Graduate School MADIS-631, University of Lille, ENSAIT & GEMTEX

School of Computing, University of Kent

Ph.D. THESIS

To obtain the grade of:

Doctor from the University of Lille and the University of Kent

Specializing in

Machine Learning

By

Ali Raza

Secure and Privacy-preserving Federated Learning with Explainable Artificial Intelligence for Smart Healthcare System

Thesis defended on 28 August, 2023 before the jury composed of:

Patrick SIARRY	Prof.	Université Paris-Est Créteil	(Reviewer/President)
Hongmei HE	Prof.	University of Salford	(Reviewer)
Rehmat ULLAH	Asst.Prof.	Cardiff Metropolitan University	(Examiner)
Ramla SADDEM	Assoc.Prof.	Université de Reims Champagne-Ardenne	(Examiner)
Peng LIU	Asst.Prof.	University of Kent	(Examiner)
Kim Phuc TRAN	Assoc. Prof. HDR	Université de Lille	(Co-Supervisor)
Ludovic KOEHL	Prof.	Université de Lille	(Supervisor)
Shujun LI	Prof.	University of Kent	(Supervisor)

Word Count: 49,219

Number of Pages: 219

Ecole doctorale MADIS-631, Université de Lille, ENSAIT & GEMTEX

École d'informatique, Université du Kent

Thèse de Doctorat

pour obtenir le grade de:

Docteur de l'Université de Lille et l'Université de Kent

dans la spécialité

Apprentissage Automatique

par

Ali Raza

Apprentissage Fédéré Sécurisé et Préservant la Confidentialité avec Intelligence Artificielle Explicable pour les systèmès intelligents dans le domaine de la Santé

Thèse soutenue le 28 août 2023 devant le jury composé de :

Patrick SIARRY	Professeur	Université Paris-Est Créteil	(Rapporteur/Président)
Hongmei HE	Professeur	University of Salford	(Rapporteur)
Rehmat ULLAH	Maître de Conférences	Cardiff Metropolitan University	(Examinateur)
Ramla SADDEM	Maître de Conférences	Université de Reims Champagne-Ardenne	(Examinateur)
Peng LIU	Maître de Conférences	University of Kent	(Examinateur)
Kim Phuc TRAN	Maître de Conférences HDR	Université de Lille	(Co-Directeur de Thèse)
Ludovic KOEHL	Professeur	Université de Lille	(Directeur de Thèse)
Shujun LI	Professeur	University of Kent	(Directeur de Thèse)

Nombre de mots: 49,219

Nombre de pages: 219

Abstract

The growing population around the globe has a significant impact on various sectors including the labor force, healthcare, and the global economy. The healthcare sector is among the most affected sectors by the growing population due to the increasing demand for resources including doctors, nurses, equipment, and healthcare facilities. Intelligent systems have been incorporated to enhance decision-making, management, prognosis, and diagnosis in order to tackle such issues and offer improved healthcare to patients. Among such systems, those based on deep learning (DL), a subclass of machine learning (ML) have outperformed many traditional statistical and ML systems owing to their capability of automatically discovering and learning related features for a given task and robustness. Therefore, the use of DL has seen a steady increase in many applications. Nevertheless, usually, the training of DL models relies on a single centralized server, which brings many challenges: (1) except for some big enterprises most of the small enterprises have limited quality data, which is insufficient to support the training of data-hungry DL models, (2) access to data, which is vital for these systems, often raises privacy concerns. The collection and analysis of sensitive patient information must be done in a secure and ethical manner to ensure the protection of individual privacy rights, (3) high communication cost and computation resources required, (4) a large number of trainable parameters make the outcome of DL hard to explain, which is required in some applications, such as healthcare. Compared to centralized ML, federated learning (FL) improves both privacy and communication costs, where clients collaboratively train a joint model without sharing the raw data directly. FL minimizes privacy breaches and safeguards sensitive data by keeping it distributed locally.

This enables collaborative model training while reducing the risk of unauthorized access and data breaches. Additionally, it promotes data diversity and scalability by involving multiple sources in joint model training and decreases communication costs by sharing only model updates instead of the entire dataset. However, FL brings its own challenges. For example, heterogeneous local data among the clients makes it challenging to train a high-performing and robust global model. Sharing updates (hundreds of thousands of parameters) still has high communication costs. Additionally, the distributed nature and access control of local data in FL make it more vulnerable to malicious attacks. Moreover, the challenge of explaining the results of DL still remains challenging, and methods are needed to be developed to bring trust, accountability, and transparency in sensitive applications, such as healthcare.

Therefore, the aim of this thesis is to create robust frameworks that are secure, high-performing, and privacy-friendly within federated settings. These frameworks will be specifically designed for end-to-end (we train our frameworks using raw data without any manual feature extraction) healthcare applications, considering the presence of non-identically distributed data among clients in FL to bring robustness. By addressing these challenges, the objective is to enhance the overall system's resilience and effectiveness. We also propose a methodology for detecting anomalies within federated settings, particularly in applications with limited available data for the abnormal class. Furthermore, clients in FL are usually resource-constrained with limited computation and communication resources available. Therefore, to support efficient computation and communication in a federated setting we propose a lightweight framework (in terms of the trainable number of parameters). Additionally, to provide explanations of the DL models' outcomes, which are usually hard to explain because of the large number of parameters,

we propose model-agnostic explainable AI modules to help explain the results of DL models. Moreover, in order to protect the proposed frameworks against cyber attacks, such as poisoning attacks, we propose a framework in federated settings, which makes the proposed healthcare frameworks more secure and trustworthy. Finally, with experimental analysis using baseline datasets for one of the most common health conditions i.e., cardiovascular diseases (arrhythmia detection, ECG anomaly detection) and human activity recognition (used for supplementing cardiovascular diseases detection), we show the superiority of the proposed frameworks over state-of-the-art work.

Keywords: Federated Learning, Edge Computing, Healthcare, privacy, security, Explainable Artificial Intelligence, Explainable Anomaly Detection, Embedded Artificial Intelligence, Clinical Decision Support Systems, Safety and Reliability of Artificial Intelligence, poisoning attacks, data poisoning, model poisoning, Byzantine attacks.

Abstract

(*French*)

Le croissance de la population dans le monde a un impact significatif sur divers secteurs, notamment la main-d'œuvre, les soins de santé et l'économie mondiale. Le secteur de la santé est l'un des secteurs les plus touchés par le vieillissement de la population en raison de la demande croissante de ressources, notamment de médecins, d'infirmières, d'équipements et d'établissements de santé. Pour résoudre ces problèmes et fournir de meilleurs soins de santé aux patients, la prise de décision, la gestion, le pronostic et le diagnostic fiables ont été complétés par des systèmes intelligents. Parmi ces systèmes, les systèmes basés sur l'apprentissage en profondeur (DL), une sous-classe de l'apprentissage automatique (ML), ont surpassé de nombreux systèmes statistiques et ML traditionnels en raison de leur capacité à découvrir et à apprendre automatiquement les fonctionnalités associées pour une tâche donnée et leur robustesse. Par conséquent, l'utilisation de DL a connu une augmentation constante dans de nombreuses applications. Néanmoins, généralement, la formation des modèles DL repose sur un seul serveur centralisé, ce qui pose de nombreux défis : (1) à l'exception de certaines grandes entreprises, la plupart des petites entreprises disposent de données de qualité limitées, ce qui est insuffisant pour prendre en charge la formation de personnes avides de données. modèles DL, (2) l'accès aux données pose généralement des problèmes de confidentialité, (3) des coûts de communication élevés et des ressources de calcul requises, (4) un grand nombre de paramètres entraînables rendent le résultat de DL difficile à expliquer, ce qui

est requis dans certaines applications, tels que les soins de santé. Par rapport au ML centralisé, l'apprentissage fédéré (FL) améliore à la fois la confidentialité et les coûts de communication, où les clients forment en collaboration un modèle commun sans partager directement les données brutes. Cependant, FL apporte ses propres défis. Par exemple, des données locales hétérogènes parmi les clients rendent difficile la formation d'un ou plusieurs modèles mondiaux performants et robustes. Le partage des mises à jour (des centaines de milliers de paramètres) a encore des coûts de communication élevés. De plus, la nature distribuée et le contrôle d'accès des données locales dans FL les rendent plus vulnérables aux attaques malveillantes. De plus, le défi d'expliquer les résultats de DL reste encore difficile.

Par conséquent, l'objectif de cette thèse est de développer des cadres robustes, performants et respectueux de la vie privée dans des environnements fédérés pour la classification de bout en bout dans les applications de soins de santé en présence de données/mises à jour hétérogènes. De plus, les clients de FL sont généralement limités en ressources avec des ressources de calcul et de communication disponibles limitées. Par conséquent, pour prendre en charge un calcul et une communication efficaces dans un cadre fédéré, nous proposons un cadre léger (en termes de nombre de paramètres pouvant être entraînés). De plus, pour expliquer les résultats des modèles DL, qui sont généralement difficiles à expliquer en raison du grand nombre de paramètres, nous proposons des modules d'IA explicables indépendants du modèle pour aider à expliquer les résultats. Pour protéger les cadres proposés contre les cyberattaques, telles que les attaques par empoisonnement, nous proposons un cadre dans des environnements fédérés, rendant les captures et analyses de données en santé plus sûrs et plus fiables. Enfin, avec une analyse expérimentale utilisant des ensembles de données de base pour l'un des problèmes de santé les plus courants, à savoir les maladies cardiovasculaires (détection d'arythmie, détection d'anomalies ECG) et la reconnaissance

de l’activité humaine (utilisée pour compléter la détection des maladies cardiovasculaires), nous montrons la supériorité de la solution proposée à la pointe de la recherche en ce domaine (SOTA).

Mots clés: Sécurité, Informatique en périphérie, intimité, système de santé, Intelligence artificielle explicable, détection d’anomalies explicables, intelligence artificielle embarquée, Systèmes d’aide à la décision clinique, Sécurité et fiabilité de l’intelligence artificielle, attaques par empoisonnement, empoisonnement de données, empoisonnement de modèle, attaques byzantines.

Declaration of Authorship

I, **Ali RAZA**, declare that this thesis titled, "Secure and Privacy-preserving Federated Learning with Explainable Artificial Intelligence for Smart Healthcare System" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at the University of Kent and the University of Lille.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

____

Date:

____

“I seem to have been only like a boy playing on the seashore, and diverting myself in now and then finding a smoother pebble or a prettier shell than ordinary, whilst the great ocean of truth lay all undiscovered before me.”

Sir Issac Newton

Acknowledgements

I wish to express my heartfelt gratitude to my supervisors, Shujun Li, Ludovic Koehl, and Kim Phuc Tran, for their unwavering support, guidance, and encouragement throughout my PhD journey. Their vast knowledge, expertise, and insightful feedback have been instrumental in shaping the direction and quality of my research. They have provided me with the necessary tools, resources, and opportunities to grow both professionally and personally, and I am deeply grateful for their dedication and commitment to my success.

I am also deeply indebted to my family members and friends for their constant support, encouragement, and love. Their unwavering belief in my abilities, their patience, and their unwavering encouragement have been a constant source of inspiration and motivation.

I would like to extend my sincere appreciation to all the participants who took part in my study, without whom this research would not have been possible. Their willingness to share their experiences, insights, and perspectives has enriched the quality of my work and contributed to the advancement of knowledge in my field.

I am also grateful to the faculty and staff of the University of Kent and ENSAIT, University of Lille for providing me with an exceptional academic environment in which to pursue my research.

Thank you all for your invaluable support, encouragement, and guidance throughout my PhD journey.

This research work was supported by the I-SITE Université Lille Nord-Europe 2021 of France under grant No. ICOTKEN-20-001-TRAN-RAZA.

Dissemination

The work presented in this thesis has been disseminated in the following journal publications, presented in reversed chronological order:

- Raza, A., Tran, K. P., Koehl, L., & Li, S. (2023). AnoFed: Adaptive anomaly detection for digital health using transformer-based federated learning and support vector data description. Engineering Applications of Artificial Intelligence, 121, 106051. https://doi.org/10.1016/j.engappai.2023.106051

- Raza, A., Tran, K. P., Koehl, L., & Li, S. (2022). Designing ECG monitoring healthcare system with federated transfer learning and explainable AI. Knowledge-Based Systems, 236, 107763. https://doi.org/10.1016/j.knosys.2021.107763

- Raza, A., Li, S., Tran, K. P., & Koehl, L. (2022). Using Anomaly Detection to Detect Poisoning Attacks in Federated Learning Applications. arXiv preprint arXiv:2207.08486. (submitted to IEEE Transactions on Dependable and Secure Computing)

- Raza, A., Tran, K. P., Koehl, L., Li, S., Zeng, X., & Benzaidi, K. (2021). Lightweight transformer in federated setting for human activity recognition. arXiv preprint arXiv:2110.00244. (submitted to IEEE Transactions on Mobile Computing)

Contents

List of Abbreviations

AE	**A**uto**E**ncoder
CAM	**C**lass **A**ctivation **M**aps
CHF	**C**hronic **H**eart **F**ailure
CNN	**C**onvoluional **N**eural **N**etworks
DL	**D**eep **L**earning
DP-SGD	**D**ifferentially **P**rivate **S**tochastic **G**radient **D**escent
ECG	**E**lectro**C**ardio **G**raph
FL	**F**ederated **L**earning
HAR	**H**uman **A**ctivity **R**ecognition
IID	**I**ndependently and **I**dentically **D**istributed
KL	**K**ullback-**L**eibler
LSTM	**L**ong **S**hort-**T**erm **M**emory
MAE	**M**ean **A**bsolute **E**rror
MIA	**M**embership **I**nference **A**ttacks
MIT-BIH	**M**assachusetts **I**nstitute of **T**echnology and **B**eth **I**srael **H**ospital
ML	**M**achine **L**earning
MSE	**M**ean **S**quared **E**rror
OC-SVM	**O**ne-**C**lass **S**upport **V**ector **M**achine
ReLU	**R**ectified **L**inear **U**nit
SGD	**S**tochastic **G**radient **D**escent
SHAP	**S**hapley **A**dditive **E**xplanations
SOTA	**S**tate **O**f **T**he **A**rt
SVDD	**S**upport **V**ector **D**ata **D**escription

VAE	**V**ariational **A**utoencoder
XAI	**E**xplainable **A**rtificial **I**ntelligence

Chapter 1

Introduction

1.1 Context

The increasing global population will soon present a significant challenge for the global economy as it will require financial resources to cover expenses such as salaries, social insurance, and healthcare. Especially, healthcare systems are increasingly being challenged due to this growing and aging population around the globe. The elderly healthcare sector will require a substantial amount of valuable resources from society, including doctors, nurses, healthcare facilities, rehabilitation centers, and new medicines [1], [2]. Hence, it is crucial to develop advanced tools and systems that can support healthcare and enhance the quality of the healthcare system and reduce the economic burden. To address such challenges and to provide better and more reliable healthcare to patients, better decision-making in healthcare by learning from the health data is being utilized. For example, cardiovascular diseases are one of the leading causes of death [3]. Various policies have been introduced in recent years to provide tools and systems for reducing human loss to such cardiovascular events. Electrocardiography (ECG) was introduced to achieve such goals and it is the most commonly used physiological signal used to detect various cardiovascular diseases. Various types of intelligent systems have been developed to help ECG analysis, arrhythmia detection, and classification. Such methods include wavelet coefficient and fuzzy c-means clustering for disease diagnosis and classification [4], [5]. However, traditional signal processing and statistical methods rely on strong assumptions. For instance, assumptions about the distribution of the data which is hard to estimate with heterogeneous data. One more drawback of these methods is that the performance can be affected by slight changes in the input caused by factors like random noise. Therefore, there is a need to develop methods that can ensure improved resilience and accurate results.

In recent years, intelligent systems based on machine learning (ML) have been studied in a wide range of applications. For example, applications of ML have been studied for cyber security [6], economics and agriculture [7], and in healthcare [8]. Traditional ML systems rely on feature engineering, where human experts extract relevant features from the data and then use these features to train ML models such as decision trees, support vector machines, and random forests. The models learn relationships and patterns between input features and target variables which allows them to make decisions about new data. Nevertheless, traditional ML systems struggle with large-scale complex data and the need for human experts for feature engineering brings limitations to the use of ML. Hence, deep learning (DL), a subclass of ML, has seen a rapid increase with the availability of high computational resources and the massive amount of data being generated [9]. DL employs multi-layer artificial neural networks which automatically learn hierarchical representations of the training data through a process known as representation/feature learning. Therefore, allowing the DL model to learn directly from the data such as images, time-series, and audio without heavily relying on manual feature engineering. While ML is effective for certain tasks, DL outperforms in dealing with large-scale datasets and complex data representations.

Internet of Things (IoT) devices are capable of collecting an enormous amount of data each day [9], [10]. This collection of data and the exponentially increasing computational resources have unlocked new dimensions in the information technology sector, especially in DL [11]. Although DL is quite an old concept [12] owing to limited data and computational resources available in the past its use was limited. However, thanks to the internet, IoT devices, and increasing computational power, nowadays we can see DL revolutionizing nearly every field, including healthcare [13], economics [14], manufacturing [15], agriculture [16], and military [17]. In regards to healthcare applications, a lot of data have been generated across the globe and they have quite unique properties. Most of the data related to healthcare are multi-dimensional and complex, which makes the use of classical ML models, for example, decision trees and random forests, quite challenging and complex. However, the new generation machine learning models, especially the DL-based ones, can solve issues related to multi-dimensional data due to their capability of learning complex trends and patterns in data [18]. In the healthcare sector, DL has played a critical role, e.g., to help diagnose life-threatening diseases [19]. Given the right architecture and sufficient representative data, DL has the capability to find patterns in data when trained either in supervised or unsupervised settings, which are otherwise difficult for humans to find. For example, to monitor the heart condition of patients, clinical practitioners often have to go through an ECG recorded over long periods of time (hours to days). Finding patterns that could result in heart conditions in such long ECG usually becomes a hectic and time-consuming task for humans. ML algorithms could be used in such scenarios to help clinical practitioners reach quick and reliable results. Although DL can provide state-of-the-art (SOTA) performance and outperform traditional statistical methods in most of the applications, it brings its own challenges and limitations [20].

1.1.1 Limitation and Challenges of Deep Learning

DL requires a large amount of good-quality training data to be trained on to achieve desired utility. However, individual data sources like hospitals, clinics, smart devices, and smart wearables can have a minimal amount of data, so data from a single silo can be insufficient to train a high-performing DL model [21]. A solution to this is to collect data from multiple sources and then train the model on the collected data. One major issue of this approach is privacy concerns [22]. As some data (e.g., healthcare) are highly sensitive and private, some individual sources may not be willing to share their data with a central data collector, due to privacy concerns and market competition [23], [24]. Hence, it is crucial to safeguard data and comply with legal and ethical requirements. Secondly, data sources can distribute globally, and collecting such data into a central repository is a huge challenge. For example, communication cost is one of the challenges for collecting data from geographically distributed data sources. Furthermore, there exists the problem of explainability in DL. DL models are generally black box models (complex to interpret with hundreds of thousands of parameters), with no reasonable explanations for the outcomes. This ambiguity causes a limitation of DL in sensitive applications, such as healthcare. In healthcare, stakeholders such as clinical practitioners and patients should know the reason for a prediction by a DL model [25], without proper explanations of the results the application of DL in healthcare becomes limited. Explainability is important in healthcare because to convince clinical healthcare practitioners and patients we need to give them the reason behind a certain prediction for a given input. The performance of the model and explainability both are very important in healthcare applications. Explainability can also help to address the biases present in the DL model, i.e., DL can provide the right outcomes for the wrong reasons. Therefore, in clinical healthcare, practitioners and patients are unlikely to adopt ML systems without understanding the models and the reasons behind their outcomes. Hence, the explainability of the results obtained from DL model(s) to make them more understandable is of utmost importance for building trustworthy and reliable systems. In addition, explainability also brings trust and helps debug any unintended bias present in the model by allowing researchers to get insights and identify any biases in the model which can be used to refine the model to improve performance and mitigate biases.

Furthermore, the increasing adoption of DL in many applications has attracted attackers' attention. Research has shown that ML/DL models are vulnerable to advanced and sophisticated attacks [26]. Attacks such as poisoning attacks [27] attempt to achieve adversarial benefits, i.e., disabling the DL system's functionality and inferencing attacks [28]. For instance, an AI system training can be subjected to incorrectly labeled data with the aim of causing the model to be unreliable or inconsistent in the target application. Consequently, in order for DL systems to be more reliable and secure, research is needed to address security issues faced by DL systems. Additionally, inferencing attacks can be used to extract sensitive or private information from models or systems by making targeted queries or observations thereafter violating the privacy of data owners and security risks. In other words, it is vital to address the security issues of DL and such issues need much more attention with the widespread applications of DL.

1.1.2 Distributed Machine Learning and Federated Learning

Distributed machine learning (DML) trains a given ML model in multiple nodes to improve the performance and scale to large data. Nowadays, DML is an integral part of large-scale ML models to enable faster learning algorithms by utilizing a single server of the cluster in a single region that belongs to a given organization [29]. However, DML pays less focus on privacy and gives a centralized server access to raw data. Therefore, with the increase in privacy concerns over the transmission of local raw data and market competition, the use of federated learning (FL) [30], a subclass of DML, which explores the idea of training ML/DL models on the remote device has seen a growing interest. We use the term edge device/client to denote the entities such as nodes and edge devices and clients in the network. The basic idea behind FL is to collaboratively train a machine learning model without centralized training data. The clients/edge devices train a local model locally and exchange the updates (model parameters or gradients depending on the federated algorithm, we will discuss each algorithm in chapter 2). The aggregation/global serve aggregates the received updates according to a given aggregation method (mean, median, trimmed mean, etc) and exchanges the updates with the clients. FL enables devices with sufficient computational power (e.g., home computers, mobile phones, wearables, and other IoT devices) to collaboratively learn a shared machine-learning model while keeping all the training data on local devices, decoupling the ability to applying ML/DL from the need to store the data centrally at a single server or in the cloud. FL has been used in practice by many organisations [31] and it has a significant role in supporting privacy-sensitive applications by enabling the training of statistical models at the edge/client [32]. An overview of the federated setting for healthcare applications is shown in Figure 1.1.

Although FL enhances privacy, it faces many challenges to achieving high performance of jointly trained models and security. For instance, the statistical heterogeneity among the participating edge devices i.e., the device generates and collects data in a highly non-identically distributed manner.

Achieving SOTA performance and making FL robust against statistical heterogeneity is still challenging, especially when the data distribution among the clients is non-independently and identically distributed (non-IID), such as distribution skew, feature distribution skew, and quantity skew [33]. Here, robustness refers to the ability of the systems to perform consistently and effectively in the presence of various challenges, such as varying noise, non-IID data distribution, and different security attacks. Similarly, local clients in FL usually contain varying and unavoidable noise present in the data because of the experimental design and various other reasons, and training a robust global model under such conditions is highly challenging [34]. Such data distributions can lead to challenges in the aggregation of updates because models trained on non-IID may have divergent knowledge and trained parameters making it difficult to aggregate them in order to achieve desired performance. Furthermore, the problem of explanations still remains one of the main limitations. Additionally, communication of model updates while training is still vulnerable to privacy and security attack [35]–[37]. Although methods, such as encryption, and secure multiparty computation, can be leveraged to enhance security and privacy, such methods often bring reduced model performance and higher computational costs [38]. Hence, achieving a balance between performance, security, and privacy in FL is challenging.

1.2 Motivation and Problem Statement

Based on our discussion above, it can be concluded that DL has shown SOTA performance and robustness in different applications, such as healthcare. Some of the challenges faced by centralized DL can be addressed using FL, which has the potential to bring promising solutions by addressing some limitations of centralized DL. For example, it can enhance privacy and can help achieve high-performing DL models by jointly training a global model using

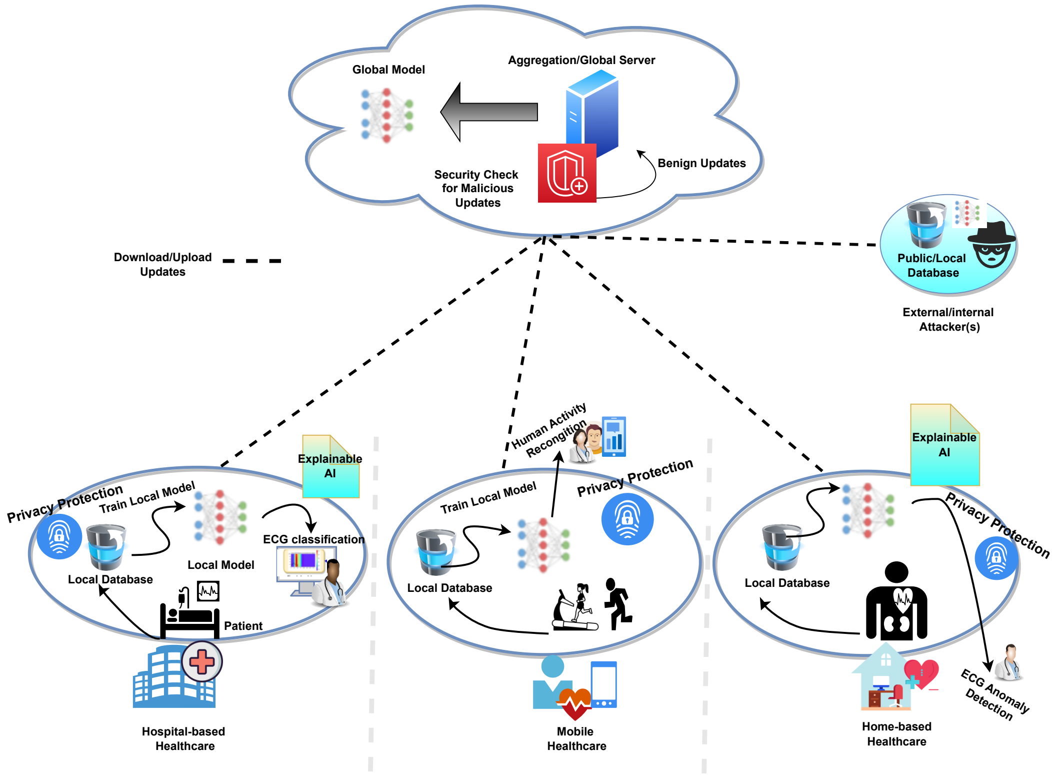


FIGURE 1.1: Overview of the architecture of FL-based healthcare

the updates of clients in the network. FL has seen growing interest where distributed parties can jointly train a global model without the need to directly share raw data with other parties. As discussed before, FL reduces communication costs and increases data diversity by allowing different data sources to jointly train a model. However, FL brings its own limitations and challenges. A key challenge in FL is the non-IID data among the participating devices in the network. When devices with heterogeneous data distribution train a joint model the model's performance is usually degraded. For example, if the amount of noise present in the raw data of each device/client is different then training a global model to achieve SOTA performance and robustness on such distribution is challenging [33]. Moreover, the distributed nature of FL makes it vulnerable to various security and privacy issues, for instance, poisoning and membership inference attacks [35]–[37]. Furthermore, in addition to achieving robust SOTA performance in privacy friendly environment, applications such as healthcare need explanations of the model's outcomes. Another challenge with FL is computation costs. Since DL usually involves higher computation and edge devices are usually more resource constrained compared to a central server, achieving both SOTA performance and low computation costs is challenging.

Hence, motivated by the promising outcomes of FL for privacy-friendly and robust DL, the aim of this thesis is to design and evaluate robust, high-performing, security and privacy-enhanced frameworks in federated settings for healthcare applications.

To achieve our goal, we have devised a strategy that involves dividing it into the following set of objectives:

1. The first objective is to develop high-performing, and privacy-friendly frameworks for end-to-end healthcare applications in federated settings that are robust enough to effectively handle non-IID data, ensuring that the models trained on diverse data sources can effectively generalize and perform well across different distributions and maintain their performance even when the input data contains unprecedented variations. In other words, we propose to make the framework robust when the data is non-IID and contains varying noise. Moreover, we emphasize the importance of computational efficiency, ensuring that the framework remains lightweight and suitable for edge devices with limited resources in a federated learning environment.

2. Furthermore, providing explanations for the outcomes is crucial, particularly in healthcare applications where explanations are highly valued. Hence, our second objective aims to ensure transparency by offering explanations that help understand the reasoning behind the model's decisions. This objective also addresses the need to mitigate any biases that may arise within the model, enhancing its fairness and reliability.

3. Finally, in order to protect the proposed frameworks against security attacks, such as poisoning attacks (we focus on poisoning attacks, see Chapter 2 for more details about poisoning attacks), our objective is to design frameworks that are robust against various poisoning attacks, such as the injection of malicious data or labels, enabling the models to maintain their integrity and accuracy even in the presence of such attacks.

1.3 Contributions

In this thesis, we propose high-performing, robust, privacy-enhanced, and secure frameworks with added explanations in federated settings for healthcare applications. We fine-tune the proposed frameworks for specific example applications for higher performance and compare them with SOTA in that application. The details of each contribution are discussed in the upcoming Chapters. Here we provide a brief overview of our contributions, which are given as follows.

1. First, we design a novel end-to-end privacy-friendly framework in a federated setting using explainable artificial intelligence (XAI) and deep convolutional neural networks (CNN). In this framework, we propose a novel two-step approach of using autoencoder and classifiers with transfer learning to achieve robustness and high performance against heterogeneous data. Additionally, we introduce an XAI-based module on top of the proposed classifier to improve explanations of the classification results, helping clinical practitioners make informed decisions.

2. In order to support lightweight computation for privacy-friendly FL and to achieve SOTA performance, we propose a transformer-based

approach for classification in federated settings, while being more computationally efficient for use on mobile devices.

3. We present a novel concept of unsupervised two-stage adaptive anomaly detection in the presence of heterogeneous data distribution in federated settings to enhance privacy protection, improve the explainability of results and support unsupervised learning in case of limited data availability.

4. We further introduce a novel framework for efficient and effective detection of poisoning attacks in federated settings.

1.4 Thesis Roadmap

An overview of the thesis roadmap is given in Figure 1.2, and the outline of the rest of the thesis is given as follows:

- Chapter 2 presents the background studies and literature review about deep neural networks, FL, XAI, and privacy and security challenges. Furthermore, we review SOTA works and identify their limitations, which we aim to address in the upcoming chapters.

- In Chapter 3, we report a novel end-to-end framework in a federated setting using XAI and deep convolutional neural networks (CNN). The federated setting is used to solve challenges such as data availability and privacy concerns. With ECG classification as an example application, we show that the proposed framework effectively classifies different arrhythmias using an autoencoder and a classifier, both based on a CNN. Additionally, we propose an XAI-based module on top of the proposed classifier for interpretability of the classification results, which helps clinical practitioners to interpret the predictions of the classifier and to make quick and reliable decisions. We train and test the

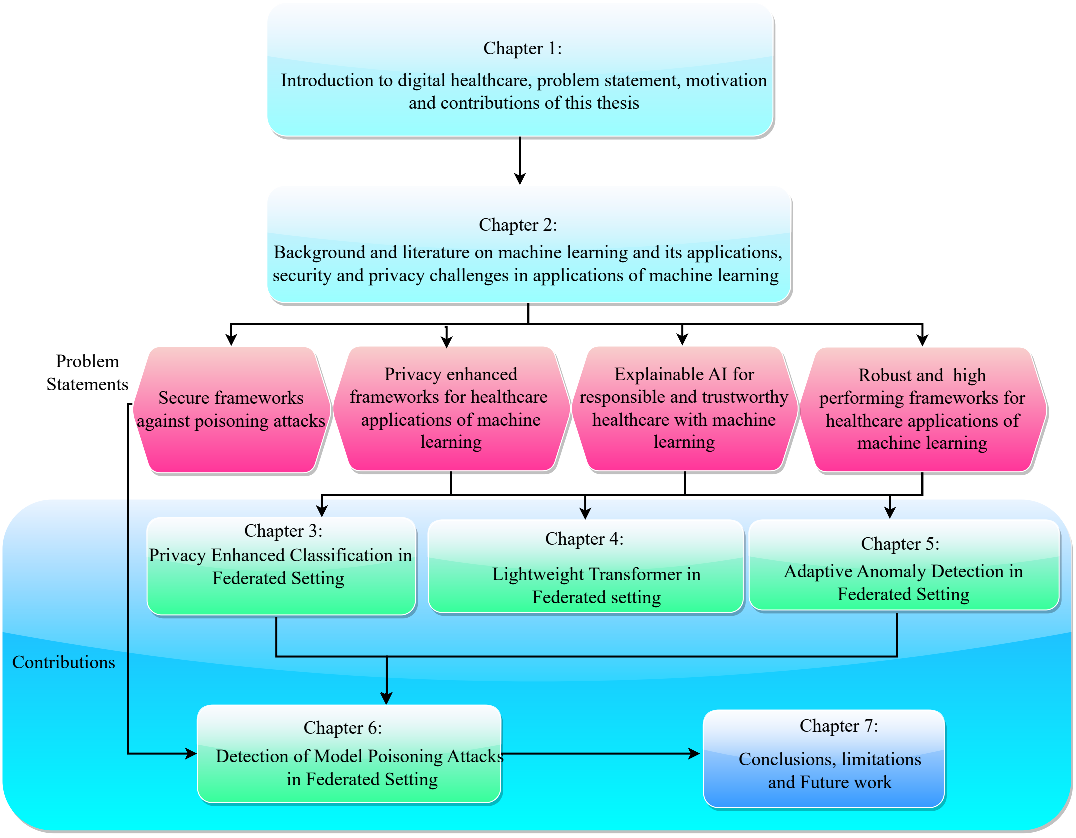

Chapter 1:
Introduction to digital healthcare, problem statement, motivation and contributions of this thesis
Chapter 2:
Background and literature on machine learning and its applications, security and privacy challenges in applications of machine learning
Problem Statements
Secure frameworks against poisoning attacks
Privacy enhanced frameworks for healthcare applications of machine learning
Explainable AI for responsible and trustworthy healthcare with machine learning
Robust and high performing frameworks for healthcare applications of machine learning
Contributions
Chapter 3:
Privacy Enhanced Classification in Federated Setting
Chapter 4:
Lightweight Transformer in Federated setting
Chapter 5:
Adaptive Anomaly Detection in Federated Setting
Chapter 6:
Detection of Model Poisoning Attacks in Federated Setting
Chapter 7:
Conclusions, limitations and Future work

FIGURE 1.2: Thesis roadmap

proposed framework using the baseline Massachusetts Institute of Technology - Boston's Beth Israel Hospital (MIT-BIH) Arrhythmia database. We show that the trained classifier outperformed existing work by achieving accuracy up to 94.5% and 98.9% for arrhythmia detection using noisy and clean data, respectively, with five-fold cross-validation showing its robustness against heterogeneous data distribution. We also propose a new communication cost-reduction method to reduce communication costs and to enhance the privacy of users' data in the federated setting.

- In Chapter 4, we propose a novel lightweight (in terms of the number of parameters) transformer, which can combine the advantages of RNNs and CNNs without their major limitations (discussed in section 2.3), and also TransFed, a more privacy-friendly, federated learning-based human activity recognition (HAR), using our proposed lightweight transformer. We designed a testbed to construct a new HAR dataset from five recruited human participants and used the new dataset to evaluate the performance of the proposed HAR classifier in both federated and centralized settings. Additionally, we use another public dataset to evaluate the performance of the proposed HAR classifier in centralized setting to compare it with existing HAR classifiers. With the experimental results show that our proposed new solution outperforms SOTA HAR classifiers based on CNNs and RNNs, while being more computationally efficient to be used on mobile computing resource constraint devices.

- Chapter 5 describes AnoFed, a novel framework for combining the transformer-based AE and VAE with the Support Vector Data Description (SVDD) (see Chapter 2 for more details) in a federated setting. It can enhance privacy protection, improve the explainability of results and support unsupervised learning (adaptive anomaly detection). Using ECG anomaly detection as a typical application of the framework in healthcare, we conducted experiments to show that the proposed framework is not only effective (in terms of the detection performance) but also efficient (in terms of computational costs), compared with a number of state-of-the-art methods in the literature. AnoFed is very lightweight in terms of the number of parameters and computation, hence it can be used in applications with resource-constrained edge devices.

- To address the challenges related to security in FL, in Chapter 6 we propose a novel framework for detecting poisoning attacks in FL, which employs a reference model based on a public dataset and an auditor model to detect malicious updates. We implemented a detector based on the proposed framework and using a one-class support vector machine (OC-SVM), which reaches the lowest possible computational complexity $\mathcal{O}(K)$ where K is the number of clients. We evaluate our detector's performance against SOTA poisoning attacks for two typical applications of FL: ECG classification and human activity recognition. With experimental results, we validate the performance of the proposed framework.

- Finally, Chapter 7 concludes the thesis and presents limitations and future research directions.

Chapter 2

Background and Literature Review

In this chapter, we provide background and literature review. In particular, we provide preliminaries and review current work and identify their limitations.

2.1 Introduction to Machine Learning

We begin with the definition and basic types of ML and then proceed towards more complex ML/DL algorithms, such as support vector data description, and multi-layer networks (convolutional neural networks, transformers, and autoencoders).

2.1.1 Machine Learning

An ML algorithm is an algorithm that is able to learn from the data. According to the definition presented in [39], an ML algorithm is said to learn if given an experience **E** to the algorithm with respect to a task **T**, and performance measure **P**, the performance **P** for task **T** improves with experience **E**. The learning algorithm can be supervised, as well as unsupervised. In unsupervised learning, the aim of the ML algorithm is to learn features and useful properties of the dataset related to a given task, for example, clustering [40]. Generally speaking, the ML algorithm learns the probability distribution for a given task. In supervised learning, the ML algorithm learns features and useful features from the dataset where each data sample is associated target or label.

Many types of tasks can be solved using ML. Some of the most common are given as follows:

Classification: In classification, the ML algorithm specifies the output category/class of a given input. To accomplish such tasks the learning algorithm learns a function $f : \mathbb{R}^d \rightarrow \{1, \ldots, n\}$. If $\mathbf{x}$ is a vector input and y is the corresponding output, the model f assigns the input to the output $y = f(\mathbf{x})$. When the number of output classes is two then it is called binary classification and when it is greater than two it is called multiclass classification. We will see some examples of applications of classification in section 2.2.2.

Anomaly detection: In this type of task, the ML algorithm searches for events that deviate from the normal events and flags the events which deviate from the normal as anomalies. A typical example of anomaly detection is spam email detection. We will see some example applications of anomaly detection in section 2.2.2.

Denoising: In this type of task, given a noisy/corrupt input $\mathbf{x}^{'} \in \mathbb{R}^d$ of a clean sample $\mathbf{x} \in \mathbb{R}^d$ obtain via an unknown process, the ML algorithm learns to reconstruct $\mathbf{x}$ from $\mathbf{x}^{'}$. In other words, the ML algorithm learns the conditional probability distribution $p(\mathbf{x}|\mathbf{x}^{'})$. We will provide more details about denoising and example applications in section 2.1.5.

2.1.2 Performance Measures

Some of the performance measures that we will be using in this thesis are given as follow:

Classification Performance: Four standard metrics found in the literature [41] are accuracy, precision, recall and F1-score. While accuracy measures the overall system performance over all classes in the dataset, the other metrics are specific to each class, and they measure the ability of the classification algorithm to distinguish certain events. For a binary classifier, each of the metrics is defined as follows:

1. **Accuracy** is the most intuitive performance measure and it is simply a ratio of correctly predicted observations to the total observations, as defined below:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}}, \tag{2.1}$$

   where TP, TN, FP and FN refer to the numbers of true positives, true negatives, false positives, and false negatives, respectively.

2. **Precision** is the ratio of correctly predicted positive observations to the total predicted positive observations, as defined below:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{2.2}$$

3. **Recall** is the ratio of correctly predicted positive observations to all the observations in actual positive class, as defined below:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{2.3}$$

4. **F1-score** is the harmonic mean of precision and recall, as defined below:

$$\text{F1-Score} = \frac{2 \times \text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}}. \tag{2.4}$$

The above definitions can be easily extended to multi-class classifiers with $m > 2$ classes. For instance, accuracy is defined as the ratio between the number of total correct predictions and the total number of samples. For other metrics, i.e., precision, recall, and F1-score, we can derive m binary classifiers, one for each given class versus all remaining classes (one-vs-rest), and then use the above definitions of the three metrics as usual for each of the m binary classifiers.

Mean Squared Error: Mean squared error (MSE) is a method used to measure the amount of difference between an observed and predicted value in statistical modeling. When the observed and predicted values are the same the MSE is zero, and as the difference between them increases, the MSE also increases. Let's suppose that y_i presents the observed value and $y_i^{'}$ presents the predicted value and n represents the total number of samples (observations), then MSE is given as follows.

$$MSE = \frac{\sum_{i=1}^{n}(y_i - y_i^{'})^2}{n} \tag{2.5}$$

Kullback-Leibler Divergence: is an asymmetric metric to measure or quantify the difference between a given sample probability distribution to a reference probability distribution. Let us suppose that $p(x)$ presents the reference probability distribution and $q(x)$ presents the sample probability distribution. Then the discrete form of KL-divergence is given as follows:

$$KL(p(x)||q(x)) = \sum_{x \in X} p(x) \ln \frac{p(x)}{q(x)} \tag{2.6}$$

2.1.3 Support Vector Data Description (SVDD)

The Support Vector Data Description (SVDD) [42] leverages a support vector classifier [43] to construct a spherical boundary around a given distribution of a dataset, with a minimum volume containing as much as possible data samples from the given data distribution. Let us suppose that $\{X_i \in \mathbb{R}^d\}_{i=1}^N$ are a set of N d-dimensional training samples, a and R denote the center and the radius of a sphere covering the training set, respectively. Huang, Chen, Zhou, *et al.* [44] formulated this goal as a constrained convex optimization problem, given as follows:

$$\min_{R,a,\xi_i} F(R,a,\xi_i) = R^2 + C\sum_i \xi_i, \text{ s.t.} \qquad \begin{cases} ||X_i - a||^2 \leq R^2 + \xi_i, \\ \xi_i \geq 0, \end{cases} \quad i = 1, \ldots, N, \tag{2.7}$$

where, the slack variable ξ_i defines the possibility of anomalous (outliers) data in the given training data. The parameter C is used to balance the trade-off between the volume inside the boundary and the errors. The Lagrangian function with Lagrange multipliers α_i and γ gives

$$\begin{aligned} L(R,a,\xi_i,\alpha_i,\gamma_i) = R^2 + c\sum_i \xi_i \\ - \sum_i \alpha[R^2 + \xi_i - (X_i - a)^2] - \sum_i \gamma_i \xi_i. \end{aligned} \tag{2.8}$$

By setting the partial derivatives of a, R, and ξ_i to zero, we can achieve the following constraints:

$$\sum_i \alpha_i = 1, A = \sum_i \alpha_i X_i \tag{2.9}$$

$$C - \gamma_i - \alpha_i = 0 \implies 0 \geq \alpha_i \geq C. \tag{2.10}$$

From the above equations, we can get

$$\max L = \sum_i \alpha_i (X_i \cdot X_i) - \sum_{i,j} \alpha_i \alpha_j (X_i \cdot X_j), \text{ s.t.}$$
$$\begin{cases} \sum_i \alpha_i = 1, \\ 0 \leq \alpha_i \leq C, \end{cases} \quad i = 1, \dots, N, \tag{2.11}$$

where, $\cdot$ operator denotes the inner product between two vectors. A training sample X_i and its corresponding α_i should follow one of the following conditions:

- $||X_i - a||^2 < R^2 \implies \alpha_i = 0;$
- $||X_i - a||^2 < R^2 \implies 0 < \alpha_i < C;$
- $||X_i - a||^2 < R^2 \implies \alpha_i = C.$

The samples whose coefficients follow $\alpha_i > 0$ are known as support vectors. The center of the sphere can be obtained by Eq. (2.9). The radius R can be obtained by calculating the distance from any support vector with $0 < \alpha_i < C$ to the center. In order to test if a given sample Z is inside or outside of the defined boundary of a sphere, the distance from the center to Z is calculated. If the distance is smaller than the radius R, then Z is considered inside and not an outlier, given as follows.

$$||Z - A||^2 = (Z \cdot Z) - 2\sum_i \alpha_i (Z \cdot X_i) + \sum_{i,j} \alpha_i \alpha_j (X_i \cdot X_j) \leq R^2. \tag{2.12}$$

The method can be made more flexible [42], [45] by employing new inner products satisfying Mercer's theorem. Moreover, a polynomial kernel and the Gaussian kernel can also be employed to achieve more flexibility as discussed in [42], [46].

2.1.4 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are feed-forward networks usually used to analyze images using grid-like topology. A CNN usually consists of convolutional layers that help to extract features of a given input. Convolutional layers comprise filter(s) to perform the convolution operation. Other than convolutional layers, CNN consists of ReLU (rectified linear unit) layer and pooling layers. ReLU layers take in the feature maps from the convolutional layers and apply ReLU activations which map the negative values in the feature map to 0 to produce a rectified feature map. The ReLU layer introduces non-linearity in the network which helps the network learn non-linear decision boundaries. Other activation functions such as sigmoid can also be used instead of ReLU, however, ReLU provides sparsity and reduced likelihood of vanishing gradient. Furthermore, pooling layers are used for up/down-sampling the dimensionality of the rectified feature map. Pooling layers help to detect different parts like edges, corners, etc. For classification, task CNNs consist of a fully connected neural network that takes in the features extracted by the convolutional layers and helps achieve a given task (e.g., classification). Figure 2.1 presents a typical CNN architecture.

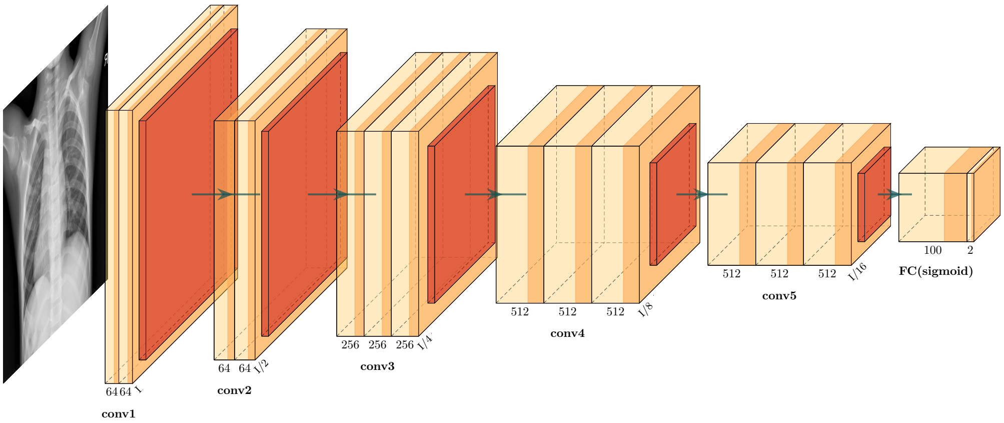


FIGURE 2.1: Architecture of Convolutional Neural Networks

2.1.5 Autoencoders and Variational Autoencoders

Autoencoders (AEs) [47] are neural networks trained mainly using unsupervised learning. They have been extensively used for data denoising and compression [48], [49]. An AE usually consists of two main components: an encoder and a decoder. The encoder learns a latent space vector representations during the training phase, while the decoder learns do reconstruct the original input given the latent vectors. We depict an AE with a single hidden layer [50] in Figure 2.2.

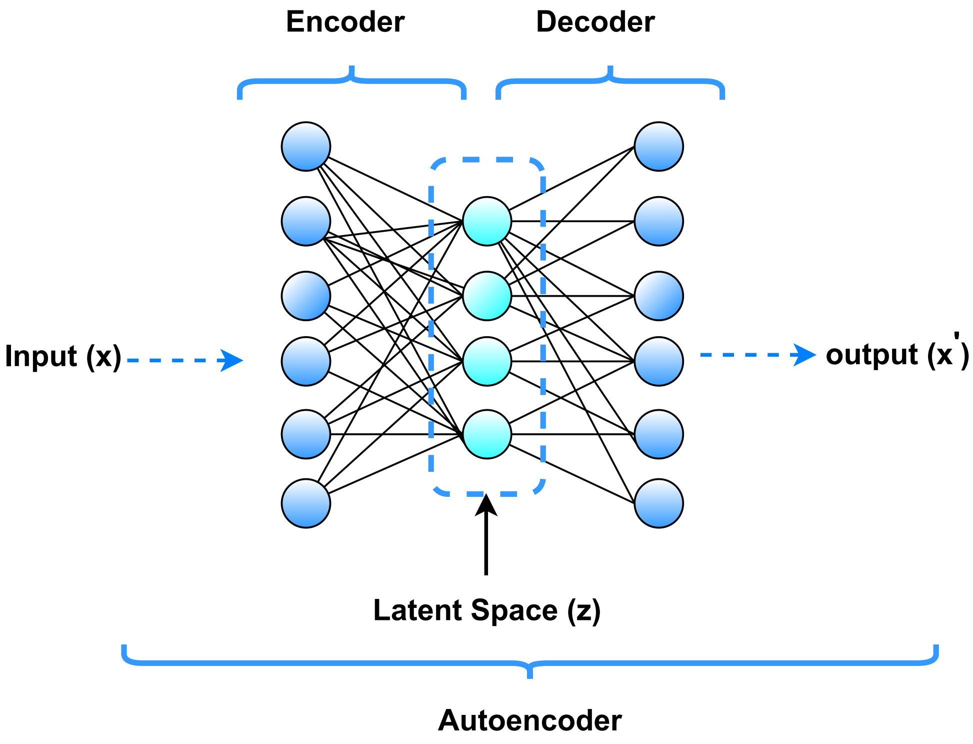


FIGURE 2.2: An AE with a single hidden layer.

The mathematical formulation of an encoder is given by the equation below:

$$Z = \phi_1(W_1 X + B_1), \tag{2.13}$$

where $X \in R^k$ is a k-dimensional input vector, z is a latent space vector, ϕ_1 is an activation function and W_1 represents the weights matrix of the encoder and B_1 is the bias vector. The parameters W_1 and B_1 are randomly initialized at the start and updated during the training phase. For the decoder, the following equation shows how it works:

$$X' = \phi_2(W_2 Z + B_2), \tag{2.14}$$

where $X' \in R^k$ is a k-dimensional output vector obtained using the latent representation given by the encoder, ϕ_2 is an activation function, W_2 and B_2 are weights matrix and the bias vector of the decoder, respectively. Each input X of the encoder part of an AE is mapped into a latent space vector. This latent space vector is used as the input of the decoder that produces X' (a reconstructed version of X) as the output. The model's internal parameters are trained by minimizing the reconstruction loss L with a suitable optimizer, given by the following equation:

$$L(\theta) = \frac{1}{N} \sum_{i=1}^{N} ||X_i - X_i'||^2, \tag{2.15}$$

where N is the total number of input vectors, θ denotes the model's parameters, and X_i and X_i' are the i-th input and output vectors, respectively. Generally, the latent representations have a lower dimensionality than the input vector, let's say k, so that the AE will keep the important and relevant information necessary to reconstruct the input [51]. When a trained AE is used for anomaly detection, the reconstruction loss is normally used to detect the anomalies [52], [53].

Variational autoencoders (VAEs) [54], [55] are structurally similar to AEs, with the only difference being that the VAEs learn a latent distribution while the AE learns a point in the latent space. This latent distribution is regularized to be close to a standard normal distribution [56]. For a given input data X, let us assume that X is computed using its corresponding latent variable

Z that cannot be observed directly. If we denote the prior distribution of Z as $p(Z)$, and consider that the input Z is sampled from the conditional likelihood $p(X|Z)$, then Bayes theorem gives the link between the prior $p(Z)$, likelihood $p(X|Z)$, posterior distribution $p(Z|X)$, as shown in the following equation:

$$p(Z|X) = \frac{p(X|Z)p(Z)}{p(X)}. \tag{2.16}$$

Given an input dataset X defined by an unknown probability function $p(X)$ and a latent vector Z, a VAE learns from the input to get a distribution $p_\theta(X)$, where θ is the set of the network parameters. Equation (2.17) represents the mathematical formulation of an unknown probability function.

$$p(X) = \int p(X, Z)dz. \tag{2.17}$$

Unfortunately, $p(X)$ is intractable distribution and hence we cannot compute it directly. However, by leveraging variational inference the problem of intractable distribution can be solved. If we consider $p(Z|X)$ to be approximated by another tractable distribution $q(Z|X)$, then the parameters of $q(Z|X)$ can be defined to be very similar to $p(Z|X)$ to infer the intractable distribution. By minimizing the KL-divergence (a metric describing the difference between two probability distributions), we can ensure that $q(Z|X)$ is similar to $p(Z|X)$,

$$\min_{\mathrm{KL}}(q(Z|X)||p(Z|X)), \tag{2.18}$$

We can minimize the KL-divergence by minimizing the following:

$$E_q(Z|X)\log p(X|Z) - \mathrm{KL}(q(Z|X)||p(Z|X)). \tag{2.19}$$

The above equation ensures that the learned distribution q is similar to the prior distribution p. A VAE mapping X to Z and reconstructing X from Z is shown in Figure 2.3.

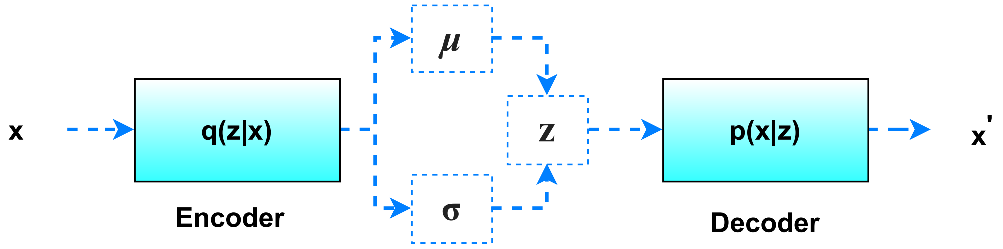


FIGURE 2.3: The general structure of a VAE.

The decoder learns to reconstruct the input from the latent space vector. Furthermore, re-parameterization is used to calculate the relationship between the model's internal parameters and the loss using backpropagation. Re-parameterization randomly samples ϵ from a standard normal distribution, and then shifts the random sample ϵ with mean μ and scales it by the variance σ of the latent distribution, given by the following equation:

$$Z = \mu + \sigma \times \epsilon. \tag{2.20}$$

The loss function for a VAE consists of two different losses (as shown in the equation below): one is used to penalize the reconstruction loss, and the second (KL-loss) is used to ensure that the learned distribution $q(Z|X)$ is similar to the true prior distribution $p(Z)$, which follows a unit normal distribution, across each dimension j of the latent space.

$$L(X, X') + \sum_j \mathrm{KL}(q_j(Z|X)||p(Z)). \tag{2.21}$$

Since being proposed, many researchers have proposed many optimized approaches of autoencoder, such as sparse autoencoder, denoising autoencoder, contractive autoencoder, and convolutional autoencoder [57]. We can achieve two main tasks from autoencoders: denoising and dimensionality reduction. In this study, we build a denoising autoencoder, which is an extension of simple autoencoders. It is worth noting that denoising autoencoders were not originally meant to automatically denoise an input. Instead, the denoising autoencoder procedure was invented to help:

1. the hidden layers of the autoencoder learn more robust filters,
2. reduce the risk of overfitting in the autoencoder, and
3. prevent the autoencoder from learning a simple identity function.

In denoising autoencoders noise is stochastically (i.e., randomly) added to the input data, and then the autoencoder is trained to recover the original, non-perturbed signal.

2.1.6 Transformers

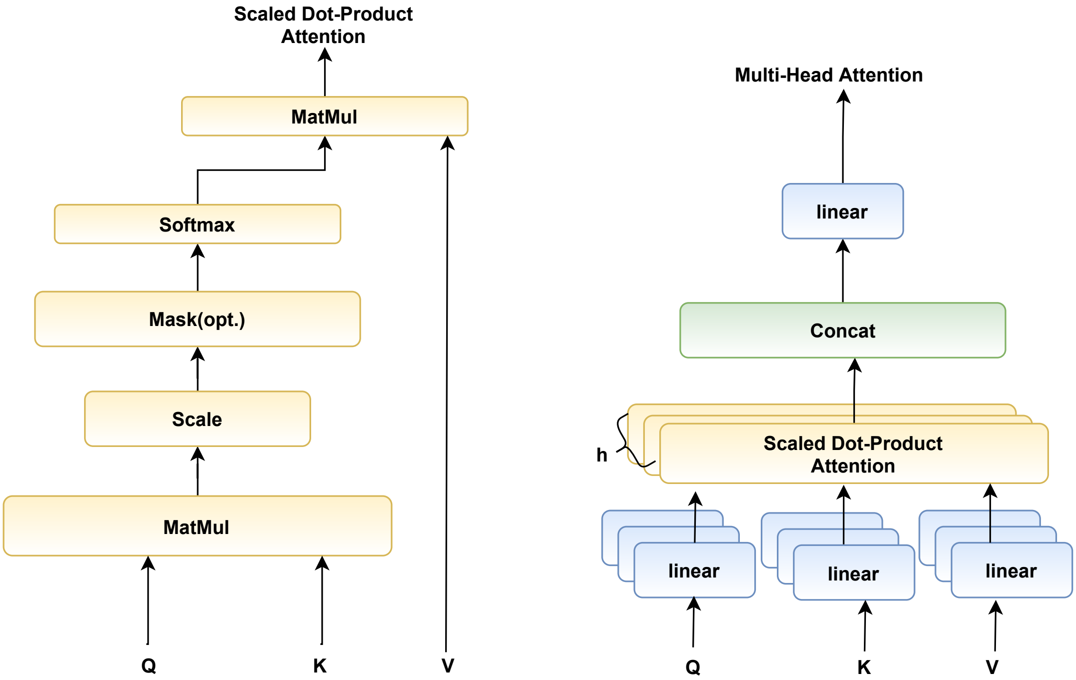


FIGURE 2.4: An illustrative diagram of the attention mechanism and its parallelization, regenerated from [58]. Left: the attention mechanism, Right: parallelization of the attention mechanism.

Vaswani et al. [58] introduced a novel architecture called Transformer for sequence-to-sequence learning. One of the key components of transformers is the attention mechanism. The attention mechanism looks at an input sequence and decides at each step which other parts of the sequence are important. Similar to LSTMs, a transformer basically transforms one sequence to another one with the help of two parts: an encoder and a decoder, but it differs from existing sequence-to-sequence methods in that it does not imply any recurrent networks (GRU, LSTM, etc.). The encoder and decoder consist of modules that can be stacked on top of each other multiple times. Each module mainly consists of multi-head attention and feed-forward layers. The input and output are first embedded into an n-dimensional space since they cannot be used directly. Another part of the model is a positional encoding of different words. Since there are no recurrent networks that can remember how a sequence is fed into the model, a relative position is encoded for every part of the input sequence. These positions are added to the embedded n-dimensional vector of each input sub-sequence.

The attention mechanism used in transformers can be described by the following equation:

$$\text{Attention}(Q, K, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{2.22}$$

where Q is the query matrix (vector representation of input sub-sequence), K are all the keys (vector representations of all the sequences) and V are the values (vector representations of all the sequences). For the encoder and the decoder, multi-head attention modules, V consists of the same word sequence as Q. However, for the attention module that is taking in the encoder and the decoder sequences, V is different from the sequence represented by Q. In other words, V is multiplied and summed with the attention weights γ, defined by the following equation:

$$\gamma = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right). \tag{2.23}$$

The self-attention mechanism is applied multiple times in parallel along with the linear projections of Q, K, and V. It helps the system to learn from different representations of Q, K, and V. The weight matrices W that are learned during the training are multiplied by Q, K, and V to learn the linear representations. Figure 2.4 gives an illustrative diagram of the attention mechanism and its parallelization, where *Matmul* stands for matrix multiplication and *Mask (opt.)* is an optional operation in the self-attention mechanism that controls which positions in the input sequence can attend to which other positions.

Moreover, positional encoding is used to keep track of the input and output sequence. Finally, transformers employ feed-forward networks. These feed-forward networks have identical parameters for each position of the input sequence, which describes each element from a given sequence as a separate but identical linear transformation.

Transformers have various applications because they use attention mechanisms, which can help the machine learning models to learn from data more effectively to improve the performance of many machine learning tasks such as natural language processing ones [59]. For example, a hybrid HAR classifier using CNNs and transformers was introduced in [60], which utilizes a two-streamed structure to capture both time-over-channel and channel-over-time features and use the multi-scale convolution augmented transformer to capture range-based patterns.

So far we have introduced different DL models. How each model is used and trained is slightly different and depends on the use case. However, how each DL model learns is similar. Hence, understanding what each model learns and how the learned patterns/ trends are distributed across multiple layers is important to enhance the performance and utility in real-world applications. Concepts such as stitching connectivity [61], and memorization capability [61], [62] are being used to understand the learned representations of DL models.

2.1.7 Stitching Connectivity

Stitching connectivity [61] is a method to measure the similarity of internal representations of different models trained using different but similar data. Consider two models A and B, which have the same architecture. For A and B to be stitched connected, they can be stitched at all the layers to each other. In other words, two models, let's say A and B with identical architecture but trained using stochastic gradient descent(or its variants) using independent random seeds and independent training sets taken from the same distribution. Then the two trained models are stitched and connected for natural architectures and data distributions. Hence, we expect the models trained on similar but different training sets of the same distribution will behave similarly.

DL models learn from the training data and the trained models can memorize patterns and trends in data which is important for their utility in real-world applications. In some cases, they can even memorize the data without learning trends and key patterns (over-fitting) which is not desired. Moreover, different layers in the model learn different patterns and trends in data. Hence, understanding how and what type of information each layer learns is of utmost significance in order to improve the utility of DL models and to understand if the patterns and trends learned by the model are aligned with the domain knowledge. Moreover, stitching connectivity helps understand how the internal representation is affected by noisy data. Such understanding can be used to address model and data poisoning attacks and the effect of noise in federated learning.

2.1.8 Memorization in Deep Neural Networks

Deep neural networks are capable of memorizing the training data in a fashion such that they prioritize learning simple patterns first [62] using the lower-level layers in the model, while the higher-level layers tend to learn more specific data characteristics. Furthermore, when a model is trained on noisy data, the first half of the layers are similar to a model trained on good-quality data [61].

Understanding the memorization patterns in DL models can help explain complex models. For example, by analyzing the activation of higher-level layers we can explain which patterns and trends in the data are significant for the mode to make predictions. In the next section, we will explain how memorization in deep learning can be used to explain the results of complex DL models.

2.1.9 Explainable Artificial Intelligence

Explainable Artificial Intelligence (XAI) [25] lets humans understand and articulate how an AI system made a decision. XAI encompasses a range of procedures and techniques enabling human users to grasp and understand the outcomes and outputs generated by machine learning algorithms. It serves as a means to elucidate an AI model, its projected influence, and possible biases. XAI aids in assessing model precision, equity, openness, and the consequences of AI-driven decision-making. For an organization, XAI plays a pivotal role in instilling trust and assurance when implementing AI models. Additionally, explainability in AI supports responsible AI development practices, while comprehending the process behind a particular output of an AI-powered system offers numerous benefits. Explainability can help researchers ensure that the is model working as expected and meets regulatory standards. I can also help the end users affected by the decisions of any model to challenge an outcome. Recent research suggests that it will be of key importance in marketing [63], healthcare, manufacturing, insurance, and automobiles [64].

To address the problem of explainability in DL models, researchers have proposed different solutions [65]–[67]. For instance, Selvaraju proposed a method called Gradient-weighted Class Activation Mapping (Grad-CAM) [68] to visualize input regions that are important for predictions. From such values, we can have an idea about where exactly the machine learning model is focusing while making a prediction and thus the reason. Explainability is important in healthcare because to convince a clinical healthcare practitioner and a patient we need to give them the reason behind a certain prediction for sample input.

Shapley additive explanations (SHAP)[69] are also widely used to explain how each feature contributes to and affects the output of each model. SHAP is based on cooperative game theory and can increase the explainability of ML models. However, SHAP is more suitable for tabular data, in this thesis our focus is more on time-series data and SHAP values are not suitable for time-series data. In addition, SHAP values are computationally expensive because of the exponentially increasing number of coalitions with the increase in the number of features which limits its use in resource-constraint devices [70]. In comparison to SHAP, Grad-CAM-based explanations are more suitable for data such as time-series and images and are computationally less expensive. Hence, in this thesis, we adopt Grad-CAM-based explanations.

2.2 Machine Learning in Healthcare

In this section, first, we provide a general overview of ML applications in healthcare and then discuss some example applications and recent work.

2.2.1 Overview

The use of machine learning in healthcare has been widely studied ranging from detection and diagnosis of different diseases, such as melanoma [71], [72] and cancer [73], [74]. Owing to the importance of machine learning in healthcare-related applications, in this section we review the literature on machine learning with a special focus on arrhythmia detection, human activity recognition, and ECG anomaly detection as well will be using them as example healthcare applications in upcoming chapters.

Certain activities in our body are governed by signals of some cognitive diseases [75]. For example, a changing gait may result from a stroke. A number of researchers proposed to monitor users' activities using wearable sensors, with the help of which different human body activities can be recognized [76]–[78]. Based on the monitoring of such activities, the early prognosis of health issues can be identified. In this regard, there has been significant development in the utilization of ML and DL technologies in healthcare. While such technologies will probably never completely replace clinical practitioners, they can transform the healthcare sector, benefiting both patients and providers [19], [79]–[81].

Since the applications of ML in healthcare are a vast topic and covering all of them in this thesis is not possible. Therefore, in this thesis owing to the impact of cardiovascular diseases on humans we use and review time-series data-based ECG classification/arrhythmia detection as an example healthcare application. Moreover, we also use and review human activity recognition using time-series data as it supplements ECG analysis i.e., ECG can vary depending on human activity [82].

2.2.2 Deep Learning-based Healthcare

In regard to healthcare, ML and DL play a vital role. Researchers have proposed many methods for different healthcare applications using time-series data, such as methods to address cardiovascular diseases, some of which are given as follows.

1. **Electrocardiograph (ECG) classification:** ECG classification into arrhythmia types [83]–[87] is one of the most important routine tasks. Rubin et al. [88] applied deep learning to the task of automated cardiac auscultation, i.e., recognizing abnormalities in heart sounds. They described an automated heart sound classification algorithm that combines the use of time-frequency heat map representations with a deep CNN. Their CNN architecture is trained using a modified loss function that directly optimizes the trade-off between sensitivity and specificity. Gjoreski et al. [89] presented a method for chronic heart failure (CHF) detection based on heart sounds. The method combines classic ML and end-to-end DL models. The classic ML model learns from expert features, and the DL model learns from a spectro-temporal representation of the signal. Moreover, in order to enable the intelligent classification of arrhythmias with high accuracy, Huang et al. [90] presented an intelligent ECG classifier using the fast compression residual convolutional neural networks (FCResNet).

2. **Human activity recognition (HAR)**: Based on data from one or more body sensors, HAR classification uses a classifier to predict human activities. HAR can be used along with ECG classification as ECG can significantly vary depending on human activity. Generally, the data contains tri-axial data from different sensors like accelerometers, gyroscopes, and magnetometers. Most modern smart devices such as

smartphones and wearables have such sensors. Initially, researchers used various hand-crafted features for training HAR classifiers.

Generally speaking, hand-crafted features can be divided into three main types: frequency domain, time domain, and time-frequency analysis. Classical machine learning models like k-means, probabilistic methods (naive Bayes), and support vector machines have been proposed for HAR classification [91].

Deep learning exploits the benefits of having a huge amount of data and highly non-linear deep models, to outperform classical models. Using deep learning, the feature extraction simply can be omitted which in the case of classical machine learning is a hectic and important task. Raw data in the form of sliding windows or simple windows can be directly fed into deep learning-based classifiers. CNNs, long short-term memory (LTSM) RNNs, and hybrid models combining RNNs and CNNs are dominating approaches proposed for HAR [92]–[99]. Figure 2.5 shows the pipeline of a typical HAR classification process.

Generally in HAR, the raw data from sensors is transformed into windows of a fixed-length size, which are fed directly to the classifier. While prediction, data is collected using the same window length and then, again depending on the model selected for a HAR classifier, features are extracted, or raw windowed data is fed into the classifier that predicts the target human activity, such as walking, sitting, etc.

3. **Anomaly Detection** The use of ML and DL for anomaly detection has been steadily growing in academia as well as in industry due to its proven performance. It finds its application in many domains such as cyber security [100], telecommunication and networking [101], and

healthcare [102]. An extensive survey of such anomaly detection methods can be found in [103], which gives a board review of different methods including those based on machine learning as well as those that do not use machine learning. Moreover, the survey also discusses applications of anomaly detection in cyber security, medical image analysis, natural language processing, wireless sensing, etc. In the cyber security research literature, intrusion detection has been the topic of many researchers. For example, a comprehensive study on anomaly detection-based intrusion detection techniques was presented in [104], covering statistical and machine learning-based techniques. Kwon, Kim, Kim, *et al.* [101] presented network anomaly detection based on restricted Boltzmann machine-based deep belief networks and deep recurrent neural networks, as well as other methods based on more traditional machine learning algorithms. Durga, Nag, and Daniel [105] presented anomaly detection using machine learning (including deep learning) algorithms in the context of the Internet of Things (IoT) based healthcare. Also focusing on healthcare-related applications, Wang, Zhao, Xiong, *et al.* [106] applied deep learning to analyze physiological signals that allow doctors to identify latent health risks. Similarly, some researchers have investigated the potential of using smartphones and wearable devices to capture data in this regard, and the latter is seen as a promising solution for healthcare [107], [108].

Although the aforementioned work seems promising, they may find limited applicability in the real world because they use centralized data collection and training techniques. A centralized approach may cause privacy concerns among users and data owners. Thereafter, traditional centralized healthcare applications find limited applicability due to privacy concerns [109]–[111]. Furthermore, since most of the real-time data, such as ECG data and HAR data are noisy, existing approaches cannot perform well in real time because

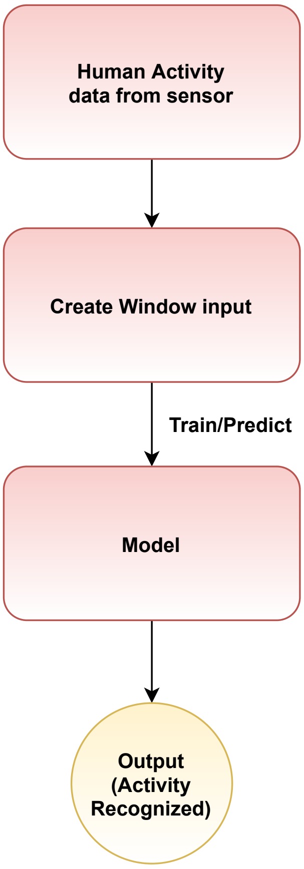


FIGURE 2.5: Overview of the HAR classification pipeline.

they are being trained on preprocessed (cleaner) data. Furthermore, they do not provide explainability, which is one of the key requirements in deep learning-based clinical healthcare [64]. Moreover, the communication cost of large datasets is also a challenge in centralized settings. Hence, this limits their real-time application.

In the next section, we will review FL, which was introduced to solve some of the key challenges in centralized ML.

2.3 Federated Learning

To address the privacy issues in centralized ML, researchers have been working on Federated learning (FL) and Transfer learning (TF). Federated machine (FL) learning was first proposed by Google [30], an overview of FL is shown in Figure 2.6. In FL settings machine learning models are trained based on distributed edge devices. The key idea is to protect user data during the process. It works like this: an edge (client) device downloads the current model, improves it by learning from data on its local data, and then summarizes the changes as a small focused update. Only this update to the model is sent to the aggregation/global server, using encrypted communication (optionally), where it is aggregated with other user updates to improve the global shared model. All the training data remains on local devices. FL allows for smarter models, lower latency, and less power consumption while ensuring privacy. This approach has another benefit: in addition to providing an update to the global shared model, the improved model on the local edge device can also be used immediately, powering experiences personalized by the use of IoT devices.

There are mainly two different approaches for making global updates: i) federated averaging where clients send the updates (learned parameters) to the global server after training the local model for multiple training epochs, and ii) federated stochastic gradient descent (SGD) where clients send updates (gradients) to the global server after each local training batch. McMahan et al. [112] compared the two approaches and showed that federated averaging can reduce communication costs by a factor of 10 to 100 times, compared to federated SGD. The merits of federated averaging make it popular in many applications [113]. There are various types of FL depending upon how the updates and collaboration are managed [32]. For example, in horizontal FL each client contains subsets of the overall data distribution. In vertical FL each client contains different sets of features but shares a common set of data samples. Another type of FL is transfer FL, which combines FL with transfer learning. In this technique, a pre-trained model trained on a large public dataset is distributed among the clients and the clients fine-tune it using their local data. There are various other types of federated learning [32]. In this thesis, we focus more on horizontal FL as it is commonly in different applications.

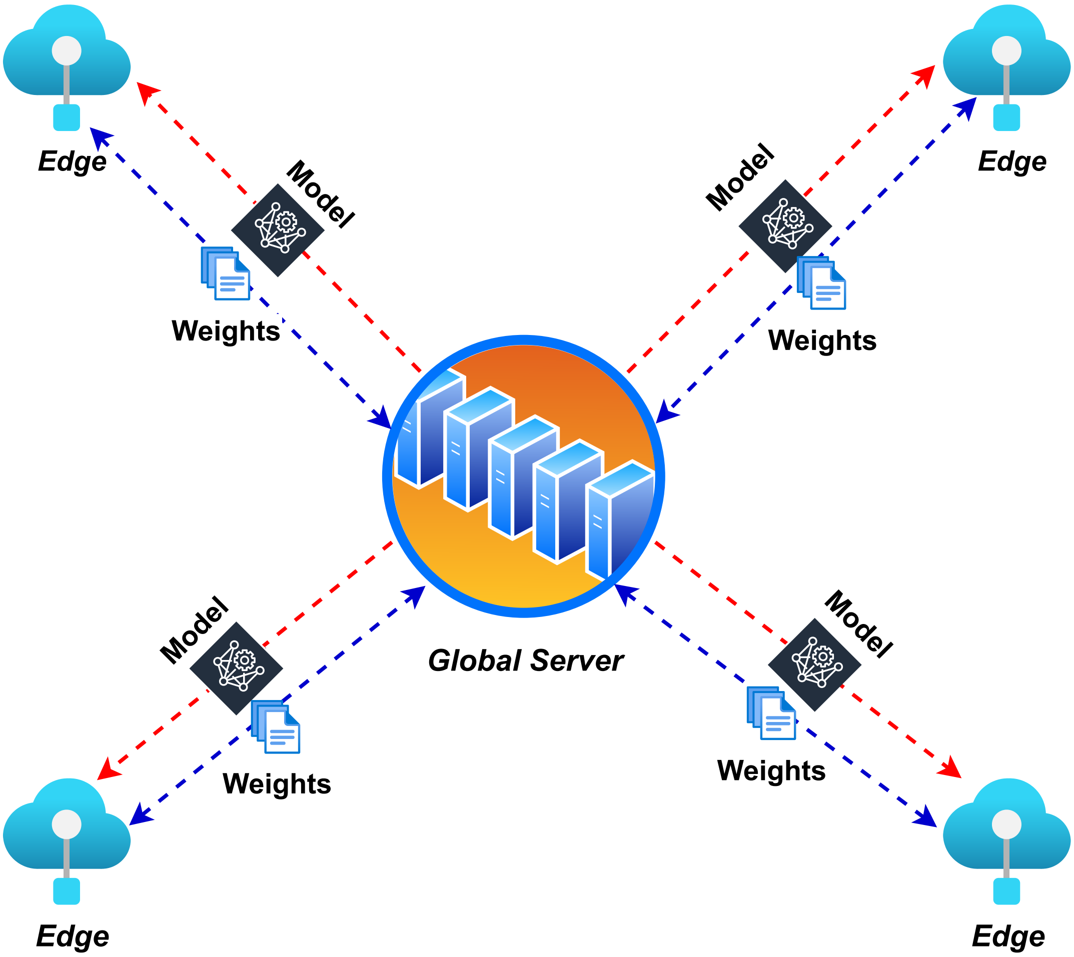


FIGURE 2.6: Architecture of Federated Learning

Federated learning has many advantages over the centralized approach. For instance, one of its advantages over the centralized approach is that it can provide more privacy protection for sensitive data. This is because the global model is trained without requiring clients to share their local (often sensitive) data directly. Moreover, it reduces communication costs because only trained parameters are shared, instead of the often large amount of data from all clients. Furthermore, FL has the ability to resolve the data islanding problems by privacy-preserving model training in the network. Due to its privacy-friendly and efficient communication constraints, FL finds a number of applications in healthcare [113]. Xu et al. [114] summarized the general solutions to the statistical challenges, system challenges, and privacy, and point out the implications and potentials of FL's application in healthcare. They show that training the model in the federated learning framework leads to comparable performance to the traditional centralized learning setting. Transfer learning (TF) aims at transferring knowledge from an existing trained model to a new model. The key idea is to reduce the distribution divergence between different models. To this end, there are mainly two general approaches: instance reweighting [115] and feature matching [116]. Recently, deep transfer learning methods have made considerable success in many application fields. Chen et al. [117] proposed FedHealth, the first federated transfer learning framework for wearable healthcare to tackle privacy and security challenges. FedHealth performs data aggregation through federated learning and then builds relatively personalized models by transfer learning. FedHealth makes it possible to do deep transfer learning in the federated learning framework without accessing the raw user data. However, there are certain limitations to it. Firstly, it does not provide the explainability of the predictions, which is often required in sensitive domains like healthcare. Secondly, it does not accommodate any mechanism to denoise the raw signals, which often contain random noise, and dealing with the random noise is quite challenging in a federated setting.

In other words, regarding the application of ML and DL healthcare, a lot of promising work has been done as discussed above both in centralized settings [67], [86], [87], [118]–[122] and in federated settings [117], [123]. However, centralized approaches are vulnerable to privacy issues. Research work like FedHealth tries to address the issues of privacy concerns using FL and TL architecture. Nevertheless, works like FedHealth have the limitation of explainability, and adaptive anomaly detection with evolving data. Furthermore, most of the existing work is based on RNNs and CNN-based models. However, RNNs are costly because of their serial computation and they generally require more computational time. CNNs reduce the cost of sequence-to-sequence modeling because they are easy to parallelize, which is not possible in RNNs. However, one disadvantage of CNNs is that they require a very large number of layers to capture the long-term dependencies (in applications, such as anomaly detection and HAR) in the sequential data, eventually making the model so large that would be impractical to use in resource-constraint devices.

In addition to the challenges mentioned above most of the existing work relies on the assumption that the clients have independent and identically distributed data which is a strong assumption for practical applications. In practical applications, clients contain noisy data samples and existing FL techniques are not robust enough to eliminate the effect of noisy data (varying across clients) resulting in significant performance reduction [124]. Since FL contains a large number of clients, the local data of each client has varying noise present in it due to various reasons, such as the different data collection setup, availability of different data handling, and preprocessing techniques. In addition, the distribution of data from each client can vary from one client to another i.e., unbalanced class skew distribution, feature skew distribution, etc. For example, in case one client can have more data samples of a particular disease class that is common in a particular geographical region compared to other clients in different geographical regions with a lesser number of cases of that particular disease class. Hence, it is challenging to achieve a robust global model in a federated setting when is local data of clients is non-IID. To achieve robust global models and reduce the impact of non-IID data on the global model, methods are needed to develop which can provide robustness against non-IID data of the federated clients.

Furthermore from a security perspective, in FL the global model can be easily manipulated, even if a single-edge device is compromised [35]–[37].

The attack surface of FL is growing due to its distributed nature. For example, malicious peers can launch data poisoning [125], [126] or model poisoning [127] attacks, in which one or more malicious edge devices manipulate their local training data or the local model trained on benign data, to impair the performance of the updated global model. Moreover, an attacker can launch inference attacks to compromise the data owner's privacy. We will discuss some SOTA security and privacy attacks later.

FL can be divided into three phases: data and behavior auditing, training, and testing. FL faces different kinds of security threats in each phase [128]. Hence, establishing secure FL needs to take effective measures at each phase to mitigate such threats. A solution before integrating a local model into the global model is to audit the local before training the local model(s). However, due to the privacy concerns and the architecture of FL, it is challenging to achieve such audits [128]. A trivial method to address model poisoning attacks could be using accuracy, i.e., using accuracy as a measure to access the quality of data being used to train the local model. Nevertheless, such methods can not be generalized as accuracy solely cannot reveal information about the underlying data. Just looking at the accuracy it cannot be claimed that the model is trained on benign or malicious data. Furthermore, models can be designed to have high accuracy for the testing samples by including them in the training dataset. A model can have low accuracy even if it has been trained on benign data depending on the amount of training data, training epochs, hyper-parameters tuning, optimization, etc. Hence, new solutions are required to detect such models and data poisoning attacks. Methods should be developed to verify that the shared local model gradients are not trained on anomalous or malicious (e.g., noisy, featured poisoning, label poisoning) data. In other words, malicious behaviors of the locally trained models should be detected before considering them in the global aggregation process in order to prevent malicious peers from compromising and manipulating the global model. Therefore, many researchers have proposed advanced poisoning attack detection methods for FL [129]–[131], which unfortunately all suffer from various weaknesses, e.g., the number of attackers has to be known or not high enough, working with non-independent and identically distributed (non-IID) data only, and high computational complexity.

In the next section, we will discuss some of the security and privacy challenges (our focus will be poisoning attacks) in FL and will review SOTA work related to the security issues in FL.

2.4 Poisoning Attacks in Federated Learning

In this section, first, we provide an overview of the poisoning attacks in FL and then discuss SOTA approaches in an attempt to address such issues.

2.4.1 Byzantine attacks

Byzantine attacks are a type of attack where a trusted device or a set of devices turn rogue and try to compromise the overall system. Such attacks can significantly reduce the performance of the global model in federated learning [132]. Researchers have shown that in the case of some aggregation algorithms, even the presence of a single malicious node can significantly reduce the performance of the global model [35]–[37]. Byzantine attacks, such as poisoning attacks [27], [133], can substantially reduce the performance (classification accuracy, precision, and recall) of FedAvg, even in the presence of a very small percentage of adversarial participants in the network. Such attacks can be classified as targeted attacks that negatively impact only one or more target (but not all) classes under attack and untargeted attacks that impact all the classes negatively. Furthermore, poisoning attacks are mainly classified into two main categories: data poisoning [27] and model poisoning [35] attacks depending on the phase where the attacks are executed. If the attacker manipulates the training data then this is called data poisoning attacks and if the attacker manipulates the trained model's parameters then such attacks are called model poisoning attacks. Further details of each type of poisoning are given as follows:

Data poisoning attacks:

Data poisoning attacks [27], [132] are those attacks in which the attacker manipulates the training data directly according to a given strategy and then trains the model using the manipulated dataset. In this study, we consider the following four types of SOTA data poisoning attacks:

1. **Random label flipping poisoning attacks:** In such attacks the attacker flips the true labels of the training instance randomly.

2. **Random label and feature poisoning attacks:** In such attacks, in addition to flipping the label randomly, the attacker adds noise to the input features of the training instances.

3. **Label swapping poisoning attacks:** In such attacks the attacker swaps the labels of selected samples of a given class with those of another class.

4. **Feature poisoning attacks:** In such attacks the attacker adds noise (enough to manipulate the global model) to the features of the training data.

Model poisoning attacks:

In the model poisoning attacks [35], the attacker trains the model using legitimate datasets and then manipulates the learned parameters before sending it to the global server. In this study, we consider the following four types of SOTA model poisoning attacks:

1. **Sign flipping attacks:** In such attacks the attacker trains the model using legitimate data and then flips the sign of trained parameters and enlarges their magnitude.

2. **Same value attacks:** In such attacks the attacker sets the parameter values as **C**, where **C** corresponds to a vector whose elements have an identical value C, which is a constant set to a value such as 100, 200, 300, etc.

3. **Additive Gaussian noise attacks:** In such attacks the attacker trains the model as expected with legitimate data but adds Gaussian noise before sharing the updates with the global server.

4. **Gradient ascent attacks:** In such attacks the attacker trains the models using a gradient ascent instead of a gradient descent optimizer.

2.4.2 Existing defense mechanisms

In order to address the above-mentioned attacks, many defense methods have been proposed. Hu et al. [132] conducted a survey of the state-of-the-art defensive methods against byzantine attacks in FL. They show that byzantine attacks by malicious clients can significantly reduce the accuracy of the global model. Moreover, their results show that existing defense solutions cannot fully be protecting FL against such attacks. In [131] Xia et al. summarized poisoning attacks and defense strategies according to their methods and targets. In this section, we discuss some of the existing SOTA defense mechanisms below.

Distance-Based Mechanisms

Such mechanisms detect malicious updates by calculating the distance between updates. Updates with a larger distance from others are discarded from global aggregation. For example, Blanchard et al. [36] proposed Krum, where the central server selects updates with minimum distance from the neighbors. Similarly, Xia et al. [134] proposed a method to discard the shared parameters with a large distance from the mean of shared parameters. Bo et al. [135] proposed FedInv, which inverses the updates from each client to generate a dummy dataset. The server then calculates Wasserstein distances for each client and removes the update(s) with exceptional Wasserstein distances from others. Gupta et al. [136] proposed a method called MUD-HoG, a method to address poisoning attacks in federated learning using long-short history of gradients of clients. Nevertheless, for such methods to work the number of attackers is needed to be known in advance or needs to be less than a certain percentage of total clients in the network.

Performance-Based Mechanisms

Methods in this category evaluate updates based on a clean dataset that is contained by the server. Updates that underperform are either assigned low weights or removed from the global aggregation. For example, Li et al. [137] proposed using a pre-trained autoencoder to evaluate the performance of an update. The performance of malicious updates will be lower as compared to that of benign updates. Nevertheless, training an autoencoder needs sufficient benign model updates which are hard to get. Xie et al. [138] proposed Zeno, which requires a small dataset on the server side. It computes the score for each update using the validation dataset server-side. A higher score implies a higher probability of the respective update being benign and vice-versa. However, Zeno requires knowledge about the number of attackers in advance to work properly.

Statistical Mechanisms

A method in this category use statistical features of the shared gradients or updates. Commonly used features are mean or median. Such features can circumvent benign updates and help to achieve a robust and probably more benign gradient. For example, a coordinate-wise median and a coordinate-wise median-based solution are provided by Yin et al. [139], which aggregates the parameters of local models independently, i.e., for the i-th model parameter the global server sorts the i-th parameter of the m other local models. Removes the largest and smallest parameters and computes the mean of the remaining parameters as the i-th parameter of the global model. Similarly, El Mhamdi et al. [140] proposed to combine Krum and a variant of trimmed mean [139]. However, these approaches are vulnerable to poisoning attacks even using robust aggregation [128]. Additionally, for such methods to work, the number of attackers should be limited by an upper bound (e.g., 50%) or should be known prior.

Target Optimization-Based Mechanisms

Target optimization-based methods optimize an objective function to improve the robustness of the global model. For example, Li et al. [141] RSA which regularizes the objective in such a way that it forces each local model in federated learning to be close to the global model. However, such methods only address data poisoning attacks and fail to eliminate model poisoning attacks.

2.5 Inference Attacks and potential solutions in Machine Learning

As discussed earlier, FL can enhance the privacy of users and it has been successfully applied to many applications as discussed above. However, FL has certain limitations: models trained in federated settings can still leak information about the local training data because machine learning models inherit properties of data or can memorize the training data in case of over-fitting. Such properties can be exploited by malicious or untrusted actors in an attempt to steal information from the shared parameters of the locally trained models [142]. For example, inferring attacks are mainly used to identify training data in a white box or a black box model. Such attacks can be divided further into two main types: membership inference attacks (MIAs) [28] and reconstruction attacks [143]. MIAs attempt to find out if a given data sample has been used to train a target model or not. For example, Nasr et al. [144] designed an MIA in a white-box setting against both centralized machine learning and FL systems. Later they successfully applied the proposed attack in a federated setting to infer training data and information via a curious but honest aggregation server or any other participating client in a federated setting. Therefore, in this section, we review MIAs attacks and their potential solutions.

2.5.1 Membership Inference Attacks

Most research on MIAs is based on the fundamental idea proposed by Shokri et al. [145], where a binary attack classifier model $\mathbf{M}_{\text{attack}}$ given a target model $\mathbf{M}$ and a data sample x_i, gives a decision if x_i has been used in the training of $\mathbf{M}$, i.e., a member or a non-member of the training dataset. Figure 2.7 presents an overview of MIAs.

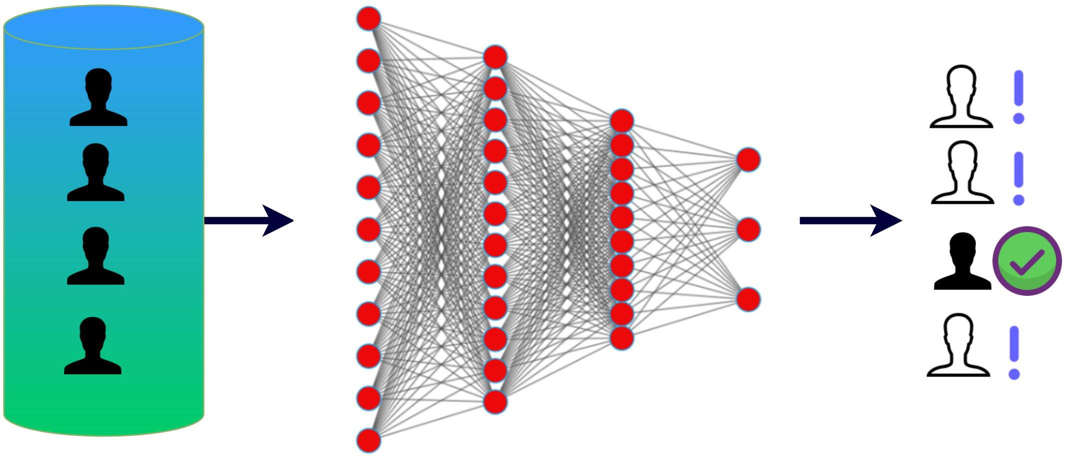


FIGURE 2.7: Overview of MIAs: Given a trained model and a data sample an attacker attempts to identify if the data sample was included in the training dataset to train the model.

We can divide MIAs into two major types: threshold-based attacks and training-based attacks, which are described as follows.

Threshold-based attacks: In a threshold-based attack [146], a threshold value is used to count the number of training and testing samples with membership probabilities larger than the given threshold. The threshold value is usually chosen to be between 0.5 (random guessing) and 1 (100% accuracy). Furthermore, the receiver operating curve is also utilized in such attacks to determine how accurately an attacker can determine a member and a non-member data sample [147].

Training-based attacks: Training-based attacks usually involve training a number of shadow attack models. The aim of the shadow models is to mimic the behavior of the target model $\mathbf{M}$. To train $\mathbf{M}_{\text{attack}}$ a number of shadow models are developed to imitate $\mathbf{M}$, where the training data of shadow models is known to the attacker. The attacker constructs training data using the training input and output pairs of shadow models, which is further used to train the $\mathbf{M}_{\text{attack}}$ in order to distinguish members and non-members of training data of $\mathbf{M}$. Shokri et al. [145] proposed that using a higher number of shadow models can improve the performance of $\mathbf{M}_{\text{attack}}$. There are a number of methods to create the training data of shadow models. For example, it can be created using noisy-real world data which is similar to training data of **M**, or synthetic data can be created with the help of **M**.

Training several shadow models is computationally insufficient. To address this issue, Salem et al. [148] proposed a method that uses **M**'s prediction on the target points to deduce its membership status, where **M** itself acts as a shadow model. Therefore avoiding training of shadow models to imitate **M**.

2.5.2 Differential Privacy as a Solution to MIA

As discussed in the previous section, privacy threats from attacks like MIAs pose serious challenges in the development of FL-based applications. In order to address such challenges, a promising method called differentially private stochastic gradient descent (DP-SGD) [149] was introduced. The basic idea of DP-SGD is to add noise into the clipped gradients computed using stochastic gradient descent which is widely used in machine learning algorithms. The two main modifications made to SGD are: random noise is added to clipped gradients in order to make it hard for the attackers to know whether or not a particular training sample was used in the training of the model or not, and secondly, the sensitivity of each gradient is bounded. It was claimed that DP-SGD can provide provable DP guarantees. Very recently (since 2022), a few other methods have been proposed as new variants of DP-SGD [150]–[152]. However, such new variants are still too new to be widely used before they are further validated. Therefore, it remains the most studied method of applying DP to FL [153].

2.6 Summary

In this chapter, we have examined various applications of DL in healthcare and the challenges associated with it. Additionally, we have explored the concept of FL and its potential to address some of the challenges faced by centralized machine learning (ML) approaches. We have highlighted how FL can enhance user privacy by enabling collaborative learning without direct access to local data. Furthermore, we have emphasized the importance of XAI in sensitive applications, as it can assist users and help researchers in mitigating potential biases in DL models.

Moreover, we have discussed the limitations of FL, including security attacks, inference attacks, and difficulties in handling non-IID data distribution. In conclusion, FL shows promise in creating privacy-preserving DL systems, but it confronts several obstacles, such as poisoning attacks, inference attacks, and suboptimal performance with non-IID data. Additionally, the need for explanations, inherent to DL itself, is also crucial. To fully unlock the potential of FL-based DL systems, it is imperative to develop frameworks that are secure, privacy-preserving, robust against non-IID data, and provide transparency in the decision-making process. Table 2.1 presents a summary of the literature review.

TABLE 2.1: Summary of Cited Literature

Topic	Citation(s)
Machine Learning in Healthcare	melanoma [71], [72] ; cancer detection [73], [74]; Arrhythmia detection [67], [83]–[87], [89], [90], [118]–[122]; Human activity recognition [91]–[99]
Anomaly detection	cyber security [100], telecommunication and networking [101], [105], and healthcare [102], [105], [106]
FL for enhanced privacy	Image classification [30]; healthcare [113], [117], [123]
Issues in FL	Varying Noise across clients [34], [124]; identically distributed (non-IID) and heterogeneous data [33] ; security and privacy [27], [35]–[37], [125]–[127], [132], [145]–[148]; computational costs [38]
Poisoning attacks detection in FL	Distance-Based Mechanisms [36], [134]–[136]; Performance-Based Mechanisms [137], [138]; Statistical Mechanisms [128], [139], [140]; Target Optimization-Based Mechanisms [141]
Explainable AI	Shapley additive explanations [69], [70]; Visual explanations [65]–[68]
Privacy enhancement in FL	differentially private stochastic gradient descent (DP-SGD) [149]–[152]

Chapter 3

Privacy Enhanced Robust End-to-End Classification with Federated Learning and Model-Agnostic Explanations

3.1 Introduction

In recent years DL has been applied for many applications. For example, solutions have been proposed for analyzing and classifying time-series health-related data: ECG data [67], [86], [87], [117]–[122], [154]–[156]. However, as we argued in Chapter 1 most of these works are based on a centralized ML architecture, thereafter they are prone to issues like privacy concerns and data availability. Moreover, since most of the real-time time-series data is noisy, they cannot perform well in real time because they are being trained on preprocessed (usually cleaner) data. Furthermore, they do not provide explainability, which is one of the key requirements in DL-based clinical healthcare. Hence, this limits their real-time application. To address all of the above-mentioned challenges, in this chapter, we propose an end-to-end privacy-friendly explainable framework in a federated setting. The proposed framework consists of three main parts: an autoencoder, a classifier, and an XAI module. Firstly, we propose a novel deep convolutional neural network (CNN) based autoencoder, which is used to denoise the raw time-series signals from the subject directly. Secondly, we propose a novel CNN-based classifier, which uses transfer learning to classify the raw time-series data. Thirdly, we adopt the Grad-CAM model [68] in the framework to explain the classification results in a novel and reliable pattern. Additionally, we propose a custom communication cost reduction approach that reduces the communication cost and increases the privacy protection of the framework.

3.1.1 Contributions

The main contributions of this chapter are as follows:

1. We propose an end-to-end framework which is the first federated transfer learning and explainable-AI-based framework for healthcare. It aggregates the data from different edge devices (hospitals, users) without compromising privacy and security, provides relatively personalized model learning through knowledge transfer, and provides explanations of the results, which is one of the key requirements in applications like healthcare. In addition to explanations, the proposed XAI module can be used to recognize new potential patterns leading to trigger heart arrhythmias.

2. We propose a novel 1-dimensional CNN-based autoencoder in a federated setting to efficiently denoise the raw time series data collected data from patients. The autoencoder provides a denoised version of the input, which we use for further classification and explanation of the predictions.

3. With the help of transfer learning, we use the encoder part of the proposed autoencoder to make a novel 1-Dimensional CNN-based classifier to classify given time-series data into respective classes.

4. We propose a novel module, called the XAI module for explanations of predictions of the proposed classifier. The proposed XAI module is combined with the proposed classifier to explain the decision-making process of the classifier. The XAI module can be used with every updated classier locally at the edge devices in the federated setting, and it does not need any pre-training.

5. We propose a new communication cost reduction method for federated learning in the proposed framework, which not only reduces the communication costs but also increases the privacy of the classical federated learning method. Furthermore, the proposed method can be integrated into existing cost optimization algorithms to enhance their cost effectiveness and privacy protection level.

6. We used the MIT-BIH Arrhythmia Database [157] to train our proposed framework. It is important to note that to make the data more realistic, we first upsample the data to create more data samples, and then add 10-30% random noise. The proposed framework shows excellent performance by providing an overall accuracy of 94.5% using noisy data and overall accuracy of 98.9% on the clean data in the original MIT-BIH database. Moreover, we evaluated the performance of the proposed framework using four standard metrics: classification accuracy, precision, recall, and F1-score.

7. The proposed framework additionally boasts desirable features: explanations of the results by using the proposed XAI module, and efficient classification of the ECG. Additionally, it provides an enhanced level

of privacy protection to users because of the federated setting and the proposed communication cost-reduction method.

The rest of the chapter is organized as follows. Section 3.2 discusses a detailed description of the proposed framework. Sections 3.3 and 3.4 present the experimental setup and performance evaluation, respectively. Section 3.5 summarizes the chapter.

3.2 The Proposed Framework

Before describing our proposed framework in detail, let us explain the research problem first. Given data on N different edge nodes (since we are using cross-silo federated learning, each edge node can represent a different organization, i.e., hospital) represented by $E = \{E_1, E_2, \ldots, E_N\}$ and the data of each E_i (here $i = 1, 2, \ldots, N$) is given by $\{D_1, D_2, \ldots, D_i\}$, respectively. A conventional ML model, denoted by ConMOD, can be trained by combining all the data $D = \{D_1, D_2, \ldots, D_i\}$. The data from different edge nodes have different distributions. However, in our problem, we want to collaborate all the data to train a federated transfer learning model, denoted by FedMOD, where any user E_i does not expose its data D_i to others. Assume that AccFed represents the accuracy of FedMOD and AccCon$_i$ represents the accuracy of each locally trained model of E_i, then one of the objectives of our proposed method is to ensure that the accuracy of AccFed is close to or superior to each AccCon$_i$.

The proposed framework aims to achieve accurate and efficient personal healthcare through federated transfer learning and XAI without compromising privacy. Figure 3.1 gives an overview of the proposed method. The proposed method consists of three major parts, the autoencoder, the classifier and the XAI module, which are discussed below in the following three subsections. The final sub-section discusses the learning process.

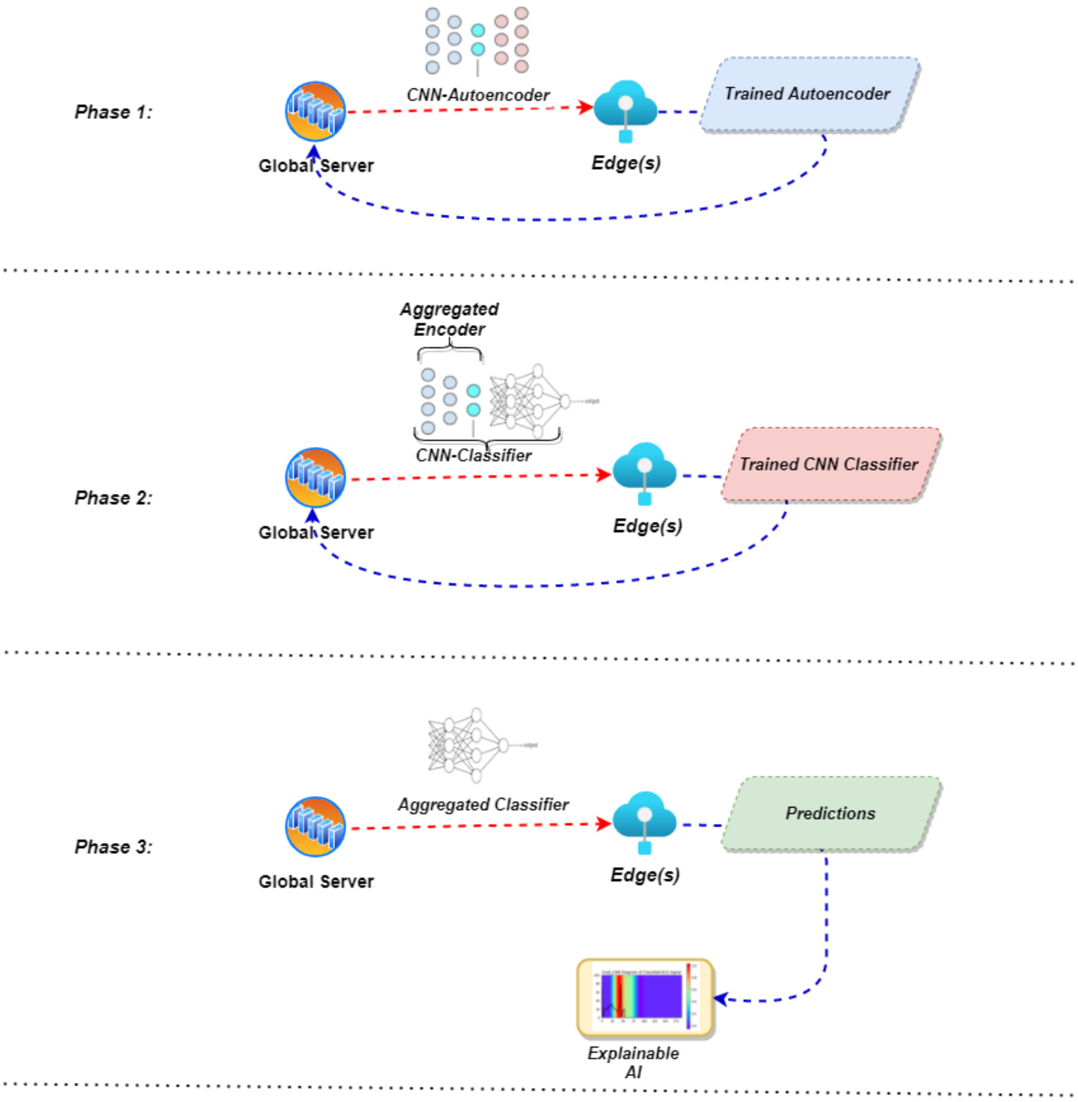


FIGURE 3.1: An overview of the proposed framework

3.2.1 CNN-based Autoencoder

In order to denoise the raw input signal from ECG devices, we proposed an autoencoder. The proposed autoencoder is shown in Figure 3.2. It consists of an input layer, an output layer, and 12 hidden layers. Among the hidden layers, there are 6 convolutional layers, 3 maxpooling layers, and 3 upsampling layers. Furthermore, the CNN-autoencoder is virtually divided into two parts: Encoder and Decoder. The encoder consists of the input layer, 3 maxpooling layers, and 3 convolutional layers in an alternate fashion. On the other hand, the decoder consists of 3 upsampling layers, 3 convolutional layers, and a convolutional output layer. In the proposed autoencoder, we use a varying learning rate lr to keep the training process efficient while keeping the reconstruction loss L as small as possible i.e., if the lr is too small the model converges very slowly, and if the lr is too large the model struggles to converges near global minima. Hence the evolving lr enables big steps at earlier epochs and after a certain epoch the lr rates become small which helps the model converge to global minima. Equation (3.1) gives the mathematical representation of the learning rate (lr) used.

$$\mathrm{lr} = \begin{cases} 0.01, & \text{if epoch} \leq 40, \\ \mathrm{lr} \times e^{-0.1}, & \text{otherwise.} \end{cases} \tag{3.1}$$

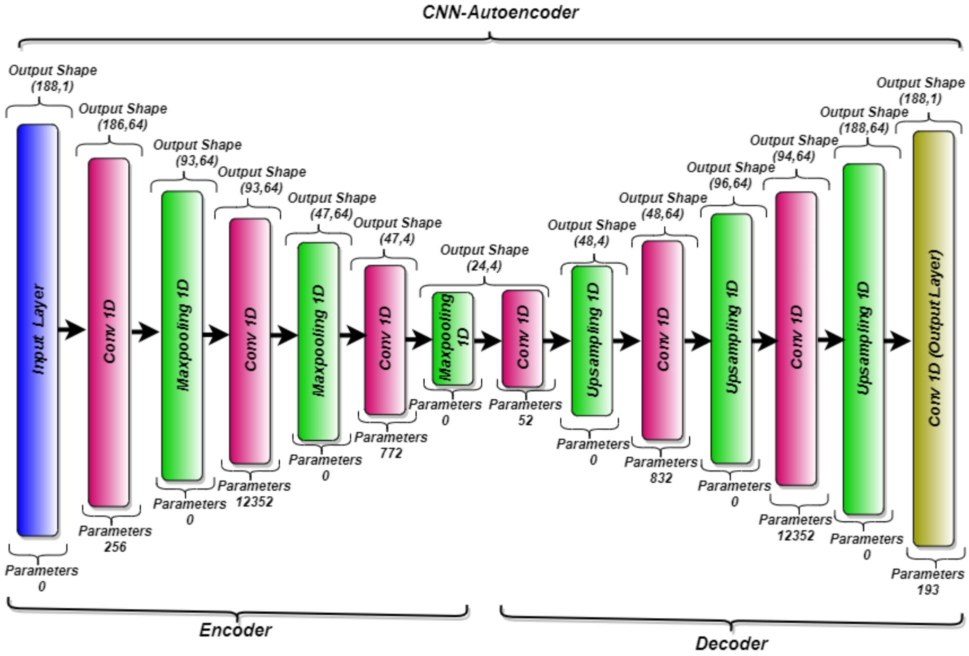


FIGURE 3.2: The architecture of the proposed denoising autoencoder

3.2.2 CNN-based Classifier

The proposed classifier is composed of 4 convolution layers, 3 max pooling layers, 2 fully connected layer,s and 1 softmax layer for classification, as shown in Figure 3.3. The classifier is designed for classifying an input ECG signal into one of the five classes, as shown in Table 3.1. We use transfer learning to transfer the encoder part of the trained autoencoder into the proposed classifier because these convolution layers aim at removing the noise from raw input data and the next layers in the classifier aim to classify the input ECG signal. Hence, the first 3 convolutional layers do not need to be trained while training the individual local classifiers. In other words, we keep the first 3 convolutional layers static during the classifier training phase, which means that no parameters are updated during backpropagation in the first 3 convolutional layers. This provides each local node E_i with the trained parameters for denoising the signal while training the classifier, which increases the performance of the classier. As for the last 2 convolution layers and the fully connected layers, since they are at a higher level, they focus on learning specific features for the classification task. Therefore, we update their parameters during the classifier training phase. The softmax serves as the classification function, and is given by the following equation:

$$y_i = \frac{\exp^{z_c}}{\sum_{c=1}^{C} \exp^{z_c}}, \tag{3.2}$$

where C is the total number of classes, z_c denotes the learned probability for a specific class c, and y_i is the final classification result for a sample i. A Sigmoid can also be used for binary classification which marks one class to 0 if output of sigmoid is <=0.5 and other to 1 if output of sigmoid is >0.5. Our classifier uses categorical cross-entropy (CE) as the loss function. This gives probability over the C classes for each input sample, given by Eq. (3.3).

Where t_c is the ground truth for each class c.

$$\mathrm{CE} = -\sum_{c}^{C} t_c log(y_i) \tag{3.3}$$

TABLE 3.1: The five classes of ECG signals

Class description	Single-letter symbol
Non-ecotic beats (normal beat)	N
Supraventricular ectopic beats	S
Ventricular ectopic beats	V
Fusion Beats	F
Unknown Beats	Q

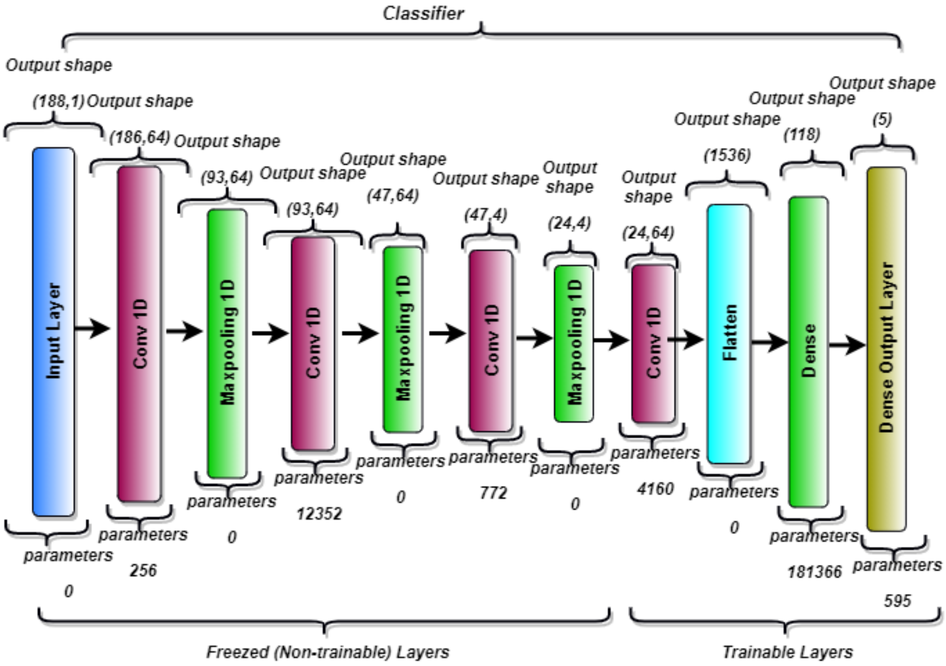


FIGURE 3.3: The proposed CNN-based classifier

3.2.3 XAI with Grad-CAM

As mentioned in 2 Grad-CAM explanations are computationally efficient and are suitable for data such as images and time-series. Hence, inspired by the work in [68] and [158], we decided to use Gradient-weighted Class Activation Mapping (Grad-CAM) and modified it for time series data on top of our classifier, which uses class-specific gradient information to localize important regions. We combine these localized regions with an existing time-series visualization map to create a high-resolution heatmap visualization. Using this visualization, practitioners can understand the reason for a certain prediction given by the classifier. The XAI with GRAD-CAM module is shown in Figure 3.4.

The creation of this heatmap visualization consists of the following steps:

1. In the first step, we compute the gradient of y^c (where y^c is the score for any class c) with respect to the feature map activations A^k for kernel k of the last convolution layer. If G_c represents the gradients for any class c, it can be represented as follow:

   $$G_c = \frac{\partial y^c}{\partial A^k}. \tag{3.4}$$

   Any particular value calculated in this step depends on the input ECG signal (sample input). The weights of the classifier are fixed at this stage. We first reshape an input sample into the batch size and feed it into the classifier, since the input determines the feature maps A_k as well as y^c.

2. The second step consists of global average pooling of the gradients G_c, both along height h and width w to obtain the neuron importance weights α_k^c also called alpha values, given by Eq. (3.5).

   $$\alpha_k^c = \frac{1}{Z}\sum_h\sum_w \frac{\partial y^c}{\partial A^k} \tag{3.5}$$

   These alpha values for class c and feature map k will be used later as a weight applied to the feature map A^k.

3. The third step consists of a weighted linear combination of the feature map activations A^k and α_k^c is calculated using the alpha values, given by Eq. (3.6).

   $$\text{Grad\_CAM}^c = \text{ReLU}(\sum_k \alpha_k^c A^k) \tag{3.6}$$

   This gives us the final Grad-CAM heatmap. A rectifier linear Unit (ReLU) function is applied to emphasize only the positive values and turn all the negative values into 0.

4. The classifier's last convolutional layer's features are quite small, and it is difficult to visualize them for analysis. To address this problem, we upsample the heatmap to the size of the input sample in width. Moreover, we feed the input sample to the autoencoder and receive a denoised version of the input sample, and overlap it on the heatmap. In the resulting heatmap, regions overlapping between the heatmap and the ECG signal show the point of focus during prediction. This gives a detailed picture to the practitioners to understand which region of the ECG input signal the classifier is looking at while making a prediction.

3.2.4 Learning Process

The learning process of the proposed method has been depicted in Figure 3.1. For a clearer explanation, we present the learning procedure in Algorithm 3.1. It should be noted that the algorithm works continuously with new emerging data. Optionally, if an E_i wants to personalize the classifier C, it can be done by keeping all the convolution layers of the final updated classifier static and by training the dense layers for personalization. This is because the convolution layers aim at extracting low-level features and for the densely connected

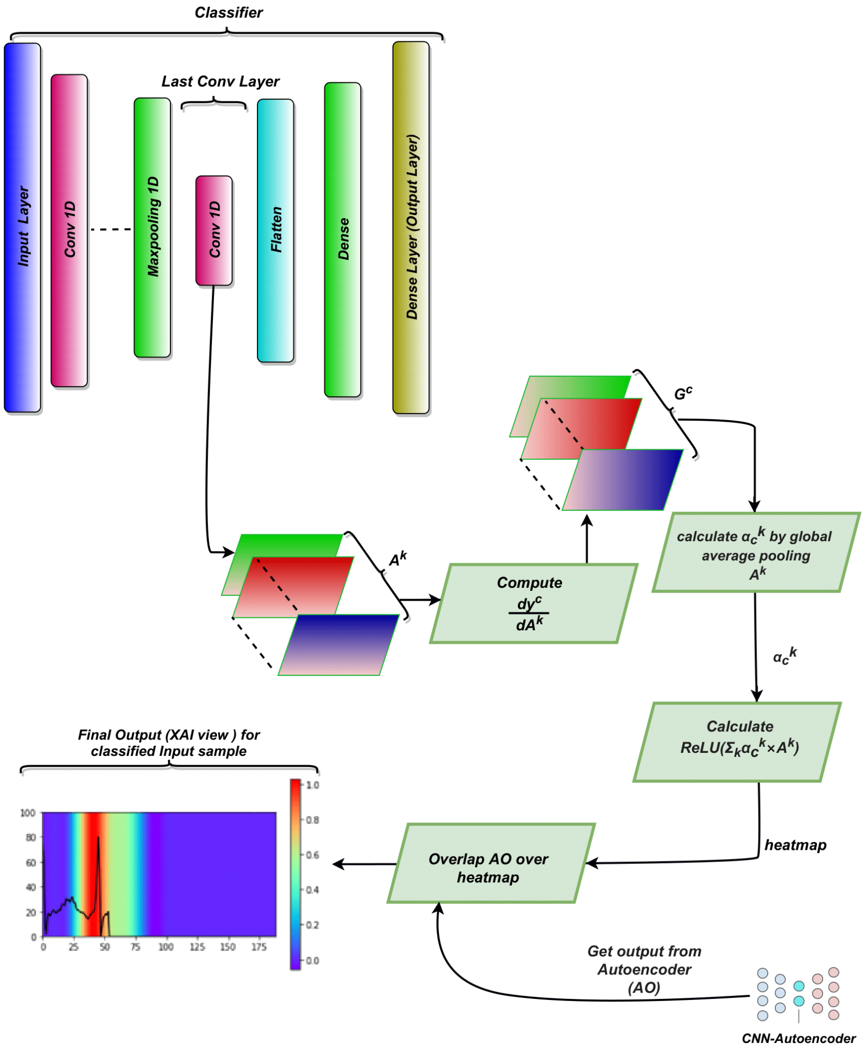

Classifier
Last Conv Layer
Input Layer
Conv 1D
Maxpooling 1D
Conv 1D
Flatten
Dense
Dense Layer (Output Layer)
G^c
A^k
Compute $\frac{dy^c}{dA^k}$
calculate α_c^k by global average pooling A^k
α_c^k
Calculate $ReLU(\Sigma_k \alpha_c^k \times A^k)$
heatmap
Overlap AO over heatmap
Final Output (XAI view) for classified Input sample
100
80
60
40
20
0
0
25
50
75
100
125
150
175
1.0
0.8
0.6
0.4
0.2
0.0
Get output from Autoencoder (AO)
CNN-Autoencoder

FIGURE 3.4: Overview of the proposed XAI module in our framework

layers since they are at a higher level, they focus on learning specific features for the task and the user.

The global server G_s (Aggregation Server) creates an autoencoder AE with predefined hyper-parameters. It should be noted that we use Keras auto-tuner to get the best possible hyper-parameters. Keras auto-tuner empirically tries to find the best possible hyper-parameters. After creating the AE, G_s waits for the client's requests. When clients request G_s, it sends the AE to the client. It is worth noting that, each global round is divided into two tiers, for the first tier G_s sends the AE, and for the second tier G_s sends the classifier C. Hence while requesting, each client mentions the tier as well. On

Algorithm 3.1: Training procedure of Proposed method

```
Input: Data from edge nodes $D_1, D_2, \dots, D_n$
Output: Trained aggregated and updated model
Global Server $G_s$ constructs the initial Global Autoencoder AE and
compiles it using the predefined hyper-parameters
$G_s$ waits for the $E_i$ to request. If the request is received, send AE to
the $E_i$
$E_i$ receives the AE, trains it on its local data $D_i$, and sends trained
weights of AE back to $G_s$
$G_s$, wait for $n$ $E_i$-s to send back their locally train AE.
if weights received form $n$ $E_i$-s then
    $F(w) = \sum_{k=1}^{n} \frac{n_k}{n_t} w_{r+1}^{k}$
$G_s$ constructs a classifier $C$
for For $i = 1, 2, 3$ do
    set Weight of convolutional Layer$_i$ of $C$ = Weight of convolutional
     Layer$_i$ of $F(w)$.
    set convolutional Layer$_i$ of $C$ trainable = False
$G_s$ sends $C$ to $E_i$
$E_i$ trains $C$ on $D_i$ and sends back the trained $C$ to $G_s$.
$G_s$, wait for $n$ $E_i$ to send back their locally trained $C$.
if weights received form $n$ $E_i$-s then
    $F(w) = \sum_{k=1}^{n} \frac{n_k}{n_t} w_{r+1}^{k}$
$G_s$ send $F(w)$ to $E_i$
$E_i$ set $F(w)$ as weight of $C$ and makes predictions
Repeat with continuously emerging data.
```

receiving the autoencoder, client E_i trains the autoencoder on its local data D_i. When the training is completed the client sends back the trained weights of the autoencoder to the global server. The server waits for a fixed number n of clients to send the weights of their locally trained AE. Here, n can be decided by mutual consensus among administrators. When the desired number of clients send their weights and are received by G_s, it aggregates the weights of all the clients by using the formula for aggregation given by Eq. (3.7) from [159].

$$F(w) = \sum_{k=1}^{n} \frac{n_k}{n_t} w_{r+1}^{k}, \text{ where } F_k(w) = \frac{1}{n_k} \sum_{i \in P_k} f_i(w)). \tag{3.7}$$

Here, $F(w)$ are the aggregated weights, n_t is the number of data samples of all participants and n_k is the number of samples of k-th participant. For a

ML problem, typically $f_i(w) = (x_i, y_i; w)$, that is, the loss of the prediction on example x_i, y_i made with model parameters w. There are n clients over which the data is partitioned, with P_k the set of indexes of data points on client k, n is a total number of participants in each round and r is the global round number.

After aggregation, G_s creates a new CNN-based classifier C for classification. Here, again we use the Keras auto-tuner for best hyper-parameters for the newly created C. Furthermore, we use the encoder part of the autoencoder for transfer learning. We transfer the weights of the updated and aggregated encoder part of AE to C and set the transferred layers to static. After this, G_s sends C to each client E_i. Upon receiving C, each E_i trains the classifier using its local data and sends it back to G_s. G_s collects the weight of n clients and aggregates them using Eq. (3.7). After aggregation, it sends the aggregated weights back to each E_i. Clients set the aggregated weights as new weights of their local C, which can be further used for predictions. During predictions, the XAI module taps the gradients and outputs the visual explanation.

3.2.5 Communication Cost Reduction and Privacy Enhancement

In the federated learning setting, training data remain distributed over a large number of clients each with unreliable and relatively slow network connections. For the synchronous protocols in federated learning [160], the total number of communication bits that are required during uplink and downlink communication by each of the N clients during training of the global model is given by:

$$\tau^{\text{up/down}} \in \mathcal{O}(U \times |w| \times (H(\Delta w^{\text{up/down}}) + \tau)), \tag{3.8}$$

where U is the total number of updates by each client, $|w|$ is the model size and $H(\Delta w^{\text{up/down}})$ is the entropy of transmitted weights during communication, τ is the difference between the update size and the minimal update size, given by entropy [161]. Generally, there are three ways to reduce communication costs: (1) reducing the number of clients N, (2) reducing the update size, and (3) reducing the number of updates U. Hence, research on communication-efficient federated learning can be divided into four groups: model compression, client selection, update reduction, and peer-to-peer learning [114]. In order to provide communication-efficient federated learning, we provide a new approach for our proposed architecture called layer selection, which comes under the model compression group. Moreover, layer selection can be added to all of the existing approaches to further reduce communication costs. The proposed layer selection (communication cost reduction) method is shown in Figure 3.5, with more details given below.

Suppose that W1 and W2 represent the weights of all layers of the encoder and decoder of the autoencoder, respectively, trained at edge devices. Since we are only concerned with the encoder part of the autoencoder, the edge devices select the weights of the encoder part (W1) and send them to the global server. The global server aggregates the received weights to obtain the global weights represented as AW1 and sent to the edges. After receiving AW1, the edge devices use transfer learning to transfer these global weights to their local classifier and freeze the transferred layers, as mentioned earlier. The edge devices train the local classifier using their local data. Suppose that WC1 and WC2 represent the weights of the trainable lower (convolutional) and higher (dense) layers of a local classifier, respectively. As the higher layers learn specific features about the underlying data [162], each edge sends only WC1 to the aggregation server that carries common and low-level features about the training data. The aggregation server performs weighted aggregation of all WC1 weights received to obtain AWC1, which are then sent to edge devices. The edge devices use AWC1 along with their individual WC2 for a more localized classification of the ECG. Since we share few weights compared to the classical method, this makes communication lighter and reduces communication costs. Moreover, since the FL framework continuously performs global training with emerging data, our communication cost reduction method can significantly reduce the overall communication costs.

Furthermore, recall that features in deep neural networks are highly transferable in the lower levels of the network since they focus on learning more common and low-level features. As the edge devices only send the weights of lower layers, the privacy of underlying data at each edge is enhanced. To be more precise, the weights of a lower layer, weights of the encoder part in the autoencoder (WC1), contain more common and low-level features about the underlying data, while the weights of higher layers, weights of the decoder part of the autoencoder (WC2), contains more specific features about the underlying data. Hence, by not communicating WC2, we can increase the privacy of the local data by sharing only weights (WC1) that contain more common and low-level (i.e., less private) features.

3.3 Experimental Results

3.3.1 Dataset

For the experimental purpose, we used the widely used MIT-BIH Arrhythmia Database [157] as our baseline dataset. This database contains 48 half-hour excerpts of two-channel ambulatory ECG recordings, obtained from 47 subjects studied by the BIH Arrhythmia Laboratory between 1975 and 1979. The dataset includes 109,446 samples. Twenty-three recordings were chosen at random from a set of 4,000 24-hour ambulatory ECG recordings collected from a mixed population of inpatients (about 60%) and outpatients

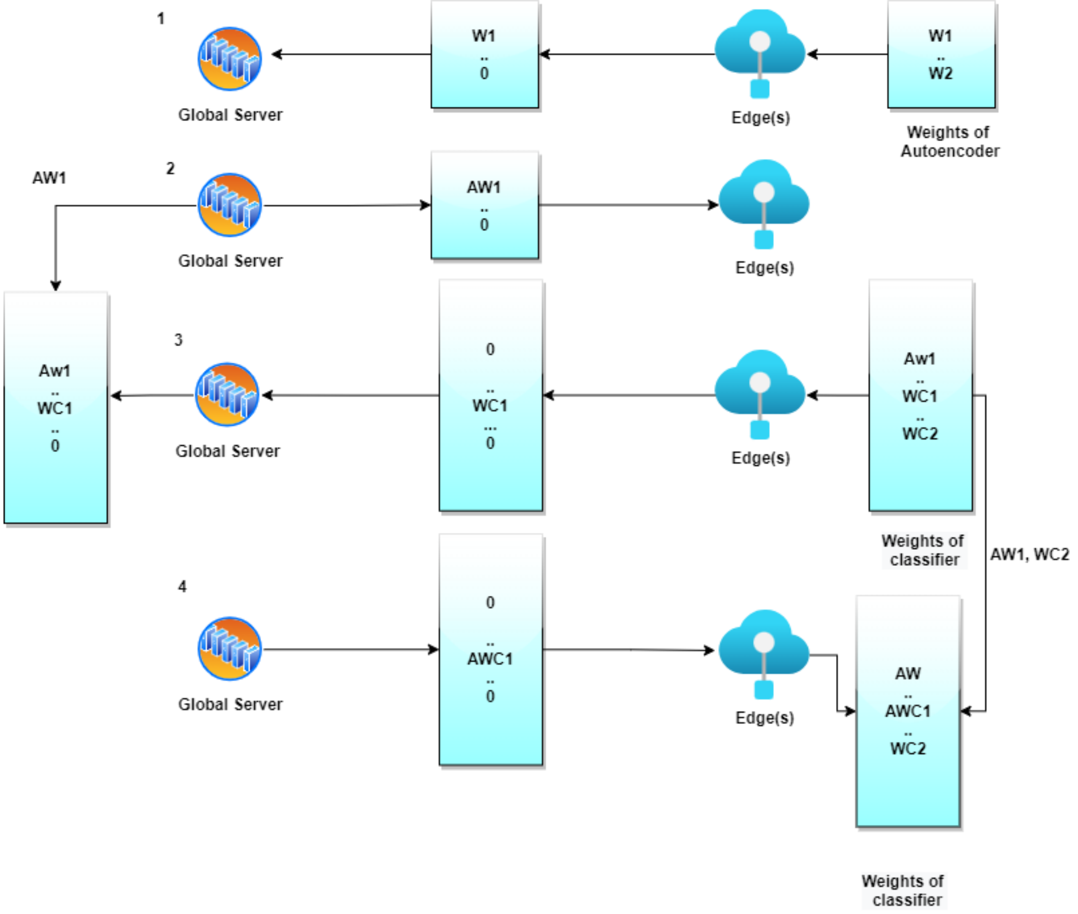


FIGURE 3.5: The layer selection method for communication cost reduction

(about 40%) at Boston's Beth Israel Hospital; the remaining 25 recordings were selected from the same set to include less common but clinically significant arrhythmias that would not be well-represented in a small random sample. In our experiment, we used ECG lead II re-sampled to the sampling frequency of 125 Hz as the input. It should be noted that this dataset has unbalanced classes. Figure 3.6 shows the distribution of the original dataset. This highly unbalanced data can cause problems like overfitting. Hence to balance the classes we used upsampling. The resulting data distribution after upsampling is shown in Figure 3.7. Furthermore, this dataset is highly preprocessed, but in real-world scenarios, the ECG data collected is always noisy. Hence, to simulate more realistic data we introduced 10-30% noise into the original dataset and trained the proposed framework on the noisy version of the dataset, too. A comparison of the original (clean) and noisy datasets are shown in Figure 3.8.

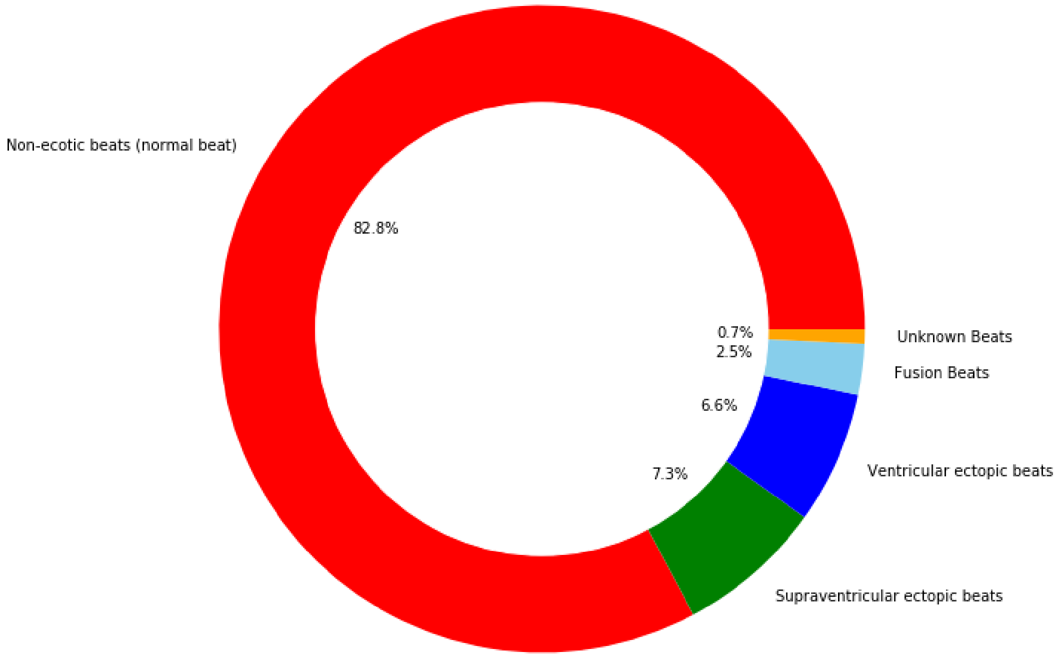


FIGURE 3.6: The distribution of the original dataset

3.3.2 Implementation Details

The Framework was implemented using Python and TensorFlow. Secure socket layer communication was used for communication between the server and edge devices. Both the autoencoder and the classifier were trained locally only on three local Raspberry Pi devices (Pi 3 Model B+ with 1.4GHz, 64-bit quad-core ARMv8 CPU and 1GB LPDDR2 SDRAM), denoted by Edge$_1$, Edge$_2$ and Edge$_3$. Furthermore, a workstation with an Intel core i-6700HQ CPU and 32 GB RAM was used as the global server G_s. It should be noted that FedHealth [117] initially trained their model at G_s, which may cause security risks in the case of a malicious global server. If the models (AE and C) are trained initially on G_s this may cause biased training. Hence,

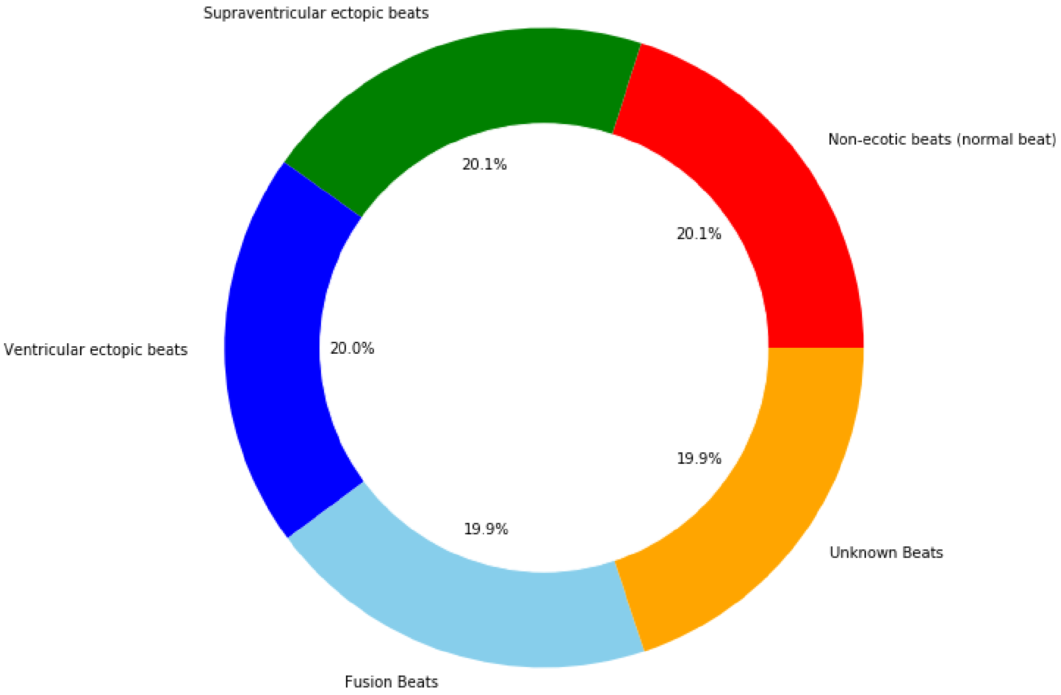
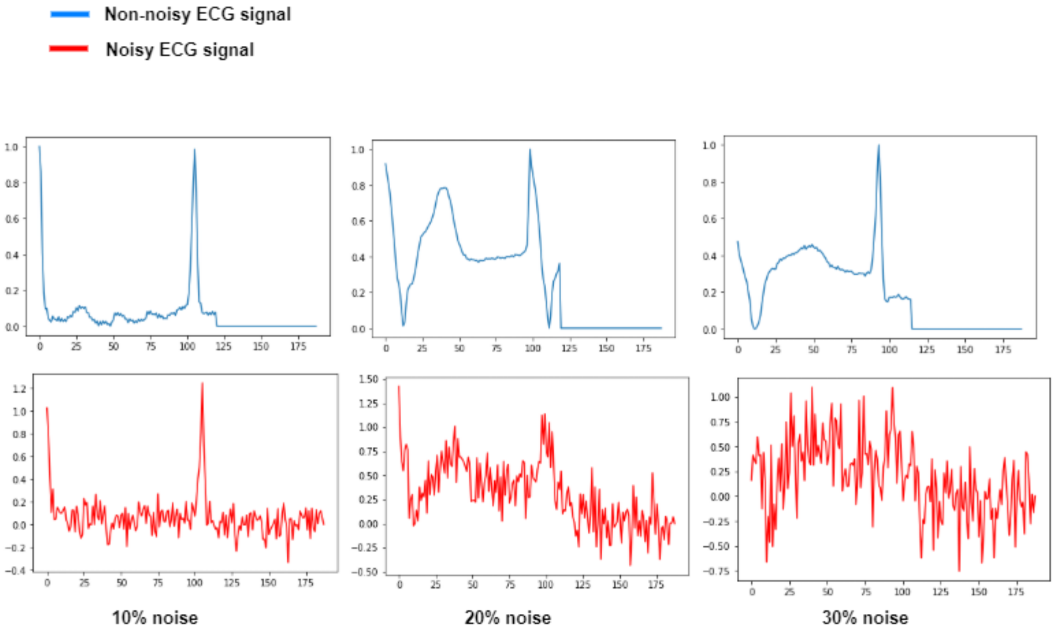


FIGURE 3.7: The distribution of the upsampled (re-balanced) dataset



10% noise 20% noise 30% noise

FIGURE 3.8: Comparison of the original and the noisy version of the dataset

to avoid such risks, we performed only aggregation at the G_s. Furthermore, AE adopted a convolution size of 3. It uses a Root Mean Square Propagation

(RMSProp) as the optimizer. Each E_i device uses 80% of data for training and 20% of data for evaluation. We distributed the dataset randomly at each edge device and introduced random noise. In this case, the data in Edge_1 contains 20% random noise, the data in Edge_2 contains 30% random noise, the data in Edge_3 contains 10% random noise. Furthermore, each edge used a fixed batch size of 100 and was trained for 50 training epochs. Moreover, each edge used an evolving learning rate, given by Eq. (3.1).

The classifier C used a batch size of 100. The learning rate was set to 0.001 with 150 training epochs. The accuracy of each of the locally trained C was calculated by using the following equation:

$$A^i_{cc} = \frac{|x : x \in D_i \wedge y'(x) = y(x)|}{|x : x \in D_i|}. \tag{3.9}$$

In regards to the execution time, given the above setting, it took an average of 745 seconds to complete one global round of training. Furthermore, it took an average of 2.32 seconds to generate a prediction and XAI results.

3.4 Performance Analysis of the Proposed Method

In this section, we analyze the performance of the proposed framework using some state-of-the-art metrics.

3.4.1 Reconstruction of Autoencoder

We introduced noise into the dataset and used the noisy sample as the input in the autoencoder and the cleaned samples as labels. The performance of the autoencoder was measured using reconstruction Mean Absolute Error (MAE). Reconstruction MAE for each locally trained AE in each of Edge_1, Edge_2, Edge_3 and aggregated AE is given in Figure 3.9. It can be seen that the reconstruction MAE of the aggregated autoencoder is nearly 0, which means that our autoencoder reconstructed the original signal very well. Moreover, it can be seen that the reconstruction MAE aggregation AE is less than or nearly equal to the reconstruction MAE of each locally trained AE.

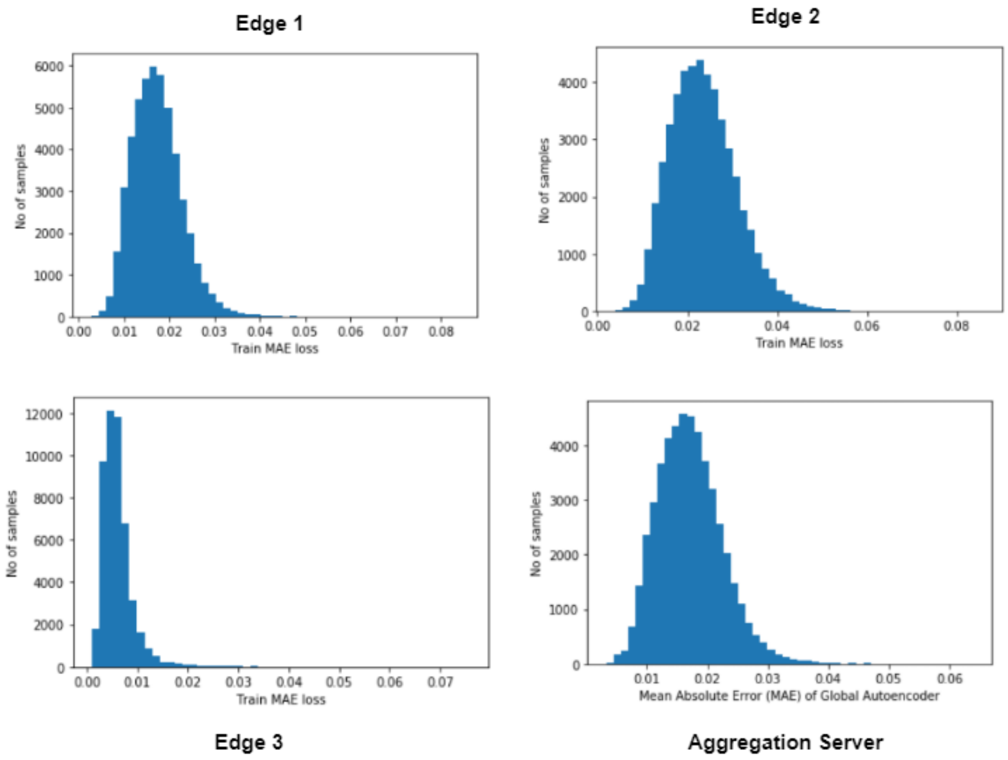


FIGURE 3.9: Reconstruction MAE

3.4.2 Classification Performance

Classification performance was measured using the performance metrics mentioned in Chapter 2.1.2. Precision, recall, F1-score metrics of each binary classifier (one for each of five class labels) at the three edge devices (Edge$_1$, Edge$_2$, and Edge$_3$) and the global server are given in Table 3.2. We also show the accuracy of the five-class classifier in the last column. It should be noted that the results shown in Table 3.2 are computed using the noisy data which we prepared earlier. We also tested the proposed classifier using the original (clean) data. With this data, it provided 98 ± 0.9% accuracy. Other metrics, such as precision, recall, and F1-score, are shown in Table 3.3. However, for

TABLE 3.2: The classification performance of the proposed framework, with the noisy version of the dataset

Class	*Precision*	*Recall*	*F1-Score*	*Accuracy*
N	89%	91%	90%	94.9%
S	94%	89%	92%	
V	93%	96%	94%	
F	95%	94%	95%	
Q	99%	99%	99%	

(A) Edge 1 (20% noise)

Class	*Precision*	*Recall*	*F1-Score*	*Accuracy*
N	85%	87%	86%	91.9%
S	91%	87%	88%	
V	91%	94%	92%	
F	93%	93%	93%	
Q	98%	98%	98%	

(B) Edge 2 (30% noise)

Class	*Precision*	*Recall*	*F1-Score*	*Accuracy*
N	94%	98%	96%	97.9%
S	98%	92%	95%	
V	95%	99%	97%	
F	99%	94%	96%	
Q	97%	100%	98%	

(C) Edge 3 (10% noise)

Class	*Precision*	*Recall*	*F1-Score*	*Accuracy*
N	90%	92%	91%	94.5%
S	94%	89%	91%	
V	93%	96%	94%	
F	96%	96%	95%	
Q	99%	99%	99%	

(D) Global/Aggregation Server

real-time use we expect the data to be noisy, which is why we proceeded with the noisy data.

TABLE 3.3: The classification performance of the proposed framework, with the original (clean) dataset

Class	*Precision*	*Recall*	*F1-Score*	*Accuracy*
N	95%	99%	97%	98.9%
S	98%	97%	98%	
V	97%	99%	98%	
F	99%	93%	96%	
Q	100%	100%	100%	

3.4.3 Qualitative Analysis

Understanding the reasons for predictions of the model decision is very important in healthcare applications. In order to validate that the decisions made by the proposed XAI module are interpretable, we use visualization to demonstrate that clinically important beats in the ECG wave are used for classification. Figure 3.11 illustrates the importance of each beat that the ECG classifier is giving while performing classification of some instance ECG signal inputs.

In order to achieve the explanations/explainability of the XAI module, it is important to understand the ECG signal [163]. Generally, the amplitude and width of the p-wave, QRS complex, and the T-wave are important features of an ECG graph, as shown in Figure 3.10. These regions play a vital role in ECG analysis [164]. The XAI module in the proposed framework shows that the proposed classifier looks at these critical features of the input sample. In Figure 3.11, the red segments show more important regions of the heartbeat for the network's decision while predicting a particular class. In other words, the red segments of the heartbeat have more influence on the detection process of the classifier while classifying the input ECG signal. These results can be used to help clinical practitioners to diagnose the underlying health issues. However, we strongly advise that these results should not be used for any medical consultation without prior discussion with a clinical professional. In other words, heat maps should be cross-checked by clinicians with prior expert knowledge.

3.4.4 Comparison With Other State-of-the-Art Methods

We compared our proposed framework with the state-of-the-art methods reported in 2020 [67], [86], [87], [117]–[122]. First, we compare the previous

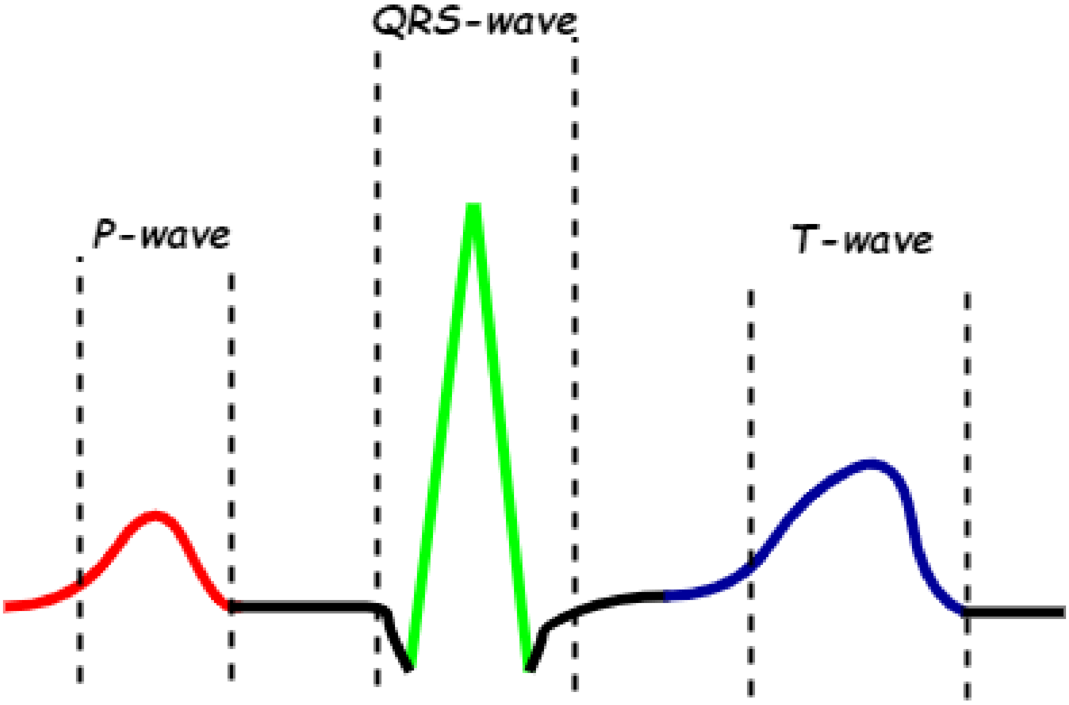


FIGURE 3.10: The major waves of a single normal ECG pattern

work with ours to show that the proposed framework provides all of the desirable properties like explanations, privacy-preserving, and working with raw data. Table 3.4 shows the comparison between our proposed method and the other methods. It can be seen that the proposed scheme outperforms others by providing all desirable properties, while others lack some of the desirable properties. Moreover, we also compare our work with existing works for ECG classification. It is to be observed that other methods used the baseline MITBIH dataset (without noise), with which better accuracy results can be achieved. Contrastingly, we introduced (10%-30%) noise into the data to make it more realistic and to show that it is robust to varying noise across different edge/client devices. Table 3.5 shows the comparison of classification performance between our proposed method and the other methods. It can be seen that our proposed method outperformed the methods in others. Observe that the proposed classifier deals with the classification tasks of five classes, while others deal with fewer classes (some with two and some with three). Additionally, the proposed method works in federated architecture

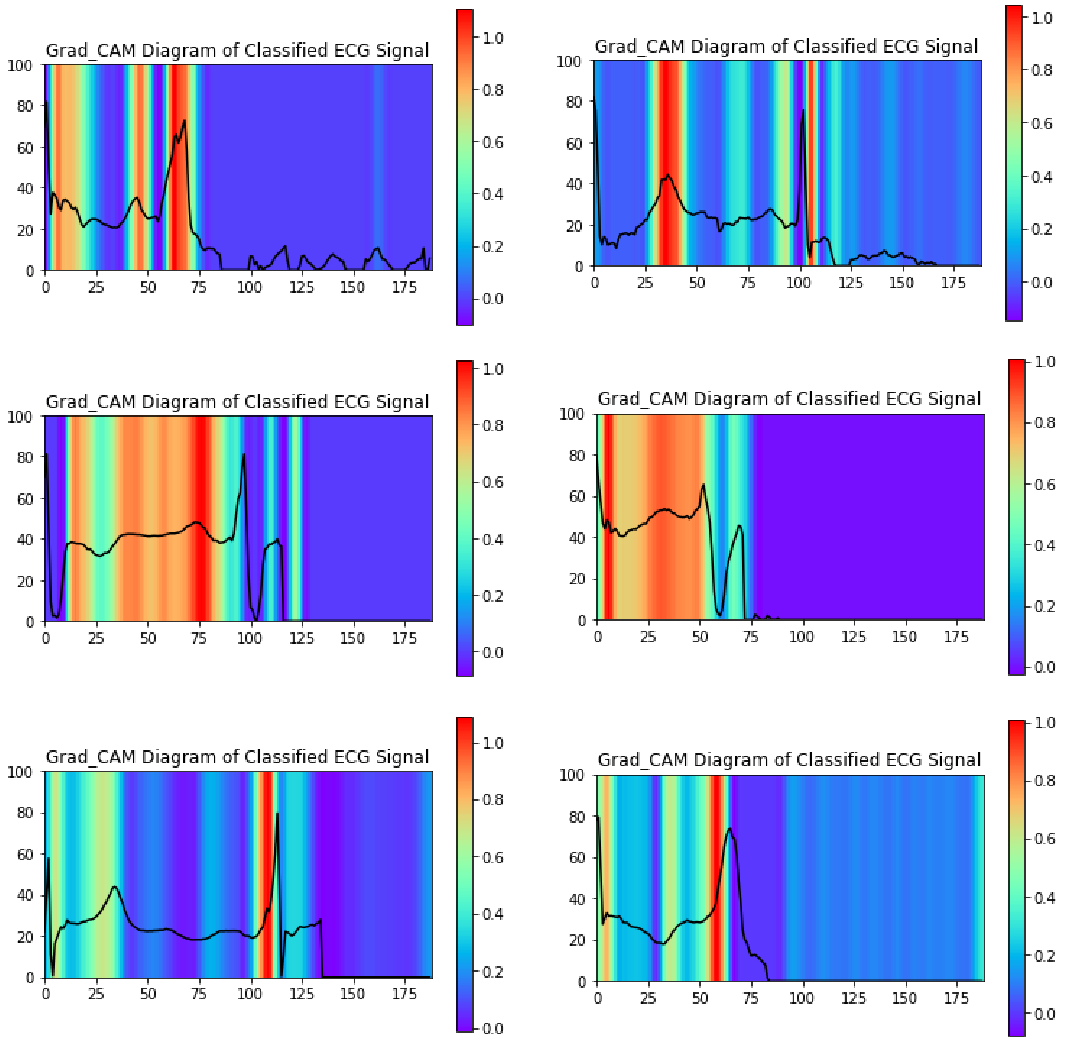


FIGURE 3.11: The outputs of the XAI module

and performs better compared to others. Our proposed method provides explainability as an additional feature. Moreover, the proposed method provides enhanced data privacy to the users via the federated setting, which is not the case for other methods. Furthermore, the proposed method provides robustness to varying noise by denoising raw signals without any further preprocessing, followed by classification and explainability.

3.4.5 Privacy Enhancement

As mentioned earlier, digital healthcare data is like digital fingerprints that carry a lot of personal information. Hence, we should protect such data as

TABLE 3.4: Comparison with the state-of-the-art work

Scheme	explanations	Raw Input	Privacy Preserving
[154]	✗	✗	✗
[155]	✗	✗	✗
[165]	✗	✗	✗
[117]	✗	✗	✓
[118]	✗	✗	✗
[87]	✗	✗	✗
[119]	✗	✗	✗
[86]	✗	✗	✗
[120]	✗	✗	✗
[121]	✗	✗	✗
[67]	✓	✗	✗
[122]	✗	✗	✗
Proposed	✓	✓	✓

much as possible while using them in ML algorithms. Most past studies on ECG classification do not provide privacy protection for such data because they are centralized and data are shared with the central model directly. Recently, Chen et al. [117] used federated learning to provide privacy protection, by only sharing the learned parameters without sharing the data. Although the shared parameters can protect privacy, there are still chances to recover some information from the shared parameters of higher-level layers in the classifier, since they can contain more data-specific information as discussed previously. As a comparison, in our proposed framework we only share the learned parameters from lower-level layers that carry only more common and low-level (i.e., less privacy-sensitive) features. Thus, our proposed framework can enhance privacy even further, and at the same time can reduce communication costs as fewer parameters are shared between the edge/local and global servers.

A comparison between existing work in federated settings for healthcare is shown in Table 3.6.

TABLE 3.5: Comparison with previous studies for ECG classification

Scheme	Centralized or Federated	Acc (clean data)	Acc (noisy data)
[154]	Centralized	86.0%	-
[155]	Centralized	96.9%	-
[118]	Centralized	98.1%	-
[87]	Centralized	96.5%	-
[86]	Centralized	93.1%	-
[119]	Centralized	98.7%	-
[120]	Centralized	94.9%	-
[121]	Centralized	98.1%	-
[166]	Centralized	98.6%	-
[165]	Centralized	98.3%	-
[67]	Centralized	98.8%	-
Proposed	Federated	98.9%	94.5%

3.4.6 Communication Cost Reduction

Here, we show the communication cost reduction using the proposed communication cost reduction method. The number of total parameters communicated between an edge device and the global server, for one global round, denoted by TPC, is given as follow:

$$\mathrm{TPC} = \mathrm{W1} + \mathrm{W2} + \mathrm{WC1} + \mathrm{WC2} + \mathrm{AW1} + \mathrm{AW2} + \mathrm{AWC1} + \mathrm{AWC2}, \quad (3.10)$$

In the proposed framework, the TPC is given as follow:

$$\begin{aligned}\mathrm{TPC} = 13386 + 13429 + 4160 + 181961 + 13386 + 13429 + 4160 + 181961 \\ = 425872, \quad (3.11)\end{aligned}$$

With the proposed communication cost reduction method, the TPC is given as follows:

$$\mathrm{TPC} = \mathrm{W1} + +\mathrm{WC1} + \mathrm{AW1} + +\mathrm{AWC1}, \quad (3.12)$$

In the proposed framework, the TPC is given as follows:

$$\text{TPC} = 13386 + 4160 + 13386 + 4160 = 35092, \tag{3.13}$$

From Eqs. (3.11) and (3.13), we can calculate that the proposed communication cost reduction method reduces the communication cost by 8.2%.

TABLE 3.6: Comparison with the state-of-the-art work in federated setting for healthcare

Scheme	Communication Cost Reduction	Privacy Enhancement
[117]	✗	✗
Proposed	✓	✓

3.4.7 Time Complexity of proposed Algorithm

In this section, we provide the time complexity of the proposed Algorithm 3.1. In a CNN-based network, the number of features in each feature map is at most a constant times the number of input features let us say n (typically the constant is < 1). Convolving a fixed-size filter across an input signal with n features takes $O(n)$ time, since each output is just the sum product between some features let's say k in the input, and a fixed number of weights w in the filter, and w and k do not vary with n. Similarly, any max or average pooling operation does not take more than a linear amount of time in the input size. Moreover, the edge node can compute in parallel, therefore, the overall run time is still linear, i.e., $O(n)$.

3.5 Summary

In this chapter, we proposed a privacy-preserving, efficient, and explainable AI-based end-to-end framework to address the limitations of DL-based classifications. Firstly, we proposed a CNN-based autoencoder in a federated architecture to denoise the raw ECG signal from patients to achieve robustness against varying noise present in the local data of edge/clients. When trained on the baseline dataset, The proposed autoencoder provided an excellent reconstruction of the raw input signals and improved the overall performance when applied in federated settings. Secondly, we proposed a new classifier for ECG classification. When the classifier was trained in federated settings it was able to improve the overall classification performance of the edge devices. Moreover, the experimental results on the baseline database revealed that the proposed framework outperformed existing algorithms, including both centralized and federated ones. Furthermore, we extended the usability of our framework by providing a novel explainable module on top of the classifier, whose usefulness is visually demonstrated by showing that clinically meaningful heartbeat segments of ECG signals are indeed behind the classification results. Additionally, we also proposed a communication cost reduction method, which can significantly reduce communication costs while increasing the level of privacy protection of users' ECG data against the global server. Hence, the proposed framework shows its applicability by providing many desirable properties including robustness, explanations, privacy protection, communication cost reduction, and high accuracy in classification. Such a combination of such properties does not hold for other existing solutions, therefore making the proposed framework a unique solution for real-world healthcare applications.

Eventually, the proposed framework will encourage (1) more healthcare data owners to participate in training a good ML model for patients and health professionals, with fewer privacy concerns, (2) more accurate diagnostic assistance in places with scarce access to cardiologists or healthcare facilities, (3) more explainable classification results that can be used to identify new potential patterns leading to trigger heart arrhythmias. Hence, the proposed framework has great potential to be added to hospital computer software platforms to support the work of health professionals and ultimately reduce mortality and save human lives.

As a future research direction, we aim at applying the proposed framework to more healthcare applications, especially HAR and anomaly detection in the context of home care, and other types of arrhythmia to extract new patterns that might be helpful for their diagnosis and monitoring. However, as discussed in chapter 2 RNNs are computationally slow and CNNs can be computationally expensive when we use them to capture long-term dependencies, such as anomaly detection and HAR which requires monitoring long-term dependencies and even higher dimensional data. Hence, in the next chapter, we will introduce privacy-friendly transformers-based classification in federated settings which can achieve the best of both CNNs and RNNs.

Moreover, the Fl architecture is still vulnerable to adversarial attacks as discussed in Chapter 1 and 2. Such attacks are also applicable to upcoming Chapter 4, and Chapter 5. Hence, we will address adversarial attacks such as poisoning attacks in Chapter 6.

Chapter 4

Lightweight Transformer-based Classification in Federated Setting for Home Care

4.1 Introduction

As discussed previously, researchers have made substantial progress on high accuracy for DL applications. For example, HAR classifiers [92], [167], [168]. Yao et al. [92] proposed a deep learning model based on data collected from accelerometer and gyroscope sensors on mobile devices, which involves a hybrid model combining a convolutional neural network (CNN) and a recurrent neural network (RNN). However, training such models on real-world data collected from smart devices leads to two major challenges. However, as argued previously deep learning requires a large amount of data for training [92], which will incur communications between the centralized server and clients (data owners). Second, collecting data from a home environment can raise privacy concerns since a lot of the sensor data include or can infer personal and sensitive data about people in the home [169], [170]. FL helps improve the privacy of local data, reduces unnecessary communications between the global server and the clients, and meets the business needs of data owners and local healthcare providers who would not share their data. Therefore, many researchers have considered using federated learning for HAR and other healthcare applications [114].

In regards to performance, commonly used deep learning techniques including CNNs and RNNs have their limitations. CNNs have an advantage over RNNs (including LSTMs) as they are easy to parallelize, while RNNs have recurrent connections and hence parallelizaion cannot be easily achieved. However, in long-term sequences like time series, capturing the dependencies can be cumbersome and unpractical using CNNs [171]. To address these challenges, transformers have been introduced recently [58]. The transformer technique is an attempt to capture the best of both worlds (CNNs and RNNs). They can model dependencies over the whole range of a sequence and there are no recurrent connections, so the whole model can be computed in a very efficient feedforward fashion. Since its introduction, transformers have been widely studied in various applications, for example, in natural language processing [172] and healthcare [173]. However, the application of transformers for home care, such as HAR needs to be explored, and developing transformers in a federated setting can potentially boost the HAR, as it enhances privacy which is much desired. Nevertheless, there are open questions regarding the implementation details such as what are the limits of the algorithm and if transformers really perform well in HAR, coupled with federated learning. Transformers are generally computationally expensive and to support FL the development of the lightweight transformers-based model(s) is challenging.

In this chapter, to answer all these questions, first, we develop a novel lightweight transformer for example time-series applications, and show that it can provide high performance in terms of accuracy and computational cost compared to existing deep models such as RNNs and CNNs. We trained and test the proposed base-line transformer using different open-source sensor data. To show its applicability to real-time data, we collect data using a prototype that we developed to collect human activity data using three different types of sensors: accelerometer, gyroscope, and magnetometer. While collecting the data we tested different sensor locations on the human body: hip, chest, and upper arm (further details about the prototype and data collected will be provided in later sections). Furthermore, to address the aforementioned challenges, such as privacy concerns and additional computational costs, we propose a novel transformer framework in a federated setting called TransFed, the first transformer-based classifier for HAR in federated settings. Moreover, we evaluate the performance of federated learning using the proposed transformer and showed that federated learning can be used instead of centralized learning for time-series classification. We compare the proposed transformer-based classifier under two training settings, using centralized learning and using federated learning. In federated learning, we use skewed data based on a non-identical independent distributed (non-IID) data distribution among clients, and in the centralized setting, we use the whole dataset which contains only slightly unbalanced classes. With experimental results, we show that the proposed TransFed outperformed existing state-of-the-art methods.

4.1.1 Contributions

The main contributions of the chapter are as follows:

1. We proposed a novel lightweight (in terms of the number of trainable parameters required) transformer for classification. We show that the proposed transformer outperformed other state-of-the-art HAR classification methods based on CNNs and RNNs when trained and tested

on a public dataset as well as a dataset we constructed, for example, a healthcare application i.e., HAR.

2. In order to address challenges related to privacy and computation costs, we introduce TransFed, the first framework for classification based on federated learning and transformers.

3. We designed a prototype to collect human activity data using three different types of body sensors: accelerometer, gyroscope, and magnetometer. We also tested different locations of each type of sensor on the human body to find the points of maximum impulse (PMIs) and evaluated the performance of the data for each location. We then constructed a new dataset for evaluating the proposed classifier.

4. We evaluated the performance of TransFed using non-IID data. Based on the data distribution we analyze how the non-IID data of clients can affect the performance of TransFed.

The rest of the chapter is organized as follows. Section 4.2 explains our proposed transformer and the federated learning framework TransFed in detail. The experimental setup and the experimental results on performance analysis are given in Sections 4.3 and 4.4, respectively. The last section summarizes the chapter.

4.2 Proposed Methods

In this section, we explain our proposed methods, including the proposed lightweight transformer and the FL-based framework TransFed for addressing privacy concerns in a federated setting. For the proposed transformer model, we first describe the model itself and then move on to explain two data formats we tested for the proposed transformer model. Both the model itself and the data formats tailored for the model help improve the transformer's performance in terms of computational complexity and classification accuracy. The following three subsections will introduce the transformer model, the data formats used, and the proposed TransFed framework, respectively.

4.2.1 Proposed Lightweight Transformer Model

We designed a novel lightweight transformer, as shown in Figure 4.1. From bottom to top, the first layer inputs the raw human activity data after preprocessing it into certain sized window (as discussed earlier). The input is then passed through the transformer layer, which extracts discriminative features from the input data. Finally, the output of the transformer is fed into a prediction layer for the classification or final output.

Unlike the traditional transformer model, we use a single patch encoding in our model. This is because we found out that, we can get state-of-the-art results even without using multiple patches, which helps make our proposed transformer simpler and therefore more computationally efficient. The proposed transformer is virtually divided into two parts: the transformer layer and the prediction layer. Both layers are composed of many sub-layers. The transformer layer starts with an augmentation sub-layer, which is used to increase the diversity of the training set by applying random (but realistic) transformations. The output of the augmentation sub-layer is fed into a normalization sub-layer to normalize the data. Following the normalization sub-layer, a multi-head attention sub-layer applies the self-attention mechanism to the input data, which is the fundamental mechanism of a transformer. The self-attention mechanism is a sequence-to-sequence operation: a sequence of vectors goes in, and a sequence of vectors comes out. Let us call the input vectors $X_1, X_2, \dots, X_l$ and the corresponding output vectors $Y_1, Y_2, \dots, Y_l$. To produce the vector Y_i, the self-attention operation applies weights averaged over the input vectors as follows:

$$y_i = \sum_j w_{ij} X_j, \tag{4.1}$$

where j indexes over the whole sequence and the sum of the weights to one overall j-s. The weight w_{ij} is not a parameter as in a normal neural network but is derived from a function over X_i and X_j. To make this operation more lightweight, we use the dot product:

$$w_{ij} = X_i^T X_j. \tag{4.2}$$

Since the dot product gives a value between $-\infty$ and $+\infty$, we apply a softmax to map the values to the range of 0 and 1, and to ensure that they sum to 1 over the whole sequence:

$$w_{ij} = \frac{\exp \mathrm{w}_{ij}}{\sum_j \exp \mathrm{w}_{ij}}. \tag{4.3}$$

The self-attention operation defines the correlation among the input features with respect to the learning task. The output of the multi-head attention sub-layer is then added with the output of the previous normalization sub-layer and fed the result into a succeeding normalization sub-layer, which is then passed through a dense sub-layer. The dense sub-layer applies a non-linear transformation for further feature extraction, given as:

$$\text{Output} = \text{activation}(\text{dot}(\text{input}, \text{kernel})), \tag{4.4}$$

where "activation" is the element-wise activation function passed as the activation argument, "kernel" is a weight matrix, and "dot" is the dot product. The output of the dense sub-layer is added with the output of the previous addition sub-layer. The transformer layer can be applied multiple times. The output from the transformer layer is fed into a flattening and dense output sub-layer with softmax activation, which gives the final classification probability distribution over the pre-defined classes.

4.2.2 Data Formats for Proposed Transformer Model

Since the input data format plays an important role in the classification and computational efficiency of the proposed transformer model, we experimented with two possible formats of the input data as explained below.

Image-based Format

Transformers introduced for computer vision tasks work with 2-D images by splitting an image into a vector of small sub-images (patches). This vector is used as the input. In order to follow the convention, we created an image of size $N \times M$ from input samples of activity in a time frame of 2 seconds, as a time frame of 2 seconds can give enough information about the activity, as shown in Figure 4.2. After this, patches of size $n \times n$ were created, where $N, M > n$. We tried images and patches of different sizes. Unfortunately, when we trained and tested the proposed transformer, using the above-mentioned data format, the results were not good enough. This is due to broken the natural boundary between consecutive 1-D samples into 2-D images, which makes it hard for the transformer to capture all features effectively by suppressing all random noise in the images because no 2-D filter of a reasonable size would cover distant pixels that are neighboring samples in the original 1-D signal. We observed that with the increase in size (the number of transformer layers), the classification performance was improved but this setting is not suitable for edge devices with a relatively low computation power. Nevertheless, this approach can be utilized in settings where

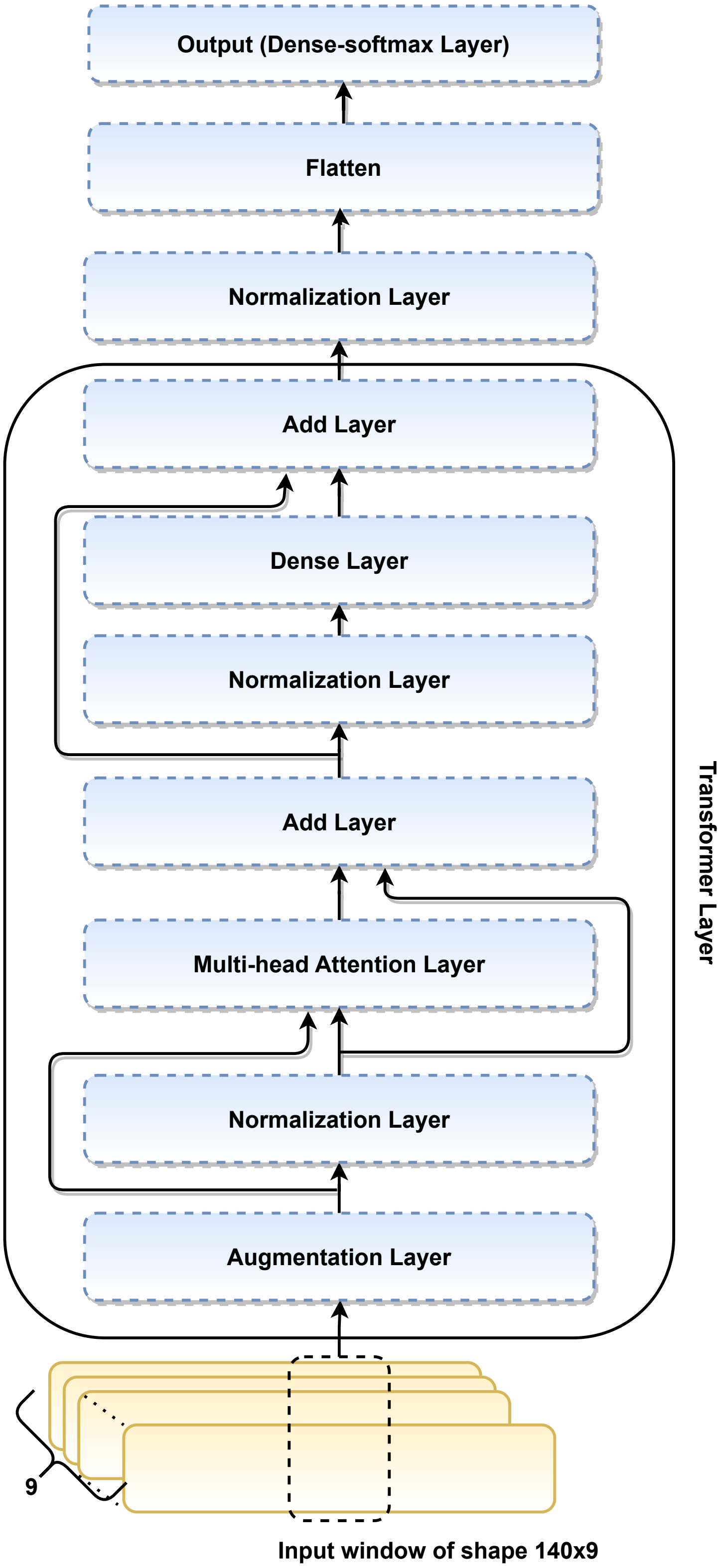


FIGURE 4.1: The proposed transformer model.

the computational power is high enough. Hence, we decided not to use the image-based format.

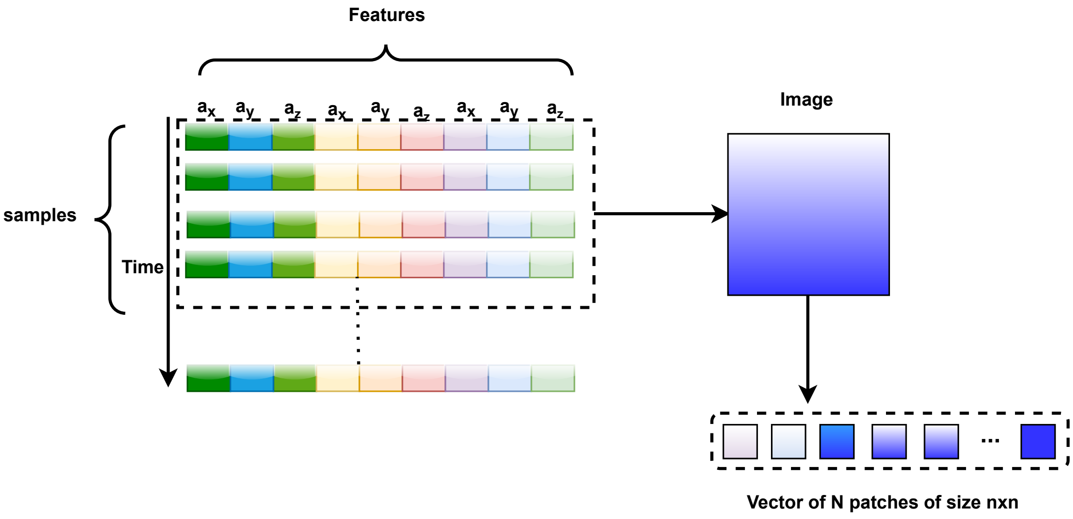

Features
a_x a_y a_z a_x a_y a_z a_x a_y a_z
samples
Time
Image
...
Vector of N patches of size nxn

FIGURE 4.2: Image-based format for the proposed transformer model.

Averaged-Window Format

In this approach, we propose to reshape the raw input data into a fixed sliding window of shape $W \times F$, where W is the number of samples in a window and F is the number of features. In our experiments, we tried to change W as well as the number of features. Since a human activity is an action that can be performed in a certain window of time, to capture a specific activity we need to optimize the window size so that the model can map the information contained in that window to a specific class. For example, going upstairs is an activity that can be recognized within a suitable window size such as 2 or 3 seconds. Hence, W (the window size) is very critical in HAR because of the following trade-off: if W is too small the classifier cannot distinguish between activities; and if W is too large the classifier will require more data and more computational resources. Let us take going upstairs as an example. If W is below one second, it would not be suitable to recognize the activity as no human being can finish going upstairs within just one second (for HAR tasks, a single temporal point is too small to be informative). In regards to F, we noticed that when there are a large number of candidate classes, a large F significantly improves the classification performance. While for a small number of classes, a small (such as 6) or large (such as 9) value of F gives almost the same classification performance. Moreover, we found that combining tri-axial data from the accelerometer and gyroscope provides a significant improvement over combining accelerometer and magnetometer data and combining magnetometer and gyroscope data. After trying many different values of the window size with grid search, we found that a window size of 140 × 9 provided optimized results for all human activities we considered when both accuracy and computational costs are taken into account. To summarize, we take samples of the input over a time frame of 2 seconds with 9 features and average them feature-wise to generate a new sample as shown in Figure 4.3. This format can work because averaging cancels the random noise.

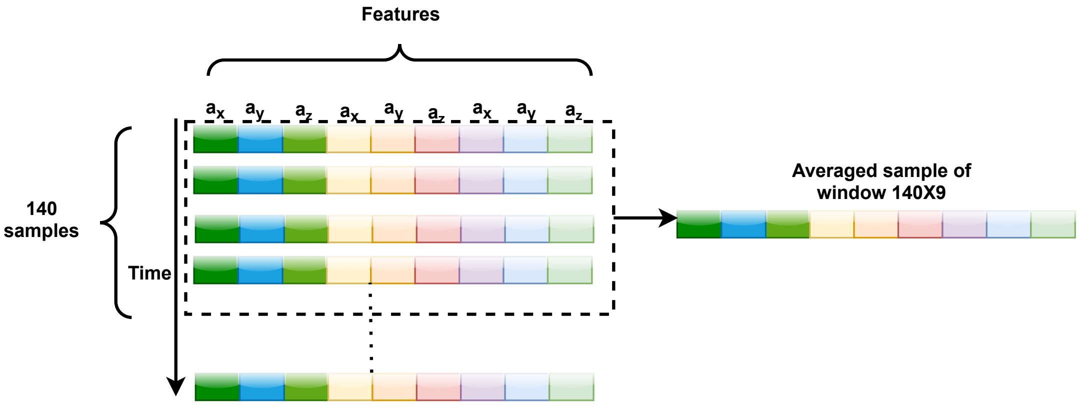

Features
a_x a_y a_z a_x a_y a_z a_x a_y a_z
140 samples
Time
Averaged sample of window 140X9

FIGURE 4.3: Averaged-window format for the proposed transformer model.

4.2.3 The Proposed FL Framework TransFed

Figure 4.4 shows the basic flowchart of the proposed TransFed. In our method, the federated setting is adopted in order to facilitate collaborative learning while preserving the privacy of the underlying data. In the federated setting, a central (global) server sends the compiled architecture of the model (which is a transformer in our case) to all edge or client devices. Each device trains its transformer locally using its local data. After all the local transformers are trained, each edge device sends the trained parameters of its transformer to the global server, which are then aggregated by the global server to construct the global model without a training process. After the global model is available, each edge device downloads the aggregated parameters of the global model and updates its local model according to its local needs. There are two expected key advantages of the federated setting, (i) it increases the overall accuracy, generalization, and robustness of the model, and (ii) it provides better privacy protection to the data owners. Algorithm 4.1 defines the workflow of the proposed TransFed framework.

4.3 Experimental Setup

In this section, we discuss the experimental setup that we designed to test the performance of the proposed transformer in federated setting using real-time data. We first discuss a testbed that we designed to construct a new dataset and to support the experiments for evaluating the performance of the proposed TransFed framework.

4.3.1 Testbed for Data Collection

TransFed can in principle work with different types of sensors from which data about human activities are collected. For our testbed, we decided to use three types of sensors available on most smart wearable devices: tri-axial accelerometers, gyroscopes, and magnetometers. These sensors provide measurements at a sampling frequency of 115 Hz. The frequency of 115 Hz establishes a sufficient condition for a sample rate that permits a discrete sequence of samples to capture all the information from a continuous-time human activity signal. The testbed is shown in Figure 4.5.

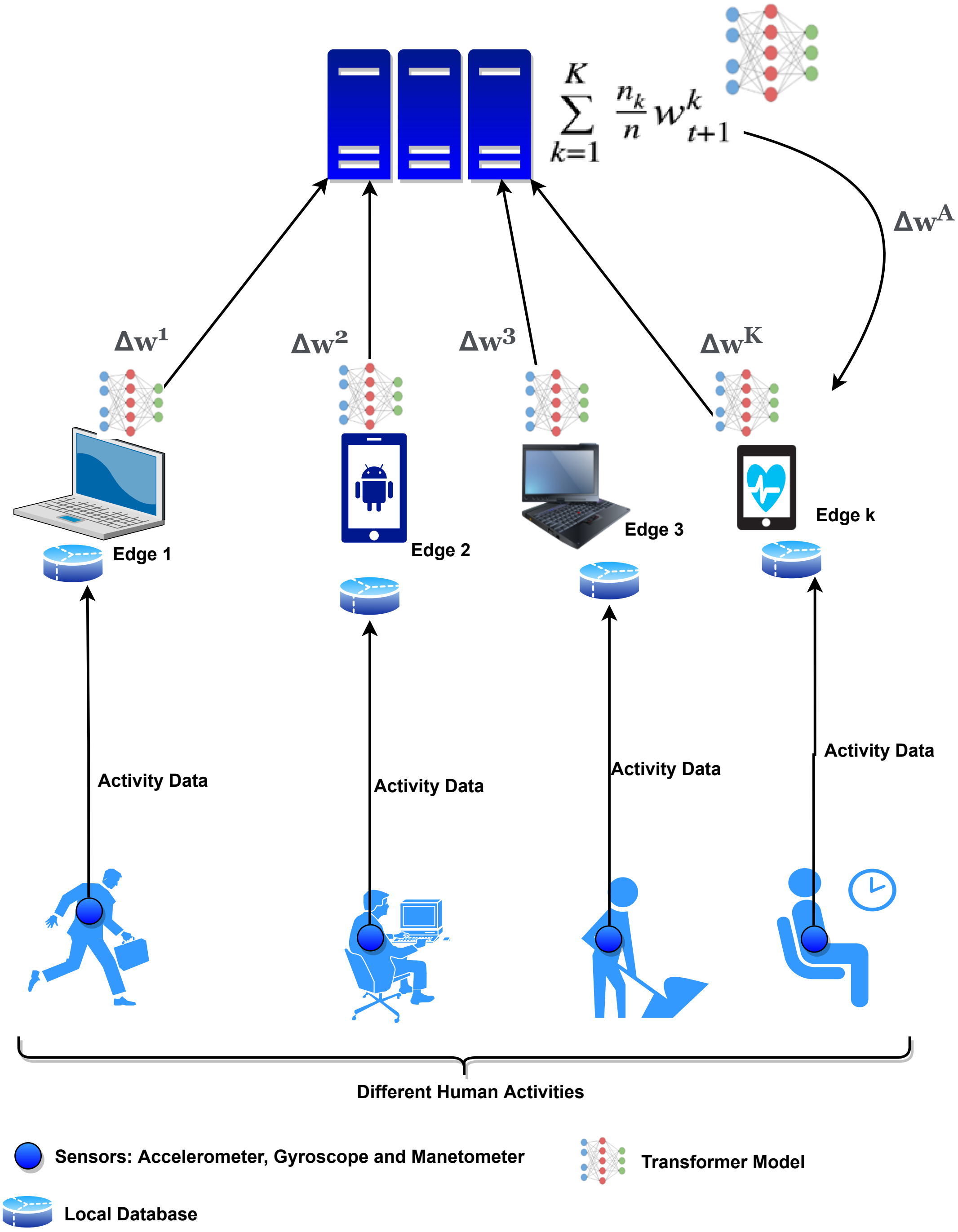


FIGURE 4.4: An illustrative diagram of the proposed framework.

Sensor Locations and Data Collection

Since the quality of data being used by an ML model can significantly impact its performance, we decided to use a data-centric approach to ensure the performance of our proposed model. One important aspect of acquiring high-quality data for HAR purposes is to identify an optimized location on the human body for each sensor used. We need to optimize the sensor locations in such a way that it provides both (i) data informative enough to be used in ML, and (ii) convenience and comfort to humans while they are wearing the sensors. Keeping both points in mind and considering what was commonly chosen in the literature [93], [98], we tried different locations on the human body: upper arm, hip, and chest. After placing each sensor on each body location in the recruited individuals, we recorded data for the following 15 activities: sitting, walking, jogging, going upstairs, going downstairs, eating, writing, using a laptop, washing face, washing hands, swiping, vacuuming, dusting a surface, and brushing teeth. Using the data collected from each location, we trained and tested the proposed transformer model, separately for the data of each location, in a centralized setting. The results showed that the model trained and tested using data collected from the hip outperformed models using data collected from two other locations (chest and upper arm). Hence, in the following, we report experimental results with data collected from the hip only.

To construct our new dataset, we recruited five human participants with different ethnic backgrounds, i.e., Pakistani, Algerian, French, Vietnamese, and Moroccan. Before collecting the data, each participant was briefed about each activity to be conducted, the health hazards of the experimental setup, and how the data will be used by the researchers. No financial compensations were made. Each participant performed the 15 activities as shown in Table 4.1. While performing each activity for a duration of around 3 minutes[1], the sensors placed on each participant generated activity data, which was sent to the ESP32 module using the I2C communication protocol. Note that the ESP32 and sensors were both located on the participant's body, powered by a lithium battery. The ESP32 module sent the data over Wi-Fi using the

[1] The precise duration was determined based on the ability and willingness of each participant.

HTTP method POST to the edge (Raspberry Pi), which was then stored in a MySQL database. The experiment was approved by the ENSAIT, GEMTEX–Laboratoire de Génie et Matériaux Textiles, University of Lille, France, from which all participants were recruited.

TABLE 4.1: Description of the 15 human activities covered in our experiments for constructing the new dataset.

Class Name	Class ID	Performed Activity	Number of Samples
Standing	0	Standing still on the floor	22,851
Sitting	1	Sitting still on a chair	23,204
Walking	2	Walking at a normal pace	23,982
Jogging	3	Running at a high speed	21,594
Going Upstairs	4	Ascending on a set of stairs at a normal pace	23,832
Going Downstairs	5	Descending from a set of stairs at a normal pace	21,836
Eating	6	Eating lunch	21,798
Writing	7	Writing on a paper	21,434
Using laptop	8	Using laptop normally	22,009
Washing face	9	Washing face standing	22,027
Washing hand	10	Washing hands standing	22,009
Swiping	11	Swiping a surface walking and standing	19,186
Vacuuming	12	Vacuuming a surface while walking and standing	22,507
Dusting a surface	13	Dusting a surface sitting	21,513
Brushing Teeth	14	Brushing teeth standing	22,495

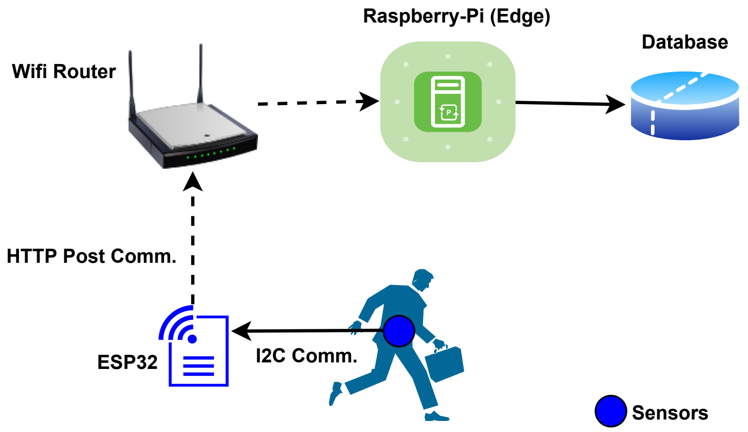


FIGURE 4.5: Our prototype for real-time data collection.

4.3.2 Federated Learning Testbed

We built a testbed to support the training and testing of the proposed HAR classifier in a federated setting (i.e., the TransFed framework). The testbed consists of one aggregation server (master node) and five clients (edge) devices. Each client trains its local model for $e = 100$ epochs on the local dataset, updates the local model, and sends the model back to the aggregation server. When all clients perform e number of epochs, the master node updates the global model and sends it again to the client workers. The process continues in r number of communication rounds. Algorithm 4.1 defines the whole implementation of the federated learning using TensorFlow and socket communication. Each client executes Lines 3-5. Whereas, the rest of the algorithm is executed by the aggregation server. In Algorithm 4.1, `train()` refers to TensorFlow's training (fit) function, and `send()` refers to the send function of the MLSocket library in Python.

To simulate more realistic scenarios, we used the SSL (Secure Socket Layer) protocol for secure communications between the server and client devices. The transformer-based classifier was trained locally only on five local Raspberry Pi devices (Pi 5 Model B+ with a 1.4GHz, 64-bit quad-core ARMv8 CPU and 1GB LPDDR2 SDRAM) as edge devices. Furthermore, a workstation with an Intel core i-6700HQ CPU and 32 GB RAM was used as the

Algorithm 4.1: The federated learning algorithm for training our proposed transformer-based HAR classifier.

```
Input: $\mathbf{GM}^r$ – the global transformer model for the $r$-th round, $\mathbf{LM}_k^r$ – the local model on the $k$-th client for the $r$-th round, $n$ – the number of data observations across all clients, $n_k$ – the number of observations on each client (edge), $\mathbf{LD}_k$ – the set of local datasets for training on each client, $r$ – the index of the current round, $e$ – the number of training epochs per one round, $b$ – the batch size of training data, $\Delta W_k^r$ – parameters of client $k$ at $r$-th round, $K$ – the number of clients participating in federated learning.
Output: Trained aggregated and updated model
while while $r \neq 0$ do
    for each cleint $k$ do
        $\mathbf{LM}_k^r = \mathbf{GM}^r$
        $\mathbf{LM}_k^{r+1} = \text{Train}(\mathbf{LM}_k^r, \mathbf{LD}_k, e, b)$
        send $(\Delta W_k^{r+1})$
    $\mathbf{GM}^{r+1} = \sum_{k=1}^{K} \frac{n_k}{n} \mathbf{LM}_k^{r+1}$
```

aggregation server. This hardware setup for clients allowed us to simulate what typical home healthcare systems can provide, in terms of computing resources.

4.3.3 Data Partitioning and Distribution

In order to analyze the performance of the proposed transformer in a federated setting we used skew data derived from a non-IID distribution. To achieve non-IID data over the clients in federated learning, we group the data by each user in the dataset and split it among 5 different clients, and selected one activity on each client to have 40-50% fewer samples. For example, Client 1 has 40-50% less samples of standing activity compared to others and Client 2 has 40-50% less samples of sitting activity compared to others.

4.3.4 Centralized Setting

In addition to testing the performance of our proposed TransFed framework based on the proposed transformer, we also tested the proposed transformer in a centralized setting, which demonstrated that it is a general technique that can work under both centralized and federated settings. We tested the proposed transformers with both our new dataset and the well-known public WISDM dataset [174], and compared its performance against other state-of-the-art methods. Since most of the existing work used the WISDM dataset in centralized setting, testing the proposed transformer-based classifier using a public dataset in a centralized setting gives a fair comparison. The details of WISDM can be found in Table 4.2.

TABLE 4.2: Basic information of the activity classes in the WISDM dataset.

Class Name	Class ID	Number of Samples
Walking	0	424,400
Jogging	1	342,177
Going Upstairs	2	122,869
Going Downstairs	3	3,100,427
Sitting	4	59,939
Standing	5	48,395

4.4 Performance Analysis

In this section, we provide the performance analysis of the proposed framework based on the above-mentioned experimental setup.

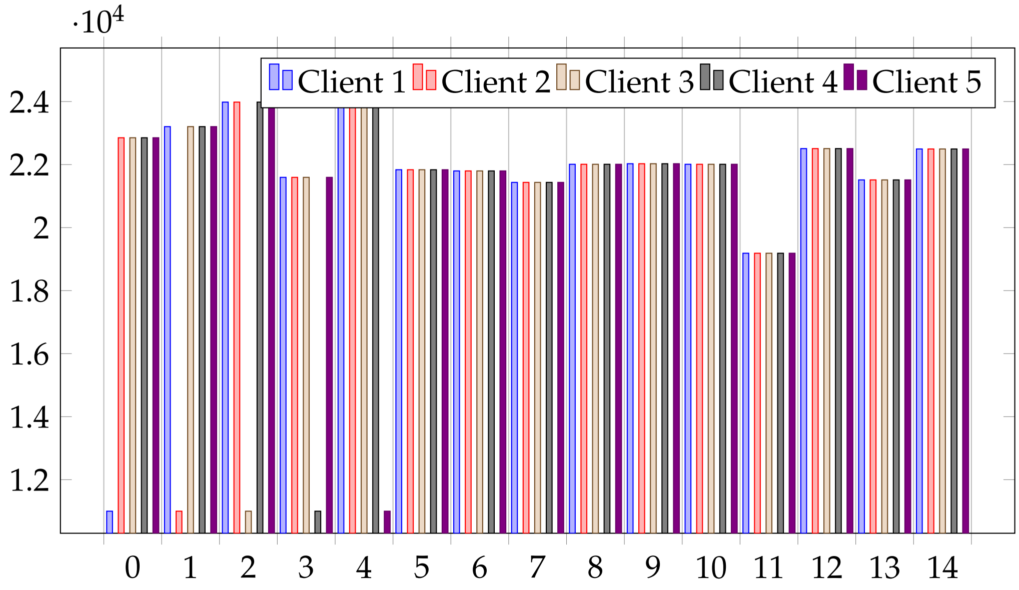


FIGURE 4.6: Data derived from a non-IID quantity distribution in the federated setting, where each client contains 40-50% less data of a given class. The x-axis represents the class ID and the y-axis represents the number of samples.

4.4.1 Accuracy (Training and validation)

To measure the classification performance of the proposed transformer-based classifier, we use the four classification metrics presented in Chapter 2.1.2. Accuracy (one-vs-rest accuracy, where we split multi-class classification into a binary classification problem per class) is often used to measure how a machine learning classifier's performance evolves during the training process. The trend over time can be used to determine whether the model is properly and ideally trained, to detect anomalies in time (such as over- or underfitting), and to make necessary adjustments.

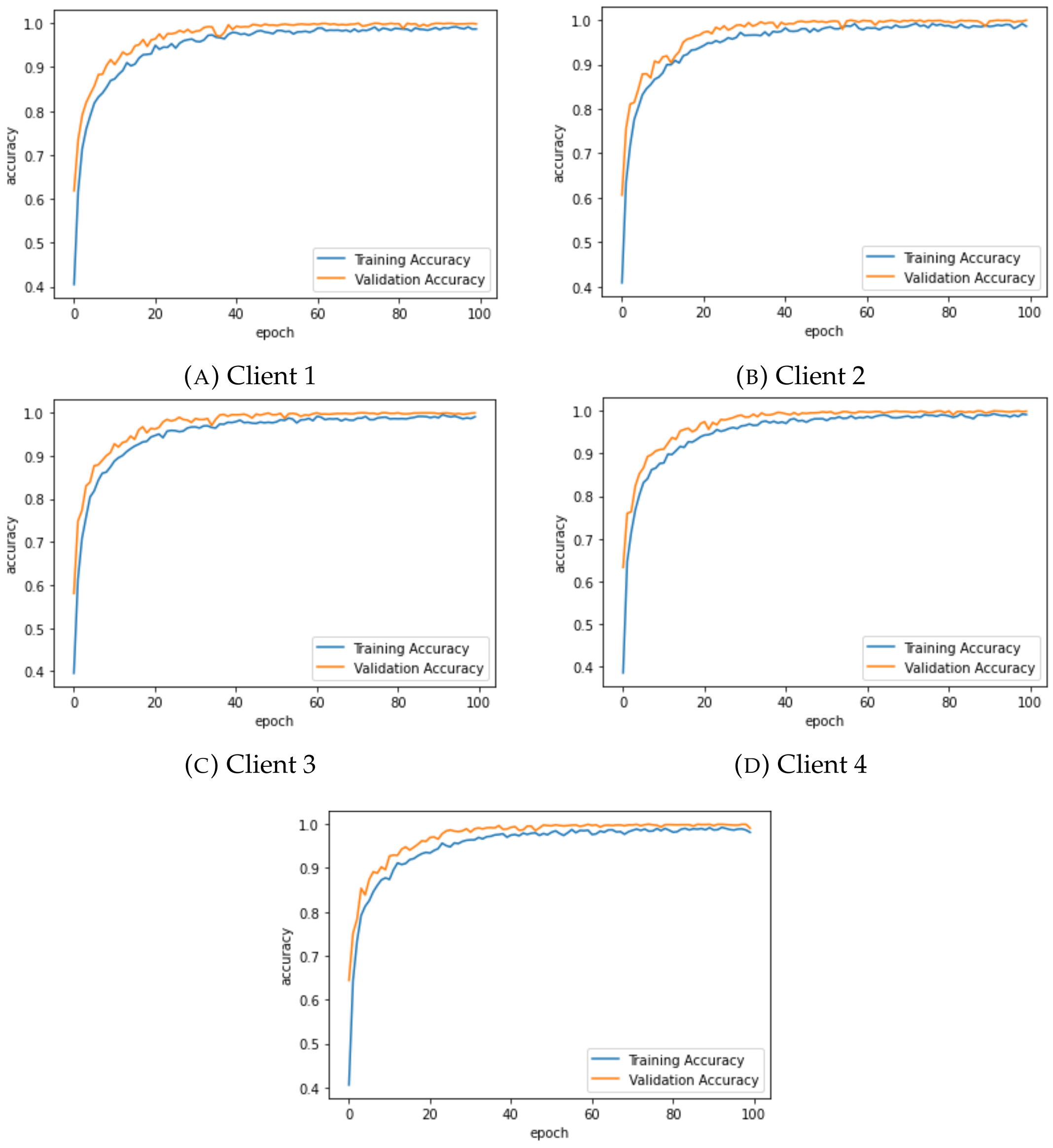
To evaluate the performance of the proposed transformer in a federated setting, we used non-IID distribution as shown in Figure 4.6. We trained the transformer at each client for 100 epochs. The hyperparameters used during the training process are given in Table 4.3. Each client used two transformer layers and a learning rate of 0.01 with an Adam optimizer.

TABLE 4.3: Hyperparameters used by each client for federated learning.

Hyper-Parameter	Value
Learning Rate	0.01
Number of Epochs	100
Batch Size	30
Weight Decay	0.001
Transformer Layers	2
Multi-attention Heads	5
Input shape	140×9

Figures 4.7 show the accuracy of each local model at the corresponding local client. For comparison purposes, we performed 100 iterations (epochs) for each model and draw a point every 10 iterations when drawing, making the curves clearer but still reflecting the overall trend. Among the local models, the performance is almost similar for the non-IID dataset, which indicates that the proposed transformer is robust against imbalanced data caused by a non-IID distribution. Overall, each local model was able to achieve more than 98 percent training and validation accuracy using the non-IID dataset.

For the centralized setting, we trained the proposed model using the public WISDM dataset as well as our collected dataset. The hyper-parameters were kept the same as mentioned earlier for the federated setting. Figures 4.8a and 4.8b present training and validation accuracy of the proposed transformer in the centralized setting using the WISDM dataset and our collected dataset, respectively.



(A) Client 1

(B) Client 2

(C) Client 3

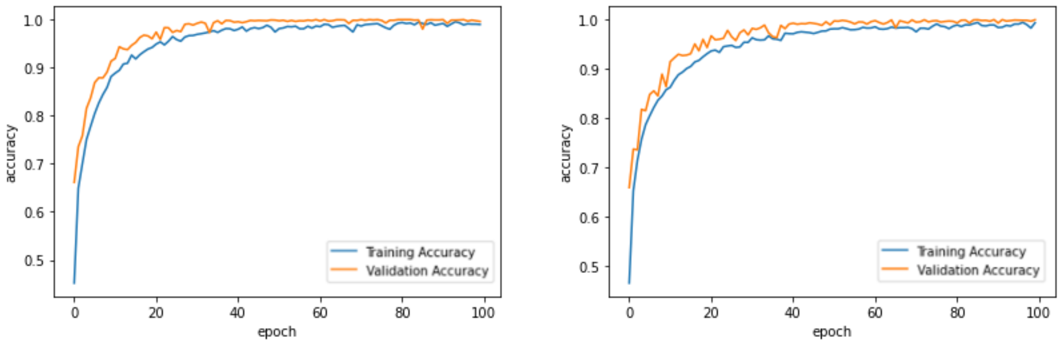
(D) Client 4

(E) Client 5

FIGURE 4.7: Training and validation accuracy of the clients using data derived from a non-IID distribution.

4.4.2 Classification Performance

Tables 4.4–4.8 present classification results obtained on all five clients using their local non-IID data in the federated setting. Table 4.9 shows the classification performance of the global model after performing federated averaging. We tested the global model using a test dataset that was not used to train any of the client models and the proposed transformer achieved an overall accuracy of 98.74%.



(A) Accuracy of the centralized model with the WISDM dataset

(B) Accuracy of the centralized model with our collected dataset

FIGURE 4.8: Training and validation accuracy of the classifier based on the proposed transformer in the centralized setting.

Moreover, Tables 4.10 and 4.11 present the classification performance of the proposed transformer-based classifier in the centralized setting, using the WISDM dataset and our collected dataset, respectively. Overall the classifier based on our proposed transformer achieved an accuracy of 99.14% and 98.89% with our collected dataset and the WISDM dataset, respectively. Since the WISDM dataset has much more unbalanced class samples than our collected dataset, it is not surprising to see the performance is (slightly) lower compared with our collected dataset, since imbalanced data are harder to learn.

4.4.3 Confusion Matrices

A confusion matrix, also known as an error matrix, is an $n \times n$ matrix or table that shows how each class is classified into all the n classes of a classifier. The diagonal elements of a confusion matrix show the correct classification results and other cells show different misclassification rates. Hence, we evaluate the proposed transformer using a confusion matrix to determine where exactly the transformer miss-classifies the classes during testing. Figure 4.9 presents the confusion matrix obtained with the updated global model in the

TABLE 4.4: Classification performance of the clients using data derived from a non-IID distribution: Client 1 with 50% less "Standing" data.

Activity	Precision	Recall	F1-score
Standing	100%	100%	100%
Sitting	100%	100%	100%
Walking	100%	100%	100%
Jogging	100%	100%	100%
Going Upstairs	100%	100%	100%
Going Downstairs	100%	100%	100%
Eating	100%	100%	100%
Writing	100%	100%	100%
Using Laptop	100%	100%	100%
Washing Face	100%	99.0%	99%
Washing Hand	99.0%	100%	99.0%
Swiping	95.0%	100%	98.0%
Vacuuming	100%	96.0%	98.0%
Dusting	100%	98.0%	99.0%
Brushing Teeth	100%	100%	100%

TABLE 4.5: Classification performance of the clients using data derived from a non-IID distribution: Client 2 with 50% less "Sitting" data.

Activity	Precision	Recall	F1-score
Standing	100%	100%	100%
Sitting	98.0%	100%	99.0%
Walking	100%	98.0%	99.0%
Jogging	99.0%	99.0%	99.0%
Going Upstairs	98.0%	96.0%	97.0%
Going Downstairs	95.0%	99.0%	97.0%
Eating	98.0%	98.0%	97.0%
Writing	100%	96.0%	99.0%
Using Laptop	96.0%	100%	98.0%
Washing Face	99.0%	100%	99.0%
Washing Hand	97.0%	97.0%	97.0%
Swiping	96.0%	95.0%	96.0%
Vacuuming	93.0%	94.0%	93.0%
Dusting	96.0%	92.0%	94.0%
Brushing Teeth	97.0%	98.0%	97.0%

federated setting, using our collected dataset. Figure 4.10 presents the confusion matrix obtained using the classifier based on our proposed transformer in the centralized setting, using our collected dataset. In both figures, the

TABLE 4.6: Classification performance of the clients using data derived from a non-IID distribution: Client 3 with 50% less “Walking” data.

Activity	Precision	Recall	F1-score
Standing	100%	100%	100%
Sitting	100%	100%	100%
Walking	99.0%	100%	100%
Jogging	98.0%	98.0%	98.0%
Going Upstairs	98.0%	97.0%	98.0%
Going Downstairs	98.0%	98.0%	98.0%
Eating	95.0%	97.0%	96.0%
Writing	99.0%	98.0%	99.0%
Using Laptop	96.0%	97.0%	96.0%
Washing Face	96.0%	99.0%	97%
Washing Hand	98.0%	95.0%	96.0%
Swiping	96.0%	92.0%	94.0%
Vacuuming	95.0%	97.0%	96.0%
Dusting	96.0%	95.0%	96.0%
Brushing Teeth	97.0%	98.0%	98.0%

TABLE 4.7: Classification performance of the clients using data derived from a non-IID distribution: Client 4 with 50% less “Jogging” data.

Activity	Precision	Recall	F1-score
Standing	100%	100%	100%
Sitting	100%	97.0%	99.0%
Walking	97.0%	100%	99.0%
Jogging	99.0%	100%	100%
Going Upstairs	99.0%	98.0%	98.0%
Going Downstairs	98.0%	99.0%	98.0%
Eating	97.0%	97.0%	97.0%
Writing	99.0%	98.0%	98.0%
Using Laptop	97.0%	99.0%	98.0%
Washing Face	98.0%	97.0%	98.0%
Washing Hand	96.0%	98.0%	97.0%
Swiping	96.0%	94.0%	95.0%
Vacuuming	93.0%	94.0%	94.0%
Dusting	95.0%	94.0%	95.0%
Brushing Teeth	99.0%	98.0%	98.0%

x-axis indicates the predicted class labels and the y-axis indicates the ground true class labels. We can see that in both settings the proposed HAR classifier

TABLE 4.8: Classification performance of the clients using data derived from a non-IID distribution: Client 5 with 50% less "Going Upstairs" data.

Activity	Precision	Recall	F1-score
Standing	100%	100%	100%
Sitting	99.0%	97.0%	98.0%
Walking	97.0%	98.0%	98.0%
Jogging	99.0%	100%	99.0%
Going Upstairs	100%	100%	100%
Going Downstairs	99.0%	99.0%	99.0%
Eating	96.0%	96.0%	96.0%
Writing	100%	97.0%	98.0%
Using Laptop	96.0%	100%	98.0%
Washing Face	97.0%	97.0%	97.0%
Washing Hand	95.0%	95.0%	95.0%
Swiping	94.0%	95.0%	95.0%
Vacuuming	97.0%	96.0%	96.0%
Dusting	96.0%	97.0%	96.0%
Brushing Teeth	99.0%	98.0%	98.0%

TABLE 4.9: Classification performance of the global model after federated averaging using data derived from a non-IID distribution.

Activity	Precision	Recall	F1-score
Standing	100%	100%	100%
Sitting	100%	100%	100%
Walking	99.0%	100%	100%
Jogging	99.0%	99.0%	99.0%
Going Upstairs	96.0%	94.0%	95.0%
Going Downstairs	97.0%	97.0%	97.0%
Eating	99.0%	97.0%	98.0%
Writing	100%	99.0%	100%
Using Laptop	99.0%	100%	99.0%
Washing Face	100%	98.0%	99.0%
Washing Hand	93.0%	100%	97.0%
Swiping	97.0%	90.0%	94.0%
Vacuuming	90.0%	98.0%	94.0%
Dusting	99.0%	94.0%	97.0%
Brushing Teeth	96.0%	99.0%	97.0%

worked very well with similar performance across all the 15 classes. Similarly, Figure 4.11 presents the confusion matrix obtained using the WISDM dataset in centralized settings. It can be seen that the proposed transformer

TABLE 4.10: Classification performance of the proposed transformer in the centralized setting using WISDM dataset [174].

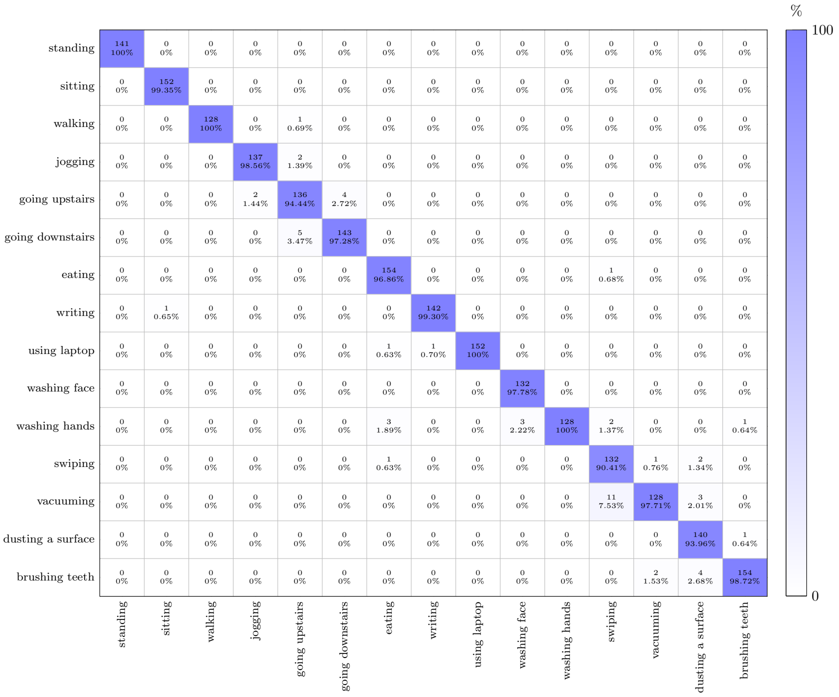
Activity	Precision	Recall	F1-score
Walking	100%	100%	100%
Jogging	100%	100%	100%
Going Upstairs	95.0%	96.0%	96.0%
Going Downstairs	98.0%	99.0%	99.0%
Sitting	100%	100%	100%
Standing	97.0%	97.0%	97.0%

TABLE 4.11: Classification performance of the proposed transformer in the centralized setting using our collected dataset.

Activity	Precision	Recall	F1-score
Standing	100%	100%	100%
Sitting	100%	99.0%	99.0%
Walking	99.0%	99.0%	99.0%
Jogging	98.0%	98.0%	98.0%
Going Upstairs	99.0%	95.0%	97.0%
Going Downstairs	97.0%	99.0%	98.0%
Eating	99.0%	98.0%	98.0%
Writing	98.0%	99.0%	98.0%
Using Laptop	97.0%	99.0%	98.0%
Washing Face	100%	97.0%	99.0%
Washing Hand	96.0%	100%	98.0%
Swiping	95.0%	90.0%	93.0%
Vacuuming	93.0%	98.0%	95.0%
Dusting	99.0%	93.0%	96.0%
Brushing Teeth	96.0%	100%	98.0%

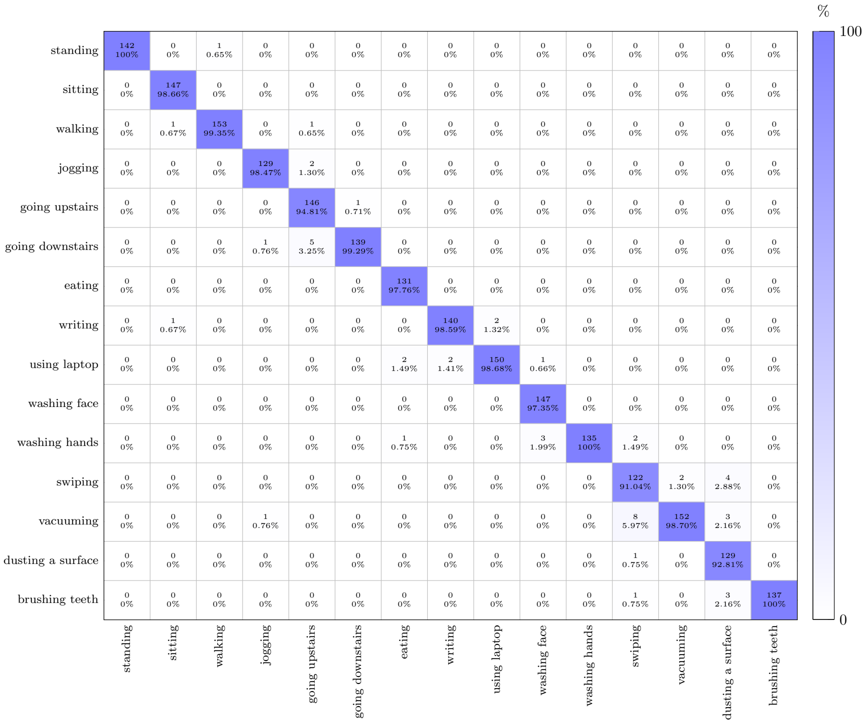
achieved almost perfect classification results for all classification activities.

From the confusion matrices, it can be observed that misclassifications occurred more between activities that involve similar body movements, e.g., swiping and vacuuming. It can also be observed that, even for these similar but different human activities, the proposed transformer achieved a very good performance.



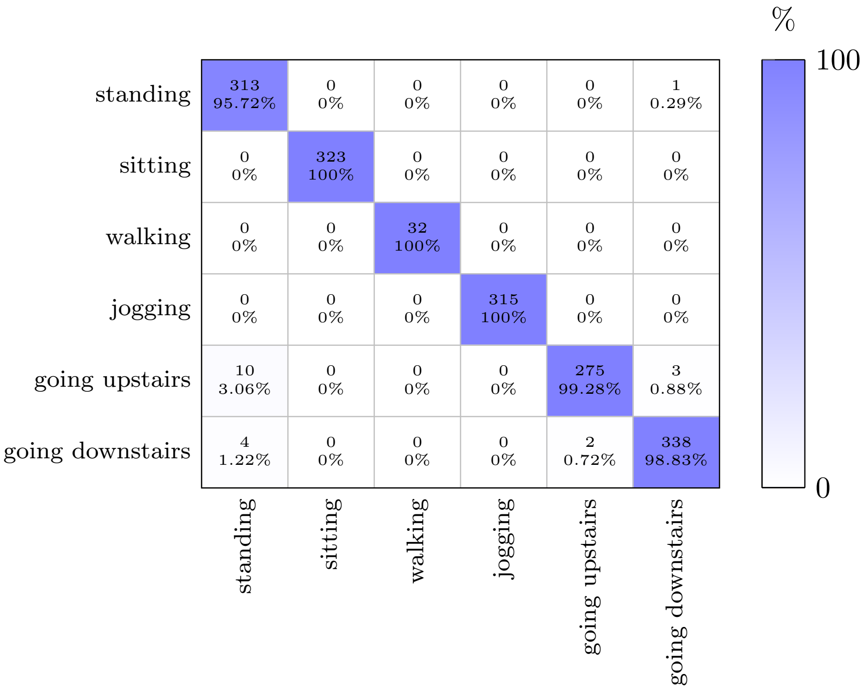
	standing	sitting	walking	jogging	going upstairs	going downstairs	eating	writing	using laptop	washing face	washing hands	swiping	vacuuming	dusting a surface	brushing teeth
standing	141 100%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
sitting	0 0%	152 99.35%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
walking	0 0%	0 0%	128 100%	0 0%	1 0.69%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
jogging	0 0%	0 0%	0 0%	137 98.56%	2 1.39%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
going upstairs	0 0%	0 0%	0 0%	2 1.44%	136 94.44%	4 2.72%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
going downstairs	0 0%	0 0%	0 0%	0 0%	5 3.47%	143 97.28%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
eating	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	154 96.86%	0 0%	0 0%	0 0%	0 0%	1 0.68%	0 0%	0 0%	0 0%
writing	0 0%	1 0.65%	0 0%	0 0%	0 0%	0 0%	0 0%	142 99.30%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
using laptop	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	1 0.63%	1 0.70%	152 100%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
washing face	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	132 97.78%	0 0%	0 0%	0 0%	0 0%	0 0%
washing hands	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	3 1.89%	0 0%	0 0%	3 2.22%	128 100%	2 1.37%	0 0%	0 0%	1 0.64%
swiping	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	1 0.63%	0 0%	0 0%	0 0%	0 0%	132 90.41%	1 0.76%	2 1.34%	0 0%
vacuuming	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	11 7.53%	128 97.71%	3 2.01%	0 0%
dusting a surface	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	140 93.96%	1 0.64%
brushing teeth	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	2 1.53%	4 2.68%	154 98.72%

FIGURE 4.9: The confusion matrix obtained with the final global transformer model in a federated setting with five clients, using data derived from a non-IID distribution.



	standing	sitting	walking	jogging	going upstairs	going downstairs	eating	writing	using laptop	washing face	washing hands	swiping	vacuuming	dusting a surface	brushing teeth
standing	142 100%	0 0%	1 0.65%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
sitting	0 0%	147 98.66%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
walking	0 0%	1 0.67%	153 99.35%	0 0%	1 0.65%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
jogging	0 0%	0 0%	0 0%	129 98.47%	2 1.30%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
going upstairs	0 0%	0 0%	0 0%	0 0%	146 94.81%	1 0.71%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
going downstairs	0 0%	0 0%	0 0%	1 0.76%	5 3.25%	139 99.29%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
eating	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	131 97.76%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
writing	0 0%	1 0.67%	0 0%	0 0%	0 0%	0 0%	0 0%	140 98.59%	2 1.32%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%
using laptop	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	2 1.49%	2 1.41%	150 98.68%	1 0.66%	0 0%	0 0%	0 0%	0 0%	0 0%
washing face	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	147 97.35%	0 0%	0 0%	0 0%	0 0%	0 0%
washing hands	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	1 0.75%	0 0%	0 0%	3 1.99%	135 100%	2 1.49%	0 0%	0 0%	0 0%
swiping	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	122 91.04%	2 1.30%	4 2.88%	0 0%
vacuuming	0 0%	0 0%	0 0%	1 0.76%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	8 5.97%	152 98.70%	3 2.16%	0 0%
dusting a surface	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	1 0.75%	0 0%	129 92.81%	0 0%
brushing teeth	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	0 0%	1 0.75%	0 0%	3 2.16%	137 100%

FIGURE 4.10: The confusion matrix obtained with the transformer model in a centralized setting using 5-fold cross-validation, using the balanced collected data.



	standing	sitting	walking	jogging	going upstairs	going downstairs
standing	313 95.72%	0 0%	0 0%	0 0%	0 0%	1 0.29%
sitting	0 0%	323 100%	0 0%	0 0%	0 0%	0 0%
walking	0 0%	0 0%	32 100%	0 0%	0 0%	0 0%
jogging	0 0%	0 0%	0 0%	315 100%	0 0%	0 0%
going upstairs	10 3.06%	0 0%	0 0%	0 0%	275 99.28%	3 0.88%
going downstairs	4 1.22%	0 0%	0 0%	0 0%	2 0.72%	338 98.83%

FIGURE 4.11: The confusion matrix obtained with the transformer model in the centralized setting, using the WISDM dataset.

4.4.4 Comparison with SOTA Methods

In this subsection, we compare the proposed transformer with existing SOTA methods. Table 4.12 compares two key features of our proposed transformers with methods based on RNNs and CNNs. RNN-based models do not allow parallelization during training because of their sequential nature, which makes the model computationally slow and expensive. CNN-based methods can perform parallel computation, but they are computationally expensive because of the convolution function. Our new method based on the proposed transformer completely eliminates recurrence and convolution and replaces them with a self-attention mechanism to establish dependencies between the input and the output. It is the first type of architecture to rely entirely on attention to calculate representations of the input and the output. In addition, transformers leave more room for parallelization. RNNs and CNNs use a large number of parameters (usually hundreds of thousands or even more), but the proposed transformer only uses 14,697 parameters, making it more suitable to be utilized in mobile computing devices. Moreover, unlike traditional transformers, the proposed transformer uses a single patch instead of using multiple-patches. Therefore, the proposed transformer is also much more computationally efficient.

TABLE 4.12: Comparison of our transformer-based approach with those based on RNNs and CNNs, in terms of computation costs.

Method	Parallelization	Computationally Expensive
RNNs	No	Yes
CNNs	Yes	Yes
Transformers	Yes	No

We also compared the performance of the TransFed global model and the centralized classifier based on the proposed transform with those of selected state-of-the-art HAR methods in the literature [94], [96], [97], including five working in a centralized setting and one in the federated setting. These three state-of-the-art methods were chosen because their performance results were reported using the WISDM dataset, which allows a direct comparison of the performance results. Table 4.13 shows the comparison results with the six selected state-of-the-art methods. It is obvious that our proposed methods achieved a substantial improvement in terms of accuracy when trained and tested using our new dataset and the WISDM dataset. Specifically, our method achieved an accuracy of 98.74% and 99.14% in the federated and centralized settings, respectively, using our collected dataset. Furthermore, using the WISDM dataset in the centralized setting, it achieved an overall accuracy of 98.89%. Hence, from Table 4.13 it can be seen that the proposed transformer outperforms existing state-of-the-art methods, in both centralized and federated settings.

TABLE 4.13: Comparison with selected state-of-the-art methods for HAR classification.

Scheme	Centralized or Federated	Number of Activities	Accuracy
[94][a]	Federated	6	89.00%
[96][a]	Centralized	6	97.63%
[97][a]	Centralized	6	96.70%
Proposed[a]	Centralized	6	98.89%
Proposed[b]	Federated	15	98.74%
Proposed[b]	Centralized	15	99.14%

[a] Using the WISDM dataset.
[b] Using our new dataset.

4.5 Summary

In this chapter, we proposed a novel single-patch lightweight transformer for home care applications e.g., HAR. We examined the use of transformers as a HAR classifier. The purpose of the lightweight transformer was to provide a state-of-the-art classification performance while keeping it computationally efficient for mobile computing devices such as smartphones. For our proposed transformer-based classifier, we examined its performance in both federated and centralized settings, under a non-IID data distribution. To test the performance of the proposed transformer in the federated setting, we developed a framework called TransFed and designed a testbed to collect data from five human participants who conducted 15 different activities in a simulated home environment.

Our extensive experimental results confirmed that the proposed transformer can provide better performance compared with a number of state-of-the-art CNN- and RNN-based HAR classifiers while providing a standardized and automated way to accomplish the feature learning step. Furthermore, the federated setting used by our proposed framework TransFed can help improve data privacy, which is a major issue in centralized approaches.

Up to this chapter, we used labeled data for healthcare applications. However, in some cases, such as anomaly detection it is hard to find data about anomalies. Therefore, we need unsupervised or semi-supervised approaches to address such challenges where data about some classes are limited and hard to obtain. Hence, in the next chapter, we will be developing a framework for adaptive anomaly detection in a federated setting with limited anomaly class data.

Chapter 5

Adaptive Anomaly Detection and Explanations with Federated Learning

5.1 Introduction

One of the important tasks we often encounter when analyzing real-world data is to determine whether a given instance is normal or an anomaly for a given environment and task. The formal process of detecting or classifying all such data instances (anomalous data points) in a data-driven fashion is known as anomaly detection or outlier detection [103], [175]. Anomaly detection is important because it can be used to detect different types of important real-world problems such as health issues, fraud, and security breaches. In some critical applications such as many related to healthcare, anomaly detection can help avoid catastrophic outcomes, such as loss of human lives. For example, anomalous cells in a Magnetic Resonance Image (MRI) or an irregular segment of an Electrocardiogram (ECG) may indicate the presence of a specific disease such as a malignant tumor or an impeding heart attack [176]–[178], respectively.

Detection of anomalies or outliers has been of great interest to the statistics and Machine Learning (ML) research communities. Many anomaly detection techniques have been developed, including general techniques and more application-specific ones. For example, an ECG (Electrocardiogram) is a quick, safe, and painless way to monitor heart conditions (e.g., arrhythmias). Nevertheless, to detect arrhythmias, longer-term ECG monitoring is often required to track the patients' heart conditions for an extended period of time (e.g., 24 hours) [179]. Recent development in sensing technologies has enabled such longer-term monitoring of patients. Smart and portable devices, such as smart watches, Omron HeartScan [180], and the recently developed Hexoskin Smart Garments [181], are revolutionizing cardiac diagnostics by monitoring cardiac activities and transmitting longer-term ECG signals to cloud services for remote analysis by medical professionals. However, such signals are often too long for medical professionals to inspect, who simply cannot spend too much time (hours or longer) looking at the ECG signal in order to detect possible abnormal signals.

To address the above-mentioned challenges in different applications, such as longer-term ECG analysis, machine learning based anomaly detection methods have been proposed [182]. Nevertheless, as we argued previously, such methods have limitations in terms of privacy concerns, communications costs, and explanations [183]–[186]. Moreover, to train a sufficiently accurate and robust machine learning model, we normally need a lot of well-labeled data, which can take a long time to collect from a single silo (organization) especially if one or more target health conditions are not very common. Another limitation of most threshold-based anomaly detection methods is that they do not determine the threshold adaptively. Instead, they rely on the standard deviation [187] or the absolution deviation around the mean [188] to determine the threshold. However, this approach works only when we know the (at least approximate) underlying distribution of data. Unfortunately, most real-time data are not normally distributed so the distribution has to be estimated first. Additionally, in addition to privacy concerns mentioned above, data owners (individual silos) often have other reasons to be unwilling to share data with a central authority, e.g., market competition. All such problems call for more adaptive and privacy-preserving machine learning in a non-centralized setting, and FL has been proposed to address such a need [112].

In this chapter, we present AnoFed, a new framework for adaptive anomaly detection in federated settings that can achieve the above-mentioned aims. Although we show the applicability of the proposed framework for ECG analysis, the general idea can be extended to other digital health applications. In AnoFed, we apply FL for two goals – to provide enhanced data privacy and to reduce communication costs. The use of FL allows incremental updates of the global model while new data from participating nodes (edge devices) come in, therefore achieving adaptivity. To support resource-constrained edge devices as local nodes, we propose novel lightweight transformer-based Autoencoders (AE) and Variational Autoencoders (VAE) as building blocks of AnoFed. Moreover, we combine the proposed models with the SVDD [42] with kernel density estimation for adaptive anomaly detection, for each global round of training in the federated setting. We also provide an XAI module to trace the most critical part(s) of the ECG that is/are responsible for the detected anomaly.

5.1.1 Contributions

The key contributions of this paper are summarized as follows.

1. To the best of our knowledge, AnoFed is the first lightweight (in terms

of the number of parameters and the number of local/global training rounds required for the desired efficacy) VAE and an AE based on transformers in federated setting (for enhanced privacy of user) for ECG anomaly detection. Owning to the use of transformers, AnoFed can combine the merits of both CNN- and RNN-based models.

2. We propose a new framework which is a combined design of the VAE/AE and SVDD with kernel density estimation for adaptive anomaly detection. The VAE/AE extracts feature from the input data in the form of error vectors which are then used to train the SVDD with kernel density estimation, which allows the proposed framework to provide state-of-the-art results even when the data distribution changes in the local clients in the federated setting.

3. We design an XAI module to improve the explainability of the results of our framework, which helps enhance the trust of users in the proposed framework.

4. We trained and tested our proposed framework using two datasets from PhysioNet [189] and a testbed with three edge devices and a global server, and the results show that our framework could achieve SOTA detection accuracy with the proven ability to automatically adapt to different distributions of the data (error vectors).

The rest of the chapter is organized as follows. Section 5.2 presents the proposed framework. Sections 5.3 and 5.4 present the experimental setup and performance analysis, respectively. Comparisons with some SOTA methods and an analysis of the time complexity of the proposed framework are also included in Section 5.4. Finally, the last section summarizes the chapter.

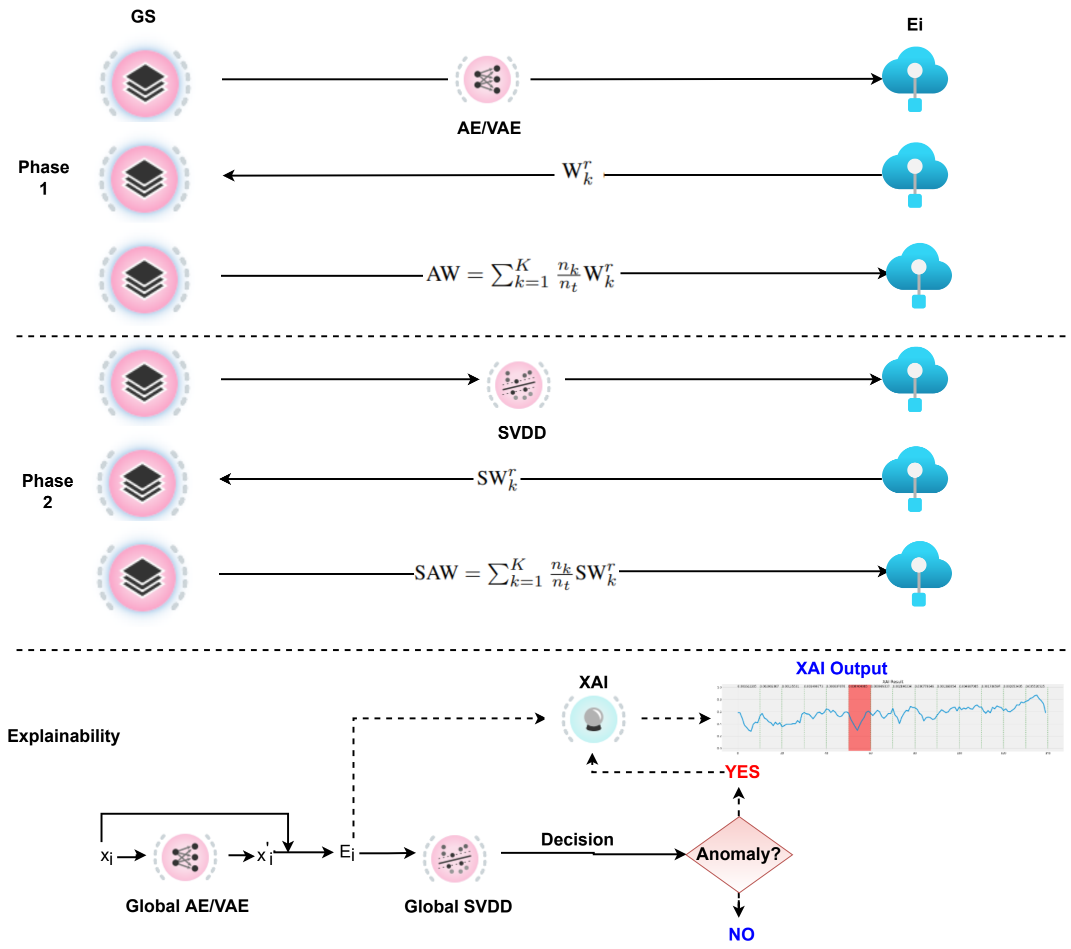


FIGURE 5.1: Overview of the proposed framework AnoFed.

5.2 Proposed Framework AnoFed

In this section, we first give an overview of the proposed framework AnoFed and then provide details of different components.

5.2.1 Overview

An overview of the proposed framework is shown in Figure 5.1. Let us assume that there are K edge devices participating in FL to jointly train an ECG anomaly detection system. In order to train a joint global model **GM**, all the edge devices connect to a central server or global server **GS**, where an edge device is represented as $\mathbf{E}_k$ and data in each edge device is represented as D_k, $k = 1, \ldots, K$. **GM** represents the global updated model and $\mathbf{LM}_k$ the local model at $\mathbf{E}_k$. We divide one global round into two phases: in Phase one, **GM** is AE/VAE, and in Phase two, **GM** is SVDD. Additionally, we denote the weights of $\mathbf{LM}_k$ in Phase one as W_k and averaged weights of **GM** as AW. In Phase two, we denote the weights of $\mathbf{LM}_k$ as SW_k and averaged weights of **GM** as SAW. In Phase one, all the edge devices train the VAE/AE and use a callback to monitor the reconstruction loss. When the reconstruction loss is not improving anymore, each edge device sends the local weights of VAE/AE to the global server and waits for the global server to send the aggregated weights for VAE/AE. After aggregating the updates, the global server sends the aggregated updates to all participating edged devices. Each edge device then computes the error vectors using the aggregated updates and starts Phase two: training of the SVDD. Similar to the first phase, each edge device monitors the classification performance of the local SVDD and sends the updates to the global server once it stops improving, which are then aggregated and sent back to all participating edge devices. The explanations can be achieved in both the global model and the local models. It should be noted that the XAI module does not require training, instead, it simply takes the output of the AE/VAE and that of the SVDD classifier to produce a visualized explanation of the detected anomaly. The training of each global round is described by Algorithm 5.1. In this algorithm, n_k is the number of training samples of the edge device E_k, and $n_t = \sum_{k=1}^{K} n_k$ is the total number of samples across all edge devices.

5.2.2 Proposed AE and VAE

Since both AE and VAE performed well according to the literature, we tested an AE and a VAE with the same number of transformer blocks in order to see which one performs better. The proposed AE and VAE are shown in Figure 5.2.

Algorithm 5.1: The training procedure of the proposed framework AnoFed (a single global round)

```
Input: Data from edge devices $D_1, D_2, \ldots, D_k$
Output: AW and SAW
GS compiles the initial GM
GS sends GM to the requesting $E_k$
$E_k$ receives GM, trains it using local data $D_k$, and sends updated
  weights $\mathrm{W}_k^r$ of $\mathbf{LM}_k^r$ to GS
GS waits to receive updates from all $K$ edge devices.
if updates received form K edge devices then
    $\mathrm{AW} = \sum_{k=1}^{K} \frac{n_k}{n_t} \mathrm{W}_k^r$
$G_s$ sends AW to $E_k$
$E_k$ updates its local model with AW
$E_k$ computes error vectors
$E_k$ trains SVDD using error vectors and sends updated weights $\mathrm{SW}_k^r$
  to GS
if updates received form K edge devices then
    $\mathrm{SAW} = \sum_{k=1}^{K} \frac{n_k}{n_t} \mathrm{SW}_k^r$
$G_s$ sends SAW to $E_k$
```

In both the AE and the VAE, the encoder module consists of the first input layer that takes the ECG segment of 140 stamps as the input. The output of the input layer is passed through the transformer layer. The transformer layer consists of many sub-layers as shown in Figure 5.2. The first one is an augmentation layer, which applies random (with loss of information) and more realistic (without significant loss of information) transformations to increase the diversity of the training set. The output from the augmentation layer is normalized using a normalization layer, and then a multihead attention layer applies the self-attention mechanism. The self-attention mechanism takes a sequence and outputs a corresponding sequence vector. Let us consider k-dimensional input vectors $X_1, X_2, \ldots, X_t$ and $X_1', X_2', \ldots, X_i'$ as their corresponding output vectors. To compute the vector X_i', the self-attention mechanism computes weights averaged over all the input vectors,

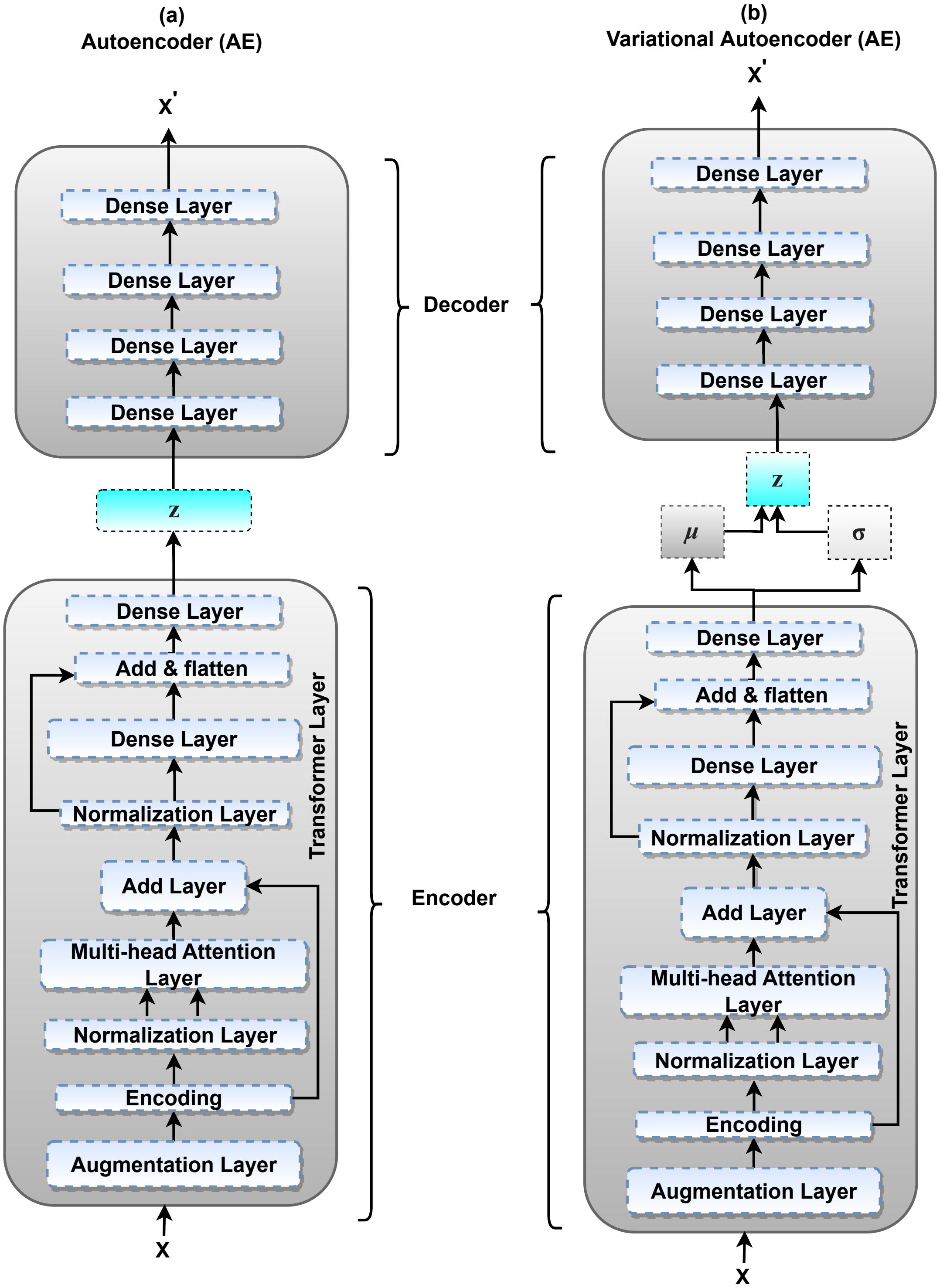


FIGURE 5.2: The AE and the VAE for our proposed framework AnoFed.

given by the following equation:

$$X_i' = \sum_j W_{ij} X_j, \tag{5.1}$$

where j indexes the whole sequence and the sum of all the indexes is equal to one. The weight W_{ij} is computed using the following function over X_i and X_j:

$$W_{ij} = X_i^T X_j. \tag{5.2}$$

As the output of a dot product is a real value, in order to map the values in the range of 0 and 1, and to ensure that the sum over the whole sequence sums to 1, we employ a softmax function, given as follows:

$$W_{ij} = \frac{\exp(W_{ij})}{\sum_j \exp(W_{ij})}. \tag{5.3}$$

The dot product in the self-attention mechanism expresses the correlation of input features. The output features are obtained by computing the weighted sum over the whole input sample. The output of the multi-head attention layer is combined with the output of the preceding normalization layer using an additional layer. Then, the output of the additional layer is fed into a succeeding normalization layer. The output of this normalization layer is then fed into a dense layer that applies a non-linear transformation to extract further features, given as follow:

$$\text{Output} = f_{\text{activation}}(\text{dot}(\text{input}, \text{kernel})), \tag{5.4}$$

where $f_{\text{activation}}$ is the activation function, and the kernel is a weight matrix. The output of the final dense layer gives latent space representation in the case of AE, i.e., the encoder maps the input into a lower-dimensional feature space Z. Whereas in the case of VAE, the output is a latent distribution with μ as the mean and σ as the standard deviation, expressing the latent space regularization (enforced to be close to a standard normal distribution). For both AE and VAE, the latent representation is then fed into the decoder. The decoder module consists of four simple dense layers and the final layer uses a sigmoid activation function that gives probability distributions of the candidate classes, whereas the activation function of the remaining layers is a rectifier linear unit (ReLU).

5.2.3 Proposed SVDD Module

In a federated setting, because the distributions of local data and the global data can differ from each other significantly, anomaly detection methods relying on a static threshold (normally manually selected based on the training data, e.g., the mean plus one standard deviation of the reconstruction loss) may not ensure the global model can still work well. To address this challenge, we propose to use SVDD along with density kernel estimation for adaptive anomaly detection, which can avoid the problem of setting a static threshold. We adopted the SVDD classifier from [42] to construct a nonlinear SVDD by employing kernel density estimation, as shown in Figure 5.3. The kernel maps the input into a new higher-dimensional feature space by applying a nonlinear transformation using a special kernel function. After that, we use the SVDD model in this new higher-dimensional feature space. Hence, the SVDD linear model in this new higher-dimensional feature space represents a nonlinear model in the input space. To train the proposed SVDD, $\mathbf{E}_k$ first computes the error vectors E_i, given by the following equation:

$$E_i = X_i - X_i'. \tag{5.5}$$

Let $E_1, E_2, \ldots, E_k$ be independent and identically distributed samples with f, where f is the unknown density at any given sample E, then kernel density estimator function of f is given by the following equation:

$$f_b(E) = \frac{1}{k}\sum_{i=1}^{k} \mathrm{Ker}_b(E, E_i), \tag{5.6}$$

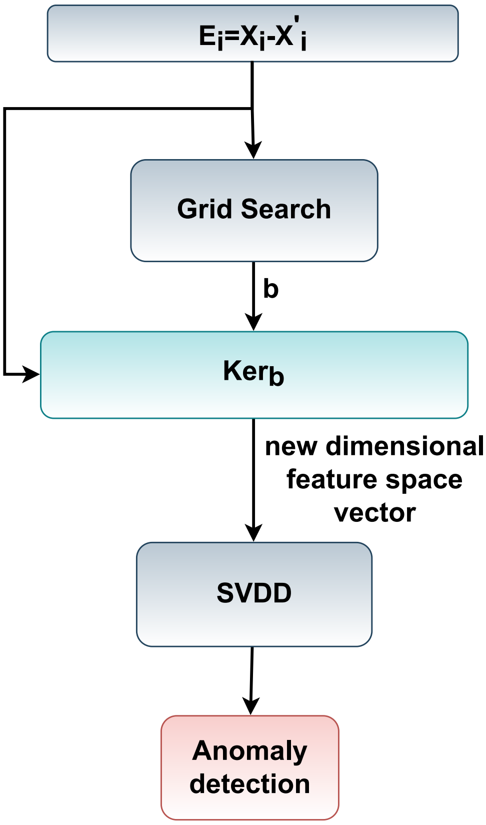


FIGURE 5.3: Adaptive anomaly detection using SVDD.

where Ker is the Laplace radial-basis kernel and b is the adaptive bandwidth. A scaled kernel with bandwidth b is defined as follows:

$$\text{Ker}_b(X) = \frac{1}{b}\text{Ker}(\frac{X}{b}).$$

In order to come up with an adaptive bandwidth for the SVDD, we use a grid search mechanism. The grid search mechanism is provided with the error vectors, which apply a grid search using pre-defined hyperparameters and outputs the best bandwidth denoted as b.

5.2.4 Proposed XAI Module

In sensitive applications such as digital healthcare, we need to inform users about why the model reached the output results and which portion(s) of the input is/are responsible for the certain output. In our case, health professionals (end users of the anomaly detection system) would be interested in knowing which portion of the input is responsible for the maximum reconstruction loss because that portion would most influence and contribute to an anomaly. Hence, identifying such a segment in the input would help identify the key pattern(s) of an anomalous ECG signal. However, deep learning-based modules are complex in nature to explain, i.e., a high number of model parameters. To address this challenge, we provide a model-agnostic XAI module to identify the segment (of desired length/window size) of the input ECG sample that contributes the most to the reconstruction loss, thereafter to the anomaly. Let $X = (x_1, \ldots, x_n)$ be the input to the AE/VAE, and $X' = (x'_1, \ldots, x'_n)$ be the corresponding reconstruction, where X is an ECG signal with n time stamps. Then, the proposed XAI module identifies the key segment of the ECG signal with the maximum reconstruction loss by Algorithm 5.2. In this algorithm, s is the number of sub-segments of the input, S_{pos} is the starting position of a sub-segment, E_{pos} is the end position of the sub-segment, max_{loss} is the maximum reconstruction loss, $\text{max}_{\text{loss-position}}$ is the position of the sub-segment in the input, and SLoss is the reconstruction loss of a given sub-segment.

5.3 Experimental Setup

5.3.1 Dataset Description

To train and test the proposed framework AnoFed, we used a combination of two datasets from PhysioNet [189]. For the anomaly class, we used the BIDMC Congestive Heart Failure Database. This database contains longer-term ECG recordings of severe congestive heart failures from 15 subjects, out

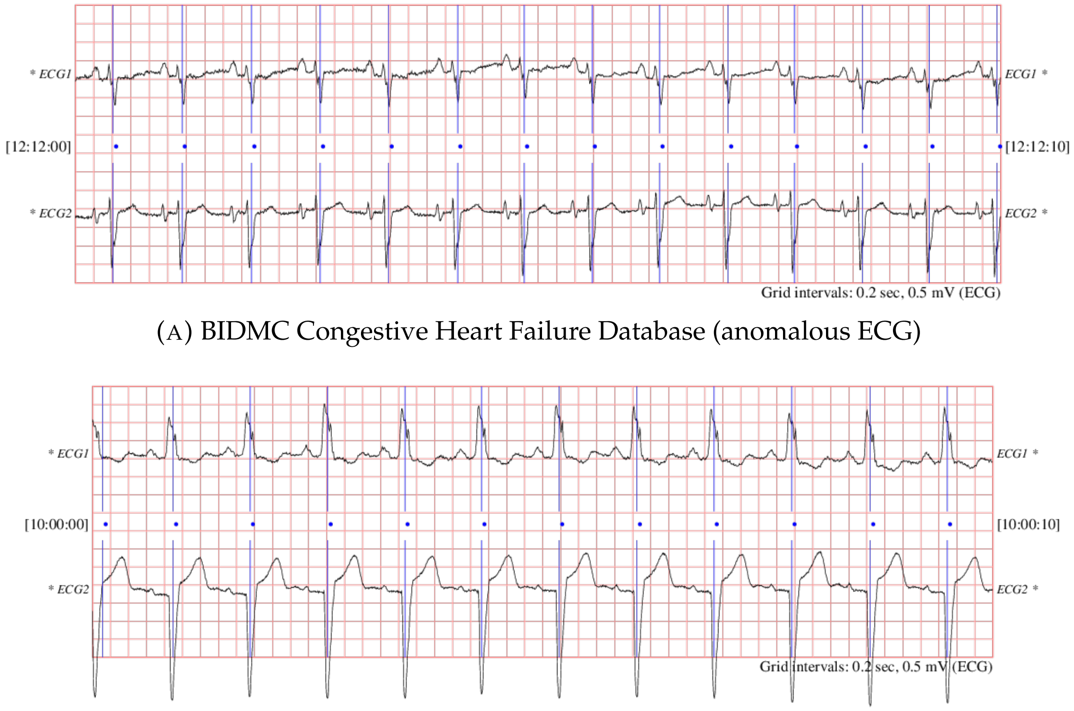
Algorithm 5.2: The XAI module used in the proposed framework AnoFed

```
**Input:** Anomalous ECG signal, desired window size $W_s \leq n$
**Output:** Segment with the maximum reconstruction loss
calculate the possible number of sub-segments $s$ and set the start
  position $S_{\text{pos}} = 0$, end the position $E_{\text{pos}} = W_s$ $\text{max}_{\text{loss}} = 0$,
  $\text{max}_{\text{loss-position}} = (0, 0)$
for $i$=1, 2, ..., $s$ do
    $\text{SLoss} = \text{abs}(\text{mean}(X[S_{\text{pos}} : E_s] - X'[S_{\text{pos}} : E_s])^2)$
    if $\text{SLoss} > \text{max}_{\text{loss}}$ then
        $\text{max}_{\text{loss-position}} = (S_{\text{pos}}, W_s)$
        $\text{max}_{\text{loss}} = \text{SLoss}$
    $S_{\text{pos}} = S_{\text{pos}} + W_s$
    $E_{\text{pos}} = E_{\text{pos}} + W_s$
Return $\text{max}_{\text{loss}}, \text{max}_{\text{loss-position}}$
```

of which 11 were men (aged 22 to 71), and 4 were women (aged 54 to 63). Further details about the data are available in [189]. The data was pre-processed by extracting each heartbeat of equal length using interpolation, and the class values were obtained by automated annotations [190]. For normal subjects, we use the Massachusetts Institute of Technology-Beth Israel Hospital (MIT-BIH) normal sinus rhythm [189]. It includes 18 longer-term ECG recordings of 18 subjects (5 men, aged 26 to 45, and 13 women, aged 20 to 50) with no significant arrhythmias. We randomly selected 5,000 (2919 normal samples and 2081 anomaly) heartbeats from each dataset to train and test the proposed framework AnoFed. Figure 5.4 shows one example sample from each of the two selected databases. Table 5.1 presents additional information about the datasets used. It should be noted that the anomaly class contains different sub-classes of anomalies in it.

TABLE 5.1: More information about the datasets used to train and test the proposed framework AnoFed.

Class Name	Class ID	Number of Samples
Normal	1	2919
Anomaly	-1	2081



(A) BIDMC Congestive Heart Failure Database (anomalous ECG)

(B) MIT-BIH Normal Sinus Rhythm Database (normal ECG)

FIGURE 5.4: An example sample from each of the two used databases.

5.3.2 Implementation Details

In order to evaluate AnoFed in real time, firstly we developed a testbed using three Raspberry Pi devices (Pi 3 Model B+ with 1.4GHz, 1GB LPDDR2 SDRAM, and 64-bit quad-core ARMv8 CPU) as clients, as shown in Figure 5.5 and a Dell workstation with 32 GB RAM and an Intel® Core™ i-6700HQ CPU as **GS**. For the initial ($r = 0$) global round, **GS** compiles the AE or the VAE. We used Adam as the optimizer for both the AE and the VAE. We distributed the above-mentioned datasets equally (but randomly selected) among the three edge devices. 75% of the data was used for training by each client or edge, whereas the rest 25% for testing. Secondly, we increase the number of clients to five. In the second setting, we used a non-IID data (unbalanced and skewed) distribution (Edges 3 and 5 with around 630 training samples each, where Edge 3 contains 60% data from the normal class and 40% data

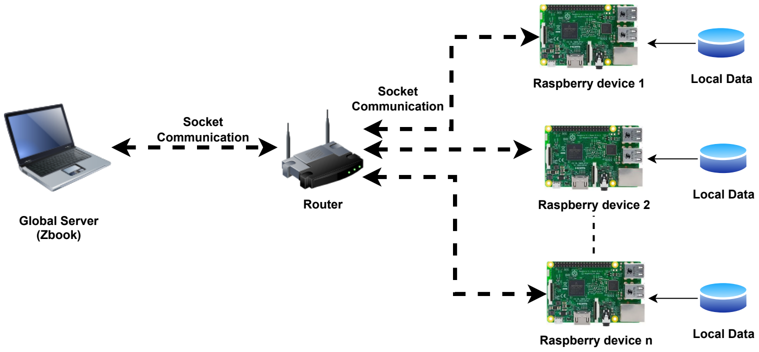


FIGURE 5.5: System diagram of the testbed

from the anomaly class. Edge 5 contains 60% data from the anomaly class and 40% data from the normal class). Each of other clients contain around 1,200 samples selected randomly. 70% of the data was used for training and 30% was used for testing in each client. In both settings, we also kept 1,000 randomly selected samples (not part of the training set in any edge device) to test the global model. Additionally, we used 10-fold cross-validation in both settings. Since our task is to use the reconstruction loss to predict anomalies, we used the normal data only for training the AE and the VAE. Furthermore, each edge used a batch size of 42, and was trained for only three epochs in each global round. The ability of our proposed AE/VAE to achieve the minimum reconstruction loss within three epochs and one global round makes it suitable for resource-constrained devices, which is much needed in many health-related applications. We used a learning rate of 0.001, with a clip value of 0.5. We used mean squared error (MSE) as the loss function. In order to find the best suitable bandwidth for kernel estimation in SVDD, we used sklearn's grid search module with bandwidth space of (3, 0.2,10), and 30-fold cross validation[1].

[1] The cross-validation parameter 30 was empirically determined to get better results for the SVDD classifier.

5.4 Performance Analysis of the Proposed Framework

In this section, we report the performance of the proposed framework AnoFed using some state-of-the-art metrics.

5.4.1 Reconstruction Loss

In order to evaluate the performance of AnoFed, we trained both the AE and the VAE in a federated setting following the experimental setup explained in the previous section. Figure 5.6 shows the reconstruction loss using the proposed AE and Figure 5.7 shows the reconstruction loss using the proposed VAE. It can be seen that both the AE and the VAE performed very well. We use the blue dotted line to show the point one standard deviation away to the right of the mean of the normal distribution, which can be chosen as a typical static threshold of the classifier. We can optimize this threshold by recursively trying other possible values. However, as mentioned previously, the anomaly detection methods using a static threshold are not compatible with the federated setting due to different distributions of local and global models. Hence, we decided to use SVDD along with density kernel estimation for adaptive anomaly detection in a federated setting, which allows us to avoid setting a static threshold. Although both the AE and the VAE have similar reconstruction losses, VAEs are considered more generalizable than AEs [191]. Therefore, we chose to use the error vectors computed using the VAE's predictions for kernel density estimation and SVDD training.

5.4.2 Classification Performance

To measure the classification performance, we used the classification performance metrics described in chapter 2.1.2.

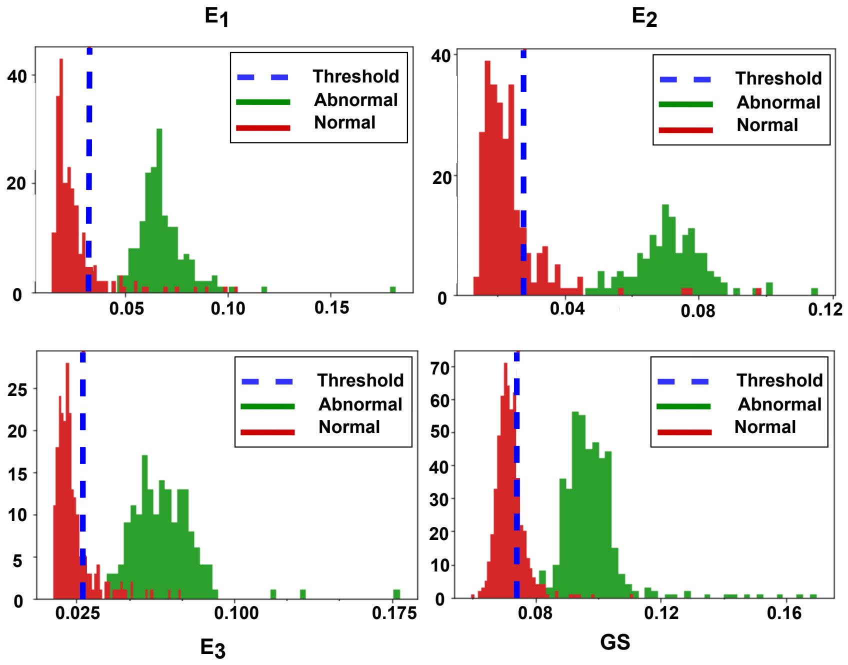


FIGURE 5.6: Reconstruction losses of the proposed AE in different models.

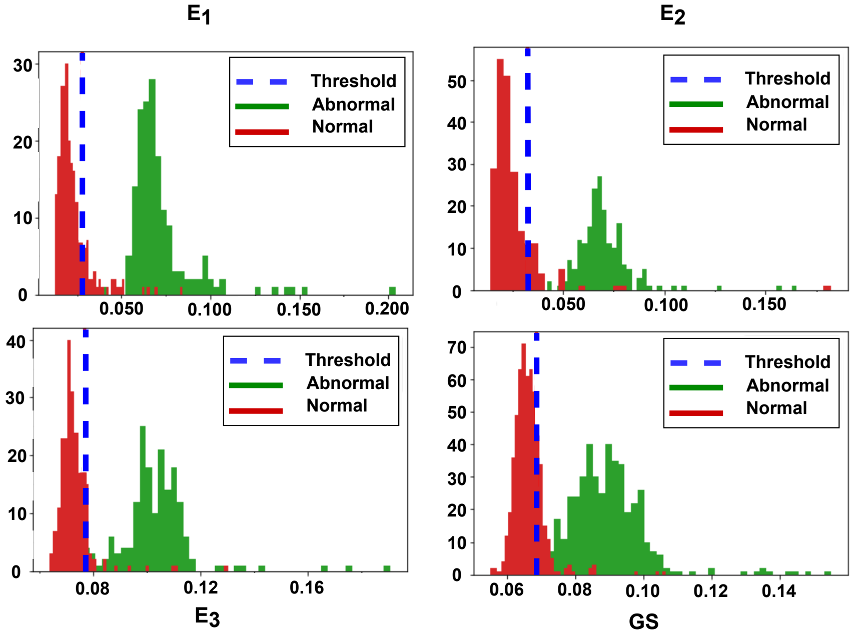


FIGURE 5.7: Reconstruction losses of the proposed VAE in different models.

Tables 5.2 and 5.3 present the anomaly detection performance of AnoFed using the proposed adaptive anomaly detection method in IID and non-IID data distribution among the clients, respectively, where the number of edge devices is also changing per data distribution setting. It can be seen that the proposed method not only achieved state-of-art performance for local models but also for the global model. As mentioned before, our method does not require prior knowledge about the distribution of the underlying data, as it can automatically adapt to the changing distribution when new data come in.

Figures 5.8 and 5.9 show that AnoFed is able to separate the normal and anomaly ECG test samples efficiently both locally and globally for both IID and non-IID settings, respectively. Any test samples with a distance more than the radius (red line) of the normal class are classified as an anomaly.

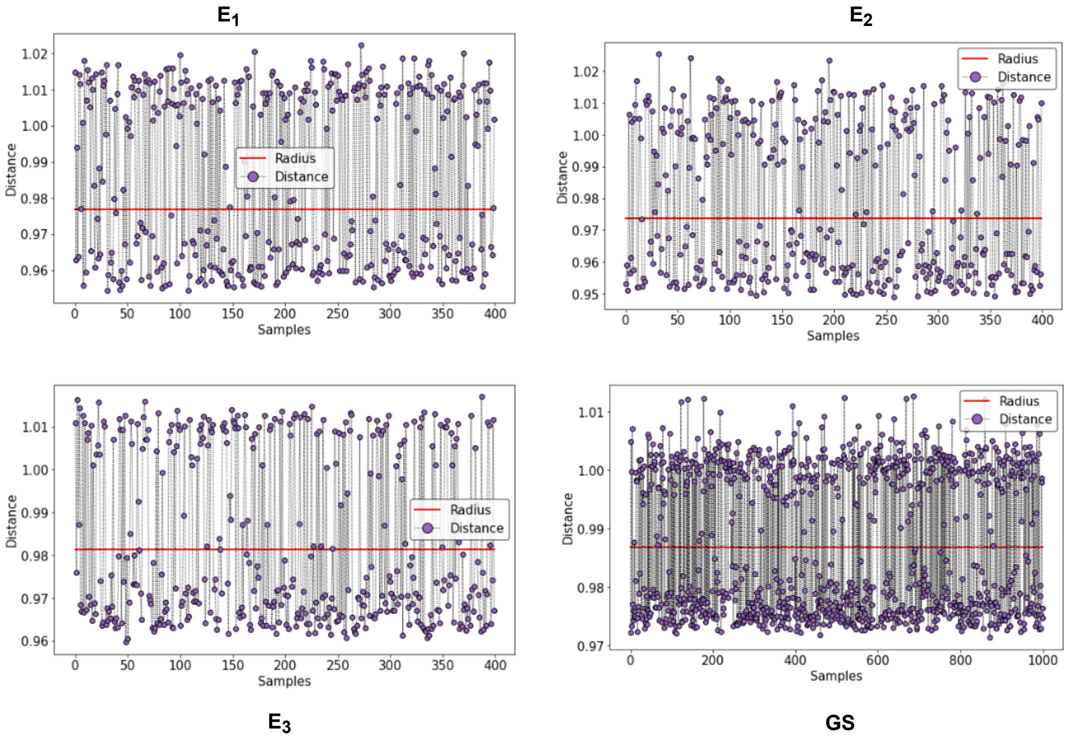


FIGURE 5.8: Hyperspheres obtained using AnoFed for IID data.

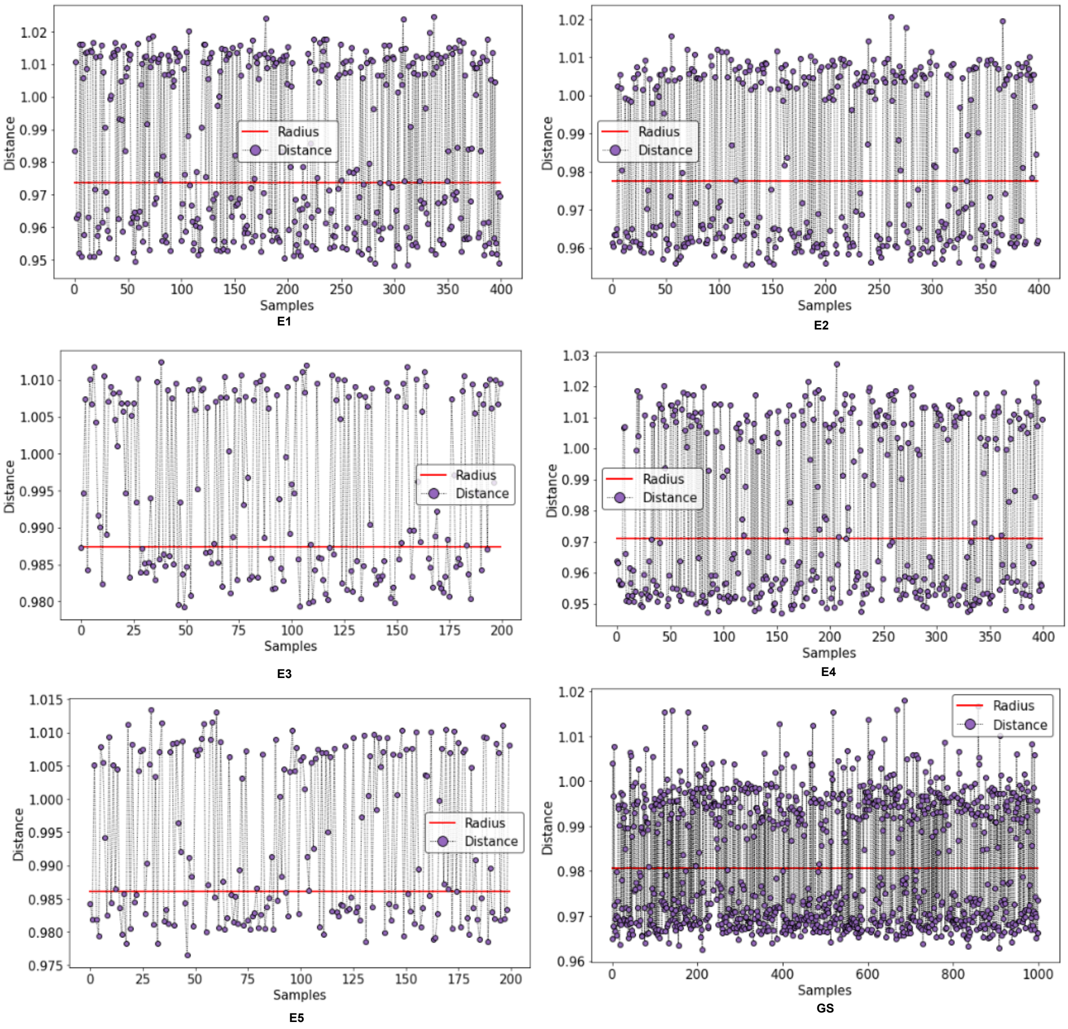


FIGURE 5.9: Hyperspheres obtained using AnoFed for non-IID data with increasing number of clients.

5.4.3 Explainability with XAI module

In order to trace back the segments of the input ECG sample to build trust among the user we proposed an XAI module as discussed previously. In this subsection, we show with a sample test example how efficiently the proposed XAI module can trace back the segments of the ECG signal responsible for maximum reconstruction loss. Figure 5.10 shows an example output of the proposed XAI module. We used a window size of 10 timestamps. Hence, each input sample is divided into 14 sub-segments. It can be seen that segments 14, 6, and 10 of samples 1, 2, and 3 are highlighted in red showing that these segments have the maximum reconstruction loss (the reconstruction

TABLE 5.2: The classification performance of the proposed framework (the adaptive approach).

Class	*Precision*	*Recall*	*F1-Score*	*Accuracy*
Normal	100%	96.0%	98.0%	97.6%
Anomaly	95.0%	99.0%	97.0%	

(A) Edge 1

Class	*Precision*	*Recall*	*F1-Score*	*Accuracy*
Normal	100%	97.0%	99.0%	98.1%
Anomaly	96.0%	100%	98.0%	

(B) Edge 2

Class	*Precision*	*Recall*	*F1-Score*	*Accuracy*
Normal	99.0%	98.0%	98.0%	98.0%
Anomaly	96.0%	99.0%	97.0%	

(C) Edge 3

Class	*Precision*	*Recall*	*F1-Score*	*Accuracy*
Normal	100%	97.0%	98.0%	98.8%
Anomaly	97.0%	99.0%	98.0%	

(D) Global Server (GS)

loss for each segment is given on the top of each column). In other words, the segments highlighted in red contribute more to the reconstruction loss as compared to others, thereafter for the anomaly.

5.4.4 Comparison

In this subsection, we compare AnoFed with some state-of-the-art methods [106], [192]–[197], in terms of desired features provided and the overall detection accuracy. Table 5.4 presents the results of the comparison. It can be seen that AnoFed provides desirable properties such as enhanced privacy protection (because FL employed in the framework allows peers in the network to train a global model without sharing local healthcare data, but only sharing trained parameters that reveal less information compared with the case when the raw local data is shared with the global server directly), explainability, and adaptive anomaly detection, while others lack some of the

TABLE 5.3: The classification performance of the proposed framework (the adaptive approach) with non-IID data.

Class	*Precision*	*Recall*	*F1-Score*	*Accuracy*
Normal	98%	97.0%	98.0%	97.0%
Anomaly	97.0%	99.0%	97.0%	

(A) Edge 1

Class	*Precision*	*Recall*	*F1-Score*	*Accuracy*
Normal	100%	96.0%	98.0%	98.0%
Anomaly	96.0%	99%	98.0%	

(B) Edge 2

Class	*Precision*	*Recall*	*F1-Score*	*Accuracy*
Normal	94.0%	97.0%	95.0%	94.0%
Anomaly	95.0%	91.0%	93.0%	

(C) Edge 3

Class	*Precision*	*Recall*	*F1-Score*	*Accuracy*
Normal	99.0%	96.0%	97.0%	97.0%
Anomaly	95.0%	99%	97.0%	

(D) Edge 4

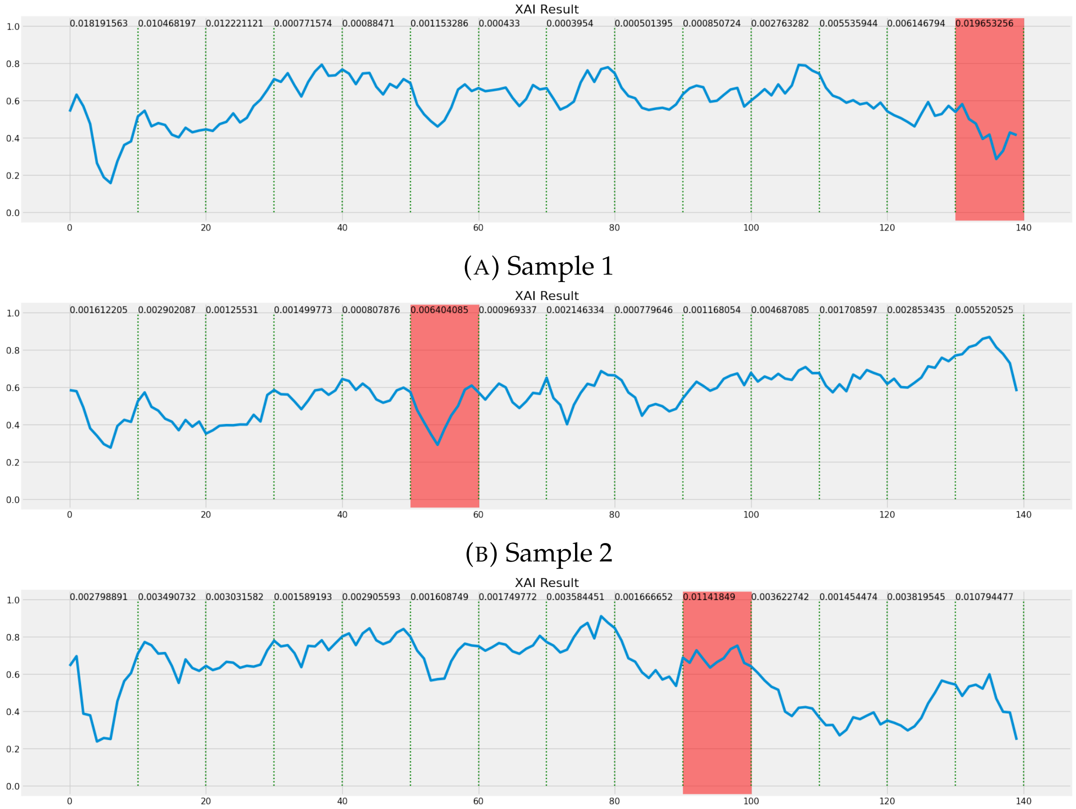
Class	*Precision*	*Recall*	*F1-Score*	*Accuracy*
Normal	91.0%	99.0%	94.0%	93.0%
Anomaly	98.0%	87.0%	92.0%	

(E) Edge 5

Class	*Precision*	*Recall*	*F1-Score*	*Accuracy*
Normal	100.0%	96.0%	98.0%	98.0%
Anomaly	96.0%	100%	98.0%	

(F) Global Server (GS)

desired properties. Table 5.5 presents comparison of selected state-of-the-art methods [106], [192]–[196] with AnoFed in terms of the overall detection accuracy. It can be seen that AnoFed achieved a performance either comparable to or better than the compared methods, with an overall accuracy of 98.8%. Moreover, we achieved state-of-the-art accuracy by training AnoFed with just three local rounds and one global round, which makes it computationally efficient for resource-constrained devices. It should be noted that AnoFed was evaluated in a federated setting, while all others are centralized so less privacy-friendly as mentioned previously.



(A) Sample 1

(B) Sample 2

(C) Sample 3

FIGURE 5.10: An example sample showing how the XAI module helps achieve explainability.

TABLE 5.4: Comparison with selected state-of-the-art methods (key features).

Scheme	*Explainability*	*Adaptivity*	*Enhanced Privacy Protection*
[106]	✗	✗	✗
[192]	✗	✗	✗
[193]	✗	✗	✗
[194]	✗	✓	✗
[195]	✗	✗	✗
[196]	✗	✗	✗
[197]	✗	✓	✗
Proposed	✓	✓	✓

5.4.5 Time Complexity of AnoFed

In this subsection, we present the time complexity for the entire pipeline of AnoFed. Since the number of transformer layers and the multi-head attention is constant, i.e., they do not depend on the input size, the dot product in multi-head attention for a given input of size n features takes $O(n)$ time.

TABLE 5.5: Comparison with selected state-of-the-art methods (detection accuracy).

Scheme	*Centralized or Federated*	*Accuracy (%)*
[106]	centralized	–
[192]	centralized	99.8
[193]	centralized	95.0
[195]	centralized	96.0
[196]	centralized	99.3
[197]	centralized	95.0
[198]	federated	70.0 (F1-score)
[199]	federated	96.94 (Multi-class)
Proposed	federated	98.8

Moreover, each output is the sum product of k features of the input, with a fixed number of weights, which are not dependent on n. Similarly, computation on the activation function takes a linear amount of time. Furthermore, all the edge or client devices perform in parallel, therefore, the overall run-time is linear, i.e., bounded by $O(n)$.

In our experiments, AnoFed took 74.3 seconds to complete one global round of training. This time is further divided as follows: training the AE/VAE for 3 rounds took around 35.4 seconds and training the SVDD took around 38.9 seconds in a federated setting. Additionally, the framework took 0.93 seconds to test a sample. It should be noted that the actual run-time performance of the entire pipeline can vary depending on the implementation details such as the hardware used, and the number of samples each local model has.

5.5 Summary

Anomaly detection is one of the important tasks to address when it comes to digital healthcare with machine learning. Deep learning-based models can achieve state-of-the-art results, but when being applied in a centralized setting, they suffer from data privacy, data availability, trust, and unknown data distribution issues when used for sensitive applications like anomaly detection in healthcare. In this paper, to address such challenges, we proposed AnoFed, a FL-based anomaly detection framework, to facilitate collaborative learning with distributed edge servers working with a global server. In order to facilitate FL with resource-constraint edge devices, we proposed a lightweight VAE and an AE based on transformers, which are used to minimize the reconstruction loss within three training epochs of each global round. To enhance the performance of the FL with a static threshold, we proposed to use kernel density estimation-based SVDD, which can provide adaptive anomaly detection without setting a hard threshold. AnoFed can address issues such as estimating the underlying data distribution automatically with each global round of FL for efficient and accurate anomaly detection. Additionally, we proposed an XAI module to provide some level of explainability to the results of AnoFed, by tracing back the major segments of the input that are responsible for a detected anomaly. Lastly, we tested the proposed framework by combining two benchmark datasets from PhysioNet's repository and showed that AnoFed achieved up to 98.8% test accuracy with 10-fold cross-validation with changing distributions of data. We also compared AnoFed with a number of selected state-of-the-art methods, showing comparable results on the performance, but with new desired features such as better privacy protection, adaptive anomaly detection, and enhanced explainability.

Although the proposed frameworks in this chapter and previous chapters can provide explanations, lightweight computation, and enhanced privacy, with SOTA performance, we still need methods to address security issues such as poisoning attacks which can potentially degrade the performance of the global model and thereafter limits the application of FL if such issues are not addressed. Consequently, to overcome these limitations, in the upcoming chapter, we will present a comprehensive framework specifically designed to address poisoning attacks in Fl. By identifying and mitigating the risks associated with poisoning attacks, this framework aims to enhance the effectiveness and security of Fl applications. Through our proposed framework, we will provide practical strategies and techniques to detect, prevent, and mitigate the adverse impact of poisoning attacks, thereby ensuring the reliability and integrity of the FL process

Chapter 6

Using Anomaly Detection to Detect Poisoning Attacks in Federated Learning Applications

6.1 Introduction

Privacy and security are among the top issues to be addressed in privacy and security-sensitive applications of ML, such as healthcare, autonomous vehicles, etc. As mentioned earlier, FL enhances data privacy in ML applications [112]. Nevertheless, the global model in FL can be easily manipulated, even if a single-edge device is compromised [35]–[37]. The attack surface of FL is growing due to its distributed nature. For example, malicious peers can launch data poisoning [125], [126] or model poisoning [127] attacks, in which one or more malicious edge devices manipulate their local training data or the local model trained on benign data, to impair the performance of the updated global model. In other words, in FL, data is not collected at a single server and there are multiple devices for collecting and analyzing data. Such distributed settings increase the chances of data poisoning attacks, hence, methods should be developed for data integrity and authentication

6.1.1 Contributions

To overcome the weaknesses of the SOTA defense measures mentioned in chapter 2, in this chapter we made the following key contributions:

- We propose a novel framework for detecting poisoning attacks in FL, which employs a reference model (used to check stitching connectivity) based on a public dataset and an auditor model to detect malicious updates. It can reach the lowest possible computational complexity $\mathcal{O}(K)$ where K is the number of clients.
- We implemented a detector based on the proposed framework to detect poisoning attacks.
- We evaluate our detector's performance against state-of-the-art (STOA) poisoning attacks for two typical applications of FL: electrocardiograph (ECG) classification and human activity recognition (HAR). Our experimental results showed that our detector can indeed overcome all the above-mentioned weaknesses.

The rest of the Chapter is organized as follows. Section 6.2 presents the proposed framework. Sections 6.3 and 6.4 cover the performance evaluation and comparison of the proposed framework, respectively. Finally, the last section summarizes the Chapter.

6.2 Proposed Framework

6.2.1 Threat Model

Attacker's goal: Similar to many other studies [125], [129], [130], we consider an attacker whose goal is to manipulate the global model in such a way that it has low performance (i.e., high error rates) and/or misbehaves in a particular way. Such attacks make the global model underperform. For example, an attacker can attack competitor FL systems. We consider both targeted and untargeted [200] attacks, as discussed previously.

Assumptions: We consider the following assumptions about the threat model:

1. There are one or more malicious edge devices who try to launch a model poisoning attack against the global model, possibly in a collaborative manner (i.e., launching a colluding attack).

2. All the attackers follow the FL algorithm, i.e., they train their local model using their local data and share the parameters with the global model.

3. All the attackers have two strategies: manipulating their local training data and train the local model using the manipulated data, and manipulating the model parameters after training it on benign data.

4. All the attackers have the full knowledge about the aggregation rules, the global model architecture, the auditor model (see Section 6.2.2 for more details about the auditor model), and the detection results, i.e., the whole detection framework is a white box and can be used as an oracle to adapt the attackers' strategy.

5. The global server can trust a third party or an isolated component, which maintains a public dataset that has data representing all classes of the underlying application for training the auditor model, and the attackers cannot poison this public dataset or the auditor model.

Note that the final assumption may look very strong, but the existence of a public dataset is common in many fields (e.g., a public health dataset maintained by the scientific community).

6.2.2 Overview

In this subsection, we describe the proposed framework. An overview of the proposed method has been shown in Figure 6.1. Let us assume K edged devices (hospitals, organizations, etc.) collaborate to train a joint global model **GM**. An edge $\mathbf{E}_k$ trains a local model $\mathbf{LM}_k$ using its local data $\mathbf{D}_k$, where $k = 1, 2, \ldots, K$. The global server **GS** is responsible for receiving the updates from edge devices and aggregation. We assume that a trusted third party or **GS** (We consider trusted third party to be a different entity from **GS**) also has an open-source dataset which is called public data and we represent it as **DP**. **DP** is supposed to be a representative dataset of all the classes in a classification problem, i.e., it has samples from each candidate class. The training for a global round is given as follows.

1. Trusted third party creates an audit model **AM** and a reference model **RM**. Where **RM** has the same architecture as global model **GM**.

2. Trusted third party splits **DP** into train $\mathbf{DP}_{\text{train}}$ and test $\mathbf{DP}_{\text{test}}$ datasets and trains the **RM** using $\mathbf{DP}_{\text{train}}$.

3. After training, trusted third party makes predictions with **RM** using $\mathbf{DP}_{\text{train}}$ and $\mathbf{DP}_{\text{test}}$. During the predictions for each dataset, trusted third party taps the activations of the last hidden layer for each input sample using Algorithm 6.1 to create a dataset $\mathbf{DA}_{\text{train}}$, and $\mathbf{DA}_{\text{test}}$ for each $\mathbf{DP}_{\text{train}}$ and $\mathbf{DP}_{\text{test}}$, respectively.

4. Trusted third-party trains the audit model (a one-class classifier) using $\mathbf{DA}_{\text{train}}$. Here, we treat all the samples of $\mathbf{DA}_{\text{train}}$ as a single class. Trusted third party sends the trained **AM**, **RM**, $\mathbf{DP}_{\text{test}}$ and a value P to **GS**. We define P later in the section.

5. Each $\mathbf{E}_i$ trains $\mathbf{LM}_i$ using $\mathbf{D}_i$.

6. Each $\mathbf{E}_i$ sends updates W_i of trained $\mathbf{LM}_i$ to **GS**.

7. **GS** sets W_i as the parameters of **RM** and makes predictions using $\mathbf{DP}_{\text{test}}$. During the predictions, **GS** taps the activations of **RM** for each input sample using Algorithm 6.1 to create a dataset $\mathbf{DA}_i$.

8. **GS** makes predictions using **AM** and $\mathbf{DA}_i$ as input data. For every input sample $X \in \mathbf{DA}_i$, **AM** outputs $y_i \in \{1, -1\}$ and creates a set $Y = \{y_1, y_2, \ldots, y_z\}$, where z is the total number of samples in $\mathbf{DA}_i$.

9. **GS** computes poisoned rate $h_i = \frac{o \times 100}{z}$, where o is the total number of $-1's$ in Y.

10. **GS** includes W_i in global aggregation if $h_i \leq P$, otherwise discards W_i. Here, $P = h_{\text{test}} + \alpha\sigma$, and it is the percentage of poison that we want to tolerate. σ is called deviation tolerance and it is given as $\sigma = |h_{\text{test}} - h_{\text{train}}|$. h_{test} and h_{train} are calculated using $\mathbf{DP}_{\text{train}}$ and $\mathbf{DP}_{\text{test}}$, respectively. α is a parameter to make the threshold flexible. In our experiments, we set $\alpha = 1$.

Algorithm 6.1: Creation of Audit Dataset(s) $\mathbf{DA}_i$

```
Input: A dataset $\mathbf{D}_i$ and a trained model $\mathbf{M}$
Output: a new dataset $\mathbf{DA}_i$
for $X \in \boldsymbol{D}_i$ do
    Transform $X$ into batch size
    Make Prediction using $X$ as a test sample in $\mathbf{M}$
    Get activation maps $A^k_{l,w}$ of the last convolutional layer. where $k$ is
      the number of activation maps with length $l$ and width $w$ each.
    Get probability score $y^c$, where $c$ is the class label of $X$.
    Reshape $A^k$ as an array of $1 \times j$, where $j = l \times w \times k$.
    Reshape $X$ as an array of $1 \times l$, where $l$ is the product of the height
      and width of $X$
    Compute $s = X \parallel A^k \parallel y^c$
    Append $s$ in $\mathbf{DA}_i$ as a new data sample
Output $\mathbf{DA}_i$
```

The proposed framework works based on the following two propositions.

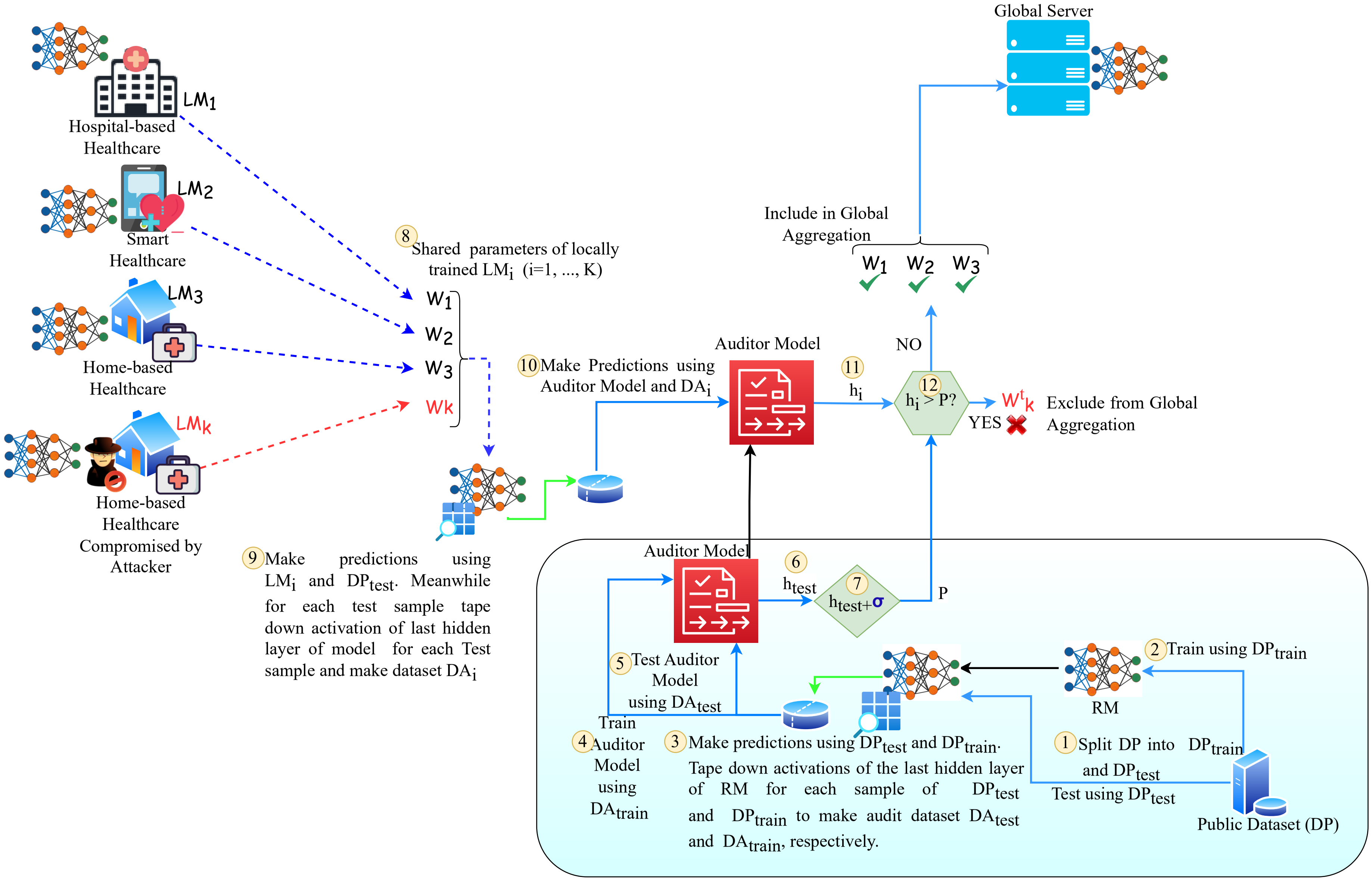

Global Server
LM1
Hospital-based Healthcare
LM2
Smart Healthcare
LM3
Home-based Healthcare
LMk
Home-based Healthcare Compromised by Attacker
8 Shared parameters of locally trained LMi (i=1, ..., K)
W1
W2
W3
Wk
Include in Global Aggregation
W1 W2 W3
✓ ✓ ✓
9 Make predictions using LMi and DPtest. Meanwhile for each test sample tape down activation of last hidden layer of model for each Test sample and make dataset DAi
10 Make Predictions using Auditor Model and DAi
Auditor Model
11 hi
NO
12 hi > P?
YES
W'k
Exclude from Global Aggregation
Auditor Model
6 htest
7 htest+σ
P
5 Test Auditor Model using DAtest
Train
4 Auditor Model using DAtrain
3 Make predictions using DPtest and DPtrain. Tape down activations of the last hidden layer of RM for each sample of DPtest and DPtrain to make audit dataset DAtest and DAtrain, respectively.
2 Train using DPtrain
RM
1 Split DP into DPtrain and DPtest
Test using DPtest
Public Dataset (DP)

FIGURE 6.1: Overview of the proposed framework

Proposition 1 *When a model is trained on noisy data (malicious/poisoned), the first half of the layers are similar to a model trained on good-quality data (benign).*

Proposition 2 *Different models with the same architecture but random initial seeds, trained on different training sets of a similar distribution, have similar internal representations and thereafter similar activations for a given test input sample.*

Based on Proposition 2, we expect that the models trained on different datasets of similar distributions will behave similarly. Since the information specific to the data is learned by the higher-level layers of the model, we take the activation of the last hidden layer (in the case of CNN's last convolutional layer) to capture more information (features) related to the training data. We only consider the activations of the last as it provides sufficient information about the underlying training data to detect poisoning attacks. Activations from other higher-level layers can also be incorporated for this purpose, nevertheless, this increases the computational costs with no significant improvement in the results for detecting poisoning attacks. Hence, we only consider the activations of the last convolutional hidden layer. We capture this property by training another model (**AM**) to learn the behavior of a model trained on the benign dataset (**RM** in our case). Hence, models behaving similarly to the model trained on the benign datasets are probably free from poisoning attacks, and at least they do not degrade the performance of the global model.

6.3 Performance Evaluation

We evaluated the proposed framework using two healthcare applications, i.e., ECG classification and HAR.

6.3.1 Experimental Setup

Datasets: For ECG classification, we use the widely known MIT-BIT arrhythmia dataset [157]. The dataset contains 48-half-hour two-channel ECG recordings. These recordings were obtained from 47 subjects. The dataset contains 109,446 samples, sampled at a frequency of 125 Hz. Further, the dataset contains five classes of ECG: non-ecotic beats (normal beats), supraventricular ectopic beats, ventricular ectopic beats, fusion beats, and unknown beats.

For HAR, we used the dataset in [174], [201]. The dataset contains time-series data related to 14 different human activities (standing, sitting, walking, jogging, going up-stairs, going down-stairs, eating, writing, using a laptop, washing face, washing hands, swiping, vacuuming, dusting, and brushing teeth) collected using sensors such as accelerometers, magnetometers, and gyroscopes.

Classifiers: We developed a convolution neural networks (CNN) based classifier for each application. The developed classifiers do not achieve the optimum classification for the considered datasets. This is because our objective is to show that our proposed framework can detect anomalous (poisoning attacks), not to achieve the best performance in terms of classification. For ECG classification we developed a five-class classifier and the aim of FL here is to learn a global five-class classifier, and for HAR we developed a fourteen-class classifier, and the aim of FL here is to learn a global fourteen-class classifier.

Federated Setting: To simulate the federated setting, we simulated a network with three edge devices (two benign ones and one attacker) where the three edge devices were implemented using Tensorflow 2.11 and Python 3 in a Dell latitude laptop with $12^t h$ Gen Intel® Core™ i7-1265U processor and 16 MB DDR4 RAM and a global server using Tensorflow 2.11 and Python 3 in a Dell workstation with 32 GB RAM and an Intel® Core™ i-6700HQ

CPU. Furthermore, we divided the dataset equally, but randomly among the participating edge devices for each application, i.e., ECG classification and HAR.

6.3.2 Performance Evaluation

In order to evaluate the performance of the proposed framework, we first tested the proposed framework to check its ability to differentiate samples in $\mathbf{DA}_i$ of benign edges from samples in $\mathbf{DA}_i$ of malicious edges. We trained a **RM** and an **AM** using a public dataset (which is excluded from the training data of the edge devices and the test data). We distributed each candidate dataset (ECG and HAR) equally (in the case of the HAR dataset, benign nodes contain slightly more data) but randomly among the benign and malicious nodes. For each dataset, we did the following. We created $\mathbf{DA}_i$ using each benign edge and labeled each sample as 1 (for benign). Similarly, for the malicious edge, we created $\mathbf{DA}_i$ using each type of model and data poisoning attack discussed in Section 2.4.1 and selected random samples from each attack's $\mathbf{DA}_i$ to make a new $\mathbf{DA}_i$ and labeled each sample as -1. We combined the two $\mathbf{DA}_i$'s that we created and labeled as -1 and 1. Then, this lastly created dataset was used to test the **AM**. Note that here we just show the ability of the proposed framework to classify benign and malicious samples not the detection of updates. We will show the detection of updates later. Table 6.1 shows the classification accuracy of the proposed framework to differentiate samples generated using shared parameters of benign edged devices and the samples generated using shared parameters of malicious edge devices for ECG classification. Similarly, Table 6.2 shows the classification accuracy of the proposed framework to differentiate samples generated using shared parameters of benign edged devices and the samples generated using shared parameters of malicious edge devices for HAR. For both types of applications, it can be seen that the proposed framework can differentiate samples of benign and malicious edged devices very well, with an overall accuracy of 94% and 99% for HAR and ECG classification, respectively. As mentioned previously, here we calculate the overall accuracy of **AM** to classify benign and malicious samples which is 94% and 99% for HAR and ECG respectively. It should be noted that the poison attack detection is still 100% as updates with 94% and 99% poisoned samples will have a poison rate of around 94% and 99%, respectively. Hence, updates with 94% and 99% poison rates will be classified as malicious and will be removed from global aggregation.

TABLE 6.1: Classification accuracy of **AM** for ECG classification

Class	*Precision*	*Recall*	*F1-Score*	*#(Samples)*
Benign	100	97	98	9,000
Malicious	97	100	99	9,000
Accuracy			99	18,000
Micro average	99	99	99	18,000
Weighted average	99	99	99	18,000

TABLE 6.2: Classification accuracy of **AM** for HAR

Class	*Precision*	*Recall*	*F1-Score*	*Support*
Benign (1)	96	95	95	22,280
Malicious (-1)	90	91	91	11,140
Accuracy			94	33,420
Micro average	93	93	93	33,420
Weighted average	94	94	94	33,420

Now, we show the ability of the proposed framework to detect malicious updates and remove them from the global aggregation. We simulated the proposed framework as follows. First, the trusted third party prepares the **RM** and the **AM** and trains the **RM** and the **AM** using the public data and sends it to **GS**. Then the benign devices and the attacker send the updates to the **GS**. The attacker launches a given attack at each global round. For example, for round r it uses a random label flipping attack and for $r + 1$ the attacker launches a feature poisoning attack, and so on. We tested the performance of our proposed framework under all the different types of model and data poisoning attacks listed in Section 2.4.1. Moreover, we distributed the data among the edged devices randomly but using non-independent and identically distributed unbalanced data distribution, where each edge has 40-50% fewer data for a given class. For example, edge 1 has 40% fewer samples of non-ecotic beats class for the ECG dataset as compared to other edge devices, and edge 2 has 50% fewer samples of supraventricular ectopic beats class compared to other edge devices. In comparison, edge 3 (attacker) has an equal number of samples for each class. This is done in order to give more attacking power to the edge3. For example, if the attacker has an equal number of samples of each class it can launch a random label flipping attack which will eventually affect all the classes and will affect the global model significantly. We follow similar distribution for the HAR dataset.

First, we present the comparison of the accuracy of the global model after one global round with and without our proposed framework, under four different data poisoning attacks. Note that our global model achieves optimum accuracy after one global round, hence we launched the attack in each global round. Otherwise, the attacks can be launched at any given global round. Figure 6.2 presents a comparison of the accuracy of the global model under different data poisoning attacks and with and without the proposed framework for ECG classification. Here, *GM* is a global model with our proposed framework under different data poisoning attacks, *LS* represents the global model under a label swapping attack without our proposed framework, *RLF* represents the global model under a random label and feature poisoning attack without our proposed framework, *RL* represents the global model under a random label attack without our proposed framework, and *FP* represents the global model under a feature poisoning attack without our

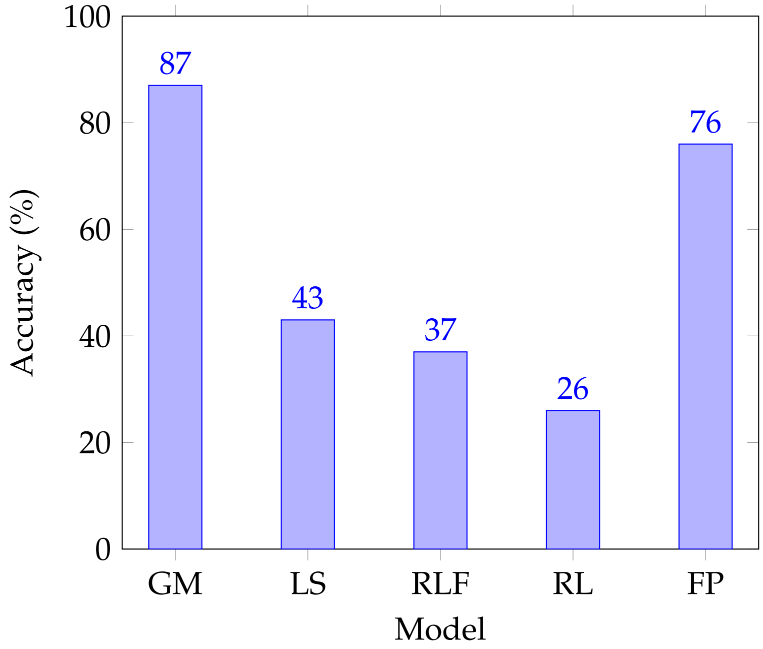


FIGURE 6.2: Performance of the global model with and without our proposed framework under four different data poisoning attacks on ECG classification. Here, GM presents the Proposed Framework, LS presents Label Swapping, RLF presents Random Label Feature poisoning, RL presents Random Label and FP presents Feature Poisoning.

proposed framework, all in a federated setting. It can be seen that the performance of the global model deteriorates significantly under the data poisoning attacks. It can also be seen that the performance of the global model was not affected by the poisoning attacks when we applied our proposed framework that could detect and eliminate almost all poisoned updates from the malicious edge before global aggregation. As mentioned previously, even if the accuracy of **AM** for benign and malicious sample detection is not 100%, it can accurately detect all the malicious updates because the poisoning rate is high. Hence, its malicious updates detection is still 100%. Similarly, Figure 6.3 presents the performance of the global model compared under four different data poisoning attacks with and without adopting the proposed framework for HAR. Since all the malicious updates are detected and removed before aggregation, the accuracy of the global model remains almost similar for a given dataset.

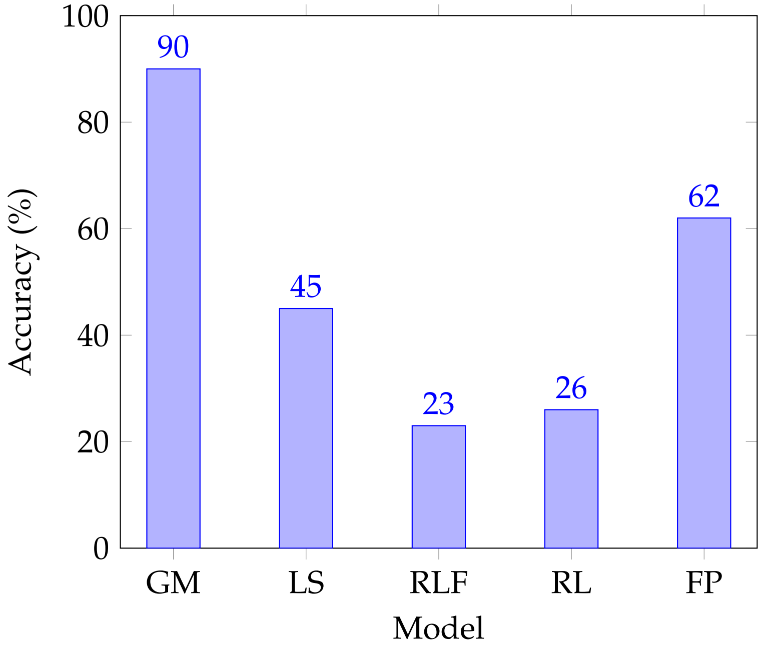


FIGURE 6.3: Performance of the global model with and without proposed framework under four different data poisoning attacks on HAR. Here, GM presents the Proposed Framework, LS presents Label Swapping, RLF presents Random Label Feature poisoning, RL presents Random Label and FP presents Feature Poisoning.

Moreover, Figure 6.4 presents a comparison of the accuracy of the global model under four different model poisoning attacks with and without our proposed framework for ECG classification. Here, *GM* shows the performance of the global model with our proposed framework, and *SF* shows the performance of the global model under a sign flip attack without our proposed framework, *SV* shows the performance of the global model under a same value attack without our proposed framework, *AGA* shows the performance of the global model under a performance additive Gaussian noise attack without a proposed framework, and *GA* shows the performance of the global model under a gradient ascent attack without our proposed framework, all in a federated setting. Similarly, Figure 6.5 presents a comparison of the accuracy of the global model under the four different model poisoning attacks with and without our proposed framework for HAR.

Table 6.3 shows the poisoning rate identified by the proposed framework

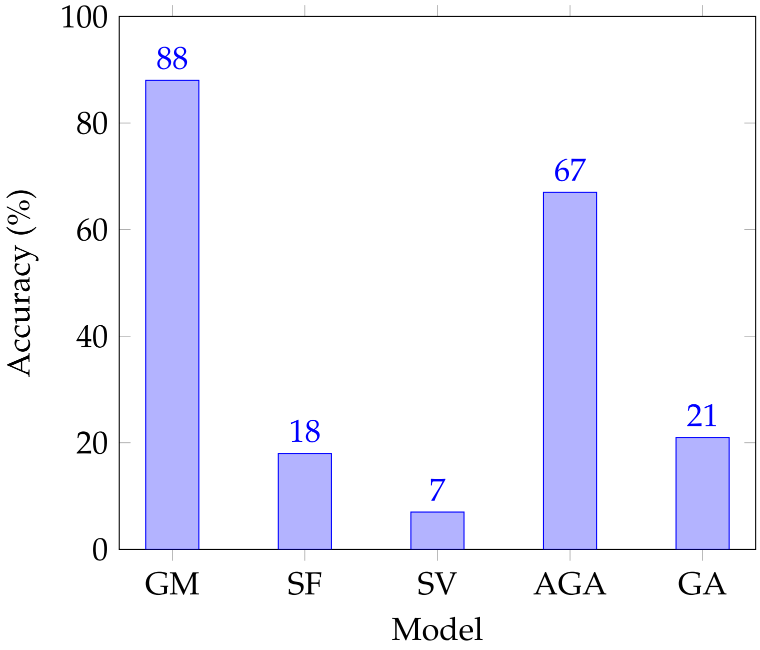


FIGURE 6.4: Performance of the global model with and without proposed framework under four different model poisoning attacks on ECG classification. Here, GM presents the Proposed Framework, AGA presents Additive Gaussian Noise, GA presents Gradient Ascent, SF presents Sign Flipping, and SV presents the Same Value attack

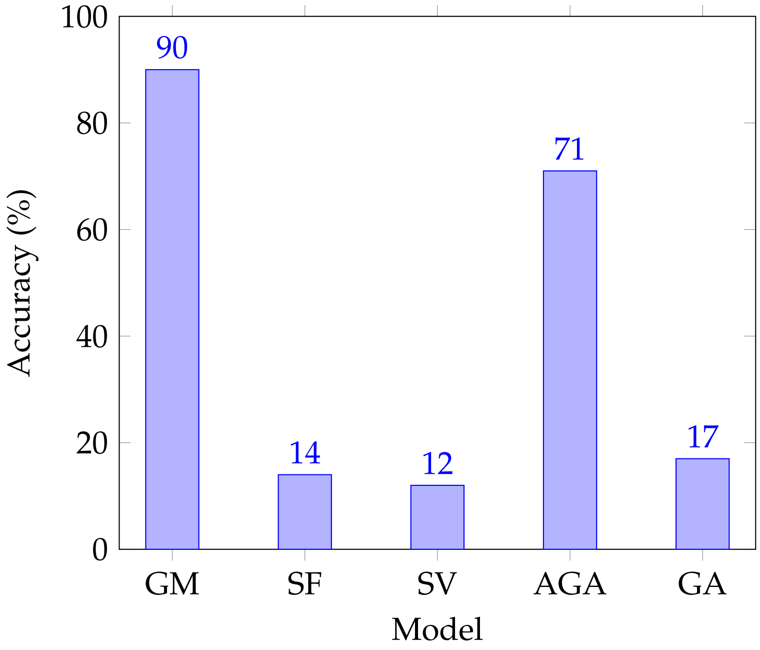


FIGURE 6.5: Performance of the global model with and without proposed framework under four different model poisoning attacks on HAR. Here, GM presents the Proposed Framework, AGA presents Additive Gaussian Noise, GA presents Gradient Ascent, SF presents Sign Flipping, and SV presents the Same Value attack

for each edge in the federated setting for HAR. Here Edge 1 and Edge 2 are benign devices and Edge 3 is the malicious device that launches different data poisoning attacks in each global round, whereas the P value in the table shows the threshold, which is calculated using h_{test} and σ. Any updates (W_i) from an edge device for which it's corresponding DA_i has a value of h_i greater than P will be marked as malicious or poisoned and removed from global aggregation. It can be seen that in Table 6.3, h_1 and h_2 for Edge 1 and Edge 2, respectively, have a value smaller than P. Hence, the updates W_1 and W_2 will be included in global aggregation, while W_3 will be excluded from global aggregation as its corresponding h_3 is greater than P. It can be seen that our proposed framework can not only detect the attacks but also provide insights into the percentage (a general idea about possible poison percentage) of poisoned data being used to train a malicious local model. For example, for LS we swapped the label of some classes (for ECG classification two classes and for HAR 10 classes) while keeping the rest of the labels in their original form, thereafter it gives a value of 47.7 for ECG and 81.2 for HAR. Similarly, Tables 6.4, 6.6 and 6.5 present the performance of our proposed framework to detect data poisoning attacks on ECG classification, model poisoning attacks on ECG classification, and model poisoning attacks on HAR, respectively. Hence, these results show that the proposed framework can identify malicious updates which are then removed from the global aggregation.

TABLE 6.3: Detection of data poisoning attacks on HAR

Attack Type	P ($h_{\text{test}} = 20.0, \sigma = 10.0$)	*Edge 1 (h_1)*	*Edge 2 (h_2)*	*Edge 3 (Malicious) (h_3)*
RLF	30.0	23.3	11.5	94.5
RL	30.0	23.5	10.0	100
LS	30.0	21.1	19.7	81.2
FP	30.0	18.3	16.1	91.0

TABLE 6.4: Detection of data poisoning attacks on ECG classification

Attack Type	P ($h_{\text{test}} = 15.0, \sigma = 10.0$)	*Edge 1* (h_1)	*Edge 2* (h_2)	*Edge 3 (Malicious)* (h_3)
RLF	25.0	10.2	14.5	100
RL	25.0	14.5	19.0	100
LS	25.0	12.1	21.3	47.7
FP	25.0	13.3	22.1	100

TABLE 6.5: Detection of model poisoning attacks on HAR classification

Attack Type	P ($h_{\text{test}} = 20.0, \sigma = 10.0$)	*Edge 1* (h_1)	*Edge 2* (h_2)	*Edge 3 (Malicious)* (h_3)
SF	30.0	20.3	19.5	100
SV	30.0	20.2	20.1	100
AGA	30.0	20.1	19.7	90.1
GA	30.0	20.3	20.1	100

TABLE 6.6: Detection of model poisoning attacks on ECG classification

Attack Type	P ($h_{\text{test}} = 15.0, \sigma = 10.0$)	*Edge 1* (h_1)	*Edge 2* (h_2)	*Edge 3 (Malicious)* (h_3)
SF	25.0	10.0	11.7	100
SV	25.0	10.2	10.1	100
AGA	25.0	9.9	9.7	96.1
GA	25.0	10.1	10.4	100

6.4 Comparison

Table 6.7 shows a comparison of our proposed framework with some of the STOA methods for detecting poisioning attacks in FL [36], [134], [135], [137], [138], [140], [141], [202]–[206], where the *Category* column presents the type of detection mechanism, attack type presents type of attack, the *Model Accuracy* column presents the accuracy of global model under attack compared with the accuracy of global model without any attack, the *Data Distribution* column presents the type of data distribution among clients: independent identically distributed (IID), non identically distributed (Non-IID), d represents model size (depth) of each client and K is the total number of edge devices/clients. It can be seen that our proposed framework provide more desirable features than all other STOA methods. Our proposed framework does not need knowledge about the number of attackers. It can detect both types of attacks, i.e., model and data poisoning attacks. Moreover, its time complexity is the least compared to others. Additionally, the proposed method can be used in both IID and non-IID data distribution with high accuracy.

6.5 Summary

In this Chapter, we present a novel framework to detect poisoning attacks in FL applications. Our proposed framework can efficiently detect SOTA data and model poisoning attacks by observing the activations of the shared weights of the local models. Unlike most of the existing methods, our proposed framework can detect poisoning attacks without degrading the global model's performance. In addition, while most of the existing methods need knowledge about the number of attackers in the network, which can limit their applications, our proposed method can detect poisoning attacks without any knowledge about the number of attackers in the network. Moreover, in principle, our proposed method can detect any number of attackers, especially if they do not collude, thanks to the use of the auditor model based on the public dataset, while many SOTA methods can only work up to a particular number of attackers. Additionally, the time complexity of our proposed framework is dependent only on the number of edge devices, with makes it suitable to be used with any size (in terms of depth) of the network, whereas the time complexity of existing SOTA methods depends on the number of edge devices as well as the size of the network. We tested our proposed framework under four different data poisoning attacks and four different model poisoning attacks for two healthcare applications, showing that the proposed framework could efficiently detect malicious updates and can exclude them from the global aggregation.

TABLE 6.7: Comparison with state-of-the-art methods.

Scheme (Year)	Category	Attack Type	Total Number of Attackers	Model Accuracy	Data Distribution	Time Complexity
[134] (2019)	Distance based	Data/Model	Less than 50%	Medium	IID	$\mathcal{O}(K^2d)$
[202] (2019)	Distance based	Data	Less than 30%	Medium	IID	$\mathcal{O}(K^2d)$
[140] (2018)	Statistic based	Data/ Model	Less than 50%	High	IID/Non-IID	$\mathcal{O}(K^2d)$
[36] (2017)	Distance based	Data/Model	Less than 50%	Medium	IID	$\mathcal{O}(K^2d)$
[203] (2018)	Distance-based	Data/Model	No limitation	High/Medium	IID/Non-IID	$\mathcal{O}(K^2d)$
[137] (2019)	Performance-based	Data/Model	less than 50%	High	IID/Non-IID	$\mathcal{O}(Kd)$
[138] (2019)	Performance-based	Data/Model	one honest user (At-least)	High	IID/Non-IID	$\mathcal{O}(Kd)$
[204] (2019)	Performance-based	Data/Model	No limitation	High	IID	$\mathcal{O}(Kd)$
[205] (2019)	Statistic based	Data/Model	less than 50%	High	IID	$\mathcal{O}(K^2d)$
[141] (2019)	Target optimization based	Data/Model	No limitation%	High	IID	$\mathcal{O}(Kd)$
[206] (2019)	statistic based	Data/Model	less than 50%	Medium	IID	$\mathcal{O}(Kd)$
[135] (2022)	Distance based	Data	less than 50%	High	IID/Non-IID	$\mathcal{O}(Kd)$
[136] (2022)	Distance based	Data/Model	less than 50%	High	IID/Non-IID	$\mathcal{O}(Kd)$
[207] (2022)	Distance based	Data/Model	less than 50% or known number of attackers	High	IID/Non-IID	$\mathcal{O}(Kd)$
[208] (2022)	Distance based	Model	less than 50%	High	IID/Non-IID	$\mathcal{O}(Kd)$
Proposed	Machine learning based	Data/Model	No limitation	High	IID/Non-IID	$\mathcal{O}(K)$

Chapter 7

Conclusions, Limitations and Future Work

7.1 Conclusions

In this thesis, we developed frameworks for high-performing, robust, and privacy and security-enhanced healthcare applications by leveraging the strengths of FL. We address the challenges faced by centralized DL i.e., privacy concerns, high communication costs for data collection, and the need for high-quality data for data-hungry DL models. We addressed these challenges by building high-performing and robust frameworks in federated settings which can enhance privacy, reduce communication costs and provide SOTA performance. Additionally, we provided XAI modules to help explain the outcome of complex DL models which are needed in some applications, such as healthcare applications. Furthermore, we made our proposed framework robust against non-IID data distribution and data poisoning attacks in a federated setting. In particular, we make the proposed framework robust against varying noisy, unbalanced and skewed data, data poisoning and model poisoning attacks. In Chapter 3, we developed and validated a CNN-based two-phase approach for robust end-to-end classification in a federated setting with model-agnostic explanations based on GradCAM. In Chapter 4, in order to support applications that have long-term dependencies, such as HAR we proposed a lightweight transformer-based classifier, which attained SOTA performance while being computationally lightweight to support federated learning with resource-constrained clients. In Chapter 5, for adaptive anomaly detection in federated settings, we designed and validated a novel two-phase approach. First, we proposed transformer base AE/VAE and trained them using the normal class signal, from which we get the error vectors and we use the obtained error vectors along with Kernel density estimation and one class support vector data description for anomaly detection. Additionally, we designed and validated an explainable AI module to detect the region of the input with the highest impact on the output, which helps decision-making by tracking back the regions with the highest anomaly with adaptive window size. Finally, in Chapter 6, we designed and validated a framework for poisoning attack detection in federated settings which can detect poisoning attacks without the need for access to local data with low time complexity.

To summarize, the thesis introduces innovative frameworks for healthcare applications using Federated Learning (FL). Key contributions include enhancing privacy/security, overcoming centralized DL challenges, and providing explainability. The work ensures robustness in federated settings against data distribution variations and attacks. Chapter-wise, we develop a CNN-based classification approach with model-agnostic explanations, a lightweight transformer-based classifier for HAR application, a novel adaptive anomaly detection approach with an explainable AI module, and a novel low time-complexity poisoning attack detection framework with SOTA performance. Overall, the research advances privacy-sensitive applications of embedded AI by tackling privacy, communication, and robustness issues while enhancing security, interpretability, and performance.

7.2 Limitations and Future work

In this section, we discuss some limitations and perspectives of the proposed frameworks. Our proposed frameworks provide SOTA performance, robustness, explanations, and enhanced security and privacy. However, there are certain limitations, which can be explored in the future to make the proposed frameworks more reliable and trustworthy. Limitations and perspectives are given as follows:

1. The proposed frameworks consider the data devices in distributed edges to be homogeneous (same device), but in some cases, the devices may be heterogeneous, and device-specific characteristics (measuring units, sampling rate, etc) may limit the generalizability of the local models from device to device and may reduce the accuracy of the aggregated model.

2. In addition, a major limitation of our work is that datasets such as our newly constructed dataset were based on only five human participants and a more artificially constructed home care scenario. It is therefore important to evaluate the proposed work in more realistic home healthcare settings, and ultimately move the proposed frameworks into real-world usage. For such future work, the involvement of patients, carers, and health professionals is vital in all stages of the research process, following well-established standard procedures and guidelines such as the UK Standards for Public Involvement [209].

3. In regards to HAR, one important aspect of the real-world-facing research is to carefully evaluate the acceptability and usability of body sensors used to ensure that they are the right ones for the target patients. This suggests that different sets of body sensors may have to be used for patients with different conditions or preferences, so we need to

investigate how the proposed frameworks will change w.r.t. the different sets of sensors. These include scenarios where nobody sensors can be put on the body of a patient, so computer vision-based approaches relying on monitoring cameras and microphones will need investigation, which will involve very different ML/DL models from those we used for the proposed work in this thesis. Validating the performance of our proposed frameworks with a larger dataset covering more patients and people with normal conditions will be useful to consolidate the evidence presented in this thesis.

4. Furthermore, our experiments were based on a small number of edge devices (3-10) and simulated local data, so it will be important to revalidate the overall performance of the proposed frameworks in a more real-world setting. Doing both will require close collaboration with healthcare organizations, which will not be trivial to achieve and will be our long-term future work.

5. From a security perspective, although our proposed framework in Chapter 6 has a number of merits, it also has the following limitations:

   - *The need to maintain a stable and validated public dataset*: The dependence on a well-validated public dataset means that our method will not work if drastically different new data samples and even new classes keep emerging. Examples include automatic detection of rare diseases or rapidly changing viruses (e.g., COVID or flu viruses), and intrusion detection systems where attackers keep changing their attack behaviors. In other words, our proposed method is suitable only for applications where 1) class labels are stable and representative samples of each class do not change too rapidly, or 2) a public dataset can be easily maintained by an active community despite the fast evolution of data samples. In the future, we will study how to relax the dependency on the public dataset, which likely requires new ideas to update the **RM**, e.g., based on some distributed consensus protocols.

- *Dependence on a trusted third party or component*: Conceptually speaking, our proposed model relies on a trusted third party or a trusted component of the global server – the auditor model + the reference model + the public dataset. Such dependencies are widely used in cryptography. In some applications, such a trusted party or model may be difficult to find or establish, e.g., when the clients do not have trust on each other and cannot agree on anything they can trust collectively. If it is possible to remove the trusted party/component or constructed a similar entity under a "zero trust" environment is an interesting future research direction.

- *Sensitivity to "small" poison in updates*: The anomaly detection component of our proposed method still relies on a threshold to detect malicious clients, therefore it will not be able to detect "small" poisoning attacks. While in this case, a single attacker may find it difficult to poison the global model, carefully coordinated colluding attacks of many malicious clients may be able to collectively make a difference. Detecting such small colluding poisoning attacks requires different approaches, and will be one interesting future research direction.

- *Limited experimental results*: Although we conducted many experiments, the existence of many poisoning attacks and different configurations of the FL system means that our experimental results do not cover all aspects of our proposed method. For instance, we used only three edge devices and one attacker and therefore did not consider different combinations of multiple attackers who could launch different attacks with different parameters and some of them could collude. In the future, we plan to conduct a more comprehensive set of experiments to investigate more aspects of our proposed method's performance.

- *More advanced attacks*: Our experiments were conducted with known poisoning attacks that are unaware of our proposed method. Some advanced attacks may be developed to target our proposed method with a better evasion rate. For example, backdoor trigger attacks inject a trigger into the training data, and the malicious behavior of the attack is invoked only in the presence of the trigger. Hence, advanced attacks and how our proposed method can be further improved to mitigate such attacks will be another direction of future research.

6. Adoption of proposed techniques to other types of federated learning such as vertical FL will require signification modification in the design and revalidation of the experiments.

7. As we discussed in Chapter 2, FL can be subjected to difference inference attacks that aim at compromising privacy, DP is a useful privacy protection mechanism, and research has shown that there exists a tradeoff between privacy and efficacy [210]. If we increase the noise (i.e., decrease the privacy budget), the accuracy of the model will decrease. If we use a high privacy budget (i.e., add low noise), we can achieve high accuracy, but in this case, the inference attacks can be launched successfully, i.e., the privacy budget can affect the attack's performance. Moreover, ML models may be able to ignore the added noise in training data if they have been sufficiently well-trained because ML models can generalize well if trained on noisy data, thus ignoring the noise and adapting key features of data [211]. Despite these complicated aspects

with the application of DP to FL, to the best of our knowledge, no past research has systematically investigated how different FL aggregation methods and/or privacy budgets interact with DP mechanisms such as DP-SGD, therefore it remains unknown if DP-SGD can survive MIAs effectively when different FL aggregation methods are used. Hence, methods are needed to fill this research gap via a number of experiments, which can reveal that if using DP-SGD alone cannot always protect against MIA attacks and the relationship between the privacy budget and the performance of MIAs is *non-monotonic*. In such cases, additional methods should be developed to enhance privacy protection.

References

[1] R. Pastorino, C. De Vito, G. Migliara, K. Glocker, I. Binenbaum, W. Ricciardi, and S. Boccia, "Benefits and challenges of big data in healthcare: An overview of the european initiatives," *European Journal of Public Health*, vol. 29, no. Supplement_3, pp. 23–27, Nov. 2019.

[2] H. Sahoo, D. Govil, K. James, and R. D. Prasad, "Health issues, health care utilization and health care expenditure among elderly in India: Thematic review of literature," *Aging and Health Research*, vol. 1, no. 100012, 2 Jun. 2021.

[3] G. R. Dagenais, D. P. Leong, S. Rangarajan, F. Lanas, P. Lopez-Jaramillo, R. Gupta, R. Diaz, A. Avezum, G. B. Oliveira, A. Wielgosz, *et al.*, "Variations in common diseases, hospital admissions, and deaths in middle-aged adults in 21 countries from five continents (PURE): A prospective cohort study," *The Lancet*, vol. 395, no. 10226, pp. 785–794, 2020.

[4] P. de Chazal, B. Celler, and R. Reilly, "Using wavelet coefficients for the classification of the electrocardiogram," in *Proceedings of the 22nd Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, vol. 1, 2000, pp. 64–67.

[5] Y. Özbay, R. Ceylan, and B. Karlik, "Integration of type-2 fuzzy clustering and wavelet transform in a neural network based ecg classifier," *Expert Systems with Applications*, vol. 38, no. 1, pp. 1004–1010, 2011.

[6] A. Handa, A. Sharma, and S. K. Shukla, “Machine learning in cybersecurity: A review,” *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 9, no. 4, e1306, 2019.

[7] H. Storm, K. Baylis, and T. Heckelei, “Machine learning in agricultural and applied economics,” *European Review of Agricultural Economics*, vol. 47, no. 3, pp. 849–892, 2020.

[8] K. Shailaja, B. Seetharamulu, and M. A. Jabbar, “Machine learning in healthcare: A review,” in *Proceedings of the 2018 2nd international conference on electronics, communication and aerospace technology*, IEEE, 2018, pp. 910–914.

[9] M. Marjani, F. Nasaruddin, A. Gani, A. Karim, I. A. T. Hashem, A. Siddiqa, and I. Yaqoob, “Big IoT data analytics: Architecture, opportunities, and open research challenges,” *IEEE Access*, vol. 5, pp. 5247–5261, 2017.

[10] D. Mourtzis, E. Vlachou, and N. Milas, “Industrial big data as a result of IoT adoption in manufacturing,” *Procedia CIRP*, vol. 55, pp. 290–295, 2016.

[11] M. Mohammadi, A. Al-Fuqaha, S. Sorour, and M. Guizani, “Deep learning for IoT big data and streaming analytics: A survey,” *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 2923–2960, 2018.

[12] J. W. Shavlik, T. Dietterich, and T. G. Dietterich, *Readings in Machine Learning*. Morgan Kaufmann, 1990.

[13] A. Esteva, A. Robicquet, B. Ramsundar, V. Kuleshov, M. DePristo, K. Chou, C. Cui, G. Corrado, S. Thrun, and J. Dean, “A guide to deep learning in healthcare,” *Nature Medicine*, vol. 25, no. 1, pp. 24–29, 2019.

[14] A. M. Ozbayoglu, M. U. Gudelek, and O. B. Sezer, "Deep learning for financial applications: A survey," *Applied Soft Computing*, 106384:1–106384:29, 2020.

[15] J. Wang, Y. Ma, L. Zhang, R. X. Gao, and D. Wu, "Deep learning for smart manufacturing: Methods and applications," *Journal of Manufacturing Systems*, vol. 48, pp. 144–156, 2018.

[16] A. Kamilaris and F. X. Prenafeta-Boldú, "Deep learning in agriculture: A survey," *Computers and Electronics in Agriculture*, vol. 147, pp. 70–90, 2018.

[17] M. D. Z. Hossain, F. Sohel, M. F. Shiratuddin, and H. Laga, "A comprehensive survey of deep learning for image captioning," *ACM Computing Surveys*, vol. 51, no. 6, 118:1–118:36, 2019.

[18] T. Georgiou, Y. Liu, W. Chen, and M. Lew, "A survey of traditional and deep learning-based feature descriptors for high dimensional data in computer vision," *International Journal of Multimedia Information Retrieval*, vol. 9, no. 3, pp. 135–170, 2020.

[19] R. Miotto, F. Wang, S. Wang, X. Jiang, and J. T. Dudley, "Deep learning for healthcare: Review, opportunities and challenges," *Briefings in Bioinformatics*, vol. 19, no. 6, pp. 1236–1246, 2018.

[20] V. Kumar and M. Garg, "Deep learning in predictive analytics: A survey," in *Proceedings of the 2017 International Conference on Emerging Trends in Computing and Communication Technologies*, IEEE, 2017.

[21] R. S. Antunes, C. André da Costa, A. Küderle, I. A. Yari, and B. Eskofier, "Federated Learning for Healthcare: Systematic Review and Architecture Proposal," *ACM Transactions on Intelligent Systems and Technology*, vol. 13, no. 4, 2022.

[22] Z. Ji, Z. C. Lipton, and C. Elkan, *Differential privacy and machine learning: A survey and review*, arXiv:1412.7584 [cs.LG], 2014.

[23] A. Raghuvanshi, U. K. Singh, and C. Joshi, "A Review of Various Security and Privacy Innovations for IoT Applications in Healthcare," *Advanced Healthcare Systems: Empowering Physicians with IoT-Enabled Technologies*, pp. 43–58, 2022.

[24] L. Van Zoonen, "Privacy concerns in smart cities," *Government Information Quarterly*, vol. 33, no. 3, pp. 472–480, 2016.

[25] D. Gunning and D. Aha, "DARPA's explainable artificial intelligence (XAI) program," *AI Magazine*, vol. 40, no. 2, pp. 44–58, 2019.

[26] H. Hu, Z. Salcic, L. Sun, G. Dobbie, P. S. Yu, and X. Zhang, "Membership inference attacks on machine learning: A survey," *ACM Computing Surveys*, vol. 54, no. 11s, 2022.

[27] V. Tolpegin, S. Truex, M. E. Gursoy, and L. Liu, "Data poisoning attacks against federated learning systems," in *Computer Security – ESORICS 2020: 25th European Symposium on Research in Computer Security, ESORICS 2020, Guildford, UK, September 14–18, 2020, Proceedings, Part I*, Springer, 2020, pp. 480–501.

[28] H. Hu, Z. Salcic, L. Sun, G. Dobbie, P. S. Yu, and X. Zhang, "Membership inference attacks on machine learning: A survey," *ACM Computing Surveys*, vol. 54, no. 11s, p. 235, 2022.

[29] J. Verbraeken, M. Wolting, J. Katzy, J. Kloppenburg, T. Verbelen, and J. S. Rellermeyer, "A survey on distributed machine learning," *ACM Computing Survey*, vol. 53, no. 2, 2020.

[30] J. Konečn, H. B. McMahan, D. Ramage, and P. Richtárik, *Federated optimization: Distributed machine learning for on-device intelligence*, arXiv:1610.02527 [cs.LG], 2016.

[31] K. Bonawitz, H. Eichner, W. Grieskamp, D. Huba, A. Ingerman, V. Ivanov, C. Kiddon, J. Konečn, S. Mazzocchi, B. McMahan, *et al.*, "Towards federated learning at scale: System design," *Proceedings of machine learning and systems*, vol. 1, pp. 374–388, 2019.

[32] C. Zhang, Y. Xie, H. Bai, B. Yu, W. Li, and Y. Gao, "A survey on federated learning," *Knowledge-Based Systems*, vol. 216, p. 106 775, 2021.

[33] Q. Li, Y. Diao, Q. Chen, and B. He, "Federated learning on Non-IID Data Silos: An Experimental Study," in *Proceeding of the IEEE 38th International Conference on Data Engineering*, 2022, pp. 965–978.

[34] X. Fang and M. Ye, "Robust Federated Learning With Noisy and Heterogeneous Clients," in *Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 072–10 081.

[35] M. Fang, X. Cao, J. Jia, and N. Gong, "Local model poisoning attacks to Byzantine-robust federated learning," in *Proceedings of the 29th USENIX Security Symposium*, USENIX Association, 2020, pp. 1605–1622.

[36] P. Blanchard, E. M. El Mhamdi, R. Guerraoui, and J. Stainer, "Machine learning with adversaries: Byzantine tolerant gradient descent," in *Advances in Neural Information Processing Systems*, vol. 30, Curran Associates, Inc., 2017, pp. 119–129.

[37] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[38] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, H. B. McMahan, S. Patel, D. Ramage, A. Segal, and K. Seth, "Practical secure aggregation for privacy-preserving machine learning," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 2017, pp. 1175–1191.

[39] T. M. Mitchell, "Does Machine Learning Really Work?" *AI magazine*, vol. 18, no. 3, pp. 11–11, 1997.

[40] A. E. Ezugwu, A. M. Ikotun, O. O. Oyelade, L. Abualigah, J. O. Agushaka, C. I. Eke, and A. A. Akinyelu, "A comprehensive survey of clustering algorithms: State-of-the-art machine learning applications, taxonomy, challenges, and future research prospects," *Engineering Applications of Artificial Intelligence*, vol. 110, p. 104 743, 2022.

[41] Y. H. Hu, S. Palreddy, and W. J. Tompkins, "A patient-adaptable ECG beat classifier using a mixture of experts approach," *IEEE Transactions on Biomedical Engineering*, vol. 44, no. 9, pp. 891–900, 1997.

[42] D. M. J. Tax and R. P. W. Duin, "Support vector data description," *Machine Learning*, vol. 54, no. 1, pp. 45–66, 2004.

[43] C.-C. Chang and C.-J. Lin, "Training v-support vector classifiers: Theory and algorithms," *Neural Computation*, vol. 13, no. 9, pp. 2119–2147, 2001.

[44] G. Huang, H. Chen, Z. Zhou, F. Yin, and K. Guo, "Two-class support vector data description," *Pattern Recognition*, vol. 44, no. 2, pp. 320–329, 2011.

[45] V. N. Vapnik, *The Nature of Statistical Learning Theory*. Springer, 1995.

[46] K. Lee, D.-W. Kim, D. Lee, and K. H. Lee, "Improving support vector data description using local density degree," *Pattern Recognition*, vol. 38, no. 10, pp. 1768–1771, 2005.

[47] P. Baldi, "Autoencoders, unsupervised learning, and deep architectures," in *Proceedings of the 2012 ICML Workshop on Unsupervised and Transfer Learning*, ML Research Press, 2012, pp. 37–49.

[48] S. Smys, J. I. Z. Chen, and S. Shakya, "Survey on neural network architectures with deep learning," *Journal of Soft Computing Paradigm*, vol. 2, no. 3, pp. 186–194, 2020.

[49] O. Yildirim, R. San Tan, and U. R. Acharya, "An efficient compression of ECG signals using deep convolutional autoencoders," *Cognitive Systems Research*, vol. 52, pp. 198–211, 2018.

[50] D. Cozzolino and L. Verdoliva, "Single-image splicing localization through autoencoder-based anomaly detection," in *Proceedings of the 2016 IEEE International Workshop on Information Forensics and Security*, IEEE, 2016.

[51] M. S. Seyfioğlu, A. M. Özbayoğlu, and S. Z. Gürbüz, "Deep convolutional autoencoder for radar-based classification of similar aided and unaided human activities," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 54, no. 4, pp. 1709–1723, 2018.

[52] C. Baur, B. Wiestler, S. Albarqouni, and N. Navab, "Deep autoencoding models for unsupervised anomaly segmentation in brain MR images," in *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries – 4th International Workshop, BrainLes 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 16, 2018, Revised Selected Papers, Part I*, ser. Lecture Notes in Computer Science, vol. 11383, Springer, 2018, pp. 161–169.

[53] D. Y. Oh and I. D. Yun, "Residual error based anomaly detection using auto-encoder in SMD machine sound," *Sensors*, vol. 18, no. 5, p. 1308, 2018.

[54] A. Oussidi and A. Elhassouny, "Deep generative models: Survey," in *Proceedings of the 2018 International Conference on Intelligent Systems and Computer Vision*, IEEE, 2018.

[55] J. Zhai, S. Zhang, J. Chen, and Q. He, "Autoencoder and its various variants," in *Proceedings of the 2018 IEEE International Conference on Systems, Man, and Cybernetics*, IEEE, 2018, pp. 415–419.

[56] H. Takahashi, T. Iwata, Y. Yamanaka, M. Yamada, and S. Yagi, "Variational autoencoder with implicit optimal priors," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 5066–5073.

[57] P. Li, Y. Pei, and J. Li, "A comprehensive survey on design and application of autoencoder in deep learning," *Applied Soft Computing*, vol. 138, p. 110 176, 2023, ISSN: 1568-4946.

[58] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, NeurIPS, 2017, pp. 5998–6008.

[59] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, *BERT: Pre-training of deep bidirectional transformers for language understanding*, arXiv:1810.04805 [cs.CL], 2018.

[60] B. Li, W. Cui, W. Wang, L. Zhang, Z. Chen, and M. Wu, "Two-stream convolution augmented transformer for human activity recognition," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 1, pp. 286–293, 2021.

[61] Y. Bansal, P. Nakkiran, and B. Barak, "Revisiting model stitching to compare neural representations," in *Advances in Neural Information Processing Systems*, vol. 34, Curran Associates, Inc., 2021.

[62] D. Arpit, S. Jastrzębski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio, and S. Lacoste-Julien, "A closer look at memorization in deep networks," in *Proceedings of the 34th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 70, PMLR, 2017, pp. 233–242.

[63] R. Yilmazer and D. Birant, "Shelf auditing based on image classification using semi-supervised deep learning to increase on-shelf availability in grocery stores," *Sensors*, vol. 21, no. 2, p. 327, 2021.

[64] F. K. Došilović, M. Brčić, and N. Hlupić, "Explainable artificial intelligence: A survey," in *Proceedings of the 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics*, IEEE, 2018, pp. 210–215.

[65] W. Samek, T. Wiegand, and K.-R. Müller, *Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models*, arXiv:1708.08296 [cs.AI], 2017.

[66] J. Choo and S. Liu, "Visual analytics for explainable deep learning," *IEEE Computer Graphics and Applications*, vol. 38, no. 4, pp. 84–92, 2018.

[67] S. Mousavi, F. Afghah, and U. R. Acharya, "HAN-ECG: An interpretable atrial fibrillation detection model using hierarchical attention networks," *Computers in Biology and Medicine*, vol. 127, p. 104 057, 2020.

[68] R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra, *Grad-CAM: Why did you say that?* arXiv:1611.07450 [stat.ML], 2016.

[69] K. E. Mokhtari, B. P. Higdon, and A. Başar, "Interpreting financial time series with SHAP values," in *Proceedings of the 29th Annual International Conference on Computer Science and Software Engineering*, IBM Corp., 2019, pp. 166–172.

[70] R. I. Hamilton and P. N. Papadopoulos, "Using shap values and machine learning to understand trends in the transient stability limit," *IEEE Transactions on Power Systems*, pp. 1–12, 2023.

[71] N. Razmjooy, F. R. Sheykhahmad, and N. Ghadimi, "A hybrid neural network–world cup optimization algorithm for melanoma detection," *Open Medicine*, vol. 13, no. 1, pp. 9–16, 2018.

[72] A. Parsian, M. Ramezani, and N. Ghadimi, "A hybrid neural network-gray wolf optimization algorithm for melanoma detection," *Biomedical Research*, vol. 28, no. 8, pp. 3408–3411, 2017.

[73] Z. Xu, F. R. Sheykhahmad, N. Ghadimi, and N. Razmjooy, "Computer-aided diagnosis of skin cancer based on soft computing techniques," *Open Medicine*, vol. 15, no. 1, pp. 860–871, 2020.

[74] K. Kourou, T. P. Exarchos, K. P. Exarchos, M. V. Karamouzis, and D. I. Fotiadis, "Machine learning applications in cancer prognosis and prediction," *Computational and Structural Biotechnology Journal*, vol. 13, pp. 8–17, 2015.

[75] H. H. Atkinson, C. Rosano, E. M. Simonsick, J. D. Williamson, C. Davis, W. T. Ambrosius, S. R. Rapp, M. Cesari, A. B. Newman, T. B. Harris, S. M. Rubin, K. Yaffe, S. Satterfield, and S. B. Kritchevsky, "Cognitive function, gait speed decline, and comorbidities: The health, aging and body composition study," *The Journals of Gerontology Series A: Biological Sciences and Medical Sciences*, vol. 62, no. 8, pp. 844–850, 2007.

[76] Y. Chen, J. Wang, M. Huang, and H. Yu, "Cross-position activity recognition with stratified transfer learning," *Pervasive and Mobile Computing*, vol. 57, pp. 1–13, 2019.

[77] S. C. Mukhopadhyay, "Wearable sensors for human activity monitoring: A review," *IEEE Sensors Journal*, vol. 15, no. 3, pp. 1321–1330, 2014.

[78] O. D. Lara and M. A. Labrador, "A survey on human activity recognition using wearable sensors," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 3, pp. 1192–1209, 2012.

[79] R. Bhardwaj, A. R. Nambiar, and D. Dutta, "A survey on human activity recognition using wearable sensors," in *Proceedings of the 2017 IEEE 41st Annual Computer Software and Applications Conference*, vol. 2, IEEE, 2017, pp. 236–241.

[80] G. Manogaran and D. Lopez, "A survey of big data architectures and machine learning algorithms in healthcare," *International Journal of Biomedical Engineering and Technology*, vol. 25, no. 2/3/4, pp. 182–211, 2017.

[81] R. Fakoor, F. Ladhak, A. Nazi, and M. Huber, "Using deep learning to enhance cancer diagnosis and classification," *Proceedings of the WHEALTH Workshop at the 30th International Conference on Machine Learning*, vol. 28, 2013.

[82] M. Simoons and P. Hugenholtz, "Gradual changes of ECG waveform during and after exercise in normal subjects.," *Circulation*, vol. 52, no. 4, pp. 570–577, 1975.

[83] S. H. Jambukia, V. K. Dabhi, and H. B. Prajapati, "Classification of ECG signals using machine learning techniques: A survey," in *Proceedings of the 2015 International Conference on Advances in Computer Engineering and Applications*, IEEE, 2015, pp. 714–721.

[84] C. K. Roopa and B. S. Harish, "A survey on various machine learning approaches for ECG analysis," *International Journal of Computer Applications*, vol. 163, no. 9, pp. 25–33, 2017.

[85] S. Sahoo, M. Dash, S. Behera, and S. Sabut, "Machine learning approach to detect cardiac arrhythmias in ECG signals: A survey," *IRBM*, vol. 41, no. 4, pp. 185–194, 2020.

[86] D. K. Atal and M. Singh, "Arrhythmia classification with ECG signals based on the optimization-enabled deep convolutional neural

network," *Computer Methods and Programs in Biomedicine*, vol. 196, p. 105 607, 2020.

[87] S. Liaqat, K. Dashtipour, A. Zahid, K. Assaleh, K. Arshad, and N. Ramzan, "Detection of atrial fibrillation using a machine learning approach," *Information*, vol. 11, no. 12, p. 549, 2020.

[88] J. Rubin, R. Abreu, A. Ganguli, S. Nelaturi, I. Matei, and K. Sricharan, *Recognizing abnormal heart sounds using deep learning*, arXiv:1707.04642 [cs.SD], 2017.

[89] M. Gjoreski, A. Gradišek, B. Budna, M. Gams, and G. Poglajen, "Machine learning and end-to-end deep learning for the detection of chronic heart failure from heart sounds," *IEEE Access*, vol. 8, pp. 20 313–20 324, 2020.

[90] J.-S. Huang, B.-Q. Chen, N.-Y. Zeng, X.-C. Cao, and Y. Li, "Accurate classification of ECG arrhythmia using MOWPT enhanced fast compression deep learning networks," *Journal of Ambient Intelligence and Humanized Computing*, pp. 5703–5720, 2020.

[91] A. Mannini and A. M. Sabatini, "Machine learning methods for classifying human physical activity from on-body accelerometers," *Sensors*, vol. 10, no. 2, pp. 1154–1175, 2010.

[92] S. Yao, S. Hu, Y. Zhao, A. Zhang, and T. Abdelzaher, "DeepSense: A unified deep learning framework for time-series mobile sensing data processing," in *Proceedings of the 26th International Conference on World Wide Web*, International World Wide Web Conference Committee (IW3), 2017, pp. 351–360.

[93] N. Sikder and A.-A. Nahid, "KU-HAR: An open dataset for heterogeneous human activity recognition," *Pattern Recognition Letters*, vol. 146, pp. 46–54, 2021.

[94] K. Sozinov, V. Vlassov, and S. Girdzijauskas, “Human activity recognition using federated learning,” in *Proceedings of the 8th IEEE International Conference on Big Data and Cloud Computing*, IEEE, 2018, pp. 1103–1111.

[95] M. M. Hassan, M. Z. Uddin, A. Mohamed, and A. Almogren, “A robust human activity recognition system using smartphone sensors and deep learning,” *Future Generation Computer Systems*, vol. 81, pp. 307–313, 2018.

[96] A. Ignatov, “Real-time human activity recognition from accelerometer data using Convolutional Neural Networks,” *Applied Soft Computing*, vol. 62, pp. 915–922, 2018.

[97] A. Murad and J.-Y. Pyun, “Deep recurrent neural networks for human activity recognition,” *Sensors*, vol. 17, no. 11, p. 2556, 2017.

[98] X. Zhou, W. Liang, K. I.-K. Wang, H. Wang, L. T. Yang, and Q. Jin, “Deep-learning-enhanced human activity recognition for internet of healthcare things,” *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 6429–6438, 2020.

[99] H. Yu, Z. Chen, X. Zhang, X. Chen, F. Zhuang, H. Xiong, and X. Cheng, “Fedhar: Semi-supervised online learning for personalized federated human activity recognition,” *IEEE Transactions on Mobile Computing*, vol. 22, no. 6, pp. 3318–3332, 2021.

[100] J. Vanerio and P. Casas, “Ensemble-learning approaches for network security and anomaly detection,” in *Proceedings of the 2017 Workshop on Big Data Analytics and Machine Learning for Data Communication Networks*, ACM, 2017.

[101] D. Kwon, H. Kim, J. Kim, S. C. Suh, I. Kim, and K. J. Kim, “A survey of deep learning-based network anomaly detection,” *Cluster Computing*, vol. 22, no. 1, pp. 949–961, 2019.

[102] M. Krichen, "Anomalies detection through smartphone sensors: A review," *IEEE Sensors Journal*, vol. 21, no. 6, pp. 7207–7217, 2021.

[103] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Computing Surveys*, vol. 41, no. 3, 15:1–15:58, 2009.

[104] Y. Yu, "A survey of anomaly intrusion detection techniques," *Journal of Computing Sciences in Colleges*, vol. 28, no. 1, pp. 9–17, 2012.

[105] S. Durga, R. Nag, and E. Daniel, "Survey on machine learning and deep learning algorithms used in Internet of Things (IoT) healthcare," in *Proceedings of the 2019 3rd International Conference on Computing Methodologies and Communication*, IEEE, 2019, pp. 1018–1022.

[106] K. Wang, Y. Zhao, Q. Xiong, M. Fan, G. Sun, L. Ma, and T. Liu, "Research on healthy anomaly detection model based on deep learning from multiple time-series physiological signals," *Scientific Programming*, vol. 2016, p. 5 642 856, 2016.

[107] M. B. Amin, O. Banos, W. A. Khan, H. S. Muhammad Bilal, J. Gong, D.-M. Bui, S. H. Cho, S. Hussain, T. Ali, U. Akhtar, T. C. Chung, and S. Lee, "On curating multimodal sensory data for health and wellness platforms," *Sensors*, vol. 16, no. 7, 980:1–980:27, 2016.

[108] H. Banaee, M. U. Ahmed, and A. Loutfi, "Data mining for wearable sensors in health monitoring systems: A review of recent trends and challenges," *Sensors*, vol. 13, no. 12, pp. 17 472–17 500, 2013.

[109] N. Inkster, *China's Cyber Power*. IISS, 2018.

[110] B. Liu, M. Ding, S. Shaham, W. Rahayu, F. Farokhi, and Z. Lin, "When machine learning meets privacy: A survey and outlook," *ACM Computing Surveys*, vol. 54, no. 2, 31:1–31:36, 2021.

[111] N. Waheed, X. He, M. Ikram, M. Usman, S. S. Hashmi, and M. Usman, "Security and privacy in IoT using machine learning and blockchain: Threats and countermeasures," *ACM Computing Surveys*, vol. 53, no. 6, 122:1–122:37, 2020.

[112] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, ML Research Press, 2017, pp. 1273–1282.

[113] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Transactions on Intelligent Systems and Technology*, vol. 10, no. 2, 12:1–12:19, 2019.

[114] J. Xu, B. S. Glicksberg, C. Su, P. Walker, J. Bian, and F. Wang, "Federated learning for healthcare informatics," *Journal of Healthcare Informatics Research*, vol. 5, no. 1, pp. 1–19, 2021.

[115] P. Huang, G. Wang, and S. Qin, "Boosting for transfer learning from multiple data sources," *Pattern Recognition Letters*, vol. 33, no. 5, pp. 568–579, 2012.

[116] X. Qin, Y. Chen, J. Wang, and C. Yu, "Cross-dataset activity recognition via adaptive spatial-temporal transfer learning," *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, vol. 3, no. 4, 148:1–148:25, 2019.

[117] Y. Chen, X. Qin, J. Wang, C. Yu, and W. Gao, "Fedhealth: A federated transfer learning framework for wearable healthcare," *IEEE Intelligent Systems*, vol. 35, no. 4, pp. 83–93, 2020.

[118] S. L. Oh, E. Y. K. Ng, R. San Tan, and U. R. Acharya, "Automated diagnosis of arrhythmia using combination of CNN and LSTM techniques with variable length heart beats," *Computers in Biology and Medicine*, vol. 102, pp. 278–287, 2018.

[119] U. Erdenebayar, H. Kim, J.-U. Park, D. Kang, and K.-J. Lee, "Automatic prediction of atrial fibrillation based on convolutional neural network using a short-term normal electrocardiogram signal," *Journal of Korean medical science*, vol. 34, no. 7, e64:1–e64:10, 2019.

[120] U. R. Acharya, H. Fujita, O. S. Lih, Y. Hagiwara, J. H. Tan, and M. Adam, "Automated detection of arrhythmias using different intervals of tachycardia ECG segments with convolutional neural network," *Information Sciences*, vol. 405, pp. 81–90, 2017.

[121] Z. Yao, Z. Zhu, and Y. Chen, "Atrial fibrillation detection by multiscale convolutional neural networks," in *Proceedings of the 2017 20th International Conference on Information Fusion*, IEEE, 2017.

[122] S. Nurmaini, A. E. Tondas, A. Darmawahyuni, M. N. Rachmatullah, R. U. Partan, F. Firdaus, B. Tutuko, F. Pratiwi, A. H. Juliano, and R. Khoirani, "Robust detection of atrial fibrillation from short-term electrocardiogram using convolutional neural networks," *Future Generation Computer Systems*, vol. 113, pp. 304–317, 2020.

[123] D. C. Nguyen, Q.-V. Pham, P. N. Pathirana, M. Ding, A. Seneviratne, Z. Lin, O. Dobre, and W.-J. Hwang, "Federated learning for smart healthcare: A survey," *ACM Comput. Surv.*, vol. 55, no. 3, Feb. 2022.

[124] K. Tam, L. Li, B. Han, C. Xu, and H. Fu, *Federated noisy client learning*, arXiv preprint arXiv:2106.13239, 2021.

[125] B. Biggio, B. Nelson, and P. Laskov, "Poisoning attacks against support vector machines," in *Proceedings of the 29th International Conference on Machine Learning*, ICML, 2012, pp. 1467–1474.

[126] M. Jagielski, A. Oprea, B. Biggio, C. Liu, C. Nita-Rotaru, and B. Li, "Manipulating machine learning: Poisoning attacks and countermeasures for regression learning," in *Proceedings of the 2018 IEEE Symposium on Security and Privacy*, IEEE, 2018, pp. 19–35.

[127] X. Zhou, M. Xu, Y. Wu, and N. Zheng, “Deep model poisoning attack on federated learning,” *Future Internet*, vol. 13, no. 3, 73:1–14:14, 2021.

[128] P. Liu, X. Xu, and W. Wang, “Threats, attacks and defenses to federated learning: Issues, taxonomy and perspectives,” *Cybersecurity*, vol. 5, no. 1, 4:1–4:19, 2022.

[129] J. Chen, L. Zhang, H. Zheng, X. Wang, and Z. Ming, “DeepPoison: Feature transfer based stealthy poisoning attack for DNNs,” *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 68, no. 7, pp. 2618–2622, 2021.

[130] A. Shafahi, W. R. Huang, M. Najibi, O. Suciu, C. Studer, T. Dumitras, and T. Goldstein, “Poison frogs! targeted clean-label poisoning attacks on neural networks,” in *Advances in neural information processing systems*, vol. 31, Curran Associates, Inc., 2018, pp. 6103–6113.

[131] G. Xia, J. Chen, C. Yu, and J. Ma, “Poisoning attacks in federated learning: A survey,” *IEEE Access*, vol. 11, pp. 10 708–10 722, 2023.

[132] S. Hu, J. Lu, W. Wan, and L. Y. Zhang, *Challenges and approaches for mitigating byzantine attacks in federated learning*, arXiv:2112.14468 [cs.CR], 2022.

[133] P. M. Mammen, *Federated learning: Opportunities and challenges*, arXiv:2101.05428 [cs.LG], 2021.

[134] Q. Xia, Z. Tao, Z. Hao, and Q. Li, “FABA: An Algorithm for Fast Aggregation against Byzantine Attacks in Distributed Neural Networks,” *IJCAI*, 2019.

[135] B. Zhao, P. Sun, T. Wang, and K. Jiang, “FedInv: Byzantine-robust federated learning by inversing local model updates,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 8, pp. 9171–9179, 2022.

[136] A. Gupta, T. Luo, M. V. Ngo, and S. K. Das, "Long-short history of gradients is all you need: Detecting malicious and unreliable clients in federated learning," 2022.

[137] S. Li, Y. Cheng, Y. Liu, W. Wang, and T. Chen, *Abnormal client behavior detection in federated learning*, arXiv preprint arXiv:1910.09933, 2019.

[138] C. Xie, S. Koyejo, and I. Gupta, "Zeno: Distributed stochastic gradient descent with suspicion-based fault-tolerance," in *Proceedings of the 36th International Conference on Machine Learning*, PMLR, 2019, pp. 6893–6901.

[139] D. Yin, Y. Chen, R. Kannan, and P. Bartlett, "Byzantine-robust distributed learning: Towards optimal statistical rates," in *Proceedings of the 35th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 80, PMLR, 2018, pp. 5650–5659.

[140] E. M. El Mhamdi, R. Guerraoui, and S. Rouault, "The hidden vulnerability of distributed learning in Byzantium," in *Proceedings of the 35th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 80, PMLR, 2018, pp. 3521–3530.

[141] L. Li, W. Xu, T. Chen, G. B. Giannakis, and Q. Ling, "Rsa: Byzantine-robust stochastic aggregation methods for distributed learning from heterogeneous datasets," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 1544–1551.

[142] W. Wei, L. Liu, M. Loper, K.-H. Chow, M. E. Gursoy, S. Truex, and Y. Wu, *A framework for evaluating gradient leakage attacks in federated learning*, arXiv:2004.10397 [cs.LG], 2020.

[143] N. Rodrıguez-Barroso, D. Jiménez-López, M. V. Luzón, F. Herrera, and E. Martınez-Cámara, "Survey on federated learning threats: Concepts, taxonomy on attacks and defences, experimental study and challenges," *Information Fusion*, vol. 90, pp. 148–173, 2023.

[144] M. Nasr, R. Shokri, and A. Houmansadr, "Comprehensive privacy analysis of deep learning: Passive and active white-box inference attacks against centralized and federated learning," in *Proceedings of the 2019 IEEE Symposium on Security and Privacy*, IEEE, 2019, pp. 739–753.

[145] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *Proceedings of the 2017 IEEE Symposium on Security and Privacy*, IEEE, 2017, pp. 3–18.

[146] L. Song and P. Mittal, "Systematic evaluation of privacy risks of machine learning models," in *Proceedings of the 30th USENIX Security Symposium*, USENIX Association, 2021, pp. 2615–2632.

[147] ——, *Systematic evaluation of privacy risks of machine learning models*, arXiv:2003.10595 [cs.CR], 2020.

[148] A. Salem, Y. Zhang, M. Humbert, P. Berrang, M. Fritz, and M. Backes, "ML-Leaks: Model and data independent membership inference attacks and defenses on machine learning models," in *Proceedings of the 26th Annual Network and Distributed System Security Symposium*, Internet Society, 2019.

[149] M. Abadi, A. Chu, I. Goodfellow, H. B. McMahan, I. Mironov, K. Talwar, and L. Zhang, "Deep learning with differential privacy," in *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, ACM, 2016, pp. 308–318.

[150] J. Wei, E. Bao, X. Xiao, and Y. Yang, "DPIS: An enhanced mechanism for differentially private SGD with importance sampling," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, ACM, 2022, pp. 2885–2899.

[151] E. Bao, Y. Zhu, X. Xiao, Y. Yang, B. C. Ooi, B. H. M. Tan, and K. M. M. Aung, "Skellam mixture mechanism: A novel approach to federated

learning with differential privacy," *Proceedings of the VLDB Endowment*, vol. 15, no. 1, pp. 2348–2360, 2022.

[152] S. Guo, X. Wang, S. Long, H. Liu, L. Hai, and T. H. Sam, "A federated learning scheme meets dynamic differential privacy," *CAAI Transactions on Intelligence Technology*, 2023.

[153] Z. Bu, S. Gopi, J. Kulkarni, Y. T. Lee, H. Shen, and U. Tantipongpipat, "Fast and memory efficient differentially private-SGD via JL projections," *Advances in Neural Information Processing Systems*, vol. 34, pp. 19 680–19 691, 2021.

[154] B. Pyakillya, N. Kazachenko, and N. Mikhailovsky, "Deep learning for ECG classification," vol. 913, 012004:1–012004:5, 2017.

[155] S. M. Mathews, C. Kambhamettu, and K. E. Barner, "A novel application of deep learning for single-lead ECG classification," *Computers in Biology and Medicine*, vol. 99, pp. 53–62, 2018.

[156] F. Murat, O. Yildirim, M. Talo, U. B. Baloglu, Y. Demir, and U. R. Acharya, "Application of deep learning techniques for heartbeats detection using ECG signals-analysis and review," *Computers in Biology and Medicine*, 103726:1–103726:14, 2020.

[157] G. B. Moody and R. G. Mark, "The impact of the MIT-BIH Arrhythmia Database," *IEEE Engineering in Medicine and Biology Magazine*, vol. 20, no. 3, pp. 45–50, 2001.

[158] R. Assaf, I. Giurgiu, F. Bagehorn, and A. Schumann, "MTEX-CNN: Multivariate Time series EXplanations for predictions with convolutional neural networks," in *Proceedings of the 2019 IEEE International Conference on Data Mining*, IEEE, 2019, pp. 952–957.

[159] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statisticss (AISTATS) 2017*, JMLR, 2017, pp. 1273–1282.

[160] V. Smith, C.-K. Chiang, M. Sanjabi, and A. Talwalkar, *Federated multi-task learning*, arXiv:1705.10467 [cs.LG], 2017.

[161] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, "Robust and communication-efficient federated learning from non-i.i.d. data," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 9, pp. 3400–3413, 2019.

[162] D. Arpit, S. Jastrzębski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio, *et al.*, "A closer look at memorization in deep networks," in *Proceedings of the 34th International Conference on Machine Learning*, JMLR, 2017, pp. 233–242.

[163] M. B. Conover, *Understanding Electrocardiography*. Elsevier Health Sciences, 2002.

[164] S. K. Berkaya, A. K. Uysal, E. S. Gunal, S. Ergin, S. Gunal, and M. B. Gulmezoglu, "A survey on ECG analysis," *Biomedical Signal Processing and Control*, vol. 43, pp. 216–235, 2018.

[165] C. Yuan, Y. Yan, L. Zhou, J. Bai, and L. Wang, "Automated atrial fibrillation detection based on deep learning network," in *Proceedings of the 2016 IEEE International Conference on Information and Automation*, IEEE, 2016, pp. 1159–1164.

[166] Y. Xia, N. Wulan, K. Wang, and H. Zhang, "Detecting atrial fibrillation by deep convolutional neural networks," *Computers in Biology and Medicine*, vol. 93, pp. 84–92, 2018.

[167] F. J. Ordóñez and D. Roggen, “Deep convolutional and LSTM recurrent neural networks for multimodal wearable activity recognition,” *Sensors*, vol. 16, no. 1, 115:1–115:25, 2016.

[168] N. Y. Hammerla, S. Halloran, and T. Plötz, *Deep, convolutional, and recurrent models for human activity recognition using wearables*, arXiv:1604.08880 [cs.LG], 2016.

[169] E. De Cristofaro, “A critical overview of privacy in machine learning,” *IEEE Security & Privacy*, vol. 19, no. 4, pp. 19–27, 2021.

[170] D. S. Char, N. H. Shah, and D. Magnus, “Implementing machine learning in health care – addressing ethical challenges,” *New England Journal of Medicine*, vol. 378, no. 11, pp. 981–983, 2018.

[171] W. Yin, K. Kann, M. Yu, and H. Schütze, *Comparative study of CNN and RNN for natural language processing*, arXiv:1702.01923 [cs.CL], 2017.

[172] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. Le Scao, S. Gugger, M. Drame, Q. Lhoest, and A. Rush, “Transformers: State-of-the-art natural language processing,” in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, ACL, 2020, pp. 38–45.

[173] X. Yang, J. Bian, W. R. Hogan, and Y. Wu, “Clinical concept extraction using transformers,” *Journal of the American Medical Informatics Association*, vol. 27, no. 12, pp. 1935–1942, 2020.

[174] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, “Activity recognition using cell phone accelerometers,” *ACM SIGKDD Explorations Newsletter*, vol. 12, no. 2, pp. 74–82, 2011.

[175] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection for discrete sequences: A survey," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 5, pp. 823–839, 2012.

[176] M. Ahmed, A. N. Mahmood, and J. Hu, "A survey of network anomaly detection techniques," *Journal of Network and Computer Applications*, vol. 60, pp. 19–31, 2016.

[177] H. Li and P. Boulanger, "A survey of heart anomaly detection using ambulatory electrocardiogram (ECG)," *Sensors*, vol. 20, no. 5, 1461:1–1461:33, 2020.

[178] T. Fernando, H. Gammulle, S. Denman, S. Sridharan, and C. Fookes, "Deep learning for medical anomaly detection – a survey," *ACM Computing Surveys*, vol. 54, no. 7, 141:1–141:37, 2021.

[179] P. Libby, R. O. Bonow, D. L. Mann, G. F. Tomaselli, D. Bhatt, S. D. Solomon, and E. Braunwald, *Braunwald's Heart Disease: A Textbook of Cardiovascular Medicine*, 12th. Elsevier, 2021.

[180] G. Kaleschke, B. Hoffmann, I. Drewitz, G. Steinbeck, M. Naebauer, A. Goette, G. Breithardt, and P. Kirchhof, "Prospective, multicentre validation of a simple, patient-operated electrocardiographic system for the detection of arrhythmias and electrocardiographic changes," *Europace*, vol. 11, no. 10, pp. 1362–1368, 2009.

[181] M. Haddad, S. Hermassi, Z. Aganovic, F. Dalansi, M. Kharbach, A. O. Mohamed, and K. W. Bibi, "Ecological validation and reliability of Hexoskin wearable body metrics tool in measuring pre-exercise and peak heart rate during shuttle run test in professional handball players," *Frontiers in Physiology*, 957:1–957:8, 2020.

[182] A. Adler, M. Elad, Y. Hel-Or, and E. Rivlin, "Sparse coding with anomaly detection," *Journal of Signal Processing Systems*, vol. 79, no. 2, pp. 179–188, 2015.

[183] H. Jin, Y. Luo, P. Li, and J. Mathew, “A review of secure and privacy-preserving medical data sharing,” *IEEE Access*, vol. 7, pp. 61 656–61 669, 2019.

[184] S. Al-Janabi, I. Al-Shourbaji, M. Shojafar, and S. Shamshirband, “Survey of main challenges (security and privacy) in wireless body area networks for healthcare applications,” *Egyptian Informatics Journal*, vol. 18, no. 2, pp. 113–122, 2017.

[185] O. Asan, A. E. Bayrak, and A. Choudhury, “Artificial intelligence and human trust in healthcare: Focus on clinicians,” *Journal of Medical Internet Research*, vol. 22, no. 6, e15154:1–e15154:7, 2020.

[186] F. Xu, H. Uszkoreit, Y. Du, W. Fan, D. Zhao, and J. Zhu, “Explainable AI: A brief survey on history, research areas, approaches and challenges,” in *Natural Language Processing and Chinese Computing: 8th CCF International Conference, NLPCC 2019, Dunhuang, China, October 9–14, 2019, Proceedings, Part II*, Springer, 2019, pp. 563–574.

[187] F. Maussang, J. Chanussot, A. Hétet, and M. Amate, “Mean–standard deviation representation of sonar images for echo detection: Application to SAS images,” *IEEE Journal of Oceanic Engineering*, vol. 32, no. 4, pp. 956–970, 2007.

[188] C. Leys, C. Ley, O. Klein, P. Bernard, and L. Licata, “Detecting outliers: Do not use standard deviation around the mean, use absolute deviation around the median,” *Journal of Experimental Social Psychology*, vol. 49, no. 4, pp. 764–766, 2013.

[189] A. L. Goldberger, L. A. N. Amaral, L. Glass, J. M. Hausdorff, P. C. Ivanov, R. G. Mark, J. E. Mietus, G. B. Moody, C.-K. Peng, and H. E. Stanley, “PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals,” *Circulation*, vol. 101, no. 23, e215–e220, 2000.

[190] Y. Chen, Y. Hao, T. Rakthanmanon, J. Zakaria, B. Hu, and E. Keogh, "A general framework for never-ending learning from time series streams," *Data Mining and Knowledge Discovery*, vol. 29, no. 6, pp. 1622–1664, 2015.

[191] G. San Martin, E. López Droguett, V. Meruane, and M. das Chagas Moura, "Deep variational auto-encoders: A promising tool for dimensionality reduction and ball bearing elements fault diagnosis," *Structural Health Monitoring*, vol. 18, no. 4, pp. 1092–1128, 2019.

[192] M. Wess, P. D. Sai Manoj, and A. Jantsch, "Neural network based ECG anomaly detection on FPGA and trade-off analysis," in *Proceedings of the 2017 IEEE International Symposium on Circuits and Systems*, IEEE, 2017.

[193] D.-H. Shin, R. C. Park, and K. Chung, "Decision boundary-based anomaly detection model using improved AnoGAN from ECG data," *IEEE Access*, vol. 8, pp. 108 664–108 674, 2020.

[194] D. Carrera, B. Rossi, P. Fragneto, and G. Boracchi, "Online anomaly detection for long-term ECG monitoring using wearable devices," *Pattern Recognition*, vol. 88, pp. 482–492, 2019.

[195] M. Lenning, J. Fortunato, T. Le, I. Clark, A. Sherpa, S. Yi, P. Hofsteen, G. Thamilarasu, J. Yang, X. Xu, H.-D. Han, T. K. Hsiai, and H. Cao, "Real-time monitoring and analysis of zebrafish electrocardiogram with anomaly detection," *Sensors*, vol. 18, no. 1, 61:1–61:16, 2018.

[196] S. Chauhan and L. Vig, "Anomaly detection in ECG time signals via deep long short-term memory networks," in *Proceedings of the 2015 IEEE International Conference on Data Science and Advanced Analytics*, IEEE, 2015.

[197] H. Zhou and C. Kan, “Tensor-based ECG anomaly detection toward cardiac monitoring in the Internet of Health Things,” *Sensors*, vol. 21, no. 12, 4173:1–4173:17, 2021.

[198] M. Zhang, Y. Wang, and T. Luo, “Federated learning for arrhythmia detection of non-iid ecg,” in *Proceedings of the 2020 IEEE 6th International Conference on Computer and Communications*, IEEE, 2020, pp. 1176–1180.

[199] D. Lin, Y. Guo, H. Sun, and Y. Chen, “Fedcluster: A federated learning framework for cross-device private ecg classification,” in *Proceedings of the IEEE Conference on Computer Communications Workshops*, 2022, pp. 1–6.

[200] O. Suciu, R. Marginean, Y. Kaya, H. Daume III, and T. Dumitras, “When does machine learning FAIL? generalized transferability for evasion and poisoning attacks,” in *Proceedings of the 27th USENIX Security Symposium*, USENIX Association, 2018, pp. 1299–1316.

[201] A. Raza, K. P. Tran, L. Koehl, S. Li, X. Zeng, and K. Benzaidi, *Lightweight transformer in federated setting for human activity recognition*, arXiv:2110.00244 [cs.CV], 2021.

[202] D. Cao, S. Chang, Z. Lin, G. Liu, and D. Sun, “Understanding distributed poisoning attack in federated learning,” in *Proceedings of the 25th International Conference on Parallel and Distributed Systems*, 2019, pp. 233–239.

[203] C. Fung, C. J. Yoon, and I. Beschastnikh, *Mitigating sybils in federated learning poisoning*, 2018.

[204] X. Cao and L. Lai, “Distributed gradient descent algorithm robust to an arbitrary number of byzantine attackers,” *IEEE Transactions on Signal Processing*, vol. 67, no. 22, pp. 5850–5864, 2019.

[205] L. Muñoz-González, K. T. Co, and E. C. Lupu, *Byzantine-robust federated machine learning through adaptive model averaging*, arXiv preprint arXiv:1909.05125, 2019.

[206] K. Pillutla, S. M. Kakade, and Z. Harchaoui, *Robust aggregation for federated learning*, arXiv preprint arXiv:1912.13445, 2019.

[207] C. Xu, Y. Jia, L. Zhu, C. Zhang, G. Jin, and K. Sharif, "Tdfl: Truth discovery based byzantine robust federated learning," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 12, pp. 4835–4848, 2022.

[208] H. Jeong, H. Son, S. Lee, J. Hyun, and T.-M. Chung, "Fedcc: Robust federated learning against model poisoning attacks," *arXiv preprint arXiv:2212.01976*, 2022.

[209] Chief Scientist Office (Scotland), Health and Care Research Wales, the Public Health Agency (Northern Ireland) and the National Institute for Health and Research (England), UK, *UK standards for public involvement in research*, Website, 2016.

[210] M. Naseri, J. Hayes, and E. De Cristofaro, "Local and central differential privacy for robustness and privacy in federated learning," in *Proceedings of the 29th Annual Network and Distributed System Security Symposium*, Internet Society, 2022.

[211] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.