Graph Based ECG Classification: A Continual Learning Framework for Evolving Cardiac Diagnosis

By
Al Shahariar Siam
213-15-4569
And
Jesmin Akter Sukhi
213-15-4563

FINAL YEAR DESIGN PROJECT REPORT

This Report Presented in Partial Fulfillment of the Requirements forthe **Degree of Bachelor of Science in Computer Science and Engineering**

Supervised by
Dr.Sheak Rashed Haider Noori
Professor & Head, CSE
Department of Computer Science and Engineering
Daffodil International University

Co- Supervised by
Dr. Md Zahid Hasan
Associate Professor
Department of Computer Science and Engineering
Daffodil International University

DAFFODIL INTERNATIONAL UNIVERSITY
Dhaka, Bangladesh

September 09, 2025

APPROVAL

This Project titled **"Graph-Based ECG Classification: A Continual Learning Framework for Evolving Cardiac Diagnosis"** submitted by **Al Shahariar Siam**, ID No: **213-15-4569 and Jesmin Akter Sukhi**, ID No:**213-15-4563** to the Department of Computer Science and Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Engineering and approved as to its style and contents. The presentation has been held on **16 September, 2025**.

BOARD OF EXAMINERS

16.09.25

Dr. Bimal Chandra Das (BCD) — **Chairman**
Professor and Dean (in-charge)
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Most. Hasna Hena (HH) — **Internal Examiner**
Assistant Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

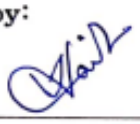
Mr. Mayen Uddin Mojumdar (MUM) — **Internal Examiner**
Assistant Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Nazibur Rahman — **External Examiner**
Technical Lead - Database Administrator
Telenor - Grameen Phone Account

DECLARATION

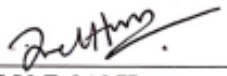
We hereby declare that this project has been done by us under the supervision of **Dr.Sheak Rashed Haider Noori, Professor & Head,** Department of Computer Science and Engineering, Daffodil International University. We also declare that neither this project nor any part of this project has been submitted elsewhere for the award of any degree or diploma.

Supervised by:



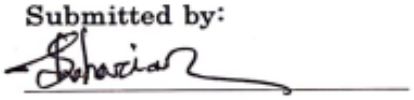
Dr.Sheak Rashed Haider Noori
Professor & Head, CSE
Department of Computer Science and Engineering
Daffodil International University

Co-Supervised by:

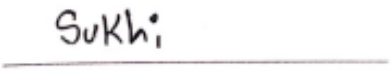


Dr. Md Zahid Hasan
Associate Professor
Department of Computer Science and Engineering
Daffodil International University

Submitted by:



Al Shahariar Siam
ID: 213-15-4569
Department of Computer Science and Engineering
Daffodil International University



Jesmin Akter Sukhi
ID: 213-15-4563
Department of Computer Science and Engineering
Daffodil International University

©Daffodil International University

ACKNOWLEDGEMENTS

This work would not have been possible without the support and contributions of many individuals over the past two semesters. We are deeply grateful to everyone who has assisted us in one way or another.

First, we express our heartfelt thanks and gratefulness to the almighty for His divine blessing making it possible for us to complete the **Final Year Design Project(FYDP)** successfully.

We are grateful and wish our profound indebtedness to **Dr. Sheak Rashed Haider Noori**, **Professor & Head**, Department of Computer Science and Engineering, Daffodil International University, Dhaka, Bangladesh. Deep knowledge and keen interest of our supervisor in the field of **Machine Learning** to carry out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts, and correcting themat all stages have made it possible to complete this project.

We would like to express our heartfelt gratitude to the Head of the Department of Computer Science and Engineering, for his kind help in finishing our project and also to other faculty members and the staff of the Department of Computer Science and Engineering, Daffodil International University.

We would like to thank our entire course-mates at Daffodil International University, who took part in this discussion while completing the coursework.

Finally, we must acknowledge with due respect the constant support and patience of our parents.

ABSTRACT

Recognition of ECGs is essential to early detection and tracking of heart conditions. Standard machine learning algorithms are not well suited to adapting to new data distributions and they do not generalize well to the real world clinical world where patient information continues to arrive. To eliminate this, we suggest a robust model that can be implemented as a combination of Graph Neural Networks (GNNs) and continual and lifelong learning to identify severe cardiac disorders: Normal (NORM), Myocardial Infarction (MI), ST/T Change (STTC), Conduction Disturbance (CD), and Hypertrophy (HYP). Based on the PTB-XL dataset, we clean ECG data and obtain significant time-domain features, entropy features, and complexity characteristics. These characteristics provide the global patient similarity graph where a patient is a node and the edges are built based on K-Nearest Neighbor (KNN) similarity. This graph is subsequently fed to GNN models that each learn a representation of patients and are then continuously adapted to new ECG classes. The system adapts itself on the arrival of new information related to the simulation of the deployment conditions to provide scalable and specific diagnosis of ECG. In our experiments, GraphSAGE has the highest accuracy, 97.29%, in comparison with GCN and GAT. It means that GraphSAGE is more confident with assigning various cardiac conditions. To increase flexibility, GraphSAGE model was used with continual learning. The model was trained step-by-step instead of training on all classes at once, with every step the model was presented with new classes and was to learn these classes with maintaining knowledge of previous classes. This has been effective in blocking catastrophic forgetting which is typical of machine learning where earlier learning is overridden when training new tasks. The conclusions were quite encouraging There was a slight increase in training accuracy overall, with the accuracy increasing by one percentage point between the first and the last task 92.57 to 96.94. This indicates the fact that the model maintained an accuracy and even increased with the addition of each class. Therefore, the approach of graph-based computation, innovative GNNs, and continual learning offer an ECG classification system that is effective, yet flexible. The proposed framework has a great potential of being applied in the real world clinical practice where medical data is variably changing with time and new disease patterns are created. It allows a variety of existing devices to be scaled, adapted, and be long-term data-drift-resistant, providing a very strong basis of intelligent, dynamic clinical decision support systems.

Table of Contents

List of Figures

List of Tables

Chapter 1
Introduction

We present some basic information about our work in this chapter such as introduction, motivation, goal, methodology and project outcome.

1.1 Introduction

Electrocardiogram (ECG) signal classification can help diagnose and monitor onset of cardiac injury, such as Myocardial Infarction (MI), Hypertrophy (HYP) and Conduction Disturbances (CD). The electrocardiogram signals have a wide usage in hospitals and clinics as a tracking and diagnosing tool of various cardiac problems. The appropriate categorization of these signals helps the physicians in identifying symptoms like heart attacks (Myocardial Infarction), hypertrophy and conduction problems in their opportune stages. Traditional machine learning models have challenges in updating to the changes in patient data in real time and can lack the ability to keep archived knowledge intact over time. To overcome such drawbacks, the proposed project proposes a new model that applies Graph Neural Networks (GNNs) and both continual and lifelong learning. Our system can learn and advance based on the patient's data consecutively. We have employed the famous PTB-XL ECG set to conduct training and testing operations. The system is built with the aim to categorize the patients into the five most common heart condition indicators and will have the flexibility to adjust to new ones when they come along.

1.2 Motivation

Heart disease is one of the leading causes of death worldwide. Detecting it early using ECG signals can save many lives. However, ECG data collected from patients keeps changing, and new patterns emerge over time. Most traditional systems fail to keep up with such changes. We were motivated to solve this problem by designing a system that learns from data in a continuous way just like a human doctor learns from experience. We also wanted to use graph-based models because they can capture relationships between patients based on their ECG patterns.

1.3 Objectives

The main objective of this project are:

i. Classify ECG signals into multiple diagnostic categories using GNN
ii. Build a graph from ECG data where each patient is represented as a node.
iii. Extract meaningful ECG features from time-domain, entropy and complexity analysis.
iv. Visualize feature distributions, evaluate their importance, and analyze model performance.
v. Train and evaluate Graph Neural Network (GNN) models like GCN, GAT and GraphSAGE.
vi. Implement a continual learning technique so that the model can learn new classes over time without forgetting the previous ones.

1.4 Methodology

In this work we proposed ECG classification framework consists of five key components: patient level dataset preparation, ECG signal preprocessing, feature extraction, graph construction, and GNN-based model development with continual learning integration. Raw 12-lead ECG signals were first de-noised using wavelet transforms and baseline removal to suppress high-frequency noise and artifacts. These de-noised signals were then merged into a single representative signal by averaging across all 12 leads, resulting in a unified waveform per ECG record. From each of these averaged signals, clinically relevant and hand-crafted features were extracted. A global patient similarity graph was constructed using K-Nearest Neighbors (KNN) based on the extracted feature. Each node represents a patient, and edges reflect similarity in cardiac characteristics. In our work we train three GNN models, including GCN, GraphSAGE, and GAT, to classify patients into five diagnostic classes: Normal, Myocardial Infarction, ST/T Change, Conduction Disturbance, and Hypertrophy [1]. To address the evolving nature of clinical data, continual and lifelong learning strategies were integrated into the training process, allowing the model to adapt to newly arriving patient data while preserving knowledge from earlier patients. This end to end approach ensures accurate, scalable, and adaptive classification of ECG conditions in a graph based learning.

1.5 Project Outcome:

NORM, MI, STTC, CD, and HYP are the five heart conditions that can be identified by this graph-based ECG classification system. We extracted 18 key features and applied signal preprocessing before using K-Nearest Neighbors (KNN) to create a patient similarity graph. For node level classification, three Graph Neural Network models (GCN, GraphSAGE, and GAT) were trained. A task based continuous learning approach was used, which allowed the model to learn new classes over time without forgetting previous ones. Cohen's d and t-tests were used to analyze feature significance, and performance metrics and visualizations were used to support the findings. Future clinical applications may benefit from the system's scalability.

1.6 Organization of the Report

This report is divided into six chapters where Chapter 1 Gives an introduction to the project. It explains the background, why the work is important, the goals, the method we plan to use, what we expect to achieve, and how the report is organized. Chapter 2 Reviews previous research on ECG classification, Graph Neural Networks (GNNs), and continual learning. It also shows a gap analysis to explain what is missing in past studies and how our system solves these problems. Chapter 3 Explains the method we used. This includes how we prepared the dataset, cleaned the ECG signals, extracted features, built the patient graph, selected models (GCN, GAT, GraphSAGE), added continual learning, and measured performance. Chapter 4 Presents how we implemented the system, the results we got, comparisons between models, and a discussion of the findings. Chapter 5 Talks about the engineering standards we followed, the challenges we faced, and the effects of our system on society, the environment, and ethics. Chapter 6 Gives the conclusion, points out the limitations of our work, and suggests ideas for future improvements.

Chapter 2
Background

In this chapter we write about related literature, those we analysis for our work.

2.1 Introduction

In this section, we look at recent research studies on how electrocardiogram (ECG) signals can be classified using modern machine learning methods. These include Graph Neural Networks (GNNs), deep learning models, and techniques that allow a system to keep learning over time without forgetting old information. The goal of this review is to understand what other researchers have done, what worked well, and what problems still remain. This helps us see the gap that our own work will try to fill. The studies we review here use different approaches such as graph models based on signal connections, hierarchical GNNs, transformer networks, and models that explain how they make decisions. They have been tested on well-known datasets like MIT-BIH Arrhythmia, PTB-XL, PhysioNet/CinC 2020, and ECG5000. Many of these methods have achieved high accuracy, but there are still some common challenges. These include dealing with noisy ECG signals, adapting to new disease types, reducing the complexity of models, and making better use of the relationships between ECG leads. Table 2.1 gives a summary of each study, including the methods used and the main findings. This summary forms the basis for identifying where improvements can be made and how our proposed method addresses those issues.

2.2 Literature Review

Classification of ECG signals has been an important research area in biomedical Engineering. Recent advancements focusing on Graph neural network (GNNs), deep learning and continual learning techniques. In Han proposed an approach "Arrhythmia classification using Graph neural network based on confusion matrix. Han et al. [2] utilizes Graph Neural Networks (GNNs) to identify irregular heartbeats. It transforms ECG signal features into a graph based on their connections and applies a GraphSAGE model to classify different types of heartbeats. Lead II data from 44 ECG recordings in the MIT-BIH Arrhythmia database were used to evaluate the approach. The three types of heartbeats (normal, supraventricular, and ventricular) were the primary focus. The model performed well for normal and ventricular heartbeats, but not easily for supraventricular beats. Using a single ECG lead and a small dataset is one of its main problems. Using more leads and a larger dataset for improved accuracy are possible future improvements. Zhao et al. [3] wants to improve the automatic identification of heart problems using 12-lead ECG signals and created a model called ECGNN. The model uses a feature extractor to find ECG signals and a graph neural network to show how the leads are connected. ECGNN was tested on the big ECG datasets, PTB-XL and ICBEB2018, and achieved a high accuracy with 90% better than other models. However, the model struggles with noisy ECG data and may not work well in real-world data. Also, having 12 leads makes it complex to make good graphs and train the neural network. Śmigiel et al. [4] want the automatic classification of ECG signal using deep learning methods and compares three models (CNN, SincNet model, and CNN that adds entropy feature to better understand the signal). These models were tested on the large PTB-XL dataset, which contains over 21,000 ECG records. The CNN with entropy features achieves 90%

accuracy in separating healthy and unhealthy hearts. However, these models are very large and more complex. On the other hand, SincNet model didn't work well because it's designed for speech rather than cardiovascular signals. Zhou et al [5] proposed a deep learning model called MTECG based on the transformer architecture, its breaks the ECG signal and learned from each piece step by step for better understanding. Using a large dataset named Fuwai dataset and PTB-XL and MTECG model performed very well, rather than the latest models, also improved F1 score by up to 27.5%. On the other hand, Fuwai dataset also gave good output. However, the main problems were overfitting noisy data and the complexity of using transformer models for ECG signals. MTECG performed very well, but also needs development to be practical use. Sethi et al. [6] want training a computer is to find heart problems from the ECG signal, and also create a model named ProtoECGNet that helps to how it makes decisions. The model evaluates the ECG signal from three aspects (heartbeat rhythm, tiny changes, and overall issues) and compares the new pattern with the previously learned prototype. It performed 92.5% accuracy on 21,000 ECGS from the PTB-XL dataset. But there were issues that disease groups can't be changed, and some health conditions are similar to each other. Still mandatory for more testing in a real hospital, and uses old neural networks. Peng et al. [7] Created a new model called ECGNet that finds heart rhythm problems using 12-lead ECG signals. It uses a graph-based approach to find patterns over time and identify relationships between various heart diseases. Using different type of datasets (PhysioNet/CinC 2020, PTB-XL, and CPSC) and finds 98% high accuracy. The trouble is model doesn't use exact position of ECG leads and is struggling with rare heart issues, also careful coordination is required. The model requires further testing in hospitals. Xiu et al [8] mentioned privacy-secure way to keep patient information safe and private while analyzing ECG signals. The model named TLVG-GIN first turns the heart signal into a graph that shows how signals change over time, then uses a deep learning method to classify signals. When applied to the ECG5000 dataset, which contains five different signals, it achieves 93.02% accuracy at a time. Here, the issue was that when the signals are converted into a graph, some useful information might be lost. Since the dataset is small and clean, it's applied to larger and more complex ECG data before using it. Wang et al. [9] how graph neural networks (GNNs) learn so that they can learn so much information about new data without forgetting old information. Created a new method named the feature graph, where the features of the nodes are used as new nodes. Through this, the model can easily train in small batches and can use lifelong learning similar to CNNs. The model was evaluated on four popular datasets and performed better than previous models, also 24% reduce errors in image matching. Gao et al [10] wants to improve ECG signal analysis so that the model can understand small patterns of the heart signal and overall heart rhythm more accurately. To find a better development, they created a model named ECG-CL, which helps to learn multiple ECG signal datasets without forgetting old information that they've already learned. The model tested four types of datasets and achieved 99% accuracy in segmenting signals and 92%-98% identifying heart disease. Using a lifelong learning method, it becomes more effective and more accurate than multiple model training. To improve Graph Neural Networks in a way that researchers can learn so much information about new data without forgetting old one, which is known as lifelong or continual learning. The researchers present a new model called RAM-CG, and this model helps to find strong connections between the nodes in a graph, which does not change even after the data changes. This model uses three datasets and improves 2.2% to 6.6% results more accurately, and it reduces the problem of forgetting previous information while learning. [11]

Table 2.1: Summary of Literature Reviewed.

Author (s)	Year	Title	Methodology	Key Findings
Han et al. [2]	2024	Arrhythmia Classification Using Graph Neural Networks Based on Correlation Matrix	Used GraphSAGE model on correlation-based graphs from ECG features; Lead II data from MIT-BIH Arrhythmia dataset	Found significant trends in data science applications.
Zhao et al. [3]	2022	ECGNN: Enhancing Abnormal Recognition in 12-lead ECG with Graph Neural Network	Combined feature extractor with GNN for lead-relationship modeling; tested on PTB-XL & ICBEB2018	Achieved 90% accuracy; struggled in noisy environments; complexity due to 12-lead data
Śmigiel et al. [4]	2021	ECG Signal Classification Using Deep Learning Techniques Based on the PTB-XL Dataset	Compared CNN, SincNet, and CNN with entropy features on PTB-XL dataset	CNN with entropy features achieved 90% accuracy; SincNet underperformed; models large and complex
Zhou et al. [12]	2023	Masked Transformer for Electrocardiogram Classification	Transformer-based MTECG model segmenting ECG signals; tested on Fuwai & PTB-XL datasets	Improved F1-score by up to 27.5%; risk of overfitting noisy data; high computational cost
Sethi et al. [6]	2025	ProtoECGNet: Case-Based Interpretable Deep Learning for Multi-Label ECG Classification with Contrastive Learning	Case-based prototype comparison from three ECG perspectives; tested on PTB-XL	Achieved 92.5% accuracy; cannot dynamically change disease groups; needs more hospital testing
Peng et al. [7]	2024	EGCNet: A Hierarchical Graph Convolutional Neural Network for Improved Classification of Electrocardiograms	GNN-based temporal pattern analysis on multiple datasets	98% accuracy; does not use exact lead positions; limited performance on rare conditions

Xiu et al. [8]	2022	Time Labeled Visibility Graph for Privacy-Preserved Physiological Time Series Classification	Converts ECG to time-labeled visibility graph; applied GIN for classification	Achieved 93.02% accuracy on ECG5000; may lose information on larger/noisy datasets
Wang et al. [9]	2022	Lifelong Graph Learning	Proposed "feature graph" for continual learning in GNNs; tested on 4 datasets	Reduced errors by 24%; supports incremental training without forgetting
Gao et al. [10]	2023	ECG-CL: A Comprehensive Electrocardiogram Interpretation Method Based on Continual Learning	Lifelong learning ECG model tested on 4 datasets	99% segmentation accuracy; 92–98% classification accuracy; better than multi-model training
Shen et al. [11]	2023	Graph Relation Aware Continual Learning	RAM-CG method to preserve strong graph connections under data changes	Accuracy improved by 2.2–6.6%; reduced catastrophic forgetting

2.3 Gap Analysis

Table 2.2 show us the gaps of existing work

Table2.2: Gap Analysis

Features	**ML Models**	**Deep Learning Models**	**GNN Based Models**	**Proposed System**
Learns new classes over time	No	No	No	YES
Keeps past knowledge when new data comes	NO	NO	NO	YES
Uses patient-to-patient connection (graph)	NO	NO	YES	YES
Explains which ECG features are important	NO	SOMETIMES	SOMETIMES	YES
Good for real hospital settings	NO	LIMITED	NO	YES
Easy to update with new diseases or patients	NO	NO	NO	YES

2.4 Summary

This chapter reviewed recent research on ECG classification using machine learning, deep learning, and Graph Neural Networks (GNNs). High accuracy was accomplished in many studies, however, the majority still has problems with expanding new disease classes, retention of previous knowledge, contamination with noisy data, and applicability to realistic real-world hospital application. Although GNN models can capture patient-to-patient relationships, they typically do not have lifelong learning or flexibility to new data. Our proposed system bridges these shortcomings because the GNNs can learn new classes as time passes and can keep track of previous information, explanations of the key aspects of ECGs, and it can be quickly updated to be used in hospitals. Our proposed model is based on combining continual learning and graph-based learning approaches, the system can Learn new disease classes over time without losing past knowledge, maintain interpretability by highlighting important ECG features, handle noisy and changing ECG signals more effectively, can be updated and scaled easily for use in real clinical environments, this integration makes the proposed system not only accurate but also flexible, robust, and closer to real-world medical needs.

Chapter 3
Research Methodology

This chapter describes the complete process used to build our ECG classification system. It covers data preprocessing, feature extraction, graph construction, and the training of GNN models. Finally, it explains how continual learning was applied to make the system adaptable and able to learn new disease classes over time.

3.1 Methodology

3.1.1 Overview:

In this work, a continuous learning framework [13] for automatic ECG classification using Graph Neural Networks (GCN, GAT, GraphSAGE) is developed using the PTB-XL ECG dataset (21,799 recordings, 18,869 patients). The raw 12-lead ECG signals were cleaned by removing slow drifts (Butterworth filter, 0.5 Hz), reducing high-frequency noise (wavelet method, db6, level-3), and adjusting amplitude (±0.3 mV, averaged across leads). From these clean signals, we extracted 13 time-based features, 3 entropy features, and 2 complexity features. A patient to patient connection graph was made using KNN, where each node is a patient and edges link patients with similar ECG patterns. The models were trained on balanced data and tested for accuracy, F1-score, and confusion matrix. A continual learning approach was used, adding new heart condition classes in stages (NORM, MI, CD, STTC, HYP) so the model could learn new diseases without forgetting older ones, similar to how doctors gain experience over time.

3.1.2 Proposed Methodology

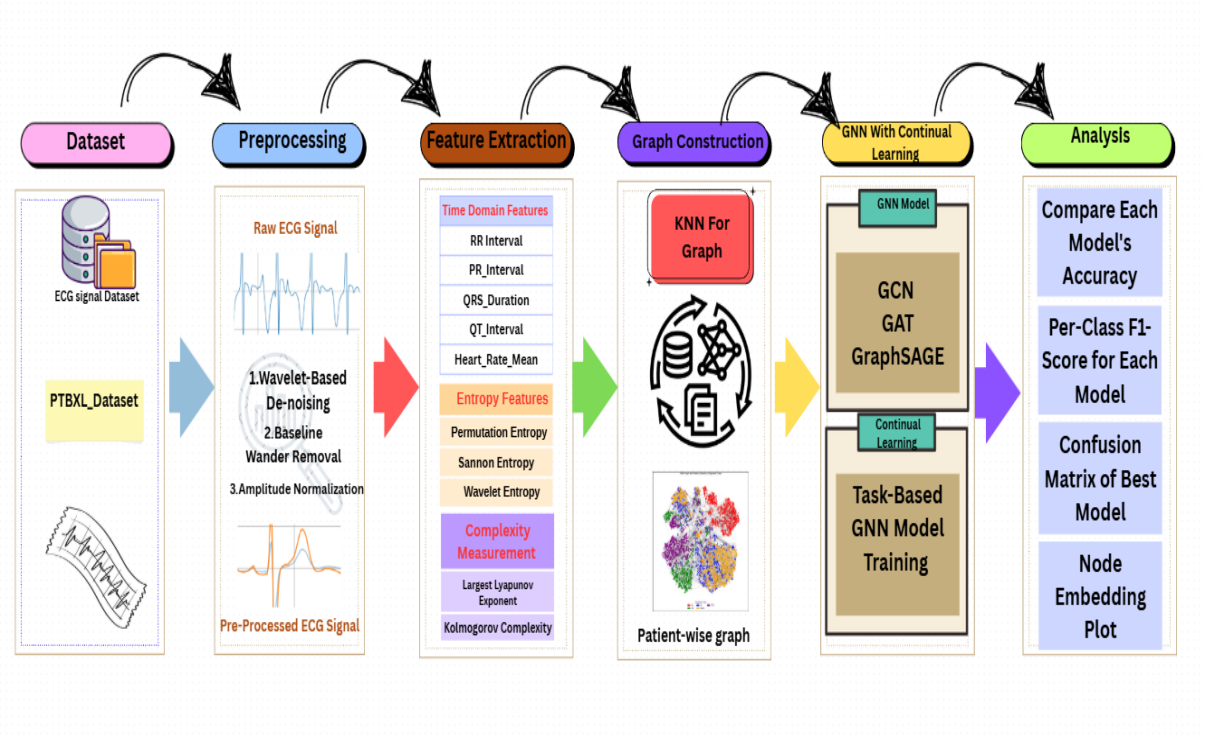


Figure 3.1: Proposed Methodology

3.2 Detailed Methodology

3.2.1 Dataset:

we select publically available dataset PTB_XL dataset from physioNet. The PTB-XL ECG dataset contains 21,799 12-lead ECG recordings from 18,869 patients, each 10 seconds long. There are 71 different ECG statements based on the SCP-ECG standard. Data includes: 12-lead ECG recordings, diagnostic labels, metadata like patient ID, recording date [14].

3.2.2 Data Preprocessing:

To enhance signal quality and facilitate reliable feature extraction, the raw ECG signals underwent a structured preprocessing pipeline consisting of the following stages:

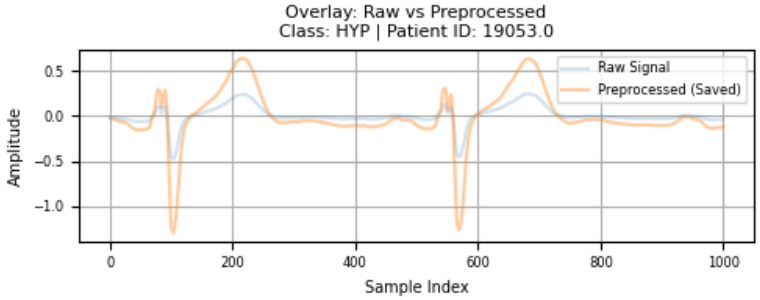


Figure 3.2 : Row VS Preprocessed Signal

Baseline Wander Removal:
Baseline drift, typically introduced by respiration and electrode motion, was removed using a fourth-order high-pass Butterworth filter with a cutoff frequency of 0.5 Hz. This effectively suppresses low-frequency components without distorting critical waveforms [15].
Wavelet-Based De-noising:
To eliminate high-frequency noise, a discrete wavelet transform (DWT) using the Daubechies-6 (db6) wavelet at level 3 decomposition was applied. The universal threshold was estimated based on the standard deviation of the noise, and soft thresh holding was used to suppress irrelevant detail coefficients [16].

3.2.3 Amplitude Normalization

The de-noised signal was standardized to have zero mean and unit variance, then scaled to a fixed physiological range of ±0.3 mV. To derive a robust, patient-level representation, signals from all 12 leads were averaged point-wise. ensured that the ECG signals were free from baseline wander, high-frequency noise, and amplitude inconsistencies, thereby improving the accuracy and stability of subsequent diagnostic analysis. Row vs Preprocessed data example shown in figure-3.2

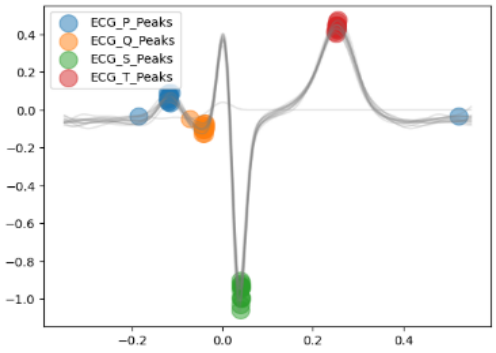


Figure3.3:Signals Peak

3.2.3 Feature Extraction:

All of those feature is extracted from Preprocessed Signal,
Duration Based Time-Domain Features:
To capture physiological characteristics of cardiac cycles, three duration-based features were computed by identifying key ECG landmarks using NeuroKit2's peak detection outputs: PR Duration: Median time between P wave onset and Q wave onset. QRS Duration: Median time between Q and S wave peaks, representing ventricular depolarization. QT Duration: Median time between Q wave onset and T wave peak, indicating total ventricular activity. [17]
These durations were derived from peak index pairs (s, e) as follows:

$$Duration = meian(\frac{e-s}{fs}) \qquad \text{Equation 1}$$

where fs=500Hz is the sampling frequency. The method ensured physiological validity by enforcing e>s and computed the median of the resulting intervals to mitigate outliers [18].

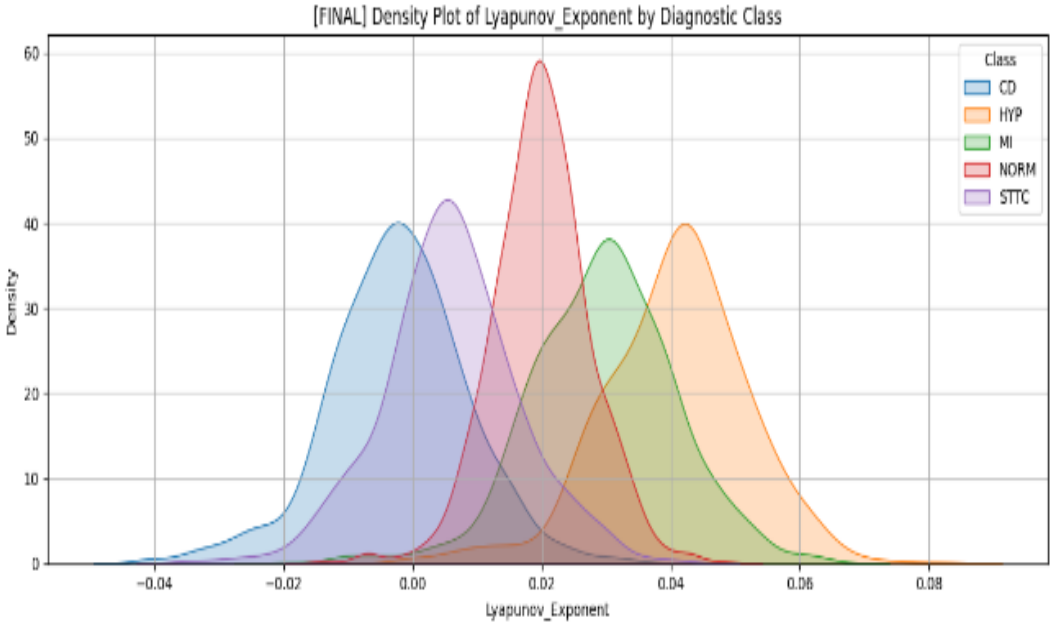


Figure 3.4: Example Density plot of class

After this we go for main feature extraction process. Those feature are:
Time domain features:
We extracted 13 time domain feature including RR interval, PR duration, QT interval, QRS duration, wave amplitudes, U-wave count, etc.
Entropy Measures:
We extracted Permutation Entropy, Shannon Entropy, Wavelet Entropy [19].
Complexity Measures:
We extracted Largest Lyapunov Exponent, Kolmogorov Complexity [20].
Feature Analysis and Visualization:
We make density plots and apply statistical Testing T-test: for feature significance between classes and Effect Size: Use cohen's d to determine clinical relevance. Finally apply density plot for class-wise plotting. Example shown in figure-3.4 and figure-3.5

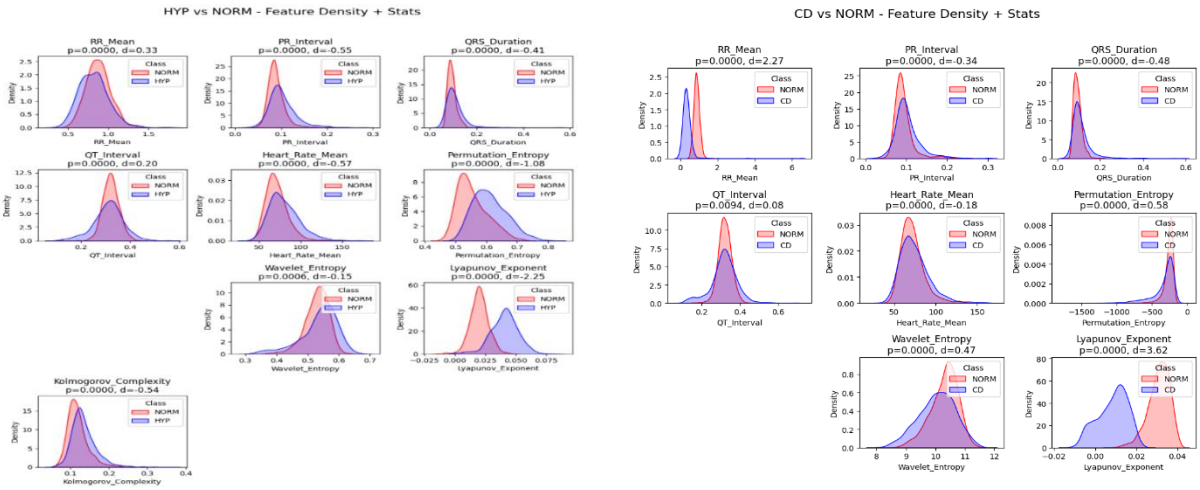


Figure 3.5: Example density plot with T-test and Effect Size

After complete analysis of our extracted feature we move forward for graph constriction. For graph constriction. We select KNN (K-nearest neighbors) algorithm. KNN graph constructed using Euclidean distance of feature vectors, where each node represent one patient and edge are connected through similar patient and nodes colored based on diagnostic class for visualization [5]. Shown on figure 3.6.

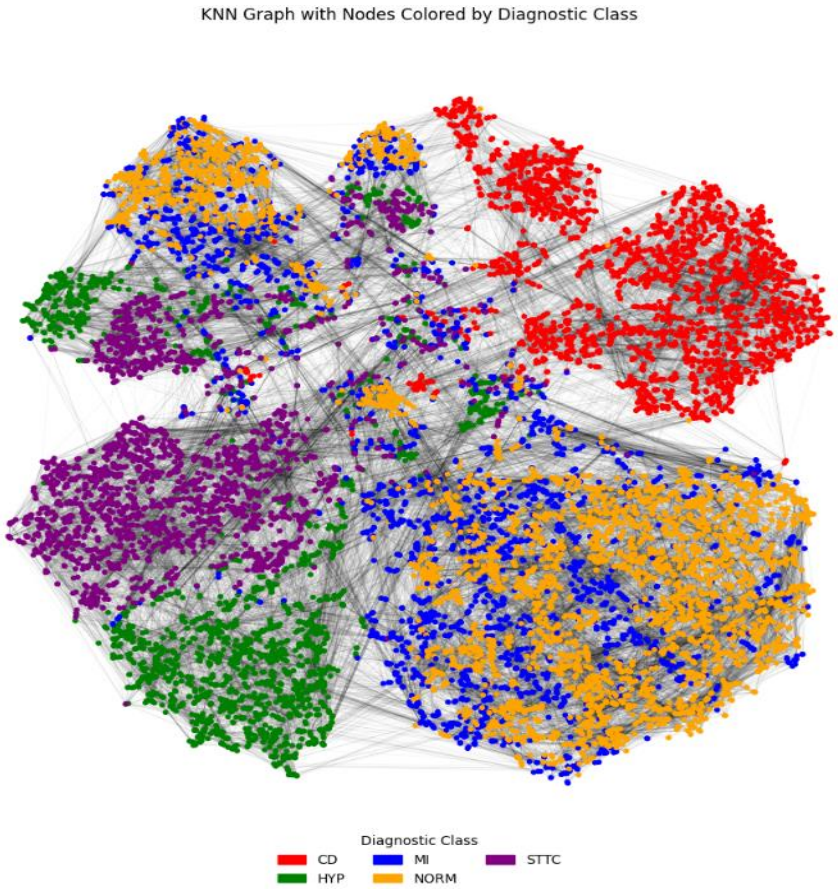


Figure 3.6: KNN Graph

3.2.4 Model Training

In this study, we employed Graph Neural Networks) to perform node-level classification on an ECG-based patient graph. Three prominent GNN architectures were selected for comparative analysis, those are Graph Convolutional Network [21], Graph Attention Network [22], GraphSAGE [23].

These models were chosen for their proven effectiveness in learning representations from structured data like graphs. Each node in our graph represents a patient, and edges are defined based on feature similarity using a K-Nearest Neighbor (KNN) approach.

The GCN model leverages spectral graph convolution by aggregating features from neighboring nodes, allowing it to learn smooth and localized patterns in the graph domain.

The GAT model introduces a self-attention mechanism to assign different weights to different neighboring nodes, which helps the model focus on the most informative connections.

The GraphSAGE model enables inductive learning by aggregating neighbor features through mean, max-pooling, or LSTM-based operations. This makes it more scalable and adaptable to unseen nodes during test time.

All models were trained using the cross-entropy loss function, optimized using the Adam optimizer for 200 epochs. Performance was evaluated on accuracy, F1-score, and confusion matrix for the multi-class classification task involving five diagnostic classes: Normal (NORM), Myocardial Infarction (MI), ST/T Change (STTC), Conduction Disturbance (CD), and Hypertrophy (HYP).

To prevent overfitting and ensure robust generalization, we applied stratified splitting (60% training, 20% validation, 20% test) using class labels.

3.2.5 Continual Learning:

To make our model behave more like a real doctor who keeps learning with each new patient, we used a Continual Learning (CL) approach. Instead of training on all ECG classes at once, we trained the model step by step, starting with only 2 classes (like NORM vs MI), then added more (like STTC, CD, etc.) in later steps. The model was not retrained from the beginning. It kept previous knowledge and learned new classes gradually. This simulates a real clinical environment where new conditions appear over time. [5]

Continual Learning Task Setup:

We developed a task-incremental learning situation in which several tasks were utilized for teaching ECG classes one at a time which is shown in table 3.1.

Table 3.1: Define Task

Task	Classes Included
1	NORM, MI
2	NORM, MI, CD
3	NORM, MI, CD, STTC
4	NORM, MI, CD, STTC, HYP

The model learned new classes yet retained previous acquired expertise through an organized training process.

Task-Based Data Preparation and Model Training:

Our system was trained by means of a series of incremental tasks that established one or more new ECG diagnostic classes in order to allow continuous learning. The methods for creating task-specific data, training the model across tasks, and storing trained models for later use are all covered in this section.

To simulate a real world clinical environment where new types of ECG data arrive over time, we divided the classification problem into four learning tasks. Each task included a growing number of diagnostic classes. Task and classes are shown in Table 1.

To extract the relevant data for each task, we implemented a function called get_task_data() which performed the following steps:

Class Filtering: It took the whole dataset and only kept the ECG records that were part of the desired classes for the current task.

Feature Extraction: It chose the node features 10 features per patient from the graph that matched the filtered subset.

Label Encoding: It used a Label Encoder to turn string labels like "MI" and "HYP" into numbers so that they could be used in PyTorch models.

Edge Retention: To keep things simple, the full graph structure all the edges between patients was kept the same during each task to keep the graph topology the same.

This task-specific data preparation helped ensure that each learning phase used only relevant classes, while still benefiting from the overall graph structure.

Training Strategy and Implementation:

To mimic the process of continuous learning and handle model training across different tasks, we came up with two training routines.

We used two training routines to make the model learn step-by-step, similar to how doctors gain experience over time.

(i) train_up_to_task()

This function trained the selected GNN models from scratch up to a specific task number. Learning starts with Task 1 and moves forward in order.

Key elements of the training setup included:

i. Loss Function: CrossEntropyLoss for multi-class classification.
ii. Optimizer: We used the Adam optimizer with a fixed learning rate of 0.005, known for stable convergence.
iii. Train-Test Split: To maintain class balance, we used StratifiedShuffleSplit, which ensures each class is proportionally represented in both the training and testing sets.
iv. Epochs: 100 per task to learn without overfitting.

By training the model incrementally through multiple tasks, we enabled it to adapt to new classes while still being evaluated on previous ones, which is essential in continual learning research.

(ii) pretrain_model()

Same as above, but also saves the model after each task. These saved models are useful for later testing, comparing results, or real-world use.

For each task, only ECG data for the chosen heart conditions was used. As tasks progressed, the model learned new conditions while keeping knowledge of earlier ones. This step-by-step approach allowed gradual improvement without forgetting past learning.

3.3 Project Plan

The project was executed in a set of distinct, progressive steps to ensure that the project acted progressively till its conclusion was achieved on time. We assembled and prepared the PTB-XL database, which has thousands of recordings of ECGs of the various patients. We then washed and manipulated the ECG readings in order to eliminate noise and prepare them to be analyzed. Based on these cleaned signals, we are extracting important features that provide description of various aspects of the heart operation. We used these features to compose a patient similarity graph where each patient is a node and is linked to other similar patients in terms of ECG patterns. Having prepared the graph, we then applied three different Graph Neural Network (GNN) models GCN, GAT, and GraphSAGE to predict the best performance. Next, we added a continual learning system making it able to continue its learning process and pick up new heart conditions without forgetting what it had already learned at the same time. Finally, we created a demo application to show how the system works in practice, making it easy to understand and test in a real-world setting.

3.4 Task Allocation

Overview of the major activities performed during the project shown in table 3.2.

Table 3.2: Task Allocation

Tasks	Weeks													
	1	3	5	7	9	11	13	15	17	19	21	23	25	27
Data collection phase														
Preprocess all data														
Feature Extraction														
Graph Constriction (KNN)														
Model Apply (GNN)														
Analysis Continual Lifelong Learning(CL)														
Apply GNN with CL														
Performance Analysis														

Estimated Work Period	
Actual Work Period	

3.5 Summary

This chapter elaborated on the entire procedure that we used in the construction of our ECG classification system based on continual learning. We acquired and preprocessed the PTB-XL dataset that consists of thousands of ECG recordings. The furious signals underwent the process of processing in order to eliminate unnecessary noises, drift, and other artifacts to make the data clear and consistent to analyze it. Following cleaning of the signals, we extracted significant features that characterize activities in the heart. These characteristics were time-based parameters such as PR interval, QRS, QT, entropy to capture the irregularities in the signal and complexity to encapsulate overall signal behaviour. Features gave a detailed picture of each patient's heart condition. we created a patient similarity graph using the KNN. In this graph, every node represented a patient, and edges connected patients with similar ECG patterns. This graph structure allowed us to model relationships between patients, which is something traditional methods cannot do properly. we trained three different Graph Neural Network (GNN) models: Graph Convolutional Network (GCN), Graph Attention Network (GAT), and GraphSAGE. These models were used to classify patients into five diagnostic classes. Then we applied a continual learning approach, where the model learned new disease classes step by step instead of training on all classes at once. This made the system behave more like a real doctor, who gains knowledge over time without forgetting what was learned before. For example, the model first learned to separate normal and MI cases, and later added more conditions like CD, STTC, and HYP, while still remembering the earlier classes.

Chapter 4
Implementation and Results

The chapter presents the detailed implementation, evaluation, and results of the Graph-based ECG classification system. We focus on the Graph Neural Network (GNN) models used, their performance, and the application of continual lifelong learn in the context of ECG classification. The chapter is structured into the following sections: Environment Setup, Testing and Evaluation, Results and Discussion, and Summary.

4.1 Environment Setup

The project was developed and run on Google Colab, which allowed us to use GPU resources for faster training of the models. We used the following libraries and tools to build and evaluate the ECG classification system. Used libraries and their purpose are presented in table 4.1.

Table 4.1: Used Libraries and Purpose

Library/Tool	Purpose
PyTorch	For building and training the Graph Neural Networks (GNNs) like GCN, GAT, and GraphSAGE
PyTorch Geometric	A library for graph-based neural networks (used to define SAGEConv, GCNConv, GATConv layers for the models).
Scikit-learn	For data preprocessing, Feature extraction and generating evolution metrix.
Matplotlib &Seaborn	Used for visualizing results such as confusion matrices, ROC curves, and t-SNE/UMAP visualizations of embeddings
NeuroKit2	A toolkit for ECG signal processing, including preprocessing steps such as denoising, feature extraction, and baseline removal.
TQDM	For providing progress bars during training and evaluation loops.
Gradio	To create a web interface for interacting with the trained model. Allows users to input a patient ID and get predictions based on different task and models

The PTB-XL dataset was used for training and testing the models. The dataset was preprocessed by applying techniques like baseline removal, wavelet de noising, and feature extraction entropy metrics, time-domain features. The data was then represented as a graph, with nodes representing patients and edges reflecting similarity between patients based on their ECG features.

4.2 Testing and Evaluation

We evaluation the performance of the Graph Network “GNN” models used in the ECG classification system. The evaluation involves several performance metrics to assess how well each model classifies ECG signals into their respective categories: Normal “NORM”, Myocardial Infarction “MI”, Conduction Disturbance “CD”, ST/T Change “STTC”, and Hypertrophy “HYP”.

Evaluation Metrics: Accuracy, Precision, Recall, F1-score, Confusion Matrix.
Testing Methodology:

Preprocessing techniques were evaluated using statistical analysis. Baseline wander removal and wavelet-based de-noising were applied to the ECG signals, and the results were presented using bar charts for comparison.

MSE Comparison:

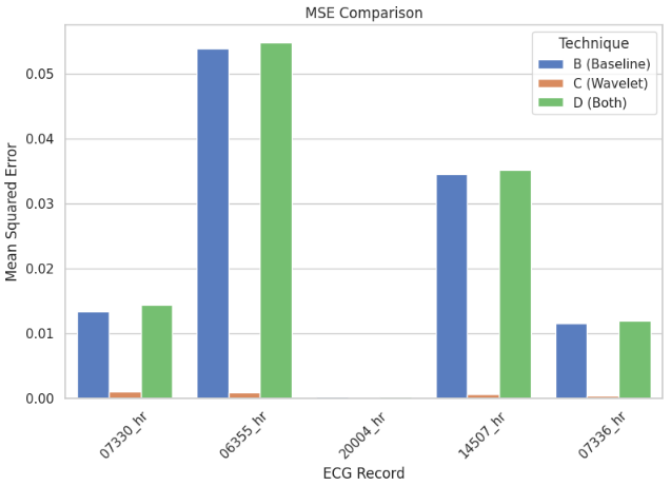


Figure 4.9: MSE Comparison

PNSR Comparison:

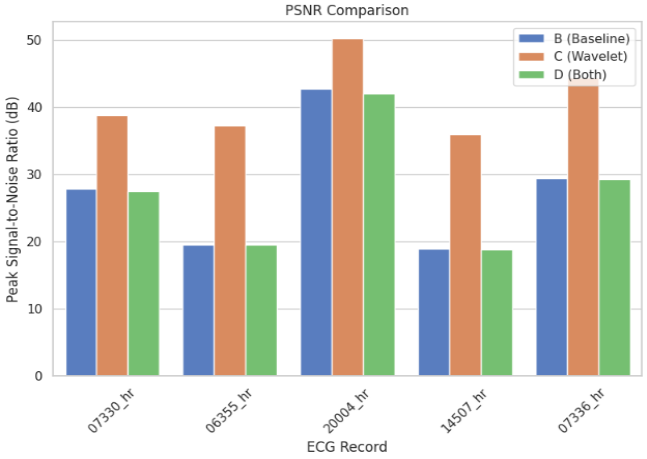


Figure 4.10: PNSR Comparison

The models were evaluated using 60% of the data for training, 20% for validation, 20% for testing, ensuring a balanced representation of all classes with Stratified Sampling. Each model was trained for 100 epochs, and performance was evaluated on the test set after each epoch. Total Three Model used in our work. The result of the evaluation is presented in the following section

4.3 Results and Discussion

4.3.1 GCN (Graph Convolutional Network) Results:

Result of GCN model shown in table 4.2.

Table 4.2: Result of GCN Model

Class	Precision	Recall	F1-Score
CD	0.99	0.98	0.98
HYP	0.95	0.93	0.94
MI	0.82	0.82	0.82
NORM	0.91	0.91	0.91
STTC	0.94	0.94	0.96

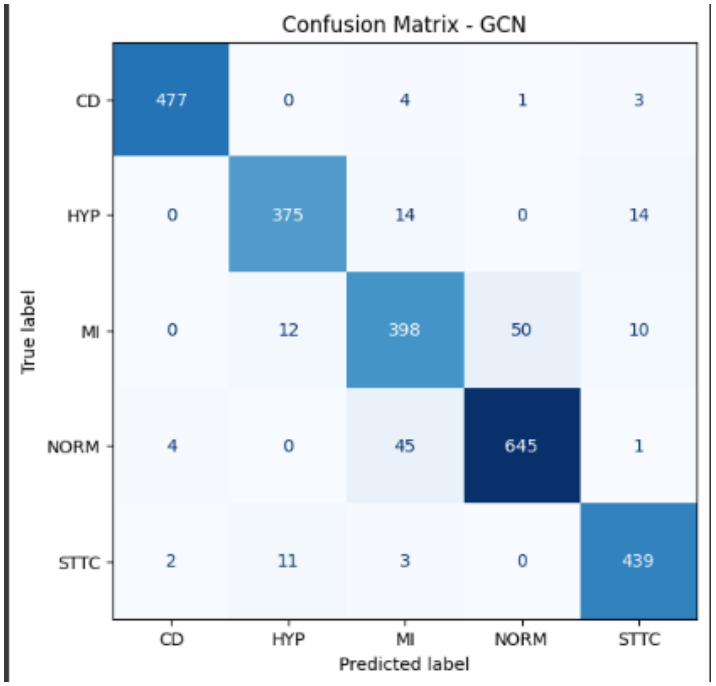


Figure 4.1: Confusion Matrix-

The GCN model performed well with a test accuracy of 92.07%, training accuracy of 92.57%, Validation accuracy of 92.22%, but showed slightly lower performance for the MI class due to its similarity with other conditions like STTC.

4.3.2 GAT (Graph Attention Network) Results:

Result of GAT model shown in table 4.3.

Table 4.3: Result of GAT Model

Class	Precision	Recall	F1-Score
CD	0.99	0.99	0.99
HYP	0.96	0.94	0.95
MI	0.85	0.84	0.85
NORM	0.92	0.93	0.93
STTC	0.95	0.96	0.96

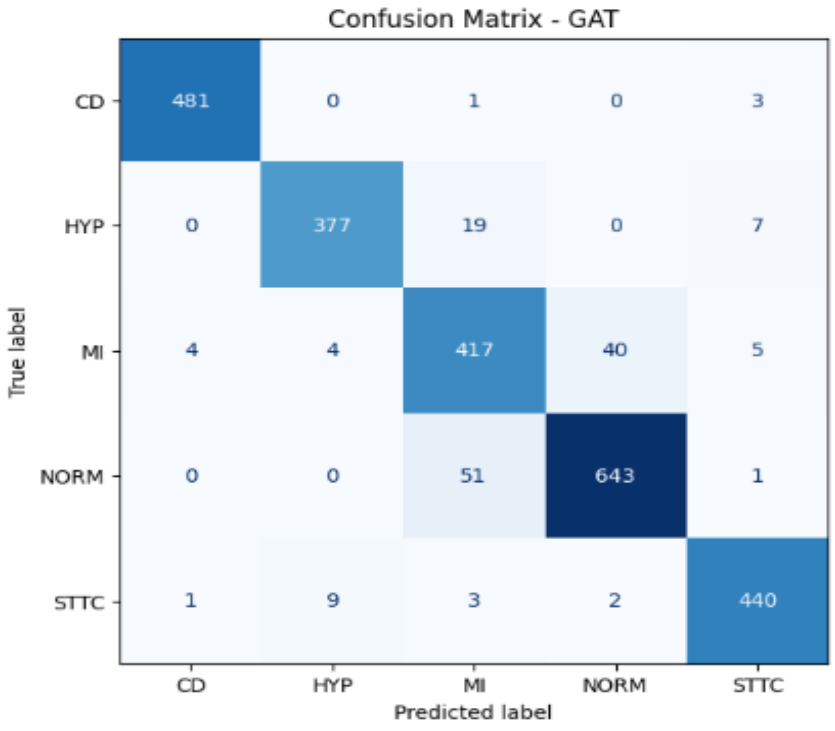


Figure 4.2: Confusion Matrix-GAT

The GAT model showed improved performance over GCN, achieving 93.26% accuracy, Training Accuracy of 93.47%, Validation Accuracy of 93.10%, The attention mechanism allowed the model to focus on important features, improving performance on challenging classes like MI.

4.3.3 GraphSAGE (Graph Sample and Aggregation) Results:

Result of GraphSAGE shown in table 4.4.

Table 4.4: Result of GraphSAGE Model

Class	Precision	Recall	F1-Score
CD	0.99	0.99	0.99
HYP	0.99	0.97	0.98
MI	0.95	0.93	0.94
NORM	0.97	0.97	0.97
STTC	0.97	1.00	0.98

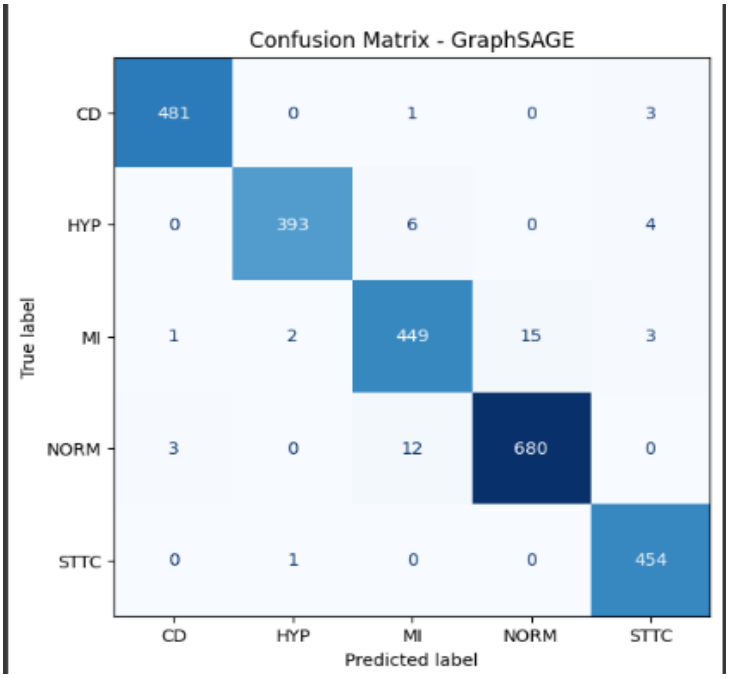


Figure 4.3: Confusion matrix- GraphSAGE

The GraphSAGE model achieved the highest accuracy of 97.29%, Training Accuracy of 97.27%, Validation Accuracy of 97.10%, Its sample-and-aggregate approach helped it generalize better and handle overlapping classes like MI and STTC.

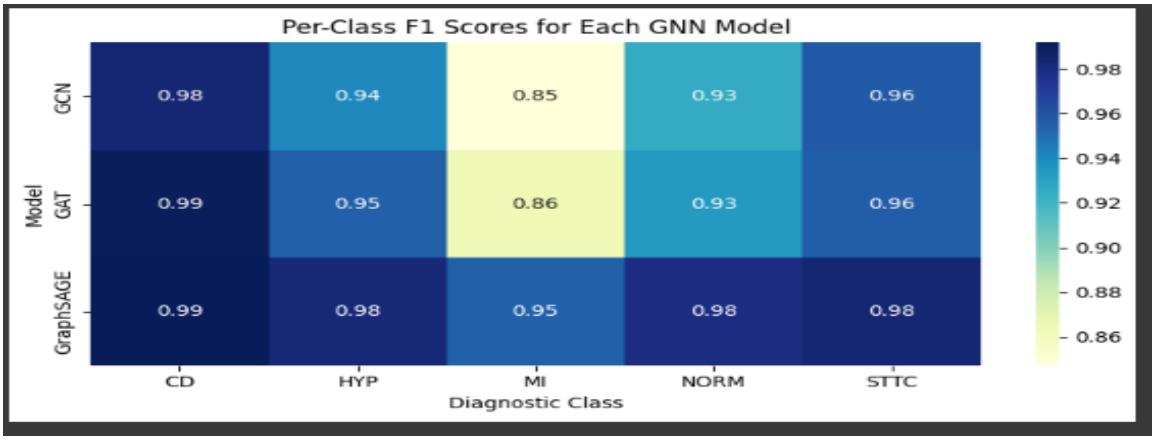


Figure 4.4: F1-Score for each GNN

4.3.4 Results After Applying Continual Learning with GNN Models:

Table 4.5 shows us the result of Continual learning with GCN.

Table 4.5: Results after applying CL with GCN

Task	Result
Initial Training Task 1(NORM,MI)	Training Accuracy 91%
Training Task 2(NORM, MI, CD)	Training Accuracy 93%
Training Task 3 (NORM, MI, CD, STTC)	Training Accuracy 93%
Training Task 4(NORM, MI, CD, STTC, HYP)	Training Accuracy 93%

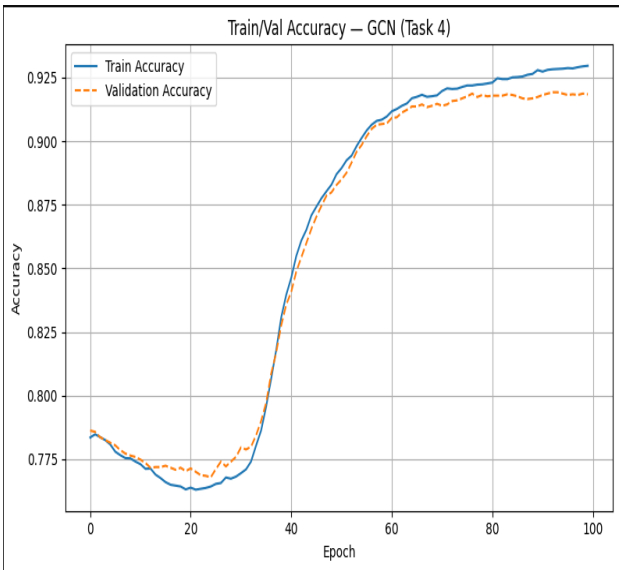


Figure 4.5: Train/VAL Accuracy GCN (Task 4)

Table 4.6 shows us the result of Continual learning with GAT.

Table 4.6: Results after applying CL with GAT

Task	Result
Initial Training Task 1(NORM,MI)	Training Accuracy 92%
Training Task 2(NORM, MI, CD)	Training Accuracy 94%
Training Task 3 (NORM, MI, CD, STTC)	Training Accuracy 93%
Training Task 4(NORM, MI, CD, STTC, HYP)	Training Accuracy 93%

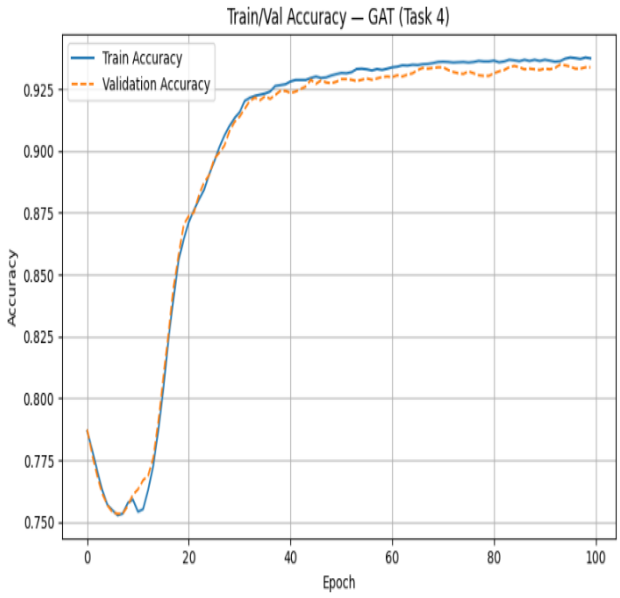


Figure 4.6: Train/VAL Accuracy GAT (Task 4)

Table 4.5 shows us the result of Continual learning with GraphSAGE.

Table 4.7: Results after applying CL with GraphSAGE

Task	Result
Initial Training Task 1(NORM,MI)	Training Accuracy 96%
Training Task 2(NORM, MI, CD)	Training Accuracy 98%
Training Task 3 (NORM, MI, CD, STTC)	Training Accuracy 98%
Training Task 4(NORM, MI, CD, STTC, HYP)	Training Accuracy 98%

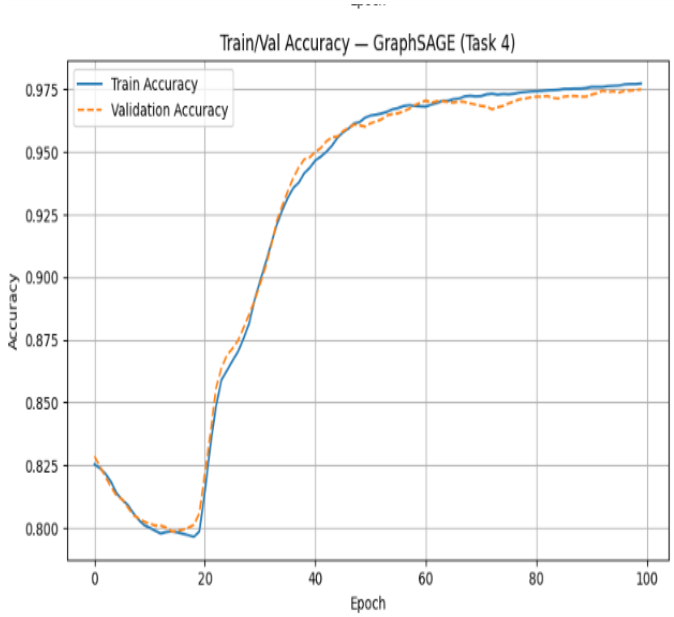


Figure 4.7: Train/VAL Accuracy GraphSAGE (Task 4)

4.3.5 Node Embedding (t-SNE cluster-wise):

We used t-SNE for dimensionality reduction and cluster visualization of node embedding [24].This allowed us to see clear class separation in the feature space

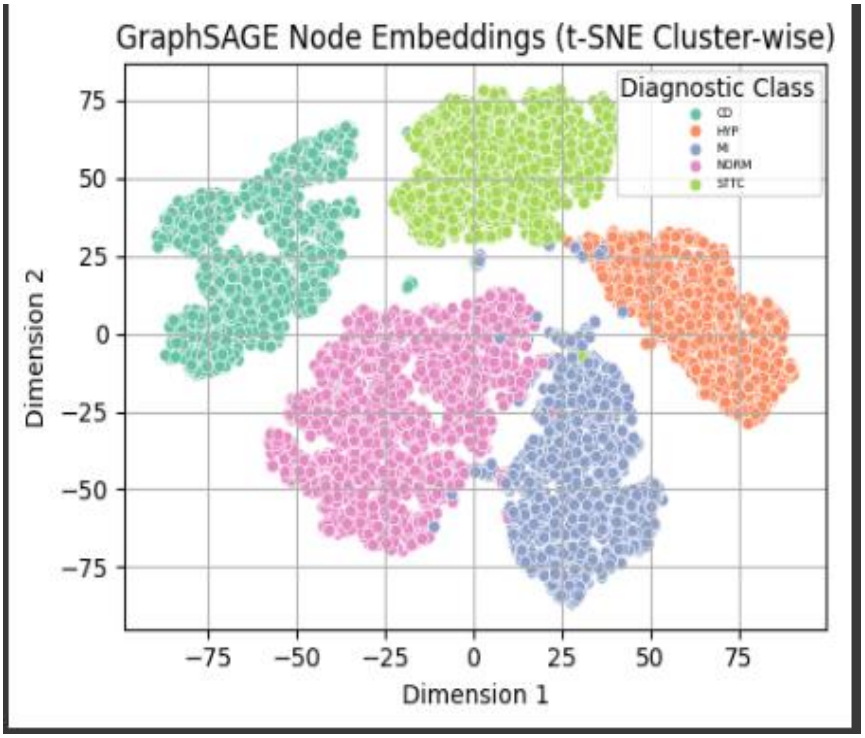


Figure 4.8: GraphSAGE Node Embedding Plot

4.4 Summary

We implemented and tested our ECG classification system using the PTB-XL dataset and three GNN models GCN, GAT, and GraphSAGE. Patient similarity graphs based on extracted ECG features were used to train the models. Of the three, GraphSAGE got the highest test accuracy of 97.29% performing well on all classes of heart conditions. We then used a method of continually learning on the GraphSage model, so that it could continually learn new conditions in the heart day by day without forgetting what was learned the previous days. After three tasks the training accuracy increased to 96.94 percent, when compared to the first task where the accuracy was 92.57 percent. The findings affirm the machine in distinction and classification of the ECG both accurately and learning new diagnostic categories with time. In short, this chapter demonstrated that GNN models maintained the benefits of continual learning by using this system to ensure scalable and robust ECG classification. The results of the performance can be considered sufficient evidence to be able to extend and deploy our approach in the future clinical application.

Chapter 5
Engineering Standards and Design Challenges

This chapter explains the standards followed while developing the ECG Classification System and the challenges faced during its design. It describes how the system meets software, hardware, and communication standards, its effects on society and the environment, and its long-term sustainability. It also outlines the major technical problems and how they were solved.

5.1 Compliance with the Standards

5.1.1 Software Standards

The software development for this project followed widely accepted industry and academic standards to ensure reproducibility, compatibility, and maintainability.

1. **Programming Language and libraries:** Python 3.10 was used because of its popularity in machine learning research and strong library support for deep learning and graph processing. Libraries included PyTorch, PyTorch Geometric, Scikit-learn, NeuroKit2, and Gradio for model training, graph operations, signal processing, and building a user interface.
2. **Data Formats:** ECG datasets were stored in structured folders, with preprocessed features saved in CSV format and results accuracy curves, confusion matrices exported as PNG images for compatibility across platforms. Model checkpoints were stored in.pth format for reproducibility.

5.1.2 Hardware Standards

Hardware Standards

1. **Development Platform:** The project was developed and tested using Google Colab with GPU acceleration for large-scale training, as well as a local system with AMD Ryzen 5 5600X CPU, 16 GB RAM, and NVIDIA GeForce GTX 1660 Super GPU.
2. **Hardware Utilization:** GPU acceleration via CUDA was used to speed up feature Extraction, model training and graph computations, especially for GNN models like GraphSAGE, GCN, and GAT.
3. **Storage:** SSD storage was used to ensure fast data loading and reduce bottlenecks during preprocessing and training.

5.2 Impact on Society, Environment and Sustainability

5.2.1 Impact on Life

The proposed system enables fast and accurate diagnosis of common heart conditions (Normal, MI, STTC, CD, HYP) using ECG signals. This can support doctors in making quicker decisions, improve patient outcomes, and potentially save lives by detecting issues early, especially in remote or under-resourced areas.

5.2.2 Impact on Society & Environment

Society: The system reduces dependence on highly specialized cardiologists for initial diagnosis, making cardiac screening more accessible to small clinics and rural healthcare facilities.

Environment: By enabling early and accurate detection, the system can reduce the need for unnecessary hospital visits and procedures, indirectly lowering energy usage and resource consumption in healthcare.

Environment: By enabling early and accurate detection, the system can reduce the need for unnecessary hospital visits and procedures, indirectly lowering energy usage and resource consumption in healthcare.

5.2.3 Ethical Aspects

The dataset used PTB-XL is publicly available and anonymized, ensuring compliance with patient privacy regulations. The system does not store personally identifiable patient information. Ethical AI principles were followed, including transparency with explainable AI techniques feature importance visualization to make model decisions understandable to clinicians.

5.2.4 Sustainability Plan

The system can be deployed as a cloud-based service or integrated into existing hospital systems, requiring minimal local computing resources. This design supports long-term use and scalability without large infrastructure costs. The continual learning framework allows the system to adapt over time as new types of heart conditions are introduced, ensuring ongoing relevance in clinical settings.

5.3 Project Management and Financial Analysis

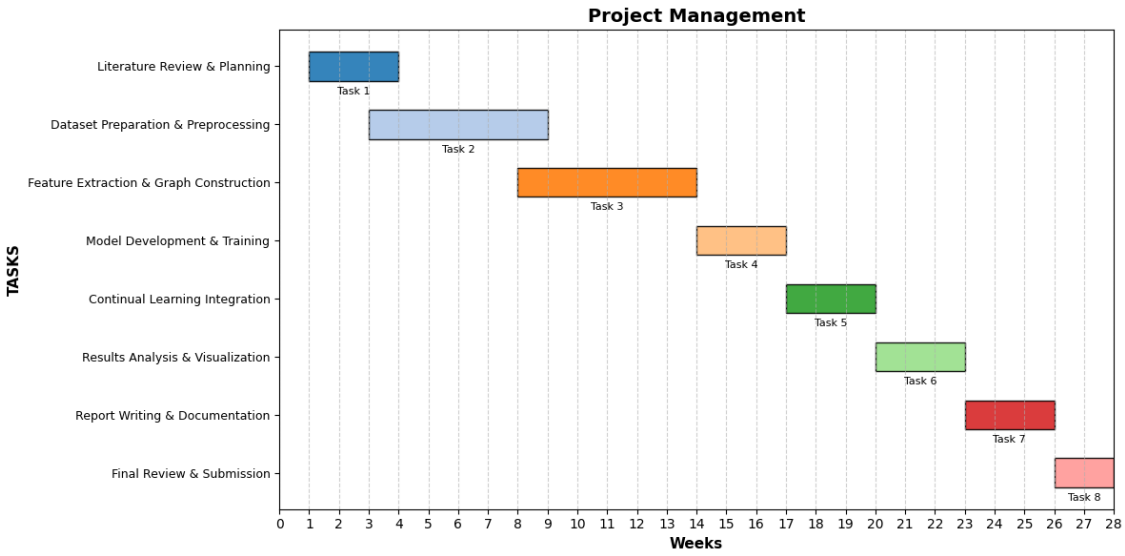


Figure 5.1: Project Management Flowchart

Cost Analysis:

Table 5.1: Cost analysis table.

Category	Resource / Tool	Cost (BDT)	Details
Hardware	AMD Ryzen 5 5600X	11,000	One-time purchase
Hardware	GTX 1660 Super GPU	22,000	One-time purchase
Hardware	16GB RAM + other parts	14,000	One-time purchase
Software	Open-source Libraries & Tools	0	Free, no licensing fees
Cloud Service	Google Colab Pro	6,600	1,100 × 06 months
Others	System Power Consumption	2160	Approx. 360 × 06 months
Total One-time Hardware Cost		47,000	CPU + GPU + RAM + others
Total Yearly Operational Cost		8,760	Colab + electricity
Grand Total (First Year)		55,760	

5.4 Complex Engineering Problem

5.4.1 Complex Problem Solving

In this section, provide mapping with problem solving categories

Table 5.2: Mapping with Complex Engineering Problem.

EP1 Dept of knowledge	EP2 Range Of Conflicting Requirements	EP3 Depth of Analysis	EP4 Familiarity of Issues	EP5 Extent of Applicable Codes	EP6 Extent Of Stake-holder Involvement	EP7 Interdependence
✓		✓	✓			✓

EP1: Required advanced understanding of ECG signal processing, machine learning, GNN architectures, and continual learning techniques.

EP3: This was a complex task because we had to deeply understand ECG data, apply GNNs models, and make sure everything worked well together.

EP4: Some challenges, like working with noisy ECG signals, Preprocessing were familiar to engineers.

EP7: Different task of the system such as preprocessing, graph construction, model training and continual learning were interconnected. A small change in one task can affect the performance of the others.

Mapping with Knowledge Profile

Table 5.3: Mapping with knowledge Profile.

K1 Natural Science	K2 Mathematics	K3 Engineering Fundamentals	K4 Specialist Knowledge	K5 Engineering Design	K6 Engineering Practice	K7 Comprehension	K8 Research Literature
	✓	✓		✓	✓		✓

K2: Strong use of statistical function, linear algebra, probability and graph theory in feature extraction, patient similarity graph construction and model Evolution.

K3: Basic knowledge about how to process ECG signal and extract useful features. This project applied signal processing fundamentals and machine learning for classification.

K5: This system was designed step by step Designed the system to handle complex data, making sure it's efficient and scalable.

K6: Practical tools and platforms were used: Python, Pytorch, Google colab, CUDA for GPU acceleration. Applied the knowledge in real world settings, including programming the model and testing them.

K8: Reviewed research to find the best methods for classifying ECG signals using machine learning and GNNs. This project included a detailed literature review of GNNs, continual learning, deep learning models and ECG preprocessing methods.

5.4.2 Engineering Activities

Mapping with Complex Engineering Activities

Table 5.4: Mapping with Complex Engineering Activities.

EA1 Range of Resource	**EA2** Level of Interaction	**EA3** Innovation	**EA4** Consequences for Society and Environment	**EA5** Familiarity
✓	✓	✓	✓	✓

EA1: We needed lots of different resources, including computing power for training models and data storage for handling ECG datasets. The project required a wide range of resources including dataset, High performance computing (Google colab, AMD Ryzen 5 system with GPU) Software (Python, Pytorch, CUDA) and advance algorithm.

EA2: The system has many parts that needed to work together, like data preprocessing, model training, and testing.

EA3: The project is innovative because we combined Graph Neural Networks (GNNs) with continual learning, allowing the system to adapt and improve over time as new data is added.

EA4: This system can help doctors diagnose heart conditions early, saving lives. It's also environmentally friendly because it uses cloud computing instead of building physical infrastructure.
EA5: Some activities were familiar, like working with ECG data and using GNN. However, the continual learning aspect was new, which made it challenging and innovative.

5.5 Summary

In this section, we explained why our ECG classification system is a complex engineering project. Building it required knowledge of ECG signal processing, machine learning, graph neural networks, and continual learning. We also had to connect several parts of the system data cleaning, feature extraction, graph building, and model training, so they work together smoothly. By mapping the work to engineering problem categories, knowledge areas, and activities, we showed that this project had to deal with different requirements, follow technical and ethical rules, and use new ideas. All of this means the system needed skills from different fields and a careful, step-by-step engineering process to make it work well.

Chapter 6
Conclusion

This chapter provides all the important points from our research. It provides a short summary of what we did, describe the limitations we faced and suggests possible future directions for improvement.

6.1 Summary

In this project, we developed a lifelong graph-based ECG classification system that combines Graph Neural Networks (GNNs) with continual learning. The goal was to classify common heart conditions Normal (NORM), Myocardial Infarction (MI), ST/T Change (STTC), Conduction Disturbance (CD), and Hypertrophy (HYP), while allowing the system to learn new classes over time without forgetting previously learned information. We worked with the PTB-XL data, applied a preprocessing of the ECG signal, selected useful features, and a graph of the similarity of patients was created based on KNN. On GNNs, three models (GCN, GAT and GraphSAGE) were tested, with GraphSAGE providing the best accuracy of 97.29%. Continual learning strategy was vocal because the system was able to maintain its stability in terms of adding new classes without forgetting the other previously mastered classes. Overall, our methodology proved that a combination of signal preprocessing, feature extraction, graph construction, GNNs, and continual learning can form an efficient and versatile ECG classification manifold that can be both simultaneously high performing and continuously developing.

6.2 Limitation

1. The model was evaluated only on one publicly accessible dataset PTB_XL. The results may be different on other dataset. Testing on real world hospital data with grater noise is still necessary.
2. The graph construction was based on the simple feature similarity KNN.

6.3 Future Work

There are several ways this work could be improved in future:

1.Test on multiple dataset from different hospitals to improve generalization.

2. Explore more advanced graph-building methods that combine clinical history and demographic data with ECG features.

3. Add explainable AI features to make model decisions more transparent to healthcare provider.

References

[1] H. Tran, " Combination of Hedge Algebra and Type-2 Fuzzy System for Electrocardiogram Signal Recognition and Classification," *JST: Smart Systems and Devices,* vol. 33, no. 1, pp. 25-33, 2023.

[2] S. Han, "Arrhythmia Classification Using Graph Neural Networks Based on Correlation Matrix," *IEEE International Conference on Bioinformatics and Biomedicine (BIBM),* no. Dec. 2024, p. 6400–6402., 2024.

[3] Z. L. L. H. a. S. P. X. Zhao, ""ECGNN: Enhancing Abnormal Recognition in 12-lead ECG with Graph Neural Network," *IEEE International Conference on Bioinformatics and Biomedicine (BIBM),,* p. 1411–1416, 2022.

[4] K. P. a. D. L. S. Śmigiel, "ECG Signal Classification Using Deep Learning Techniques Based on the PTB-XL Dataset," *Entropy,* vol. 23, no. 9, p. 1121, sep.2021.

[5] C. Z. Y. L. W. .. L. a. S. Zhou, "Graph neural networks: A review of methods and applications," *AI Open,* vol. 1, pp. 57-81, 2020.

[6] D. C. T. S. M. C. B. N. B. B. R. a. B. B.-J. S. Sethi, "ProtoECGNet: Case-Based Interpretable Deep Learning for Multi-Label ECG Classification with Contrastive Learning," *arXiv preprint,,* vol. arXiv:2504.08713, 2025.

[7] A. R. C. Y. a. H. L. J. Peng, ""EGCNet: A Hierarchical Graph Convolutional Neural Network for Improved Classification of Electrocardiograms," *EURASIP Journal on Advances in Signal Processing,* vol. 1, no. 2024, p. 93, 2024.

[8] X. R. T. Z. Y. C. L. J. D. L. X. W. L. Z. a. W. K. C. Y. Xiu, "Time Labeled Visibility Graph for Privacy-Preserved Physiological Time Series Classification," *7th International Conference on Cloud Computing and Big Data Analytics (ICCCBDA),* , pp. 280-284, 2022.

[9] Y. Q. D. G. a. S. S. C. Wang, "Lifelong Graph Learning," *in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition,* p. 13719–13728., 2022.

[10] X. W. Z. C. M. W. J. L. a. C. L. H. Gao, "ECG-CL: A Comprehensive Electrocardiogram Interpretation Method Based on Continual Learning," *IEEE Journal of Biomedical and Health Informatics,,* vol. 27, no. 11, pp. 5225-5236, 2023.

[11] W. R. a. W. Q. Q. Shen, "Graph Relation Aware Continual Learning," *arXiv preprint,,* 2023.

[12] X. D. Y. H. Y. L. X. F. a. W. Z. Y. Zhou, "Masked Transformer for Electrocardiogram Classification," *arXiv preprint,* 2023.

[13] Z. C. a. B. Liu, "Lifelong Machine Learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning,* vol. 12, no. 3, pp. 1-207, 2018.

[14] P. W. e. al., "PTB-XL, a large publicly available electrocardiography dataset," *Scientific Data,* vol. 7, 2020.

[15] B. W. a. K. E. B. M. Blanco-Velasco, "ECG signal denoising and baseline wander correction based on the empirical mode decomposition," *Computers in Biology and Medicine,* vol. 38, no. 1, pp. 1-13, 2008.

[16] P. a. N. Kumaravel, "Wavelet-based adaptive ECG signal denoising using primary components," *Digital Signal Processing,* vol. 18, no. 1, pp. 49-55, 2008.

[17] C. Z. a. C. T. C. Li, "Detection of ECG characteristic points using wavelet transforms," *IEEE Transactions on Biomedical Engineering,* vol. 42, no. 1, pp. 21-28, 1995.

[18] Z. Zhang, "Heartbeat classification using morphological and dynamic features of ECG signals," *EEE Trans. Instrum. Meas.,* vol. 63, no. 2, pp. 293-303, 2014.

[19] C. B. a. B. Pompe, "Permutation entropy: A natural complexity measure for time series," *Phys. Rev. Lett,* vol. 88, no. 17, p. 174102, 2002.

[20] A. L. G. e. al, "PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals," *Circulation,* vol. 101, no. 23, pp. 215-220, 2000.

[21] S. H. T. J. X. a. R. M. Zhang, "Graph convolutional networks: a comprehensive review," *Computational Social Networks,* vol. 6, no. 1, pp. 1-23, 2019.

[22] P. C. G. C. A. R. A. L. P. a. B. Y. Velickovic, "GRAPH ATTENTION NETWORKS," *arXiv,* pp. 10-48550, 2018.

[23] R. Y. a. J. L. W. Hamilton, "Inductive representation learning on large graphs," *NeurIPS,* 2017.

[24] L. V. d. M. a. G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research,* vol. 9, pp. 2579-2605, 2008.

*% detected as AI

AI detection includes the possibility of false positives. Although some text in this submission is likely AI generated, scores below the 20% threshold are not surfaced because they have a higher likelihood of false positives.

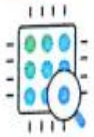
Caution: Review required.

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

Disclaimer
Our AI writing assessment is designed to help educators identify text that might be prepared by a generative AI tool. Our AI writing assessment may not always be accurate (i.e., our AI models may produce either false positive results or false negative results), so it should not be used as the sole basis for adverse actions against a student. It takes further scrutiny and human judgment in conjunction with an organization's application of its specific academic policies to determine whether any academic misconduct has occurred.

Frequently Asked Questions

How should I interpret Turnitin's AI writing percentage and false positives?
The percentage shown in the AI writing report is the amount of qualifying text within the submission that Turnitin's AI writing detection model determines was either likely AI-generated text from a large-language model or likely AI-generated text that was likely revised using an AI paraphrase tool or word spinner.



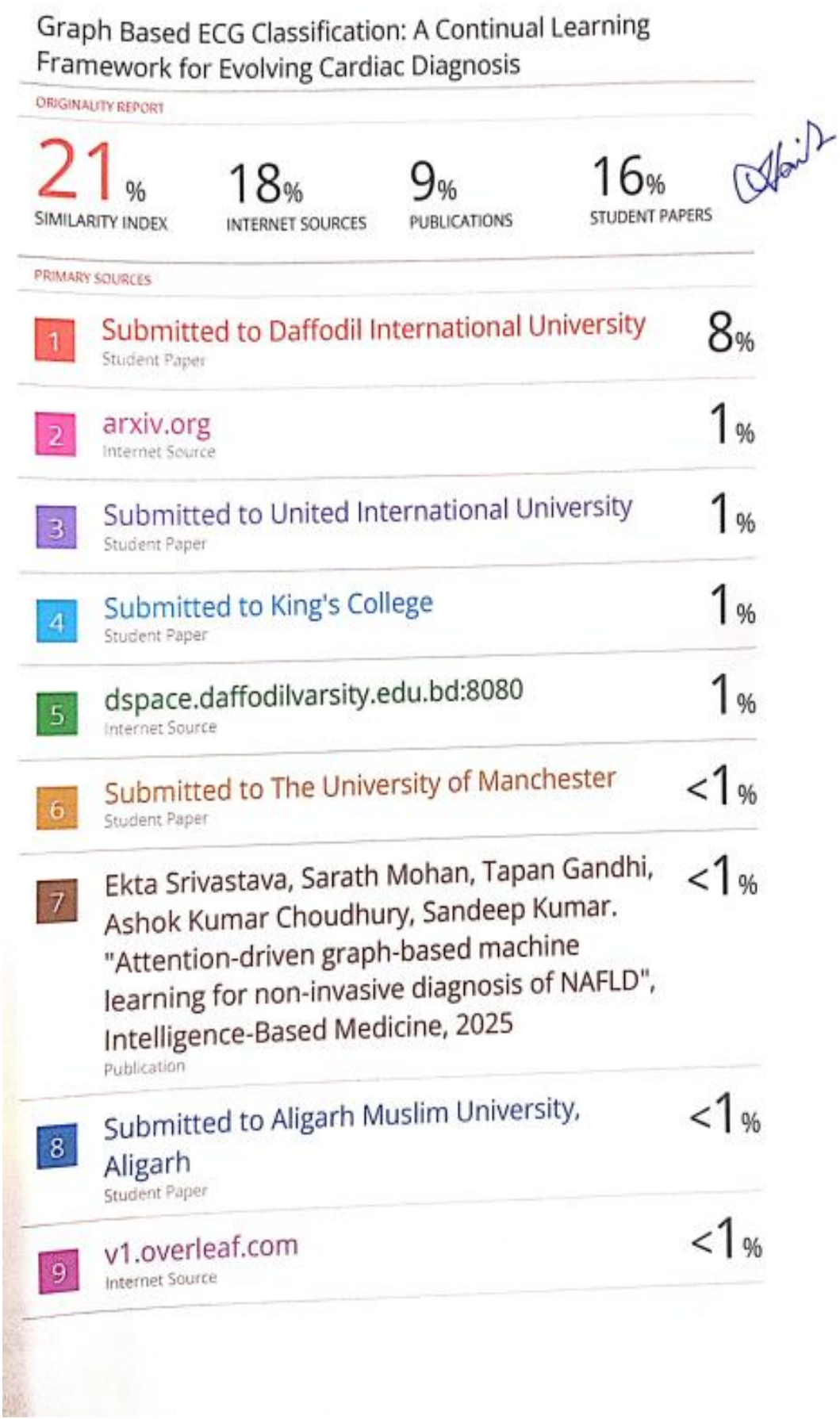
False positives (incorrectly flagging human-written text as AI-generated) are a possibility in AI models.

AI detection scores under 20%, which we do not surface in new reports, have a higher likelihood of false positives. To reduce the likelihood of misinterpretation, no score or highlights are attributed and are indicated with an asterisk in the report (*%).

The AI writing percentage should not be the sole basis to determine whether misconduct has occurred. The reviewer/instructor should use the percentage as a means to start a formative conversation with their student and/or use it to examine the submitted assignment in accordance with their school's policies.

What does 'qualifying text' mean?
Our model only processes qualifying text in the form of long-form writing. Long-form writing means individual sentences contained in paragraphs that make up a longer piece of written work, such as an essay, a dissertation, or an article, etc. Qualifying text that has been determined to be likely AI-generated will be highlighted in cyan in the submission, and likely AI-generated and then likely AI-paraphrased will be highlighted purple.

Non-qualifying text, such as bullet points, annotated bibliographies, etc., will not be processed and can create disparity between the submission highlights and the percentage shown.



Graph Based ECG Classification: A Continual Learning Framework for Evolving Cardiac Diagnosis

ORIGINALITY REPORT

21%	18%	9%	16%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Daffodil International University Student Paper	8%
2	arxiv.org Internet Source	1%
3	Submitted to United International University Student Paper	1%
4	Submitted to King's College Student Paper	1%
5	dspace.daffodilvarsity.edu.bd:8080 Internet Source	1%
6	Submitted to The University of Manchester Student Paper	<1%
7	Ekta Srivastava, Sarath Mohan, Tapan Gandhi, Ashok Kumar Choudhury, Sandeep Kumar. "Attention-driven graph-based machine learning for non-invasive diagnosis of NAFLD", Intelligence-Based Medicine, 2025 Publication	<1%
8	Submitted to Aligarh Muslim University, Aligarh Student Paper	<1%
9	v1.overleaf.com Internet Source	<1%