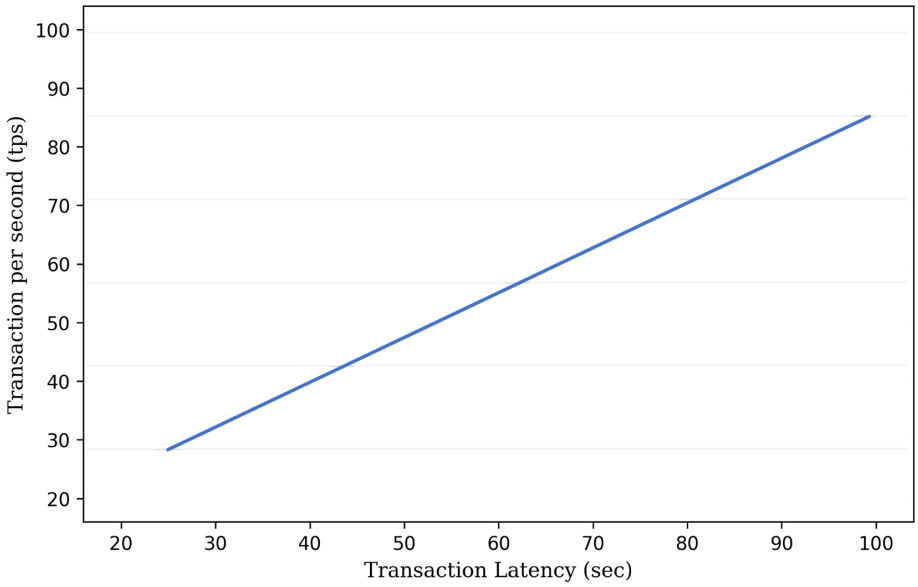


Figure 40: Transaction latency (sec) increases linearly by increasing transactions per second (tps)

With a low transaction rate, it's easier for the network to process and confirm transactions quickly. Latency tends to be lower in such cases, as there is less competition for block space, and transactions can be included in blocks sooner. However, a high transaction rate means many transactions are being submitted simultaneously. This can lead to increased competition for block space, potentially causing delays in transaction confirmation. Higher transaction rates may result in higher latency due to the time it takes to accumulate and validate transactions for block inclusion.

Proper network optimization and scaling can help mitigate the latency impact of varying block sizes and transaction rates. Techniques such as load balancing, parallel processing, and optimized consensus algorithms are helpful to maintain lower latency even as the blockchain network faces increased traffic. The availability of network resources, such as computing power, memory, and bandwidth, plays a role in determining how well a sawtooth network can handle different block sizes and transaction rates. Networks with abundant resources can handle larger blocks and higher transaction rates more efficiently, resulting in lower latency. We need dynamic adjustment of block size and other network parameters. This adaptability can help balance the trade-off between throughput, block size, and latency in response to changing network conditions.

6.3.2 Impact of Transaction rate on Throughput

The transaction rate directly and significantly impacts throughput in sawtooth network. Throughput measures the system's ability to process a certain number of transactions within a specified time period. To validate our proposed model, we create three categories of blocksizes and run parallel transactions per second. Figure 41 depicts that transaction throughput increases rapidly by increasing the transaction rate until it reaches 120-130 tps while using blocksize-20, at which point it stops increasing significantly, and only a slight difference can be seen. These results indicate that bigger blocksizes show better performance throughput until they reach their maximum throughput per block size. As an average, transaction throughput is higher

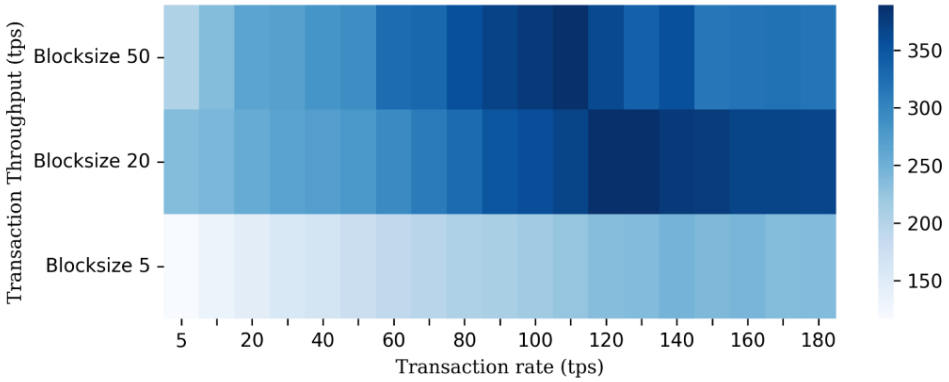


Figure 41: Impact of blocksize and transaction rate on throughput

in blocksize-20 than in 5 and 50, which shows that the more often the consensus algorithm runs, the more often it decreases the overall efficiency of the system, but we need to set the limit dynamically according to the availability of the resources. The impact on throughput also depends on the available network resources, such as bandwidth. If limited, increasing the transaction rate may not significantly increase throughput. The transaction rate increases significantly when we move towards 5 to 100 transactions per second and then increases in a non-significant style. Latency can also be decreased by increasing the number of transactions per block. In the case of blocksize-5, the average transaction rate was 306.89 ms, which increased to 270.47 ms by increasing the blocksize to 20 transactions per block. The transaction latency decreases further by increasing the size to 50 transactions per block, the transaction rate comes to 198.68 ms average. The experimental results depict that transaction throughput increases rapidly by increasing the transaction rate until it reaches 100 tps, at which point it stops increasing significantly and only a slight difference can be seen. These results indicate that bigger blocksize show better performance throughput until they reach their maximum throughput per block size. As an average, transaction throughput is higher in blocksize-50 than in 5 and 20, which

shows that the more often the consensus algorithm runs, the more often it decreases the overall efficiency of the system.

6.3.3 Security and communication Analysis

For on-chain communication, sawtooth's internal ZMQ based communication model doesn't require any authentication and passes the message requests to the validator for signature verification. Validator nodes send and receive data within the network through the use of chain codes independently. Moreover, participating entities can autonomously update their terms and conditions without requiring the involvement of other network members.

To facilitate off-chain communication within the network, we propose the implementation of a secure communication infrastructure based on transport layer security (TLS) for encrypted data sharing. This approach incorporates post-quantum essential encapsulation methods to ensure the security of private keys and employs digital signature algorithms for authentication purposes. Kyber512 is utilized as a key encapsulation method for sharing crypto keys over the network and dilithium3 for digital signatures. Dilithium3 is employed in communication to ensure messages' confidentiality, authenticity, and integrity. Section 5.4.4, shows complete process and implementation of kyber512 and dilithium3. Kyber key encapsulation algorithm (KEM) with the Kyber public-key encryption algorithm is implemented using 32-octet *cpaseed* as input for the key generation process. $cpaPublicKey$, $cpaPrivateKey$, h, and z are variables representing the components needed to derive the private key. The publicKey is derived directly from $cpaPublicKey$, while the privateKey is derived from a combination of $cpaPrivateKey$, $cpaPublicKey$, h, and z. In terms of resource utilization, the performance of post-quantum key encapsulation mechanisms (KEMs) can be compared against elliptic curves implemented in the $OpenSSL$ cryptography library. The results show that, in general, the elliptic curves perform similarly to the post-quantum KEMs, indicating the need for further optimizations to make them more suitable for devices with low resources.

6.3.4 Network Reliability Analysis

More than 1000 parallel statements were initialized and tracked to test the system's reliability; as shown in Figure 42, the system could handle 82 percent of the statements and miss only 179 requests with limited computational resources. These sets of statements include different kinds of commands and events, but missed statements are network-bound only. *Node.js* based client was responsible for initiating and deploying a different number of transactions by using *Int-key* transaction processor. These transactions are then sent to the validator through the REST-API module. P2P nodes, as well as client applications, use RESt-API to communicate between them

```
----------------------------------------------------------------------
Ran 15 tests in 0.769s

OK
Name                                                 Stmts   Miss  Cover
------------------------------------------------------------------------
cli/sawtooth_cli/__init__.py                             1      0   100%
cli/sawtooth_cli/admin_command/__init__.py               1      0   100%
cli/sawtooth_cli/admin_command/config.py                28     12    57%
cli/sawtooth_cli/admin_command/genesis.py               66      4    94%
cli/sawtooth_cli/admin_command/keygen.py                58      7    88%
cli/sawtooth_cli/exceptions.py                           4      0   100%
cli/sawtooth_cli/network_command/__init__.py             1      0   100%
cli/sawtooth_cli/network_command/compare.py            293     57    81%
cli/sawtooth_cli/network_command/fork_graph.py          44      2    95%
cli/sawtooth_cli/network_command/parent_parsers.py      23     18    22%
cli/sawtooth_cli/protobuf/__init__.py                    0      0   100%
cli/sawtooth_cli/protobuf/batch_pb2.py                  25      0   100%
cli/sawtooth_cli/protobuf/genesis_pb2.py                16      0   100%
cli/sawtooth_cli/protobuf/settings_pb2.py               45      0   100%
cli/sawtooth_cli/protobuf/transaction_pb2.py            23      0   100%
cli/sawtooth_cli/rest_client.py                         99     74    25%
cli/tests/test_admin_keygen.py                          56      5    91%
cli/tests/test_config.py                                33      0   100%
cli/tests/test_genesis.py                              112      0   100%
cli/tests/test_network.py                               75      0   100%
------------------------------------------------------------------------
TOTAL                                                 1003    179    82%
```

Figure 42: Total number of successful commits

and google protocol buffers were used to make communication easier, and curl sends data from HTTP to RESt-API. Int-key transaction processor was used to sign each transaction and combine these transactions into batches and a number of batches are combined to publish one block on the network.

CPU and network utilization performance test marks were recorded and analyzed for Amazon AWS instances and personal computer systems. Hardware specifications are described in detail in table 21. To collect the matrices of PC, cAdvisor (Container Advisor) is used, and a running daemon is used to collect the analytics against each running container separately. 11 virtual cores were used to start the network. To track the performance metrics of AWS cloud instances, cloudWatch was used. Furthermore, we aggregate these performance parameters of running containers and their impact on CPU and network consumption in charts. Figure 43 describes the CPU utilization in terms of CPU percentage, which shows sudden spikes overtime during the parallel number of transaction commits.

However, sawtooth is mainly a network-bound protocol, figure 44a and 44b shows the network packet count used during transaction requests and event handling using AWS EC2 instance. Sudden spikes show the network usage during transactions and event handling is spiked upto 15%. The following analysis demonstrates that the suggested system effectively maintains privacy and scalability, without introducing any disturbances in the areas of traceability and transaction confirmation. Moreover, the integration of privacy measures does not significantly influence the system's performance.

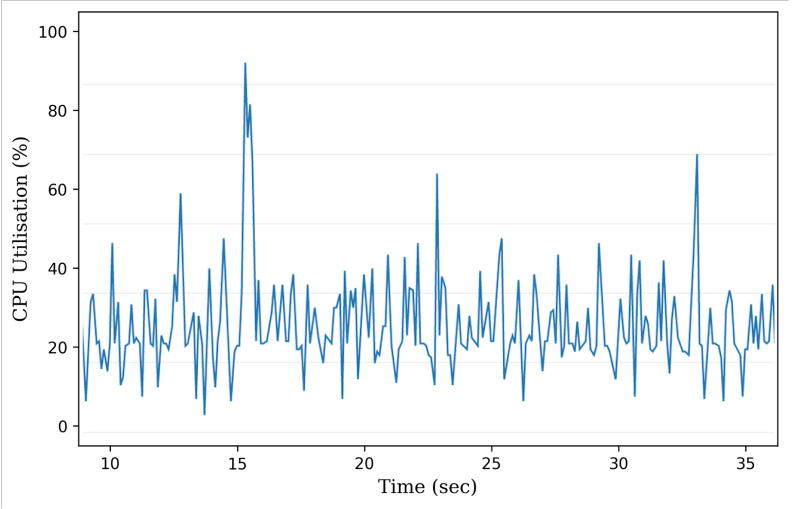


Figure 43: CPU Utilization in terms of CPU percentage

6.4 Summary

The hyperledger sawtooth system was implemented on a linux system as a local host, with AWS EC2 instances used for server communications. Node.js-based clients initiated transactions via the int-key processor, which were sent to the validator through REST-API using google protocol buffers. System performance was tested on AWS and local PCs, with cAdvisor used to collect metrics from docker containers and CloudWatch for AWS instances. Sawtooth components, including transaction processors, chain codes, events, and nodes, were launched in docker containers. CPU and network usage showed spikes during transaction handling, with the system managing 82% of 1,000 parallel statements. Despite spikes in CPU and network utilization, the system maintained performance, with network-bound operations highlighted as a factor affecting performance.

Based on our results discussed in previous sections, we have completed a comparison study of results and analysis in table 22 with recent studies. Our comparison analysis shows the reliability and sustainability of utilizing EdgeBot in real-time ubiquitous healthcare systems. The framework addresses scalability challenges by providing a solid foundation to support large-scale healthcare operations in a modular manner, catering to each participating stakeholder. Additionally, it streamlines the integration with existing systems, even those utilizing different technologies, ensuring a smoother transition to distributed ledger technologies (DLTs).

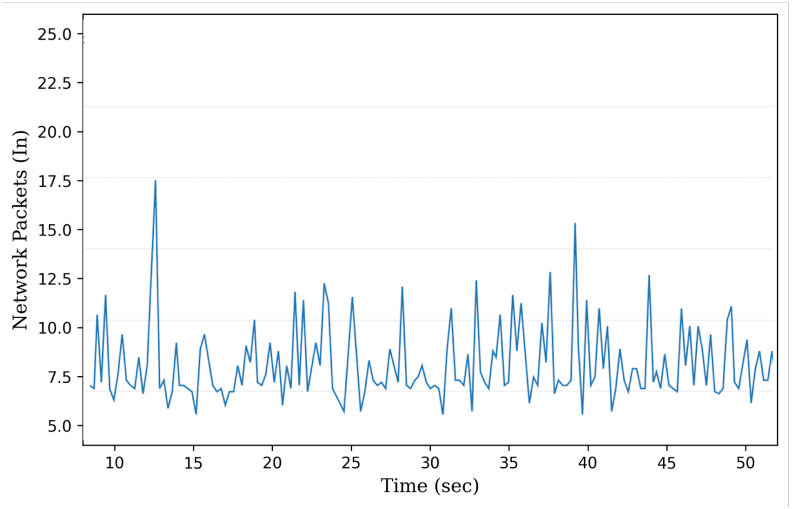
We evaluated the performance of the proposed framework using AWS services by testing key parameters, including the effect of block size on latency with varying transaction rates per second, as well as the combined impact of block size and

Table 22: Comparison of our results and analysis with existing studies

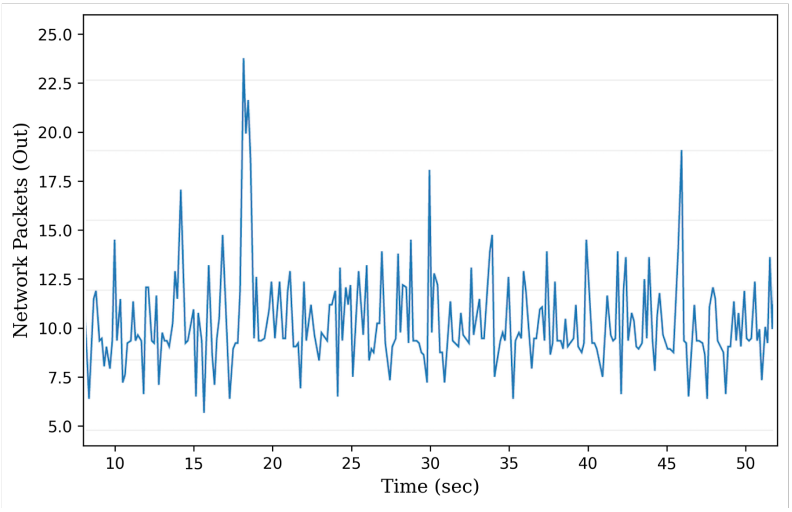
Properties	**This Study**	**[187]**	**[188]**	**[189]**	**[190]**
Impact of blocksize on TPS	**Medium**	Medium	Medium	Medium	Medium
Impact of transaction rate on TPS	**Medium**	High	Medium	Medium	High
Scalability	**High**	Medium	Medium	High	Medium
CPU utilization	**Low**	Low	Low	×	×
Network utilization	**Medium**	Medium	×	×	×
Transaction Latency	**Medium**	Medium	Medium	×	Medium
Network Reliability	**High**	×	×	×	×
On & off chain Security	**High**	×	×	Medium	×

transaction rate on throughput. Additionally, network consumption analysis indicates that bigger block sizes enhance system efficiency by reducing the frequency of consensus algorithm execution, but dynamic limits should be set based on available network resources like bandwidth. Experimental analysis shows that transaction latency increases with higher transactions per second (TPS), and throughput—the system's ability to process transactions also increases but plateaus at 120-130 TPS with blocksize-20. Larger blocksizes improve throughput until they reach their maximum capacity. As blocksize increases, transaction latency decreases, showing better performance with blocksize-50 than smaller blocks. Enabling traceability and tracking all transactions provide a complete picture of each transactional data, including date, time, location, and stakeholders involved in the operation.

Figure 44: (a)Received number of Network Packet count and (b) Outgoing Network traffic Packet count



(a) Received number of Network Packet count



(b) Outgoing Network traffic Packet count

7 Conclusion and Future Perspective

This study proposed optimized privacy protected techniques leveraging consortium and private blockchain framework as a service for ubiquitous healthcare systems in time-sensitive environments. Our main objective was to address fine-grained access control and privacy concerns related to data transmission, sharing, and trading in edge computing environments among peers by proposing resource-optimized ML algorithms service optimization, and sharding techniques. *EdgeBot* guarantees the availability of information, respects ownership rights, and protects data trading rights using fine-grained access control scheme.

First, in Chapter 4, we designed a continuous physiological signal monitoring system incorporating edge-level intelligence and ethereum to ensure privacy and security of processed information. We proposed resource-optimized 1D-CNN for multiclass arrhythmia detection, moreover, bayesian optimization algorithm was embedded within the network architecture to adaptively select the optimal combination of hyperparameters. Experiments were conducted under two main testbed configurations and a variety of bio-sensors, including a heart rate sensor, temperature, humidity sensor, and an ECG monitoring system. The first configuration collected 400 heart rate samples by utilizing photoplethysmography sensor MAX30100 in conjunction with the ESP8266-12E WiFi module. We employed 2-channel ECG system using AD8232 along with STM32F427 (32 bit microcontrollers) and raspberry pi 3 as edge gateways. It also includes digital-to-analog converters (DAC) MCP4921, general-purpose operational amplifiers (OPAMP) LF353, and a Wye resistor network. During ECG measurements, some subjects were positioned supine and instructed to relax their muscles to minimize muscular artifacts and assess pure ECG quality. High-pass and low-pass filters with cutoff frequencies of 0.1 Hz and 200 Hz were implemented and after digitization at an 800 Hz sampling frequency, notch filters at multiples of 50 Hz were used to eliminate power line interference. To ascertain the precision and credibility of AD8232 based ECG system, a comparison of HR and RR interval measurements was obtained from polysomnography (PSG) device.

Initially, for quick abnormality detection, the support vector machine (SVM) technique and naive bayes algorithm are implemented and experimental results show that SVM outperforms naive bayes in terms of accuracy for real-time scenarios. The presented approach effectively categorizes incoming signals as either normal or abnormal. Furthermore, we deployed our proposed resource optimized 1D-CNN and

base LSTMs for identifying multi-class arrhythmias and the results demonstrate that our proposed light weight 1D-CNN shows better results while comparing to base LSTMs and other related studies. The correlation results of 1D-CNN highlight the high accuracy of the 1D-CNN, with the majority of predictions correctly positioned along the diagonal, reflecting robust performance and minimal misclassifications. A comparative analysis of our proposed 1D-CNN with similar studies demonstrates an average accuracy of 97.4%, while utilizing significantly fewer resources than other studies. Average ECG processing time along with data sharing on ethereum requires only 150ms and 143ms respectively.

Using ethereum and artificial intelligence at the network's edge (edge gateways), users' privacy is improved significantly. With edge intelligence, firstly, it enables processing of raw data at the local network level, hence, mitigating the risk of raw data leakage. As a result of its application to manage access control to processed data and features, the blockchain allows end-users to have complete control over their data while allowing third parties access only to data that has already been processed. As raw data is never stored, nor is it transferred outside of the local network when analyzed directly at the edge layer, we inherently avoid breaches of sensitive information.

Second, in Chapter 5, we proposed trustless P2P data trade protocol based on edge gateways (raspberry pi 3, model B+), and STM32F427 (M3,M4 and M7) as lightweight nodes, leveraging ethereum as a service layer. Employing ECDSA for communication and data security is fortified through ECIES using X CUBE CRYPTOLIB library, generating unique keys via a child key derivation function (CKD) for every data batch. Results show an average of 17.253s, 1.462s and 1.156s execution time of M3, M4, and M7 respectively, while average power consumption by M3 and M4 was $\pm$ 200mW, whereas M7 uses an average of $\pm$ 290mW. Results indicate the superior performance of M4 cortex microcontrollers are best fit while consuming fewer resources.

Resource consumption analysis during P2P data trade shows that average computing resources utilized remained notably below 40 percent. RAM usage is approximately 26% during transaction, and 45% of CPU is utilized. The time needed to complete a transaction is divided into three stages: retrieving metadata (TRD), verifying the transaction (VTR), and confirming the transaction (TCT), it is remarkable that TRD requires only 34.6ms , VTR 36ms, and TCT 73.6ms on average. FobSim simulator was utilized to check the scalability of the proposed model. Results show that gossip protocol decreases 35% latency during parallel transactions ranging from 5 to 500. *EdgeBot* Post-quantum resistant extension was implemented and results were compared with other lightweight KEMs. Latency and memory efficiency of Kyber512 KEM with other lightweight KEMs were compared on STM32F427 microcontroller. Kyber512 required 22-25 bytes of memory, moreover, Kyber512 was found to be the best performer, balancing energy consumption and memory usage.

Results of performance analysis show *EdgeBot*, a viable option for fortifying data trade, data ownership and exchange through edge gateways.

Third, in Chapter 6, real-time data sharing and tracing among trusted stakeholders (hospitals, healthcare service providers and regulatory authorities) is proposed while preserving transparency using hyperledger sawtooth. Sawtooth system is set up on a linux platform with local host communication, while Amazon Web Services (AWS) EC2 instances handle server communication. Each component of Sawtooth, such as transaction processors, chain codes, and nodes, runs as docker containers. Implemented in Python, the system validates and commits transactions in batches. Node.js clients are responsible for initiating transactions, which are processed by the Int-key transaction processor and communicated via REST-API. The system's performance is tested on both AWS and local machines, using tools like cAdvisor for tracking container analytics and cloudWatch for monitoring AWS performance metrics. The results show spikes in CPU and network usage during heavy transaction loads. Despite these spikes, the system maintains data privacy without affecting traceability or performance significantly.

We demonstrated the performance of the proposed framework using AWS services and by testing various significant parameters, which include the impact of blocksize on latency by varying the transaction rate per second and the impact of blocksize and transaction rate on throughput. Experimental analysis shows that transaction latency increases with a higher transaction rate (TPS), affecting throughput, which measures how many transactions are processed in a certain period. Experimental results reveal that larger block sizes generally improve throughput, although network resource limitations like bandwidth can cap performance. The throughput improvement reaches a limit after approximately 120-130 TPS with blocksize-20. The study also finds that increasing transactions per block reduces latency, with a notable improvement when using blocksize-50 over smaller blocksizes. Experimental results show that transaction latency decreases linearly by increasing the blocksize and transaction rate (tps). Network bandwidth and CPU consumption metrics show that there is not much change during transaction handling or in idle conditions. However, sudden spikes in network packet count have been observed during transaction processing, and idle state systems do not require significant bandwidth, making sawtooth network protocol network-bound rather than CPU-bound. The experimental results and analysis with related studies demonstrated that our proposed model offers reliable and real-time traceability capabilities while ensuring scalability in large-scale ubiquitous healthcare systems.

To summarize, we successfully achieved the principal objective of this thesis to research, develop, and evaluate the development of such an access-control, privacy-preserved framework in various ubiquitous health aspects. However, direct implementation of a ubiquitous healthcare system is not always feasible. The inter-variability of biosignals due to individual differences poses a significant challenge

for accurately detecting each individual. Pre-trained deep learning models may experience substantial performance degradation in certain patients. Moreover, personalizing models in deep learning still demands significant manual effort and domain expertise to label a sufficient number of patient-specific data samples, which is impractical in some scenarios due to the high labor costs and specialized knowledge required. Additionally, access to personal health data is highly restricted in health monitoring applications, making it difficult to obtain the necessary data for model personalization, thereby complicating the process further.

List of References

[1] Vaidik Bhatt and Samyadip Chakraborty. Improving service engagement in healthcare through internet of things based healthcare systems. *Journal of Science and Technology Policy Management*, 14(1):53–73, 2023.

[2] Sahshanu Razdan and Sachin Sharma. Internet of medical things (iomt): Overview, emerging technologies, and case studies. *IETE technical review*, 39(4):775–788, 2022.

[3] Sai Srinivas Vellela, B Venkateswara Reddy, Kancharla K Chaitanya, and M Venkateswara Rao. An integrated approach to improve e-healthcare system using dynamic cloud computing platform. In *2023 5th International Conference on Smart Systems and Inventive Technology (ICSSIT)*, pages 776–782. IEEE, 2023.

[4] Elina Mattila, Ilkka Korhonen, Jukka H Salminen, Aino Ahtinen, Esa Koskinen, Antti Särelä, Juha Pärkkä, and Raimo Lappalainen. Empowering citizens for well-being and chronic disease management with wellness diary. *IEEE Transactions on Information Technology in Biomedicine*, 14(2):456–463, 2009.

[5] Mohammad Nuruzzaman Bhuiyan and Billah. Design and implementation of a feasible model for the iot based ubiquitous healthcare monitoring system for rural and urban areas. *IEEE Access*, 10:91984–91997, 2022.

[6] Zinab Abuwarda and Mostafa. Wearable devices: Cross benefits from healthcare to construction. *Automation in Construction*, 142:104501, 2022.

[7] Geshwaree Huzooree, Kavi Kumar Khedo, and Noorjehan Joonas. Pervasive mobile healthcare systems for chronic disease monitoring. *Health informatics journal*, 25(2):267–291, 2019.

[8] Giovanni Chiarini, Pradeep Ray, Shahriar Akter, Cristina Masella, and Aura Ganz. mhealth technologies for chronic diseases and elders: a systematic review. *IEEE Journal on Selected Areas in Communications*, 31(9):6–18, 2013.

[9] Healthcare data breach statistics. `https://www.hipaajournal.com/healthcare-data-breach-statistics/#:~:text=Between%202009%20and%202023%2C%205%2C887,disclosure%20of%20519%2C935%2C970%20healthcare%20records.`, . (Accessed on 08/23/2024).

[10] Healthcare data breach statistics. `https://www.hipaajournal.com/healthcare-data-breach-statistics/`, . (Accessed on 08/19/2023).

[11] U.s. department of health and human services office for civil rights breach portal: Notice to the secretary of hhs breach of unsecured protected health information. URL `https://ocrportal.hhs.gov/ocr/breach/breach_report.jsf`.

[12] Wei Huang, Afshar Ganjali, Beom Heyn Kim, Sukwon Oh, and David Lie. The state of public infrastructure-as-a-service cloud security. *ACM Computing Surveys (CSUR)*, 47(4):1–31, 2015.

[13] WeiTek Tsai, XiaoYing Bai, and Yu Huang. Software-as-a-service (saas): perspectives and challenges. *Science China Information Sciences*, 57:1–15, 2014.

[14] Thiyagarajan Devi and Ramachandrarao Ganesan. Platform-as-a-service (paas): model and security issues. *TELKOMNIKA Indonesian Journal of Electrical Engineering*, 15(1):151–161, 2015.

[15] Cloud computing services: Aws. URL `https://aws.amazon.com/`.

[16] Microsoft azure: Cloud computing services. URL `https://azure.microsoft.com/en-us/`.

[17] Ibm cloud computing. URL `https://www.ibm.com/cloud`.

[18] Ilhan Firat Kilincer, Fatih Ertam, Abdulkadir Sengur, Ru-San Tan, and U Rajendra Acharya. Automated detection of cybersecurity attacks in healthcare systems with recursive feature elimination and multilayer perceptron optimization. *Biocybernetics and Biomedical Engineering*, 43 (1):30–41, 2023.

[19] IK Poyner and RS Sherratt. Privacy and security of consumer iot devices for the pervasive monitoring of vulnerable people. In *Living in the Internet of Things: Cybersecurity of the IoT-2018*, pages 1–5. IET, 2018.

[20] Bader Alouffi, Muhammad Hasnain, Abdullah Alharbi, Wael Alosaimi, Hashem Alyami, and Muhammad Ayaz. A systematic literature review on cloud computing security: threats and mitigation strategies. *IEEE Access*, 9:57792–57807, 2021.

[21] Benfano Soewito, Ford Lumban Gaol, Edi Abdurachman, et al. A systematic literature review: Risk analysis in cloud migration. *Journal of King Saud University-Computer and Information Sciences*, 2021.

[22] Manoj Kumar Sasubilli and R Venkateswarlu. Cloud computing security challenges, threats and vulnerabilities. In *2021 6th International Conference on Inventive Computation Technologies (ICICT)*, pages 476–480. IEEE, 2021.

[23] A. Al-Fuqaha *et al.* Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE Communications Surveys Tutorials*, 17(4):2347–2376, Fourthquarter 2015. ISSN 1553-877X. doi: 10.1109/COMST.2015.2444095.

[24] R. Roman, J. Lopez, and M. Mambo. Mobile edge computing, fog et al.: A survey and analysis of security threats and challenges. *Future Generation Computer Systems*, 78:680–698, 2018.

[25] T. N. Gia, and M. Jiang. Exploiting fog computing in health monitoring. *Fog and Edge Computing: Principles and Paradigms*, pages 291–318, 2019.

[26] J. Peña Queralta *et al.* Edge-AI in LoRabased healthcare monitoring: A case study on fall detection system with LSTM Recurrent Neural Networks. In *2019 42nd International Conference on Telecommunications, Signal Processing (TSP)*, 2019.

[27] Hai Ziwei, Zhang Dongni, Zhang Man, Du Yixin, Zheng Shuanghui, Yang Chao, and Cai Chunfeng. The applications of edge computing and the internet of things in the healthcare and medical sectors: A bibliometric analysis. *Heliyon*, 2024.

[28] Ahmad Alzu'bi, Ala'a Alomar, Shahed Alkhaza'leh, Abdelrahman Abuarqoub, and Mohammad Hammoudeh. A review of privacy and security of edge computing in smart healthcare systems: issues, challenges, and research directions. *Tsinghua Science and Technology*, 29(4):1152–1180, 2024.

[29] Buterin, Vitalik and others. A next-generation smart contract and decentralized application platform. *white paper*, 2014.

[30] Patrick McEnroe, Shen Wang, and Madhusanka Liyanage. A survey on the convergence of edge computing and ai for uavs: Opportunities and challenges. *IEEE Internet of Things Journal*, 9 (17):15435–15459, 2022.

[31] Chuan Feng, Pengchao Han, Xu Zhang, Bowen Yang, Yejun Liu, and Lei Guo. Computation offloading in mobile edge computing networks: A survey. *Journal of Network and Computer Applications*, 202:103366, 2022.

[32] Quy Nguyen Minh, Van-Hau Nguyen, Vu Khanh Quy, Le Anh Ngoc, Abdellah Chehri, and Gwanggil Jeon. Edge computing for iot-enabled smart grid: The future of energy. *Energies*, 15 (17):6140, 2022.

[33] Jennifer S Raj and S Jennifer. Optimized mobile edge computing framework for iot based medical sensor network nodes. *Journal of Ubiquitous Computing and Communication Technologies (UCCT)*, 3(01):33–42, 2021.

[34] Al Hamid, Hadeal Abdulaziz and Rahman, Sk Md Mizanur and Hossain, M Shamim and Almogren, Ahmad and Alamri, Atif. A security model for preserving the privacy of medical big data in a healthcare cloud using a fog computing facility with pairing-based cryptography. *IEEE Access*, 5:22313–22328, 2017. ISSN 2169-3536. doi: 10.1109/ACCESS.2017.2757844.

[35] Morghan Hartmann, Umair Sajid Hashmi, and Ali Imran. Edge computing in smart health care systems: Review, challenges, and research directions. *Transactions on Emerging Telecommunications Technologies*, 33(3):e3710, 2022.

[36] A. Nawaz *et al.* Edge AI and Blockchain for Privacy-Critical and Data-Sensitive Applications. In *The 12th International Conference on Mobile Computing and Ubiquitous Networking (ICMU)*, 2019.

[37] W. Shi and S. Dustdar. The promise of edge computing. *Computer*, 49(5):78–81, May 2016. ISSN 0018-9162. doi: 10.1109/MC.2016.145.

[38] Kai Peng, Yu Yang, Shangguang Wang, Peiyun Xiao, and Victor CM Leung. Reliability-aware proactive offloading in mobile edge computing using stackelberg game approach. *IEEE Internet of Things Journal*, 2024.

[39] Xi Lin, Jianhua Li, Jun Wu, Haoran Liang, and Wu Yang. Making knowledge tradable in edge-ai enabled iot: A consortium blockchain-based efficient and incentive approach. *IEEE Transactions on Industrial Informatics*, 15(12):6367–6378, 2019.

[40] Chayakrit Krittanawong, Albert J Rogers, Mehmet Aydar, Edward Choi, Kipp W Johnson, Zhen Wang, and Sanjiv M Narayan. Integrating blockchain technology with artificial intelligence for cardiovascular medicine. *Nature Reviews Cardiology*, 17(1):1–3, 2020.

[41] J. Peña Queralta *et al.* Enhancing autonomy with blockchain and multi-acess edge computing in distributed robotic systems. In *The Fifth International Conference on Fog and Mobile Edge Computing (FMEC)*. IEEE, 2020.

[42] Eshtiak Ahmed, Ashraful Islam, Mohsena Ashraf, Atiqul Islam Chowdhury, and Mohammad Masudur Rahman. Internet of things (iot): Vulnerabilities, security concerns and things to consider. In *2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, pages 1–6. IEEE, 2020.

[43] Debe, Mazin and Salah, Khaled and Rehman, Muhammad Habib Ur and Svetinovic, Davor. Iot public fog nodes reputation system: A decentralized solution using ethereum blockchain. *IEEE Access*, 7:178082–178093, 2019.

[44] Maria Papaioannou, Marina Karageorgou, Georgios Mantas, Victor Sucasas, Ismael Essop, Jonathan Rodriguez, and Dimitrios Lymberopoulos. A survey on security threats and countermeasures in internet of medical things (iomt). *Transactions on Emerging Telecommunications Technologies*, 33(6):e4049, 2022.

[45] Eram Fatima Siddiqui, Tasneem Ahmed, and Sandeep Kumar Nayak. A decision tree approach for enhancing real-time response in exigent healthcare unit using edge computing. *Measurement: Sensors*, 32:100979, 2024.

[46] P. De Filippi, and S. McCarthy. Cloud computing: Centralization and data sovereignty. *European Journal of Law and Technology*, 3(2), 2012.

[47] Aafreen Qureshi, Shivangi Batra, Prashant Vats, Sandeep Singh, Manu Phogat, and Anupam Kumar Sharma. A review of machine learning (ml) in the internet of medical things (iomt) in the construction of a smart healthcare structure. *Journal of Algebraic Statistics*, 13(2):225–231, 2022.

[48] Ejaz Ahmed, Arif Ahmed, Ibrar Yaqoob, Junaid Shuja, Abdullah Gani, Muhammad Imran, and Muhammad Shoaib. Bringing computation closer toward the user network: Is edge computing the solution? *IEEE Communications Magazine*, 55(11):138–144, 2017.

[49] Volkan Gezer, Jumyung Um, and Martin Ruskowski. An introduction to edge computing and a real-time capable server architecture. *Int. J. Adv. Intell. Syst.(IARIA)*, 11(7):105–114, 2018.

[50] Ruby Dwivedi, Divya Mehrotra, and Shaleen Chandra. Potential of internet of medical things (iomt) applications in building a smart healthcare system: A systematic review. *Journal of oral biology and craniofacial research*, 12(2):302–318, 2022.

[51] Anoop Singh, Asha Sharma, Aamir Ahmed, Ashok K Sundramoorthy, Hidemitsu Furukawa, Sandeep Arya, and Ajit Khosla. Recent advances in electrochemical biosensors: Applications, challenges, and future scope. *Biosensors*, 11(9):336, 2021.

[52] Abid Haleem, Mohd Javaid, Ravi Pratap Singh, Rajiv Suman, and Shanay Rab. Biosensors applications in medical field: A brief review. *Sensors International*, 2:100100, 2021.

[53] Joel JPC Rodrigues, Dante Borges De Rezende Segundo, Heres Arantes Junqueira, Murilo Henrique Sabino, Rafael Maciel Prince, Jalal Al-Muhtadi, and Victor Hugo C De Albuquerque. Enabling technologies for the internet of health things. *Ieee Access*, 6:13129–13141, 2018.

[54] P Mohankumar, J Ajayan, T Mohanraj, and R Yasodharan. Recent developments in biosensors for healthcare and biomedical applications: A review. *Measurement*, 167:108293, 2021.

[55] Cristiano André Da Costa, Cristian F Pasluosta, Björn Eskofier, Denise Bandeira Da Silva, and Rodrigo da Rosa Righi. Internet of health things: Toward intelligent vital signs monitoring in hospital wards. *Artificial intelligence in medicine*, 89:61–69, 2018.

[56] Velmurugan Subbiah Parvathy, Sivakumar Pothiraj, and Jenyfal Sampson. Automated internet of medical things (iomt) based healthcare monitoring system. *Cognitive Internet of Medical Things for Smart Healthcare: Services and Applications*, pages 117–128, 2021.

[57] Benedikt Notheisen, Jacob Benjamin Cholewa, and Arun Prasad Shanmugam. Trading real-world assets on blockchain. *Business & Information Systems Engineering*, 59(6):425–440, 2017.

[58] Yang Lu. The blockchain: State-of-the-art and research challenges. *Journal of Industrial Information Integration*, 2019.

[59] Nida Malik and Alkhatib. A comprehensive review of blockchain applications in industrial internet of things and supply chain systems. *Applied Stochastic Models in Business and Industry*, 37(3):391–412, 2021.

[60] Azevedo and Gomes. Supply chain traceability using blockchain. *Operations Management Research*, 16(3):1359–1381, Sep 2023. ISSN 1936-9743. doi: 10.1007/s12063-023-00359-y. URL `https://doi.org/10.1007/s12063-023-00359-y`.

[61] Ayesha Altaf. A survey of blockchain technology: Architecture, applied domains, platforms, and security threats. *Social Science Computer Review*, 41(5):1941–1962, 2023.

[62] Maoning Wang, Meijiao Duan, and Jianming Zhu. Research on the security criteria of hash functions in the blockchain. In *Proceedings of the 2nd ACM Workshop on Blockchains, Cryptocurrencies, and Contracts*, pages 47–55, 2018.

[63] Qin Wang. Exploring web3 from the view of blockchain. *arXiv preprint arXiv:2206.08821*, 2022.

[64] Sara Saberi, Mahtab Kouhizadeh, Joseph Sarkis, and Lejia Shen. Blockchain technology and its relationships to sustainable supply chain management. *International Journal of Production Research*, 57(7):2117–2135, 2019.

[65] A. Nawaz, T. N. Gia, J. Peña Queralta, and T. Westerlund. Edge ai and blockchain for privacy-critical and data-sensitive applications. In *2019 Twelfth International Conference on Mobile Computing and Ubiquitous Network (ICMU)*, pages 1–2, 2019. doi: 10.23919/ICMU48249.2019.9006635.

[66] Anum Nawaz, Jorge Peña Queralta, Jixin Guan, Muhammad Awais, Tuan Nguyen Gia, Ali Kashif Bashir, Haibin Kan, and Tomi Westerlund. Edge computing to secure iot data ownership and trade with the ethereum blockchain. *Sensors*, 20(14):3965, 2020.

[67] Christian Killer, Bruno Rodrigues, and Burkhard Stiller. Security management and visualization in a blockchain-based collaborative defense. In *2019 IEEE international conference on blockchain and cryptocurrency (ICBC)*, pages 108–111. IEEE, 2019.

[68] Arash Kordestani, Pejvak Oghazi, and Rana Mostaghel. Smart contract diffusion in the pharmaceutical blockchain: the battle of counterfeit drugs. *Journal of Business Research*, 158:113646, 2023.

[69] Koppiahraj Karuppiah, Bathrinath Sankaranarayanan, and Syed Mithun Ali. A decision-aid model for evaluating challenges to blockchain adoption in supply chains. *International Journal of Logistics Research and Applications*, 26(3):257–278, 2023.

[70] Adam Back, Matt Corallo, Luke Dashjr, Mark Friedenbach, Gregory Maxwell, Andrew Miller, Andrew Poelstra, Jorge Timón, and Pieter Wuille. Enabling blockchain innovations with

pegged sidechains. *URL: http://www. opensciencereview. com/papers/123/enablingblockchain-innovations-with-pegged-sidechains*, 72, 2014.

[71] Joseph Poon and Vitalik Buterin. Plasma: Scalable autonomous smart contracts. *White paper*, pages 1–47, 2017.

[72] Ittay Eyal, Adem Efe Gencer, Emin Gün Sirer, and Robbert Van Renesse. Bitcoin-ng: A scalable blockchain protocol. In *13th {USENIX} symposium on networked systems design and implementation ({NSDI} 16)*, pages 45–59, 2016.

[73] S. Lee. Explaining directed acylic graph (dag), the real blockchain 3.0, 2018.

[74] David M Mosen, Julie Schmittdiel, Judith Hibbard, David Sobel, Carol Remmers, and Jim Bellows. Is patient activation associated with outcomes of care for adults with chronic conditions? *The Journal of ambulatory care management*, 30(1):21–29, 2007.

[75] Sudeep Tanwar, Karan Parekh, and Richard Evans. Blockchain-based electronic healthcare record system for healthcare 4.0 applications. *Journal of Information Security and Applications*, 50:102407, 2020.

[76] Barbara Guttman and Edward A Roback. *An introduction to computer security: the NIST handbook*, volume 800. Diane Publishing, 1995.

[77] Linda Pawczuk, Rob Massey, and Jonathan Holdowsky. Deloitte's 2019 global blockchain survey. *Deloitte Development LLC*, 2019.

[78] Akoh Atadoga, Oluwafunmi Adijat Elufioye, Toritsemogba Tosanbami Omaghomi, Opeoluwa Akomolafe, Ifeoma Pamela Odilibe, Oluwaseyi Rita Owolabi, et al. Blockchain in healthcare: A comprehensive review of applications and security concerns. *International Journal of Science and Research Archive*, 11(1):1605–1613, 2024.

[79] Olusogo Popoola, Marcos Rodrigues, Jims Marchang, Alex Shenfield, Augustine Ikpehia, and Jumoke Popoola. A critical literature review of security and privacy in smart home healthcare schemes adopting iot & blockchain: problems, challenges and solutions. *Blockchain: Research and Applications*, page 100178, 2023.

[80] Jianbing Ni, Aiqing Zhang, Xiaodong Lin, and Xuemin Sherman Shen. Security, privacy, and fairness in fog-based vehicular crowdsensing. *IEEE Communications Magazine*, 55(6):146–152, 2017.

[81] Yehia I Alzoubi, Valmira H Osmanaj, Ashraf Jaradat, and Ahmad Al-Ahmad. Fog computing security and privacy for the internet of thing applications: State-of-the-art. *Security and Privacy*, 4(2):e145, 2021.

[82] Seyedakbar Mostafavi and Waswa Shafik. Fog computing architectures, privacy and security solutions. *Journal of Communications Technology, Electronics and Computer Science*, 24:1–14, 2019.

[83] Vincent Primault, Antoine Boutet, Sonia Ben Mokhtar, and Lionel Brunie. The long road to computational location privacy: A survey. *IEEE Communications Surveys & Tutorials*, 21(3): 2772–2793, 2018.

[84] Mithun Mukherjee, Rakesh Matam, Lei Shu, Leandros Maglaras, Mohamed Amine Ferrag, Nikumani Choudhury, and Vikas Kumar. Security and privacy in fog computing: Challenges. *IEEE Access*, 5:19293–19304, 2017.

[85] Sabrina Sicari, Alessandra Rizzardi, and Alberto Coen-Porisini. Insights into security and privacy towards fog computing evolution. *Computers & Security*, page 102822, 2022.

[86] Yehia Ibrahim Alzoubi, Ahmad Al-Ahmad, and Hasan Kahtan. Blockchain technology as a fog computing security and privacy solution: An overview. *Computer Communications*, 182: 129–152, 2022.

[87] Binara NB Ekanayake, Malka N Halgamuge, and Ali Syed. Security and privacy issues of fog computing for the internet of things (iot). *Cognitive Computing for Big Data Systems Over IoT: Frameworks, Tools and Applications*, pages 139–174, 2018.

[88] Keke Gai, Yulu Wu, Liehuang Zhu, Lei Xu, and Yan Zhang. Permissioned blockchain and edge computing empowered privacy-preserving smart grid networks. *IEEE Internet of Things Journal*, 6(5):7992–8004, 2019.

[89] Jinyue Song, Tianbo Gu, and Prasant Mohapatra. How blockchain can help enhance the security and privacy in edge computing? In *2021 IEEE/ACM Symposium on Edge Computing (SEC)*, pages 448–453. IEEE, 2021.

[90] Shanhe Yi, Zhengrui Qin, and Qun Li. Security and privacy issues of fog computing: A survey. In *Wireless Algorithms, Systems, and Applications: 10th International Conference, WASA 2015, Qufu, China, August 10-12, 2015, Proceedings 10*, pages 685–695. Springer, 2015.

[91] Mohamed Amine Ferrag, Abdelouahid Derhab, Leandros Maglaras, Mithun Mukherjee, and Helge Janicke. Privacy-preserving schemes for fog-based iot applications: Threat models, solutions, and challenges. In *2018 International Conference on Smart Communications in Network Technologies (SaCoNeT)*, pages 37–42. IEEE, 2018.

[92] Mohammad Aminul Hoque and Ragib Hasan. Towards an analysis of the architecture, security, and privacy issues in vehicular fog computing. In *2019 SoutheastCon*, pages 1–8. IEEE, 2019.

[93] Geetha Kurikala, K Gurnadha Gupta, and A Swapna. Fog computing: Implementation of security and privacy to comprehensive approach for avoiding knowledge thieving attack exploitation decoy technology. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, 2(4):176–181, 2017.

[94] Arwa Alrawais, Abdulrahman Alhothaily, Chunqiang Hu, and Xiuzhen Cheng. Fog computing for the internet of things: Security and privacy issues. *IEEE Internet Computing*, 21(2):34–42, 2017.

[95] Dinh C Nguyen, Ming Ding, Quoc-Viet Pham, Pubudu N Pathirana, Long Bao Le, Aruna Seneviratne, Jun Li, Dusit Niyato, and H Vincent Poor. Federated learning meets blockchain in edge computing: Opportunities and challenges. *IEEE Internet of Things Journal*, 8(16):12806–12825, 2021.

[96] Chuanwen Luo, Liya Xu, Deying Li, and Weili Wu. Edge computing integrated with blockchain technologies. *Complexity and Approximation: In Memory of Ker-I Ko*, pages 268–288, 2020.

[97] Ruizhe Yang, F Richard Yu, Pengbo Si, Zhaoxin Yang, and Yanhua Zhang. Integrated blockchain and edge computing systems: A survey, some research issues and challenges. *IEEE Communications Surveys & Tutorials*, 21(2):1508–1532, 2019.

[98] Khalid A Fakeeh. Privacy and security problems in fog computing. *Commun Appl Electron*, 4 (7), 2016.

[99] Tian Wang, Jiyuan Zhou, Xinlei Chen, Guojun Wang, Anfeng Liu, and Yang Liu. A three-layer privacy preserving cloud storage scheme based on computational intelligence in fog computing. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2(1):3–12, 2018.

[100] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Decentralized business review*, 2008.

[101] Muneeb Ejaz, Tanesh Kumar, Ivana Kovacevic, Mika Ylianttila, and Erkki Harjula. Healthblockedge: Blockchain-edge framework for reliable low-latency digital healthcare applications. *Sensors*, 21(7):2502, 2021.

[102] Joao Amaral Santos, Pedro RM Inacio, and Bruno MC Silva. Towards the use of blockchain in mobile health services and applications. *Journal of Medical Systems*, 45:1–10, 2021.

[103] Daniele Mazzei, Giacomo Baldi, Gualtiero Fantoni, Gabriele Montelisciani, Antonio Pitasi, Laura Ricci, and Lorenzo Rizzello. A blockchain tokenizer for industrial iot trustless applications. *Future Generation Computer Systems*, 105:432–445, 2020.

[104] Cha, Shi-Cho and Chen, Jyun-Fu and Su, Chunhua and Yeh, Kuo-Hui. A blockchain connected gateway for ble-based devices in the internet of things. *IEEE Access*, 6:24639–24649, 2018.

[105] J. Bergquist. Blockchain technology and smart contracts: Privacy-preserving tools, 2017.

[106] Jason Teutsch and Christian Reitwießner. A scalable verification solution for blockchains. In *ASPECTS OF COMPUTATION AND AUTOMATA THEORY WITH APPLICATIONS*, pages 377–424. World Scientific, 2024.

[107] Xiaoxu Ren, Minrui Xu, Dusit Niyato, Jiawen Kang, Chao Qiu, and Xiaofei Wang. Paramart: Parallel resource allocation based on blockchain sharding for edge-cloud services. *IEEE Transactions on Services Computing*, 2024.

[108] Seyyed Jalaladdin Hosseini Dehshiri and Maghsoud Amiri. Evaluation of blockchain implementation solutions in the sustainable supply chain: A novel hybrid decision approach based on z-numbers. *Expert Systems with Applications*, 235:121123, 2024.

[109] Ruyan Wang, Hanyong Liu, Honggang Wang, Qing Yang, and Dapeng Wu. Distributed security architecture based on blockchain for connected health: Architecture, challenges, and approaches. *IEEE Wireless Communications*, 26(6):30–36, 2019.

[110] Manlu Liu, Kean Wu, and Jennifer Jie Xu. How will blockchain technology impact auditing and accounting: Permissionless versus permissioned blockchain. *Current Issues in Auditing*, 13(2): A19–A29, 2019.

[111] Iván Abellán Álvarez, Vincent Gramlich, and Johannes Sedlmeir. Unsealing the secrets of blockchain consensus: A systematic comparison of the formal security of proof-of-work and proof-of-stake. *arXiv preprint arXiv:2401.14527*, 2024.

[112] Christine V Helliar, Louise Crawford, Laura Rocca, Claudio Teodori, and Monica Veneziani. Permissionless and permissioned blockchain diffusion. *International Journal of Information Management*, 54:102136, 2020.

[113] Xiaolong Xu, Dawei Zhu, Xiaoxian Yang, Shuo Wang, Lianyong Qi, and Wanchun Dou. Concurrent practical byzantine fault tolerance for integration of blockchain and supply chain. *ACM Transactions on Internet Technology (TOIT)*, 21(1):1–17, 2021.

[114] Amritraj Singh, Kelly Click, Reza M Parizi, Qi Zhang, Ali Dehghantanha, and Kim-Kwang Raymond Choo. Sidechain technologies in blockchain networks: An examination and state-of-the-art review. *Journal of Network and Computer Applications*, 149:102471, 2020.

[115] Tuan Nguyen Gia, Anum Nawaz, Jorge Peña Querata, Hannu Tenhunen, and Tomi Westerlund. Artificial intelligence at the edge in the blockchain of things. In *International Conference on Wireless Mobile Communication and Healthcare*, pages 267–280. Springer, 2019.

[116] Vitalik Buterin et al. Ethereum white paper. *GitHub repository*, 1:22–23, 2013.

[117] Ian Grigg. Eos-an introduction. *White paper. https://whitepaperdatabase. com/eos-whitepaper*, 2017.

[118] Elli Androulaki, Artem Barger, Vita Bortnikov, Christian Cachin, Konstantinos Christidis, Angelo De Caro, David Enyeart, Christopher Ferris, Gennady Laventman, Yacov Manevich, et al. Hyperledger fabric: a distributed operating system for permissioned blockchains. In *Proceedings of the thirteenth EuroSys conference*, pages 1–15, 2018.

[119] Mariya Benji and M Sindhu. A study on the corda and ripple blockchain platforms. In *Advances in Big Data and Cloud Computing*, pages 179–187. Springer, 2019.

[120] Ahmad Musamih, Khaled Salah, Raja Jayaraman, Junaid Arshad, Mazin Debe, Yousof Al-Hammadi, and Samer Ellahham. A blockchain-based approach for drug traceability in healthcare supply chain. *IEEE access*, 9:9728–9743, 2021.

[121] Sanjeev Kumar Dwivedi, Ruhul Amin, and Satyanarayana Vollala. Blockchain based secured information sharing protocol in supply chain management system with key distribution mechanism. *Journal of Information Security and Applications*, 54:102554, 2020. ISSN 2214-2126. doi: https://doi.org/10.1016/j.jisa.2020.102554. URL `https://www.sciencedirect.com/science/article/pii/S2214212620301484`.

[122] Tarun Kumar Agrawal and Angelis. Demonstration of a blockchain-based framework using smart contracts for supply chain collaboration. *International Journal of Production Research*, pages 1–20, 2022.

[123] Alaa Awad Abdellatif and Al-Marridi. sshealth: Toward secure, blockchain-enabled healthcare systems. *IEEE Network*, 34(4):312–319, 2020. doi: 10.1109/MNET.011.1900553.

[124] Megha Pandey, M Velmurugan, G Sathi, Ahmed Radie Abbas, Norkuzieva Zebo, and T Sathish. Blockchain technology: Applications and challenges in computer science. In *E3S Web of Conferences*, volume 399, page 04035. EDP Sciences, 2023.

[125] Neetu Sharma and Rajesh Rohilla. A novel hyperledger blockchain-enabled decentralized application for drug discovery chain management. *Computers & Industrial Engineering*, 183:109501, 2023.

[126] Mueen Uddin. Blockchain medledger: Hyperledger fabric enabled drug traceability system for counterfeit drugs in pharmaceutical industry. *International Journal of Pharmaceutics*, 597: 120235, 2021.

[127] Vijay and Priya. Grainchain–agricultural supply chain traceability and management technique for farmers sustainability using blockchain hyper ledger. *International Journal of Intelligent Systems and Applications in Engineering*, 10(3):141–146, 2022.

[128] Krittaphas Wisessing and Napaphat Vichaidis. Iot based cold chain logistics with blockchain for food monitoring application. In *2022 7th International Conference on Business and Industrial Research (ICBIR)*, pages 359–363. IEEE, 2022.

[129] Mohit Mohit, Sanmeet Kaur, and Maninder Singh. Design and implementation of transaction privacy by virtue of ownership and traceability in blockchain based supply chain. *Cluster Computing*, 25(3):2223–2240, 2022.

[130] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

[131] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[132] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989.

[133] David H Wolpert. The lack of a priori distinctions between learning algorithms. *Neural computation*, 8(7):1341–1390, 1996.

[134] T Veeramakali, R Siva, B Sivakumar, PC Senthil Mahesh, and N Krishnaraj. An intelligent internet of things-based secure healthcare framework using blockchain technology with an optimal deep learning model. *The Journal of Supercomputing*, 77(9):9576–9596, 2021.

[135] S Neelakandan, J Rene Beulah, L Prathiba, GLN Murthy, E Fantin Irudaya Raj, and N Arulkumar. Blockchain with deep learning-enabled secure healthcare data transmission and diagnostic model. *International Journal of Modeling, Simulation, and Scientific Computing*, 13(04): 2241006, 2022.

[136] Randhir Kumar, Prabhat Kumar, Rakesh Tripathi, Govind P Gupta, AKM Najmul Islam, and Mohammad Shorfuzzaman. Permissioned blockchain and deep learning for secure and efficient data sharing in industrial healthcare systems. *IEEE Transactions on Industrial Informatics*, 18 (11):8065–8073, 2022.

[137] Prabhat Kumar, Randhir Kumar, Govind P Gupta, Rakesh Tripathi, Alireza Jolfaei, and AKM Najmul Islam. A blockchain-orchestrated deep learning approach for secure data transmission in iot-enabled healthcare system. *Journal of Parallel and Distributed Computing*, 172: 69–83, 2023.

[138] Ashish Singh and Kakali Chatterjee. Edge computing based secure health monitoring framework for electronic healthcare system. *Cluster Computing*, 26(2):1205–1220, 2023.

[139] Hemant B Mahajan and Aparna A Junnarkar. Smart healthcare system using integrated and lightweight ecc with private blockchain for multimedia medical data processing. *Multimedia Tools and Applications*, 82(28):44335–44358, 2023.

[140] Amel Ksibi, Halima Mhamdi, Manel Ayadi, Latifah Almuqren, Mohammed S Alqahtani, Mohd Dilshad Ansari, Ashutosh Sharma, and Sakli Hedi. Secure and fast emergency road healthcare service based on blockchain technology for smart cities. *Sustainability*, 15(7):5748, 2023.

[141] Bhaskara S Egala, Ashok K Pradhan, Prasenjit Dey, Venkataramana Badarla, and Saraju P Mohanty. Fortified-chain 2.0: Intelligent blockchain for decentralized smart healthcare system. *IEEE Internet of Things Journal*, 10(14):12308–12321, 2023.

[142] Hanan Naser Alsuqaih, Walaa Hamdan, Haythem Elmessiry, and Hussein Abulkasim. An efficient privacy-preserving control mechanism based on blockchain for e-health applications. *Alexandria Engineering Journal*, 73:159–172, 2023.

[143] Dhairya Jadav, Nilesh Kumar Jadav, Rajesh Gupta, Sudeep Tanwar, Osama Alfarraj, Amr Tolba, Maria Simona Raboaca, and Verdes Marina. A trustworthy healthcare management framework using amalgamation of ai and blockchain network. *Mathematics*, 11(3):637, 2023.

[144] Ibrahim Alrashdi. Fog-based deep learning framework for real-time pandemic screening in smart cities from multi-site tomographies. *BMC Medical Imaging*, 24(1):123, 2024.

[145] Mazin Abed Mohammed, Abdullah Lakhan, Dilovan Asaad Zebari, Mohd Khanapi Abd Ghani, Haydar Abdulameer Marhoon, Karrar Hameed Abdulkareem, Jan Nedoma, and Radek Martinek. Securing healthcare data in industrial cyber-physical systems using combining deep learning and blockchain technology. *Engineering Applications of Artificial Intelligence*, 129:107612, 2024.

[146] Guillermo Valle-Perez, Chico Q Camargo, and Ard A Louis. Deep learning generalizes because the parameter-function map is biased towards simple functions. *arXiv preprint arXiv:1805.08522*, 2018.

[147] Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, 2015.

[148] Viral V Acharya, Lasse H Pedersen, Thomas Philippon, and Matthew Richardson. Measuring systemic risk. *The review of financial studies*, 30(1):2–47, 2017.

[149] Jianxing He, Sally L Baxter, Jie Xu, Jiming Xu, Xingtao Zhou, and Kang Zhang. The practical implementation of artificial intelligence technologies in medicine. *Nature medicine*, 25(1):30–36, 2019.

[150] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip HS Torr, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6881–6890, 2021.

[151] Murat Yıldırım, Gökmen Arslan, and Ahmet Özaslan. Perceived risk and mental health problems among healthcare professionals during covid-19 pandemic: Exploring the mediating effects of resilience and coronavirus fear. *International Journal of Mental Health and Addiction*, pages 1–11, 2020.

[152] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[153] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[154] Sharmin Jahan and Riley. Mape-k/mape-sac: An interaction framework for adaptive systems with security assurance cases. *Future Generation Computer Systems*, 109:197–209, 2020.

[155] Verena Klös, Thomas Göthel, and Sabine Glesner. Adaptive knowledge bases in self-adaptive system design. In *2015 41st Euromicro Conference on Software Engineering and Advanced Applications*, pages 472–478. IEEE, 2015.

[156] V Buterin. Ethereum white paper: a next-generation smart contract and decentralized application platform. ethereum white paper (2014). *Ethereum White Paper*, 2020.

[157] Alessandra Gorla, Mauro Pezze, Jochen Wuttke, Leonardo Mariani, and Fabrizio Pastore. Achieving cost-effective software reliability through self-healing. *Computing and Informatics*, 29(1):93–115, 2010.

[158] Roberto Avanzi and Bos. Crystals-kyber. *NIST, Tech. Rep*, 2017.

[159] Vadim Lyubashevsky and Ducas. Crystals-dilithium. *Algorithm Specifications and Supporting Documentation*, 2020.

[160] Simon Heron. Advanced encryption standard (aes). *Network Security*, 2009(12):8–12, 2009.

[161] Tu Phan. A hierarchical deterministic wallet using ed25519 digital signature scheme. In *Future Data and Security Engineering. Big Data, Security and Privacy, Smart City and Industry 4.0 Applications: 9th International Conference, FDSE 2022, Ho Chi Minh City, Vietnam, November 23–25, 2022, Proceedings*, page 240. Springer Nature, 2022.

[162] Hugo Krawczyk and Pasi Eronen. Hmac-based extract-and-expand key derivation function (hkdf). Technical report, 2010.

[163] Jessica Vensel Rundo and Ralph Downey III. Polysomnography. *Handbook of clinical neurology*, 160:381–392, 2019.

[164] Biswadip Maity, Bryan Donyanavard, Anmol Surhonne, Amir Rahmani, Andreas Herkersdorf, and Nikil Dutt. Seams: Self-optimizing runtime manager for approximate memory hierarchies. *ACM Transactions on Embedded Computing Systems (TECS)*, 20(5):1–26, 2021.

[165] Physhik. Physhik/ecg-mit-bih: Ecg classification using mit-bih data, a deep cnn learning implementation of cardiologist-level arrhythmia detection and classification in ambulatory electrocardiograms using a deep neural network, https://www.nature.com/articles/s41591-018-0268-3 and also deploy the trained model to a web app using flask, introduced at. URL `https://github.com/physhik/ecg-mit-bih`.

[166] scikitlearn machine learning in python scikit learn documentation. `https://scikitlearn.org/stable/`. (Accessed on 08/19/2023).

[167] biosppy pypi. `https://pypi.org/project/biosppy/`. (Accessed on 08/19/2023).

[168] Muhammad Ashfaq Khan and Yangwoo Kim. Cardiac arrhythmia disease classification using lstm deep learning approach. *Computers, Materials & Continua*, 67(1), 2021.

[169] Peng Lu, Yang Gao, Hao Xi, Yabin Zhang, Chao Gao, Bing Zhou, Hongpo Zhang, Liwei Chen, and Xiaobo Mao. Kecnet: a light neural network for arrhythmia classification based on knowledge reinforcement. *Journal of Healthcare Engineering*, 2021(1):6684954, 2021.

[170] Emre Cimen. A transfer learning approach by using 2-d convolutional neural network features to detect unseen arrhythmia classes. *Eskişehir Technical University Journal of Science and Technology A-Applied Sciences and Engineering*, 22(1):1–9, 2021.

[171] Mohammed M Farag. A self-contained stft cnn for ecg classification and arrhythmia detection at the edge. *IEEE Access*, 10:94469–94486, 2022.

[172] Fredy Santander Baños, Norberto Hernández Romero, Juan Carlos Seck Tuoh Mora, Joselito Medina Marín, Irving Barragán Vite, and Gustavo Erick Anaya Fuentes. A novel hybrid model based on convolutional neural network with particle swarm optimization algorithm for classification of cardiac arrhythmias. *IEEE Access*, 11:55515–55532, 2023.

[173] Yong Xia, Yueqi Xiong, and Kuanquan Wang. A transformer model blended with cnn and denoising autoencoder for inter-patient ecg arrhythmia classification. *Biomedical Signal Processing and Control*, 86:105271, 2023.

[174] Md Rabiul Islam, Marwa Qaraqe, Khalid Qaraqe, and Erchin Serpedin. Cat-net: Convolution, attention, and transformer based network for single-lead ecg arrhythmia classification. *Biomedical Signal Processing and Control*, 93:106211, 2024.

[175] Negin Alamatsaz, Leyla Tabatabaei, Mohammadreza Yazdchi, Hamidreza Payan, Nima Alamatsaz, and Fahimeh Nasimi. A lightweight hybrid cnn-lstm explainable model for ecg-based arrhythmia detection. *Biomedical Signal Processing and Control*, 90:105884, 2024.

[176] Subba Reddy Borra, Dasari Ramesh Gari Amrutha Nayana, Sripathi Srinidhi, Surineni Bhavana, Patel Nishitha, and Voriganti Sahithi. Cadnet: cardiac arrhythmia detection and classification using unified principal component analysis and 1d-cnn model. *Research on Biomedical Engineering*, pages 1–13, 2024.

[177] N Prasanna Venkatesh, R Pradeep Kumar, Bala Chakravarthy Neelapu, Kunal Pal, and J Sivaraman. Automated atrial arrhythmia classification using 1d-cnn-bilstm: A deep network ensemble model. *Biomedical Signal Processing and Control*, 97:106703, 2024.

[178] Victor Shoup. A proposal for an iso standard for public key encryption (version 2.1). *IACR e-Print Archive*, 112, 2001.

[179] Andreas M Antonopoulos. *Mastering bitcoin: Programming the open blockchain*. " O'Reilly Media, Inc.", 2017.

[180] Darrel Hankerson, Alfred J Menezes, and Scott Vanstone. *Guide to elliptic curve cryptography*. Springer Science & Business Media, 2006.

[181] Vikram Dhillon, David Metcalf, and Max Hooper. The hyperledger project. In *Blockchain enabled applications*, pages 139–149. Springer, 2017.

[182] Diego Ongaro and John Ousterhout. The raft consensus algorithm. 2015.

[183] Amie Corso. *Performance analysis of proof-of-elapsed-time (poet) consensus in the sawtooth blockchain framework*. PhD thesis, University of Oregon, 2019.

[184] Haitz Sáez de Ocáriz Borde. An overview of trees in blockchain technology: Merkle trees and merkle patricia tries. 2022.

[185] Mr Bishakh Paul. Concept of firebase. *Intelligent Electrical Systems*, page 44, 2023.

[186] Iman Vakilinia, Weihong Wang, and Jiajun Xin. An incentive-compatible mechanism for decentralized storage network. *IEEE Transactions on Network Science and Engineering*, 2023.

[187] Erukala Suresh Babu, BV Ram Naresh Yadav, A Kousar Nikhath, Soumya Ranjan Nayak, and Waleed Alnumay. Mediblocks: secure exchanging of electronic health records (ehrs) using trust-based blockchain network with privacy concerns. *Cluster Computing*, 26(4):2217–2244, 2023.

[188] Khin Su Su Wai and Nwe Nwe Myint Thein. Performance analysis on block size valuation of hyperledger fabric blockchain. In *2023 IEEE Conference on Computer Applications (ICCA)*, pages 50–55. IEEE, 2023.

[189] Mohit, Sanmeet Kaur, and Maninder Singh. Design and implementation of blockchain-based supply chain framework with improved traceability, privacy, and ownership. *Cluster Computing*, 27(3):2345–2363, 2024.

[190] Pratima Sharma, Rajni Jindal, and Malaya Dutta Borah. Blockchain-based distributed application for multimedia system using hyperledger fabric. *Multimedia Tools and Applications*, 83(1): 2473–2499, 2024.