AI Driven personalized sleep breathing pattern classification for sleep apnea detection

Submitted in partial fulfilment of the requirements for the degree of

Bachelor of Technology
in
Computer Science & Engineering

By

Sumit Kumar Mandal
20BCE2885

Naman Agrahari
20BCE2886

Aviral Sharma
20BCE2918

Under the guidance of
Dr. Saravanaguru Ra.K

School of Computer Science & Engineering
VIT, Vellore

May, 2024

DECLARATION

I hereby declare that the thesis entitled "AI Driven Personalized Sleep Breathing Pattern Classification for Sleep Apnea Detection" submitted by me, for the award of the degree of Bachelor of Technology in Computer Science & Engineering to VIT is a record of bonafide work carried out by me under the supervision of Dr. Saravanaguru Ra.K.

I further declare that the work reported in this thesis has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place : Vellore

Date :

Signature of the Candidate

CERTIFICATE

This is to certify that the thesis entitled "AI Driven Personalized Sleep Breathing Pattern Classification for Sleep Apnea Detection" submitted by **Sumit Kumar Mandal (20BCE2885), Naman Agrahari (20BCE2886), Aviral Sharma (20BCE2918)**, **School of Computer Science & Engineering**, VIT, for the award of the degree of *Bachelor of Technology in Computer Science & Engineering*, is a record of bonafide work carried out by him / her under my supervision during the period, 04. 01. 2024 to 09.05.2024, as per the VIT code of academic and research ethics.

The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university. The thesis fulfills the requirements and regulations of the University and in my opinion meets the necessary standards for submission.

Place : Vellore

Date :

Signature of the Guide

Internal Examiner

External Examiner

ACKNOWLEDGEMENTS

I would like to express my gratitude to the Vellore Institute of Technology that has nurtured our growth and provided us with the platform to undertake our final year project, "AI Driven personalized sleep pattern classification for sleep apnea detection", as part of our capstone experience.

Additionally, I extend my sincere appreciation for the unwavering support and enriching experience provided by **Dr. Saravanaguru RA. K.** His invaluable guidance and mentorship have played a pivotal role in shaping the success of our project. Without his insightful direction and meticulous guidance, the completion of this project would not have been possible. At every phase of the project, his supervision and expertise have been instrumental in ensuring the thoroughness and excellence of this report. I also wish to thank the esteemed members of the review panel for their insightful feedback and thought-provoking questions, which have greatly enriched our project's development.

Special recognition goes to our team members, Sumit Kumar Mandal, Naman Agrahari and Aviral Sharma for their unwavering dedication and collaborative spirit throughout the project journey.

The whole experience in this project will be a big milestone in my professional development. I will apply the skills and knowledge gained in the best way possible and will continue to improve.

Team members,

Sumit Kumar Mandal
Naman Agrahari
Aviral Sharma

EXECUTIVE SUMMARY

Obstructive sleep apnea (OSA) poses a significant global health challenge, characterized by sleep disruptions and increased cardiovascular risks. Traditional diagnostic methods like polysomnography (PSG) encounter limitations in accessibility and cost, necessitating innovative alternatives. This project introduces a groundbreaking approach to OSA detection through personalized artificial intelligence (AI) methodologies. By employing advanced deep learning algorithms, we meticulously analyze individual breathing patterns extracted from PSG data, prioritizing personalization over generic parameters. The resulting AI models exhibit a nuanced understanding, enabling the identification of subtle OSA-related patterns within unique breathing characteristics. Our personalized AI-driven approach offers heightened diagnostic accuracy, especially beneficial in borderline cases. Furthermore, integration into home-based sleep monitoring devices promises widespread access and cost-effectiveness. This transformative approach not only aids in early OSA detection but also enables tailored interventions, contributing to improved global health outcomes and revolutionizing sleep medicine practices. Through continuous research and validation across diverse populations, this innovative approach holds the potential to significantly impact sleep disorder diagnosis worldwide. With ongoing refinement and validation, our personalized AI-driven solution stands poised to revolutionize the landscape of sleep disorder management, ensuring better health outcomes for individuals globally. In conclusion, our project marks a significant advancement in reshaping the landscape of sleep disorder diagnosis and management, promising more effective, accessible, and personalized solutions for individuals worldwide.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

AHI - Apnea-Hypopnea Index

AI - Artificial Intelligence

AUC - Area Under the Curve

CNN - Convolutional Neural Network

CNN-CFF - Convolutional Neural Network - Crowd Flow Forecasting

ECG - Electrocardiography

EEG - Electroencephalography

FDA - Food and Drug Administration

FN - False Negative

FP - False Positive

GDPR - General Data Protection Regulation

HIPAA - Health Insurance Portability and Accountability Act

IMU - Inertial Measurement Unit

IRB - Institutional Review Board

KNN - k-Nearest Neighbors

LR - Logistic Regression

MHR - Maximum Heart Rate

MLP - Multi-Layer Perceptron

MRI - Magnetic Resonance Imaging

OSA - Obstructive Sleep Apnea

PSG - Polysomnography

PNN50 - Percentage of NN50 divided by total number of NN intervals

RMSSD - Root Mean Square of Successive Differences

SA - Sleep Apnea

SDNN - Standard Deviation of NN intervals

SDSD - Standard Deviation of Successive Differences

SVM - Support Vector Machine

TN - True Negative

TP - True Positive

1. INTRODUCTION

1.1 Objectives

The goal of the sleep apnea detection is to categorize sleep breathing patterns with increased accuracy and customisation. Improving sensitivity and specificity will allow for more accurate case identification of sleep apnea. The goal of the project is to create an adaptive model that can dynamically adapt to each person's unique sleep patterns and enhance the individualized detection of minute alterations that may signal the onset of sleep apnea. Furthermore, the goal is to smoothly include the created AI model into the healthcare framework's current sleep monitoring systems, encouraging cooperation with healthcare platforms to ensure broad adoption. Prioritizing ethical compliance and privacy assurance means that personal health data is used ethically and that strong privacy safeguards are put in place in accordance with healthcare laws. The project's final goal is to optimize the model.

1.2 Motivation

The aim of this project is to explore the clinical relevance and potential implications of utilizing electrocardiography (ECG) signals in polysomnography (PSG) for the detection and management of sleep apnea. This includes investigating how ECG signals can provide valuable insights into the cardiovascular aspects of sleep apnea, aiding in diagnosis, risk stratification, and treatment planning. Additionally, the project aims to evaluate the effectiveness of integrating machine learning algorithms with ECG signal analysis to enhance diagnostic accuracy and efficiency, ultimately improving patient outcomes.

1.3 Background

The scope of this project encompasses a comprehensive review and analysis of existing literature on the utilization of ECG signals in PSG for sleep apnea detection and management. This includes examining the role of ECG-derived biomarkers in identifying sleep-disordered breathing events, assessing the severity of sleep apnea, and understanding its impact on cardiovascular health. Furthermore, the project will explore various machine learning techniques employed for analysing ECG signals in the context of sleep apnea, evaluating their effectiveness in improving diagnostic accuracy and facilitating personalized treatment strategies.

2. PROJECT DESCRIPTION AND GOALS

2.1 Survey on Existing System

Several studies have addressed the challenge of sleep apnea detection using machine learning methods. Anu Shilvya J and Dr. P. Subha Hency Jose (2023) employed artificial neural networks (ANNs), convolutional neural networks (CNNs), and long short-term memory (LSTM) models to analyze electrocardiogram (ECG) signals, achieving refined results through diverse classification strategies and feature engineering [12]. Similarly, Yongfeng Huang, Kuiyou Chen1, and Zhiming Zhang (2023) proposed a domain-aware spatial-temporal graph convolutional network (DAST-GCN) to capture temporal characteristics of ballisto cardiogram (BCG) signals, with a focus on adapting the model for mobile devices [10]. Shuaicong Hu et al. (2023) utilized a CNN-based auto-encoder (AE) and semi-supervised algorithm for ECG analysis, emphasizing the importance of fine-tuning data and personalized severity grading [8]. Other approaches include Amit Bhongade et al. (2023) utilizing polysomnography data for automatic IMU axis selection, and Xinlei Yan et al. (2022) refining algorithms for early detection of obstructive sleep apnea (OSA) using nasal airflow signals and ECG [14, 13]. These studies collectively contribute to the ongoing efforts to enhance sleep apnea diagnosis through the integration of advanced machine learning techniques and biomedical signal analysis.

Ref No	Year	Authors	Techniques used	Features	Directions for future investigation
[12]	2023	Anu Shilvya J, Dr. P. Subha Hency Jose	ANNs, CNNs, LSTM	RMSSD, SDNN, SDSD, NN50, PNN50, MRI, MHR using ECG signals	Refining using diverse classification strategies, feature engineering and larger datasets.
[10]	2023	Yongfeng Huang , Kuiyou Chen1 , Zhiming Zhang	Domain-aware Spatial-Temporal Graph Convolutional Network (DAST-GCN)	Temporal characteristics of individual BCG signals and their inter-sensor relationships learned via the graph structure.	Adapting the DAST-GCN model to run on devices with limited processing power, like mobile phones.
[8]	2023	Shuaicong Hu, Ya'nan Wang, Jian Liu, Cuiwei Yang	CNN- based auto-encoder (AE) and semi-supervised algorithm on the same database and cross-database.	ECG used to R-peak to R-peak (RR) interval irregularity, R-peak fluctuations, and reconstruction error using ECG	Fine-tuning data, larger training datasets, personalized severity gradin
[13]	2023	Amit Bhongade, Rohit Gupta, Tapan K. Gandhi, and Prathosh AP	Polysomnography, respiratory effort monitoring, oximetry, apnea-hypopnea inde	Peak frequency of the filtered IMU signal	Automatic IMU axis selection for optimal performance.
[14]	2022	Xinlei Yan, Lin Wang, Jiang Zhu, Shaochang Wang, Qiang Zhang, Yi Xin	Butterworth low pass filter, R-wave amplitude , Decision tree, Random Forest and XGBoost.	Nasal airflow signals for respiratory parameters and ECG Signals.	Refining the algorithm for earlier OSA detection and risk stratification.

[5]	2019	V. Vimala, K. Ramar, M. Ettappan	KNN, SVM, ANN	Decomposed EEG frequency band energy, entropy, and variance	Building a web platform for sleep apnea diagnosis using the best classifier and new EEG features.
[4]	2015	Avcı, Cafer and Akbaş, Ahmet	Random Forest, Ada Boost, Random Subspace	Daubechies wavelet family filter length 3 and CfsSubsetEval.	Developing a sleep apnea detection method based solely on ECG signals, using a larger dataset
[21]	2023	Fernando Vaquerizo-Villar,Gonzalo C. Gutiérrez-Tobal, Eva Calvo, Daniel Alvarez, Leila Kheirandish-Gozal, Félix del Campo, David Gozal, and Roberto Hornero.	CNN, CNN-Inception CNN-RNN Gradient-weighted Class Activation Mapping(Grad-CAM) for explainable AI	Single-channel EEG recordings EEG patterns associated with each sleep stage Identification of epochs likely to be misclassified	Further validation and applicability testing across populations of all ages Integration into EEG signal acquisition and processing software like Medusa© Facilitating timely and objective diagnosis of OSA
[22]	2023	Ángel Serrano Alarcón, Natividad Martínez Madrid, Ralf Seepold, and Juan Antonio Ortega	1D- CNN Grad-Cam for model interpretability	Analysis and detection of OSA events. Estimations of AHI using deep learning models Utilization of polysomnography data from the NSRR repository for training.	Develop and refine deep learning models for portable monitors (PMs) to detect OSA events. Balance model accuracy and interpretability for practical clinical use. Explore techniques to enhance model performance and interpretability.
[23]	2023	Hongjian Zhang and Katsuhiko Ogasawara	Transferred learning, Grad-Cam	Utilization of ResNet for text processing, achieving higher accuracy in classification. Grad-CAM visualization providing intuitive insight into words focused on during predictions, enhancing model interpretability.	Development of quantitative measures to assess the explainability derived from Grad-CAM Comparison of performance with state-of-the-art models. Further exploration of interpretability methods to enhance understanding and trust in AI-driven medical text processing.
[24]	2023	Micheal Dutt, SurenderReddhu, MortenGoodwin, ChristianW. Omlin	Unified CNN-CRF, Modified Gradient-Weighted Class Activation Mapping (Grad-Cam)	Enhanced Accuracy, Explainable AI	Integration Of Multimodal Data, Longitudinal Studies
[25]	2023	Serrano Alarcon, A., Martínez Madrid, N.,	1D- CNN, Grad-Cam	Optimized Hyperparameters, High Accuracy	Data Augmentation Techniques, Longitudinal studies, Integration with wearable sensors.

		Seepold, R., & Ortega, J			
[26]	2023	Jeong, H. G., Kim, T., Hong, J. E., Kim, H. J., Yun, S. Y., Kim, S., Yoo, J., Lee, S. H., Thomas, R. J., & Yun, C. H.	CNN, Grad-Cam	High Auroc Performance, Clinically Relevant Correlations	Validation with larger Datasets, Integration with clinical practice, Refinement of ground truth definition

Table 1: Literature Survey

2.2 Gaps Identified

- **Refinement of Classification Strategies and Feature Engineering:** While various machine learning techniques have been applied for sleep apnea detection using ECG signals, there is a need for refining classification strategies, feature engineering, and utilizing larger datasets to improve accuracy and customization.

- **Integration of Machine Learning Algorithms for Personalized Sleep Apnea Detection in Web Platforms:** While previous study focuses on developing a web platform for sleep apnea diagnosis but lacks focus on the integration of machine learning algorithms for personalized detection, this project aims to bridge this gap by integrating machine learning algorithms with ECG signal analysis to enhance diagnostic accuracy and efficiency.

- **Utilization of Machine Learning for Improved Accuracy in Sleep Apnea Detection Using ECG Signals:** Past studies lack focus on the integration of machine learning algorithms for improved accuracy, but this project aims to address this gap by leveraging machine learning techniques to analyse ECG signals in the context of sleep apnea detection.

2.3 Problem Statement

Traditional methods for Sleep Apnea identification, like polysomnography (PSG) in sleep labs, might lack precision and personalization. PSG monitors physiological parameters during sleep, offering a single-night snapshot in a controlled setting. This approach may not capture an individual's complete sleep patterns, leading to potential inaccuracies in detecting subtle changes indicating Sleep Apnea. Additionally, manually analysing PSG sleep data can be time-consuming and may not effectively consider individual variations. This lack of personalized

analysis could lead to delayed or inaccurate Sleep Apnea detection, especially in cases with gradual onset or subtle changes over time.

In contrast, AI-driven approaches can address these limitations by continuously adapting and learning from individual sleep patterns. This offers a more personalized and precise method for identifying Sleep Apnea. The integration of AI technologies aims to enhance sensitivity and specificity in detecting sleep breathing patterns, leading to more accurate and timely diagnoses compared to traditional methods.

3. TECHNICAL SPECIFICATION

3.1 REQUIREMENTS

3.1.1 Functional Requirements:

1. Data acquisition and processing:

- **PhysioNet Apnea-ECG dataset**

The dataset comprises 70 single-lead ECG recordings, lasting between 401 to 587 minutes and sampled at 100 Hz, are our main data. The system detects the main peaks in the heartbeats (R-peaks) from the ECG signals. [https://physionet.org/content/ucddb/1.0.0/] [27]

2. Feature extraction:

The system must extract relevant features from pre-processed ECG signals, including RR intervals (time between successive R waves on the ECG waveform) and amplitudes.

3. Normalisation:

The system should use min-max normalization to normalize the extracted features to ensure consistency across features.

4. Machine learning Models:

The system should use the extracted features to train modified Le-Net 5 model for sleep apnea detection.

5. Evaluation:

The system should evaluate the performance of machine learning models using metrics such as specificity, sensitivity, accuracy.

It should compare the performance of the proposed method with existing methods using benchmark data sets.

3.1.2 Non-functional requirements:

1. Performance:

To meet clinical criteria, the system must have high accuracy, sensitivity, and specificity in detecting sleep apnea. ECG signals must be optimally processed to obtain real-time or near-real-time results.

2. Progress:

The system must be robust to changes in input signal, noise, and artifacts commonly encountered in real-world situations. It must show consistent performance across data types and conditions.

3. Resources Used:

The system should have a user-friendly interface for easy communication with health care providers. It provides clear and interpretable results to assist in clinical decision making.

4. Reliability:

The system must be reliable and trustworthy, providing consistently accurate sleep apnea detection results. It should handle errors gracefully and have mechanisms for error detection and recovery.

5. Scalability

The system must be flexible to handle large amounts of ECG data efficiently, especially in situations with large numbers of patients.

6. Documentation and Support:

The system must have complete documentation including installation, operation, and maintenance. It provides technical support and troubleshooting support to users as needed.

3.2 FEASIBILITY STUDY

3.2.1 Technical Feasibility:

The technical efficiency of the proposed methodology for sleep apnea detection can be assessed based on available resources, technology and from the data provided:

- **Technical Features:** The project is based on a modified LeNet-5 framework for sleep apnea detection. This technology is well established in deep learning and image analysis, with many features, libraries and frameworks available for use.
- **Data Availability:** The PhysioNet Apnea-ECG dataset with codes and accompanying text is available for analysis. This dataset provides the data needed to train and test the sleep apnea detection model and to evaluate the performance.
- **Software and tools:** The role requires access to deep learning libraries (e.g., TensorFlow, Py Torch) for implementation of modelling, as well as tools for data preprocessing, training, and analysis. These software products are ubiquitous and compatible with proposed methods.
- **Hardware requirements:** The technical requirements for training and testing the SA detection model may vary depending on the size of the data set and the complexity of the neural network architecture but modern hardware resources such as GPU, cloud computing platform these tasks can be made easier to perform efficiently.

3.2.2 Economic Feasibility:

The economic viability of a project involves analysing the costs associated with its development, operation, and maintenance.

Open-source tools: The benefits of open-source tools, libraries, and data sets, such as PhysioNet and the Deep Learning Framework, can reduce the financial burden associated with software license fees and data acquisition costs.

Scalability: The scalability function is economically important, as it determines the likelihood of wide acceptance and cost-effective use in clinical settings Theoretical SA identification method scale well with increasing data volumes and computing requirements over the course of the project can remain economically viable.

3.2.3 Social Feasibility

A potential social function includes consideration of its impact on health practices, patient outcomes, and quality of life.

Clinical Context: The work addresses the important health issue of diagnosed sleep apnea, which affects a significant portion of the population and can have serious health consequences if left untreated. By providing an accurate and interpretable method for diagnosing sleep apnea, the project has the potential to improve patient outcomes and enhance sleep disorder management.

Accessibility: Using single-lead ECG signals and wearable devices to diagnose sleep apnea extends access to healthcare services beyond traditional clinical settings. This approach enables remote monitoring and sleep detection early problems, especially in underserved or remote communities where access to primary health care may be limited.

Ethical considerations: The program emphasizes data privacy and compliance, ensuring patient confidentiality is maintained throughout data collection, analysis, and storage Through adherence to ethical guidelines and standards on the project supports trust and transparency in health research and practice.

3.3 SYSTEM SPECIFICATION

3.3.1 Hardware Specification

The hardware specification for the proposed sleep apnea detection system describes the necessary computing hardware and equipment required for development, testing and operation Based on the project requirements:

Computer equipment:

High-performance computing hardware including CPUs and GPUs are required for training deep learning models, such as the modified LeNet-5 framework for sleep apnea detection.

The availability of multi-core processors and GPUs with parallel processing capabilities can speed up model training and inference tasks, improving overall system performance

The memory (RAM) capacity must be sufficient to accommodate large datasets and model parameters during the training and testing phase.

Storage:

Adequate storage is required to store raw data, pre-processed data, reduced sample loads, and intermediate results that occur during the development process

Solid-state drives (SSDs) are preferred for faster data transfer and recovery, especially when dealing with large amounts of data.

Remote devices:

Standard input/output devices, such as keyboards, mouse, and displays, are essential for user interaction and program management during development and testing.

3.3.2 Software Specification

The software specification describes the software components, tools, and configuration required to implement the sleep apnea detection system.

Deep learning Framework:

TensorFlow, Py Torch, Keras or similar deep learning libraries are needed to generate neural network architecture, training models, and inference functions.

This framework provides an API for building convolutional neural networks (CNNs), defining loss functions, optimizing model parameters, and implementing trained models

Data Processing Tools:

Python libraries such as NumPy, Panda, and SciPy facilitate data preprocessing, feature extraction, and analysis from raw ECG signal and image data.

Signal processing algorithms including filtering, segmentation, and feature extraction techniques applied using these tools to prepare input data for model training

Visualization tools:

Matplotlib, seaborn, or other data visualization libraries enable us to generate ECG signals, model performance metrics. These tools help us understand model behaviour, identify patterns in input data, and validate model predictions against ground truth observations.

Development Environment:

Integrated development environment (IDE), Google Collab provide applications that are easy to write, debug, and use.

Version control systems, such as Git, and collaborative development platforms, such as GitHub, simplify team collaboration, code sharing, and project management.

3.3.3 Standard and Policies

Compliance with appropriate standards, norms and ethical guidelines is crucial to ensure that the sleep apnea detection system is of a high standard, safe and authentic.

Data Privacy and Security:

When dealing with sensitive patient information such as ECG recordings and health data, there is need to adhere to data protection regulations like GDPR, HIPAA or local privacy laws. To protect patient confidentiality plus unauthorized access or data breaches, encryption protocols, access controls and secure transmission mechanisms should be employed.

Medical Device Regulations:

Clinical use of sleep apnea detection systems may necessitate compliance with medical device regulation such as FDA guidelines or CE marking requirements. In line with published standards, validation studies, clinical trials and regulatory submissions might be necessary to demonstrate system safety, efficacy and performance.

Ethical Considerations:

Human subject research should be cleared by an institutional review board (IRBs) or ethics committees for example before studying algorithms used in collecting and analysing patient data. The same authorization could also touch on participant confidentiality during informed consent processes, voluntariness.

4. DESIGN APPROACH AND DETAILS

4.1 SYSTEM ARCHITECTURE

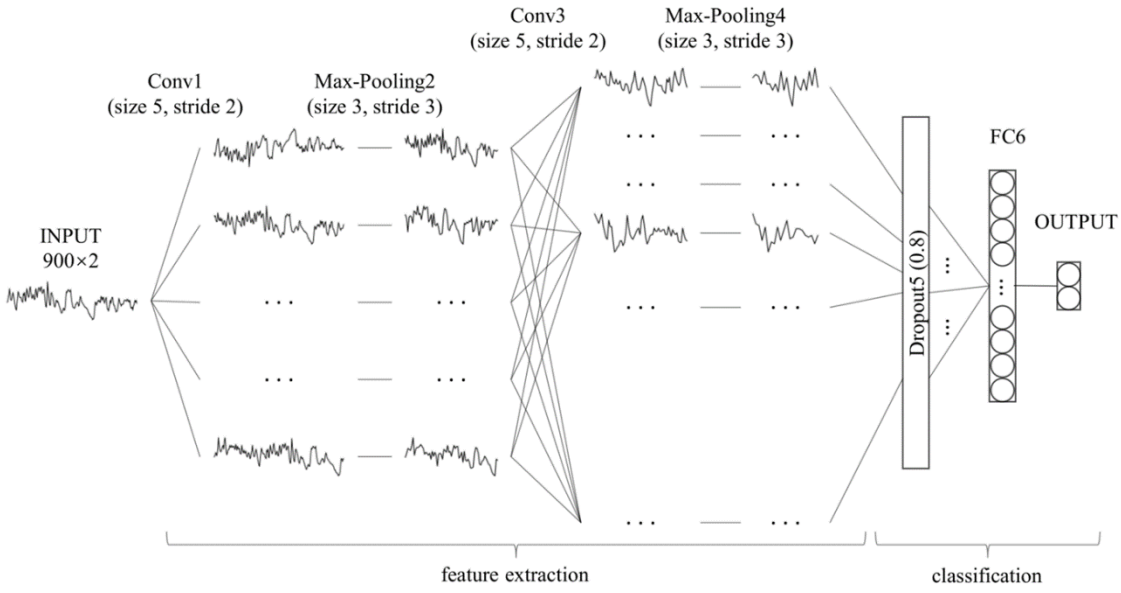

Fig 1- System Architecture

This architecture is a modified iteration of the LeNet-5 model, comprising 8 layers. Each layer operates as follows:

- Input Layer: This layer represents the grayscale single-lead ECG signals with dimensions of 32x32 pixels, serving as the input data for the model.
- Convolutional Layer 1: This layer applies 6 filters (also known as kernels) each with a size of 5x5. The convolution operation with a stride of 1 produces feature maps with dimensions 28x28x6. The activation function used here is the hyperbolic tangent (tanh) which helps in capturing basic patterns related to sleep apnea, such as variations in heartbeats.
- Average Pooling Layer 1: Following the convolutional layer, average pooling is applied with a 2x2 filter and a stride of 2. This reduces the spatial dimensions of the feature maps to 14x14x6 retaining important features.
- Convolutional Layer 2: This layer applies 16 filters of size 5x5 to extract more complex features from the ECG signals. It helps in capturing deeper patterns and abnormalities associated with sleep apnea, such as irregularities in heart rhythm.
- Average Pooling Layer 2: Like the first pooling layer, average pooling is applied with a 2x2 filter and a stride of 2, resulting in feature maps of size 5x5x16, aiding in efficient feature extraction and maintaining the most relevant information.
- Convolutional Layer 3: This layer applies 120 filters of size 5x5 and serves as a fully connected layer resulting feature map of dimensions 120x1. It helps in learning high-level representations of the input signals, which are crucial for accurate classification of sleep apnea patterns.
- Fully Connected Layer 1: This fully connected layer with 84 neurons learns complex relationships between the extracted features and their corresponding sleep patterns. It aids in capturing intricate patterns indicative of sleep apnea from the ECG signals.
- Fully Connected Layer 2 (Output Layer): The final fully connected layer with 10 neurons corresponds to the 10 classes in the sleep pattern dataset, including normal and sleep apnea patterns. The Softmax activation function generates probabilities for each class, enabling the model to classify sleep patterns accurately for sleep apnea detection.

4.2 DESIGN

4.2.1. DATA FLOW DIAGRAM

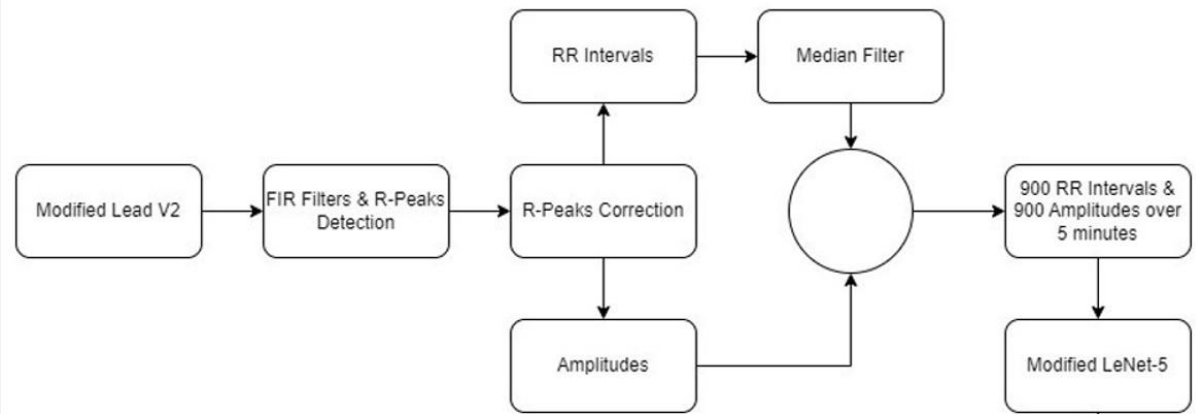

Fig 2: Dataflow diagram

Here, we start by preparing the ECG signals by extracting specific labelled segments and their neighbouring segments for analysis. We then apply the Hamilton algorithm to identify R-peaks, which are pivotal points representing heartbeats. From these R-peaks, we calculate RR intervals, which signify the time between consecutive heartbeats, and extract the corresponding R-peak values, indicating the strength of each heartbeat. To ensure reliability, we apply a median filter to smooth out any irregularities in the RR intervals. Additionally, we use cubic interpolation to ensure that our RR intervals are evenly spaced in time. Finally, we condense our data to 900 points of RR intervals and amplitudes, representing 5-minute segments, ready for in-depth analysis. Furthermore, we provided this data to the modified LeNet model.

4.2.2 USE-CASE DIAGRAM

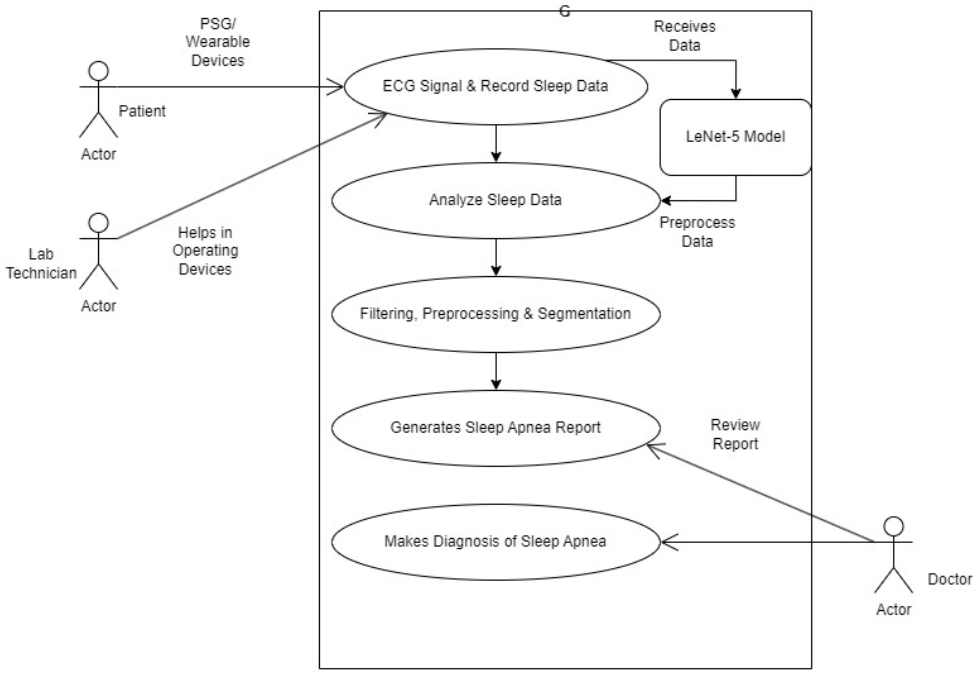

Fig 3: Use-Case diagram

The project "Sleep Apnea Detection using LeNet-5" involves the collaboration of three main actors: the Patient, Lab Technician, and Doctor.

Patients wear PSG (Polysomnography) or wearable devices to monitor their sleep, which captures signals like ECG (Electrocardiogram) to record sleep data. This data is then transmitted to the LeNet-5 model for analysis. The LeNet-5 model preprocesses the data and conducts a thorough analysis of the sleep patterns.

Following this initial analysis, filtering, preprocessing, and segmentation of the data occur to identify potential signs of sleep apnea. A detailed sleep apnea report is generated based on these findings. This report is then reviewed by a doctor who interprets the results and makes a diagnosis of sleep apnea, if present.

Additionally, Lab Technicians play a crucial role in operating the devices used for monitoring sleep, assisting in the smooth functioning of the entire process. Overall, this system aims to

provide an efficient and reliable method for detecting sleep apnea, enabling timely diagnosis and appropriate medical intervention.

4.2.3 CLASS DIAGRAM

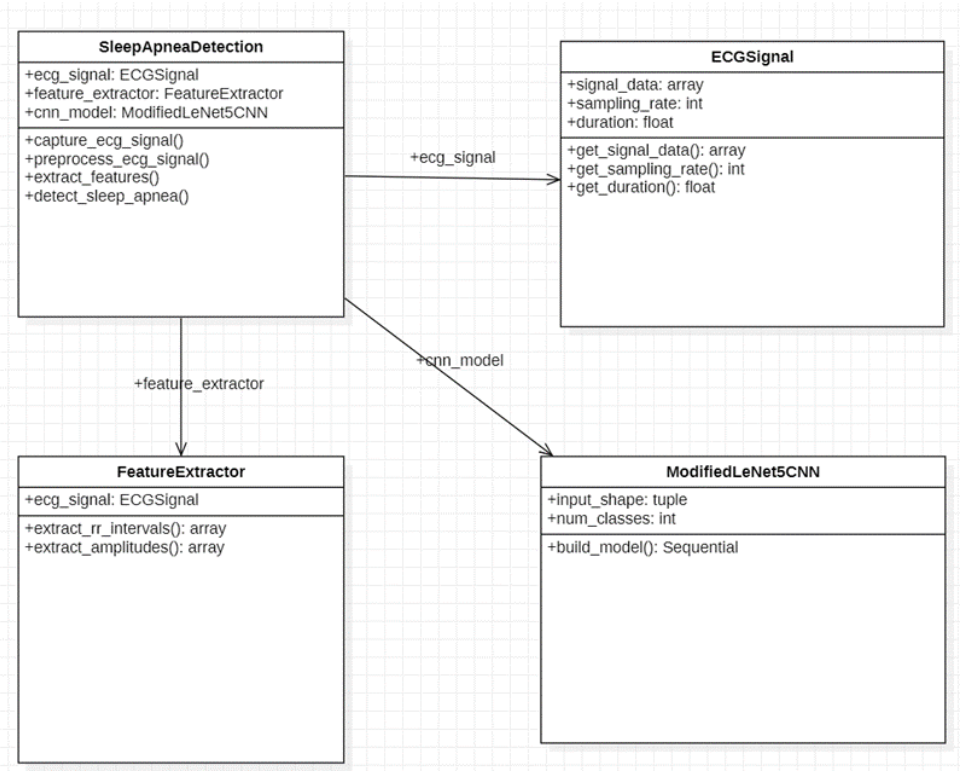

Fig 4: Class diagram

The *SleepApneaDetection* class acts as the central component, overseeing the process of detecting sleep apnea within the system. It collaborates with three other key classes: ECGSignal, FeatureExtractor, and ModifiedLeNet5CNN. Initially, the ECGSignal class manages the capture and organization of ECG signals, which are then supplied to *SleepApneaDetection* for analysis. Subsequently, the FeatureExtractor class operates in tandem with *SleepApneaDetection* to preprocess the ECG signals, extracting critical features necessary for identifying patterns indicative of sleep apnea. Finally, the ModifiedLeNet5CNN class enhances *SleepApneadetection*'s capabilities by implementing a modified LeNet-5 CNN model. This class assists in the detection process by analysing the extracted features from the ECG signals, thereby offering insights and predictions regarding the presence of Sleep Apnea. Together, these classes form a cohesive system aimed at accurately identifying Sleep Apnea through the analysis of ECG data.

4.2.4 SEQUENCE DIAGRAM

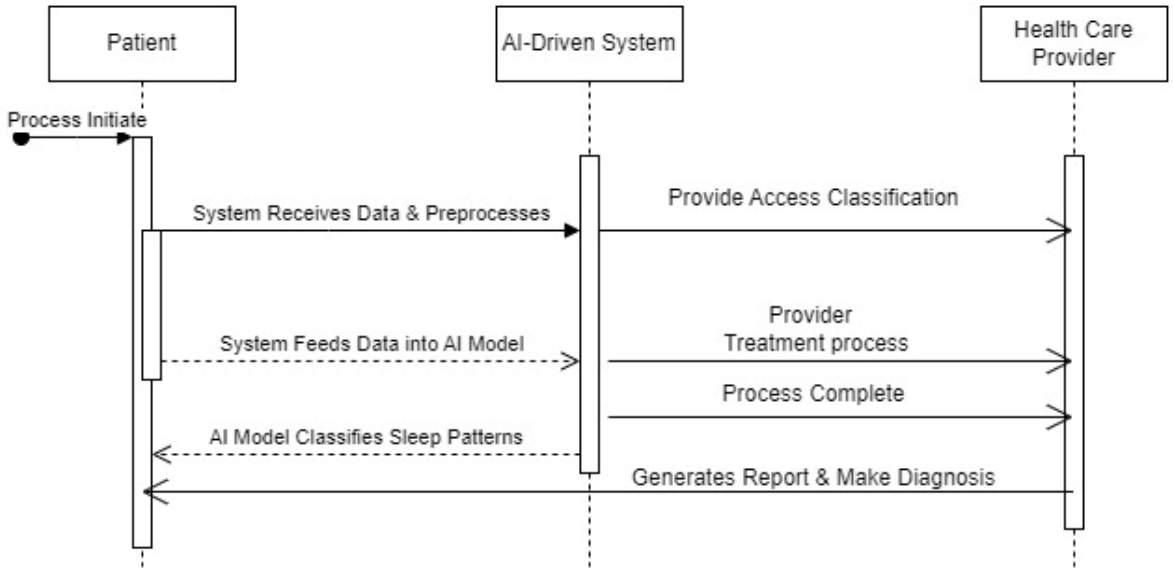

Fig 5: Sequence Diagram

This figure depicts the sequence diagram for the "Sleep Apnea Detection using LeNet-5" The process commences with the Patient initiating the monitoring or diagnostic procedure, likely by wearing PSG or a wearable device to record sleep data. This data is then transmitted to the AI-driven System, specifically designed for sleep data analysis. Upon receiving the data, the system preprocesses it to ensure proper formatting and provides access clarification to the health care provider, ensuring secure and authorized access to the results. Simultaneously, the system continues to feed the pre-processed data into the AI model, which employs algorithms to classify sleep patterns, specifically targeting indicators of sleep apnea. While this analysis is underway, the health care provider engages in the treatment process, potentially discussing treatment options with the patient or preparing for interventions based on the forthcoming results. As the process nears completion, the AI system finalizes its analysis, culminating in the generation of a comprehensive report detailing findings and recommendations. This report is then utilized by the health care provider to make a diagnosis and determine appropriate next steps for the patient's care and treatment. Throughout the entire process, collaboration between the Patient, AI-driven System, and Health care provider ensures efficient and accurate detection and management of sleep apnea.

4.3 Constraints, Alternativities and Trade-offs

Constraints: Constraints for our sleep apnea detection includes a tight timeline of one semester for requirement analysis, designing, development, testing of the entire project. On this six month of journey, different milestones and deliverables were set. Additionally, due to limited budget we were restricted access to high performance computing resources that has affected the speed of model training and testing. Domain knowledge and related expertise was also a constraint for the project because for the development of the robust sleep apnea detection, related learning and experience in ML, deep learning, signal processing, biomedical engineering is must. This constraint on domain knowledge and expertise leads to the challenges in robust algorithm development. Also, it requires supplementary training and collaboration efforts to bridge knowledges gaps.

Alternatives: During the planning and designing phase of the project, we have considered several alternatives at each step and move up with the best available solution. For the selection of model, we have explored several other machine learning models apart of modified LeNet-5. We tried with traditional machine learning models like SVM, KNN, LR, and MLP and LeNet-5 also but over here modified LeNet-5 stands out incorporating one-dimensional convolution, dropout layers, and reduced complexity for optimal performance. Similarly, we have also explored other datasets such as Sleep Heart Health Study (SHHS) dataset and many while training and testing phase of the project. Likewise, we have tried different feature sets. Beyond RR intervals and amplitudes, we have investigated other alternatives such as extracting features related to heart rate variability, spectral analysis. Exploring this diverse feature sets helped us to capture different aspects of physiological signals associated with sleep apnea and improve the model. And we have experimented with different preprocessing technique like alternative R peak detection algorithms such as Wavelet transform and Hilbert transform.

Trade Off:

a) **Data Quantity vs. Data Quality:**

Trade-off: Making use of a larger dataset with more distinct samples can improve model generalization and robustness. But if this larger data set contains noise and inaccurate data sample then the model performance and reliability will decrease significantly.

Decision: Prioritizing data quality over quantity is crucial. While a larger dataset may provide more training examples, ensuring the data is clean, reliable, and representative of the target population is essential for developing a reliable model. Quality data contributes more to model performance than sheer quantity.

b) **Algorithm Selection vs. Model Performance:**

Trade-off: Choosing between different machine learning algorithms involves trade-offs in terms of model performance, computational resources, and implementation complexity. Some algorithms may be superb in some cases some scenarios, but the same algorithm model may not be suitable in some other cases.

Decision: Conducting thorough experimentation and evaluation of multiple algorithms helps identify the most suitable approach for the specific task of sleep apnea detection. It's essential to weigh the pros and cons of each algorithm and select the one that best balances

performance, resource requirements, and ease of implementation. Here in our case a modified LeNet-5 was required with some adjustment with traditional LeNet-5.

5. SCHEDULES, TASKS AND MILESTONES

5.1 Gantt Chart

Fig 6: Gantt Chart

5.2 Module Description

5.2.1 Datasets module: This module deals with the acquisition and description of datasets used in the project.

1. PhysioNet Apnea-ECG Dataset:

Dataset: PhysioNet Apnea-ECG dataset from Philipps University.
Contents: 70 single-lead ECG signal recordings.
Sets: Released set (35 recordings) and withheld set (35 recordings).
Sampling: Signals sampled at 100 Hz.
Duration: Recordings ranged from 401 to 587 minutes.

Annotation: Expert annotations for 1-minute segments indicating apnea (labelled as SA) or normal.
Event Classification: No distinction between hypopnea and apnea; all events categorized as obstructive or mixed (central events excluded).
Recording Classes:
Class A: AHI ≥ 10, with at least 100 SA segments throughout.
Class B: AHI ≥ 5, with 5 to 99 SA segments.
Class C (Normal): AHI < 5.
https://physionet.org/content/ucddb/1.0.0/

5.2.2 Pre-processing step module: This module focuses on the preprocessing steps applied to the ECG signals before feature extraction and model training. The preprocessing involves extracting RR intervals and amplitudes from the ECG signals. The Hamilton algorithm is used to detect R-peaks, from which RR intervals and amplitudes are calculated. Preprocessing also includes handling noisy RR interval data using a median filter and ensuring even spacing with cubic interpolation.

5.2.3 Feature extraction module: Feature extraction is performed to derive informative features from the pre-processed ECG signals. Specifically, features related to RR intervals and amplitudes are extracted, as they are deemed crucial for sleep apnea detection.

We chose 12 features related to RR intervals and 6 features related to amplitudes, which had been shown to be important in sleep apnea detection. Some machine learning methods, like KNN, are sensitive to the scales of features. To make sure all features are on a similar scale, they used a technique called min-max normalization.

- $X *= \frac{X-Xmin}{X-Xmax}$

This process rescales each feature so that its values fall between 0 and 1, making it easier for the model to learn from them.

5.2.4 Machine learning model module: This module involves the implementation of machine learning models for sleep apnea detection. Two approaches are mentioned: a CNN-based approach (using LeNet-5 architecture) and traditional feature engineering-based techniques (SVM, KNN, LR, MLP). The modified LeNet-5 architecture is described, emphasizing changes such as one-dimensional convolution, dropout layer addition, and reduced complexity. Traditional machine learning models are trained and evaluated using selected features.

We used a simple and effective CNN implementation, LeNet-5, to construct our SA detection model. We will introduce both the standard LeNet-5 and our modified LeNet-5.

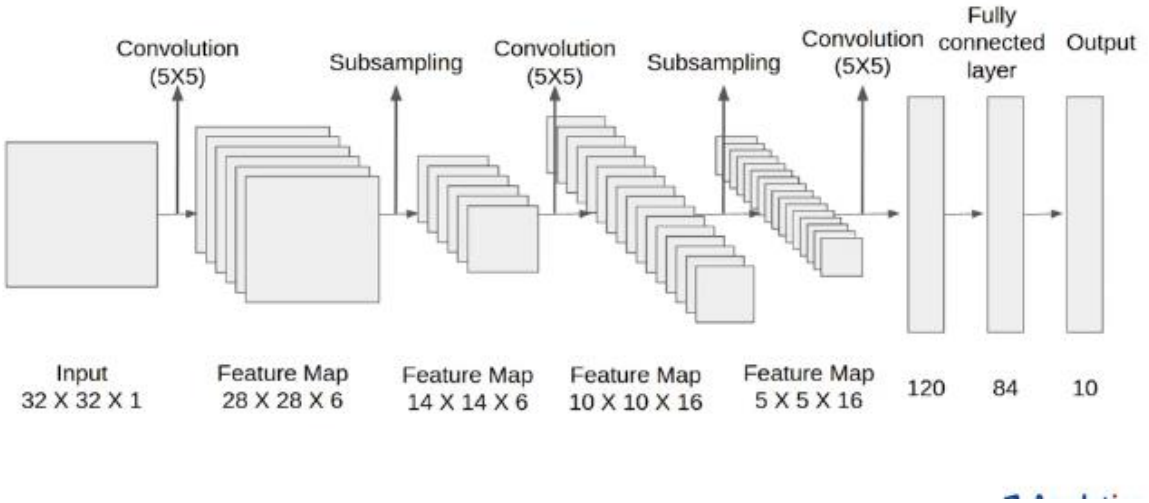

Fig 7: Standard LeNet-5

Layer	#Filters/ Neurons	Filter Size	Stride	Size of feature map	Activation Function
input	-	-	-	32*32*1	
Conv 1	6	5*5	1	28*28*6	tanh
Avg. Pooling 1		2*2	2	14*14*6	
Conv 2	16	5*5	1	10*10*16	tanh
Avg. Pooling 2		2*2	2	5*5*16	
Conv 3	120	5*5	1	120	tanh
Fully Connected 1	-	-	-	84	tanh
Fully Connected 2	-	-	-	10	SoftMax

Table 2: Standard LeNet-5

1. One-Dimensional Convolution Operation:

Change: Instead of using two-dimensional convolution (which is common in image recognition tasks like character recognition), they switched to one-dimensional convolution.

Reason: The data they were dealing with was one-dimensional (like a sequence of numbers over time), unlike images which are two-dimensional. So, one-dimensional convolution better suited their data.

2. Dropout Layer Addition:

Change: They added a dropout layer between the convolution layer and the fully connected layer.

Reason: Dropout helps prevent overfitting, where the model memorizes the training data too much and performs poorly on new, unseen data. It randomly drops out some connections during training, forcing the network to learn more robust features.

3. Reduced Complexity with One Fully Connected Layer:

Change: They kept only one fully connected layer instead of multiple layers.
Reason: This simplifies the network structure, making it easier to train and less prone to overfitting.

4. Adjusted Convolution Layer Strides and Fully Connected Layer Nodes:

Change: They modified the size of the convolution layer strides and the number of nodes in the fully connected layer.
Reason: This fine-tuning helps improve the model's performance for the specific task of sleep apnea detection.

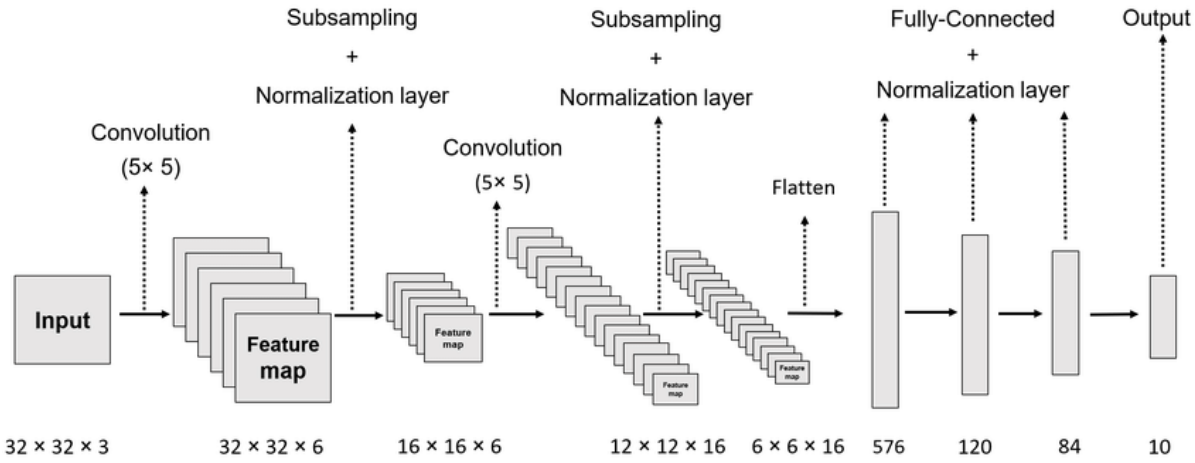

Fig 8: Modified LeNet-5

Layer	Parameter	Output Shape	Number
Input	-	(None,900,2)	0
Conv 1	32*5*2, stride 2, pad 0	(None,448,32)	352
Max Pooling 2	3, stride 3, pad 0	(None,149,32)	0
Conv 3	64*5*2, stride 2, pad 0	(None,73,64)	10304
Max Pooling 4	3, stride 3, pad 0	(None,24,64)	0
Dropout 5	0.8 rate	(None,24,64)	0
FC 6	32, ReLU	(None,32)	49184
Output	2, SoftMax	(None,2)	66

Table 3: Modified LeNet-5

5.2.5 Evaluation metrics module: This module discusses the evaluation metrics used to assess the performance of the sleep apnea detection models. The evaluation metrics used to assess the

performance of the proposed method are Specificity (Sp), Sensitivity (Sn) & Accuracy (Acc). These metrics are defined as follows:

Specificity (Sp): The proportion of true negative (TN) predictions among all actual negative instances.

- $Specificity = \frac{TN}{TN+FP}$

Sensitivity (Sn): The proportion of true positive (TP) predictions among all actual positive instances.

- $Sensitivity = \frac{TP}{TP+FN}$

Accuracy (Acc): The proportion of correctly classified instances (both positive and negative) among all instances.

- $Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$

5.3 TESTING

5.3.1 Unit Testing

Unit testing involves breaking down the software into its smallest testable parts, or units, and testing each part individually to ensure it functions correctly in isolation. In the preprocessing module of our project, various steps are involved in preparing the ECG signals for further analysis. Let's take the example of the R-peak detection algorithm. This algorithm identifies the peaks in the ECG signal, which correspond to the heart's contractions.

To conduct unit testing for the R-peak detection algorithm:

- ✓ Test Data Preparation: Manually annotate a dataset with known R-peaks. This dataset will serve as the ground truth for testing.
- ✓ Test Execution: Apply the R-peak detection algorithm to the test dataset.
- ✓ Assertion: Compare the algorithm's detected R-peaks with the manually annotated R-peaks. Verify that the algorithm's output aligns closely with the ground truth.
- ✓ Evaluation: Assess the accuracy and reliability of the R-peak detection algorithm based on the comparison results.

By conducting unit tests for each preprocessing step, such as R-peak detection, RR interval calculation, and noise filtering, we can ensure that each component of the preprocessing module operates correctly in isolation, laying a solid foundation for subsequent stages of the pipeline.

5.3.2 Integration Testing

Integration testing verifies the interactions between different modules or components of the software to ensure they function together seamlessly. In our project, the preprocessing module prepares the ECG signals, while the feature extraction module extracts informative features

from the pre-processed signals. Integration testing ensures that the interaction between these modules produces the expected results.

To conduct integration testing between the preprocessing and feature extraction modules:

- ✓ Test Data Preparation: Preprocess a set of ECG signals using the preprocessing module.
- ✓ Test Execution: Pass the pre-processed signals to the feature extraction module for feature extraction.
- ✓ Assertion: Verify that the extracted features align with the expected features based on the characteristics of the ECG signals and the preprocessing steps applied.
- ✓ Evaluation: Assess the consistency and accuracy of the feature extraction process when integrated with the preprocessing module.

Integration testing validates that the preprocessing and feature extraction modules work harmoniously together, ensuring that the extracted features accurately represent the characteristics of the ECG signals, which is essential for subsequent stages such as model training and sleep apnea detection.

6. PROJECT DEMONSTRATION

Input

The dataset consists of 70 single-lead ECG recordings with expert annotations for apnea, that we have taken from physionet apnea ecg[27]. These signals, sampled at 100 Hz and ranging from 401 to 587 minutes, serve as the primary input. These input signals are pre-processed to detects R-peaks in ECG signals, enabling the calculation of RR intervals and amplitudes.

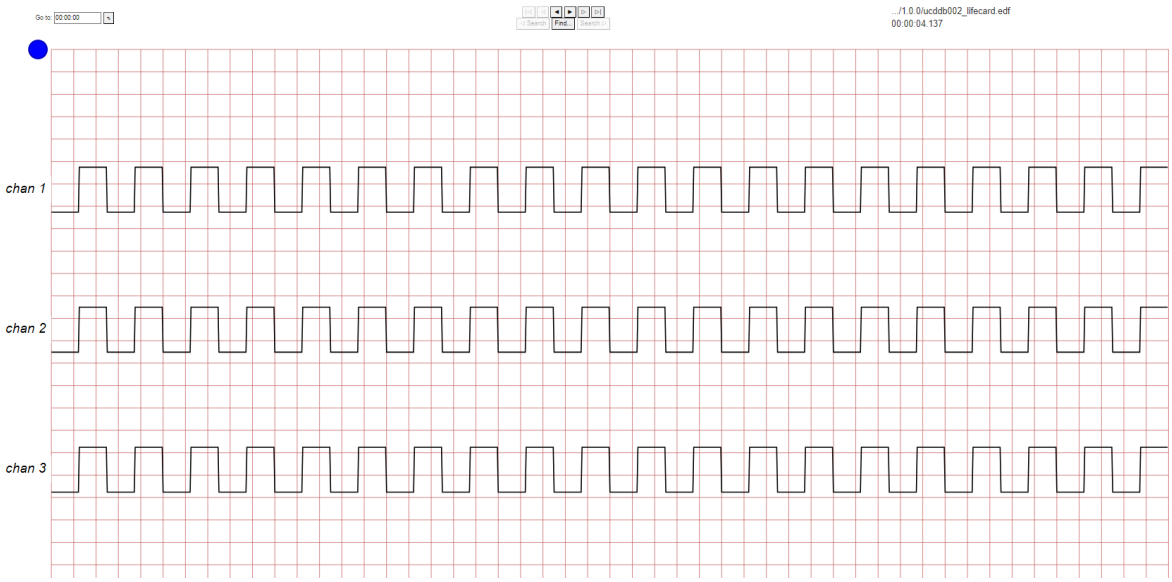

Fig 9: ECG signals

```
# PhysioNet Apnea-ECG dataset
# url: https://physionet.org/physiobank/database/apnea-ecg/
base_dir = DATA_DIR

fs = 100
sample = fs * 60  # 1 min's sample points

before = 2  # forward interval (min)
after = 2  # backward interval (min)
hr_min = 20
hr_max = 300

num_worker = 35 if os.cpu_count() > 35 else os.cpu_count() - 1  # Setting according to the number of CPU cores
```

Developed a model which is used for predicting the result.

```
def create_model(input_shape, weight=1e-3):
    """Create a Modified LeNet-5 model"""
    inputs = Input(shape=input_shape)

    # Conv1
    x = Conv1D(32, kernel_size=5, strides=2, padding="valid", activation="relu", kernel_initializer="he_normal",
               kernel_regularizer=l2(weight), bias_regularizer=l2(weight))(inputs)
    x = MaxPooling1D(pool_size=3)(x)

    # Conv3
    x = Conv1D(64, kernel_size=5, strides=2, padding="valid", activation="relu", kernel_initializer="he_normal",
               kernel_regularizer=l2(1e-3), bias_regularizer=l2(weight))(x)
    x = MaxPooling1D(pool_size=3)(x)

    x = Dropout(0.8)(x) # Avoid overfitting

    # FC6
    x = Flatten()(x)
    x = Dense(32, activation="relu")(x)
    outputs = Dense(2, activation="softmax")(x)

    model = Model(inputs=inputs, outputs=outputs)
    return model
```

The model achieved high accuracy of 93.33% on the training set and 87.33% on the testing set, with strong sensitivity (91.90%) and specificity (94.24%) during training, indicating robust performance in identifying sleep apnea instances.

However, on the testing set, sensitivity slightly decreased to 84.85%, while specificity remained high at 88.87%, suggesting room for improvement in detecting true positive cases without compromising overall accuracy. Despite this, the model demonstrated balanced precision, recall, and F1-score values for both normal (0.0) and sleep apnea (1.0) instances, showcasing its effectiveness in sleep apnea detection from ECG signals.

Training Report:

	precision	recall	f1-score	support
0.0	0.95	0.94	0.95	10236
1.0	0.91	0.92	0.91	6473
accuracy			0.93	16709
macro avg	0.93	0.93	0.93	16709
weighted avg	0.93	0.93	0.93	16709

Testing Report:

	precision	recall	f1-score	support
0.0	0.90	0.89	0.90	10455
1.0	0.83	0.85	0.84	6490
accuracy			0.87	16945
macro avg	0.86	0.87	0.87	16945
weighted avg	0.87	0.87	0.87	16945

7. RESULT & DISCUSSION

Our modified LeNet-5 model, equipped with automatic feature extraction, demonstrates impressive performance in accurately predicting the presence of sleep apnea (SA) based on electrocardiogram (ECG) segments, analysed on a minute-by-minute basis. This capability is crucial in the medical field, as it forms a fundamental aspect of diagnosing suspected SA patients. In comparison with traditional machine learning methods, our proposed model exhibits superior performance.

The withheld set evaluation showcases the robustness of our model, with notable metrics including a specificity of 90.3%, sensitivity of 83.1%, accuracy of 87.6%, and an area under the receiver operating characteristic curve (AUC) of 0.950. These metrics highlight the model's ability to effectively distinguish between individuals with and without SA, providing reliable diagnostic support.

Comparative analysis against SVM, the second-highest performing method, reveals substantial improvements across all evaluation metrics. Specifically, our model outperforms SVM by margins of 6.0% in specificity, 6.2% in sensitivity, 6.2% in accuracy, and 0.063 in AUC, respectively. This significant margin underscores the efficacy of our modified LeNet-5 approach in SA detection.

Furthermore, the comparative results indicate that KNN, another method evaluated, achieved the lowest prediction accuracy among the considered methods. This outcome is attributed to potential limitations in the feature extraction process from the ECG segments, underscoring the importance of employing advanced deep learning techniques, such as our modified LeNet-5, for enhanced diagnostic accuracy in SA detection.

Overall, the results affirm the effectiveness of our proposed method in SA detection, showcasing its superiority over traditional machine learning approaches and underscoring its potential to advance diagnostic capabilities in the field of sleep medicine.

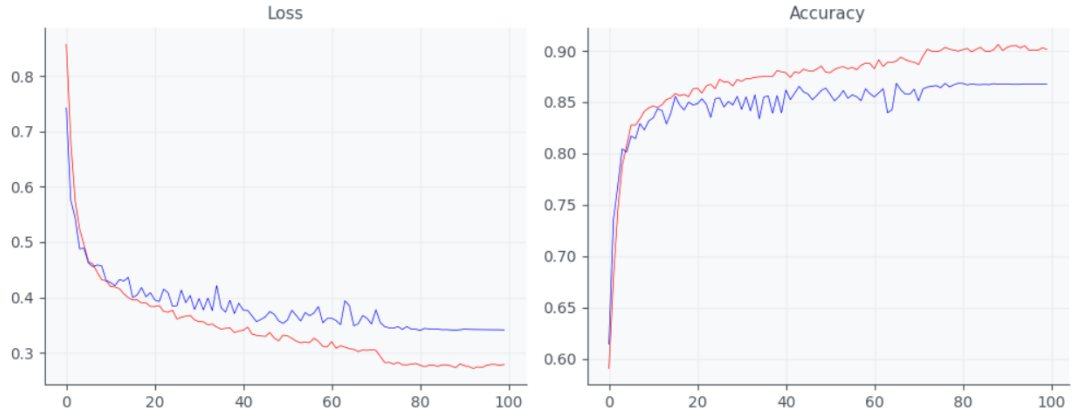

Fig 10: AUC Graph

8. SUMMARY

In our investigation, we devised a method for detecting sleep apnea (SA) using a modified version of LeNet-5 in conjunction with adjacent electrocardiogram (ECG) segments. Our experimental findings underscore the effectiveness of our approach in SA detection, surpassing the performance of both conventional machine learning methods and existing methodologies. Given the stringent precision requirements in clinical settings, further refinements to our proposed method hold the potential to expedite the development of ECG-based SA detection tools for clinical application.

Moreover, our method's reliance on a single-lead ECG signal opens avenues for its utilization in the development of SA detection solutions for home healthcare services employing wearable devices. Nonetheless, it is imperative to acknowledge certain limitations of our proposed approach. Notably, the segmentation of the Apnea-ECG dataset into 1-minute intervals may not fully capture instances where apnea/hypopnea events span across adjacent segments or when a single segment contains multiple events. Additionally, the dataset's annotations do not distinguish between hypopnea and apnea events, and they primarily encompass obstructive and mixed events, omitting central events.

Therefore, our method may face challenges in discerning between hypopnea and apnea events, as well as in detecting central events. In forthcoming research endeavours, we aim to address these limitations by incorporating additional datasets, thereby enhancing the robustness and applicability of our proposed method for SA detection.

9. References

[1]. Franklin, K., & Lindberg, E. (2015). Obstructive sleep apnea is a common disorder in the population—a review on the epidemiology of sleep apnea. Journal Of Thoracic Disease, 7(8), 1311-1322. https://doi: 10.3978/j.issn.2072-1439.2015.06.11

[2]. Platon, A. L., Stelea, C. G., Boișteanu, O., Patrascanu, E., Zetu, I. N., Roșu, S. N., Trifan, V., & Palade, D. O. (2023). An Update on Obstructive Sleep Apnea Syndrome—A Literature Review. Medicina, 59(8), 1459. https://doi.org/10.3390/medicina59081459

[3]. Eiseman, N., Westover, M. B., Mietus, J., Thomas, R., & Bianchi, M. (2011). Classification algorithms for predicting sleepiness and sleep apnea severity. Journal of Sleep Research, 21, 101-112. https://doi.org/10.1111/j.1365-2869.2011.00935.x

[4]. Avcı, C., & Akbaş, A. (2015). Sleep apnea classification based on respiration signals by using ensemble methods. Bio-Medical Materials and Engineering, 26(s1), S1703-S1710. https://doi.org/10.3233/BME-151470

[5]. Vimala, V., Ramar, K. & Ettappan, M. An Intelligent Sleep Apnea Classification System Based on EEG Signals. J Med Syst 43, 36 (2019). https://doi.org/10.1007/s10916-018-1146-8

[6]. Zhao, X., Wang, X., Yang, T. et al. Classification of sleep apnea based on EEG sub-band signal characteristics. Sci Rep 11, 5824 (2021). https://doi.org/10.1038/s41598-021-85138-0

[7]. Qureshi, A., Ballard, R. D., & Nelson, H. S. (2003). Obstructive sleep apnea. Journal of Allergy and Clinical Immunology, 112(4), 643-651. https://doi.org/10.1016/j.jaci.2003.08.031

[8]. Hu, S., Wang, Y., Liu, J., Yang, C., Wang, A., Li, K., & Liu, W. (2023). Semi-Supervised Learning for Low-Cost Personalized Obstructive Sleep Apnea Detection Using Unsupervised Deep Learning and Single-Lead Electrocardiogram. IEEE Journal of Biomedical and Health Informatics, Biomedical and Health Informatics, IEEE Journal of, IEEE J. Biomed. Health Inform, 27(11), 5281–5292. https://doi-org.egateway.vit.ac.in/10.1109/JBHI.2023.3304299

[9]. Zou, R., Yue, H., Lei, W., Fan, X., Ma, W., Li, P., & Li, Y. (2023). A Dual-Scale Convolutional Neural Network for Sleep Apnea Detection with Time-Delayed SpO2 Signals. 2023 45th Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC), IEEE Engineering in Medicine & Biology Society (EMBC), 2023 45th Annual International Conference of The, 1–4. https://doi-org.egateway.vit.ac.in/10.1109/EMBC40787.2023.10340999

[10]. Huang, Y., Chen, K., & Zhang, Z. (2023). Domain - Aware Spatial-Temporal Graph Convolutional Network for Sleep Apnea Detection via Multivariant BCG Signals. ICC 2023 - IEEE International Conference on Communications, Communications, ICC 2023 - IEEE International Conference On, 5515–5520. https://doi-org.egateway.vit.ac.in/10.1109/ICC45041.2023.10278967

[11]. Djamal, E. C., Syaukani, H., & Kasyidi, F. (2023). Sleep Apnea Detection Based on Respiratory Movement Using Harris Corner and Recurrent Neural Networks. 2023 International Conference on Computer, Control, Informatics and Its Applications (IC3INA), Computer, Control, Informatics and Its Applications (IC3INA), 2023 International Conference On, 440–445. https://doi-org.egateway.vit.ac.in/10.1109/IC3INA60834.2023.10285748

[12]. Anu Shilvya, J., & Jose, P. S. H. (2023). Obstructive Sleep Apnea Detection from ECG Signals with Deep Learning. 2023 International Conference on Sustainable Computing and Smart Systems (ICSCSS), Sustainable Computing and Smart Systems (ICSCSS), 2023 International Conference On, 384–388. https://doi-org.egateway.vit.ac.in/10.1109/ICSCSS57650.2023.10169207

[13]. Bhongade, A., Gupta, R., Gandhi, T. K., & Ap, P. (2023). A Portable Low-Cost Respiration Rate Measurement System for Sleep Apnea Detection. 2023 45th Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC), IEEE Engineering in Medicine & Biology Society (EMBC), 2023 45th Annual International Conference of The, 1–5. https://doi-org.egateway.vit.ac.in/10.1109/EMBC40787.2023.10340446

[14]. Yan, X., Wang, L., Zhu, J., Wang, S., Zhang, Q., & Xin, Y. (2022). Automatic Obstructive Sleep Apnea Detection Based on Respiratory Parameters in Physiological Signals. 2022 IEEE International Conference on Mechatronics and Automation (ICMA), Mechatronics and Automation (ICMA), 2022 IEEE International Conference On, 461–466. https://doi-org.egateway.vit.ac.in/10.1109/ICMA54519.2022.9856347

[15] H. Azimi et al., "Machine Learning-Based Automatic Detection of Central Sleep Apnea Events from a Pressure Sensitive Mat," IEEE Access, vol. 8, pp. 160726-160735, 2020. https://www.researchgate.net/publication/346891313_Machine_Learning-Based_Automatic_Detection_of_Central_Sleep_Apnea_Events_From_a_Pressure_Sensitive_Mat

[16] A. Goldstein et al., "Artificial intelligence in sleep medicine: background and implications for clinicians," J. Clin. Sleep Med., vol. 16, no. 2, pp. 161–172, Feb. 2020. DOI: 10.5664/jcsm.8388.Artificial intelligence in sleep medicine: background and implications for clinicians - PubMed (nih.gov)

[17] D. R. Mazzotti et al., "Opportunities for utilizing polysomnography signals to characterize obstructive sleep apnea subtypes and severity," in IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 26, no. 11, pp. 2205-2213, Nov. 2018. DOI: 10.1109/TNSRE.2018.2860353. https://pubmed.ncbi.nlm.nih.gov/30047487/

[18] C.-Y. Lin, Y.-W. Wang, F. Setiawan, N. T. H. Trang, and C.-W. Lin, "Sleep Apnea Classification Algorithm Development Using a Machine-Learning Framework and Bag-of-Features Derived from Electrocardiogram Spectrograms," J. Clin. Med., vol. 11, no. 1, p. 192, Jan. 2022, doi: 10.3390/jcm11010192. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8745785/

[19] A. Bandyopadhyay and C. Goldstein, "Clinical applications of artificial intelligence in sleep medicine: a sleep clinician's perspective," Sleep Breath., vol. 26, no. 4, pp. 1561-1562, Dec. 2022, doi: 10.1007/s11325-022-02592-4. https://pubmed.ncbi.nlm.nih.gov/35262853/

[20] J. Levy, D. Álvarez, F. Del Campo, and J. A. Behar, "Deep learning for obstructive sleep apnea diagnosis based on single channel oximetry," Nat. Commun., vol. 14, no. 1, p. 1378, Mar. 2023, doi: 10.1038/s41467-023-40604-3. https://www.nature.com/articles/s41467-023-40604-3

[21] Vaquerizo-Villar, F., Gutiérrez-Tobal, G. C., Calvo, E., Alvarez, D., Kheirandish-Gozal, L., del Campo, F., Gozal, D., & Hornero, R. (2023). An explainable deep-learning model to stage sleep states in children and propose novel EEG-related patterns in sleep apnea. Computers in Biology and Medicine, 165, 107419. https://doi.org/10.1016/j.compbiomed.2023.107419

[22] Serrano Alarcón, Á., Martínez Madrid, N., Seepold, R., & Ortega, J. A. (2023). Obstructive sleep apnea event detection using explainable deep learning models for a portable monitor. Computers in Biology and Medicine. https://doi.org/10.3389/fnins.2023.1155900

[23] Zhang, H., & Ogasawara, K. (2023). Grad-CAM-Based Explainable Artificial Intelligence Related to Medical Text Processing. Graduate School of Health Science, Hokkaido University, N12-W5, Kitaku, Sapporo 060-0812, Japan. https://doi.org/10.3390/bioengineering10091070

[24] Dutt, M., Redhu, S., Goodwin, M. *et al.* Sleep XAI: An explainable deep learning approach for multi-class sleep stage identification. *Appl Intell* **53**, 16830–16843 (2023). https://doi.org/10.1007/s10489-022-04357-8

[25] Serrano Alarcon, A., Martínez Madrid, N., Seepold, R., & Ortega, J. (2023, July 14). Obstructive sleep apnea event detection using explainable deep learning models for a portable monitor. Frontiers in Neuroscience, 17. https://doi.org/10.3389/fnins.2023.1155900

[26] Jeong, H. G., Kim, T., Hong, J. E., Kim, H. J., Yun, S. Y., Kim, S., Yoo, J., Lee, S. H., Thomas, R. J., & Yun, C. H. (2023). Automated deep neural network analysis of lateral cephalogram data can aid in detecting obstructive sleep apnea. Journal of clinical sleep medicine: JCSM: official publication of the American Academy of Sleep Medicine, 19(2), 327–337. https://doi.org/10.5664/jcsm.10258

[27] Dataset Link: http://www.physionet.org/physiobank/database/apnea-ecg

APPENDIX A – SAMPLE CODE

```
from google.colab import drive
drive.mount('/content/gdrive')
# Setting up the cloud Environment
## Run only once to setup the data
# %%shell
# DATA_DIR="/content/gdrive/MyDrive/datasets/"
# OUT_DIR="/content/gdrive/MyDrive/output/apnea-ecg"
# mkdir -p $DATA_DIR
# mkdir -p OUT_DIR
# wget https://physionet.org/static/published-projects/apnea-ecg/apnea-ecg-database-1.0.0.zip
# unzip -d $DATA_DIR apnea-ecg-database-1.0.0.zip
# mv /content/gdrive/MyDrive/datasets/apnea-ecg-database-1.0.0
/content/gdrive/MyDrive/datasets/apnea-ecg
# # Code to copy models into gdrive
# %%shell
# rm -rf /content/gdrive/MyDrive/apnea-ecg/
# mkdir -p /content/gdrive/MyDrive/apnea-ecg/
# rm -rf Sleep-apnea-detection-through-a-modified-LeNet-5-convolutional-neural-network
# cp -r Sleep-apnea-detection-through-a-modified-LeNet-5-convolutional-
neuralnetwork/models /content/gdrive/MyDrive/apnea-ecg/
# unzip -d /content/gdrive/MyDrive/apnea-ecg/models
/content/gdrive/MyDrive/apneaecg/models/model.final.h5.zip
# cp -r Sleep-apnea-detection-through-a-modified-LeNet-5-convolutional-neuralnetwork/utils
/content/gdrive/MyDrive/apnea-ecg
# unzip -d /content/gdrive/MyDrive/apnea-ecg/utils /content/gdrive/MyDrive/apnea-
ecg/utils/code_for_calculating_per-recording.zip
%%shell
pip uninstall keras tensorflow -y
```

```
pip install keras tensorflow
pip install biosppy wfdb tqdm
#Preprocess the Data
from google.colab import drive
drive.mount('/content/drive') import os
DATA_DIR = "/content/gdrive/MyDrive/datasets/apnea-ecg"
OUT_DIR = "/content/gdrive/MyDrive/output/apnea-ecg"
MODEL_DIR = "/content/gdrive/MyDrive/apnea-ecg/models"
if not os.path.exists(DATA_DIR):
os.makedirs(DATA_DIR, exist_ok=True)
if not os.path.exists(OUT_DIR):
os.makedirs(OUT_DIR, exist_ok=True)
if not os.path.exists(MODEL_DIR):
os.makedirs(MODEL_DIR, exist_ok=True)

DATA_DIR
import pickle
import sys
from concurrent.futures import ProcessPoolExecutor, as_completed
import biosppy.signals.tools as st
import numpy as np
import os
import wfdb
from biosppy.signals.ecg import correct_rpeaks, hamilton_segmenter
from scipy.signal import medfilt
# from sklearn.utils import cpu_count
from tqdm import tqdm
# PhysioNet Apnea-ECG dataset
# url: https://physionet.org/physiobank/database/apnea-ecg/
base_dir = DATA_DIR
fs = 100
sample = fs * 60 # 1 min's sample points
before = 2 # forward interval (min)
after = 2 # backward interval (min)
hr_min = 20
hr_max = 300
num_worker = 35 if os.cpu_count() > 35 else os.cpu_count() - 1 # Setting according to the
number of CPU cores
def worker(name, labels):
X = []
y = []
groups = []
signals = wfdb.rdrecord(os.path.join(base_dir, name), channels=[0]).p_signal[:, 0]
for j in tqdm(range(len(labels)), desc=name, file=sys.stdout):
if j < before or \
(j + 1 + after) > len(signals) / float(sample):
continue
```

```
signal = signals[int((j - before) * sample):int((j + 1 + after) * sample)]
signal, _, _ = st.filter_signal(signal, ftype='FIR', band='bandpass', order=int(0.3 * fs),
frequency=[3, 45], sampling_rate=fs)
# Find R peaks
rpeaks, = hamilton_segmenter(signal, sampling_rate=fs)
rpeaks, = correct_rpeaks(signal, rpeaks=rpeaks, sampling_rate=fs, tol=0.1)
if len(rpeaks) / (1 + after + before) < 40 or \
len(rpeaks) / (1 + after + before) > 200: # Remove abnormal R peaks signal
continue
# Extract RRI, Ampl signal
rri_tm, rri_signal = rpeaks[1:] / float(fs), np.diff(rpeaks) / float(fs)
rri_signal = medfilt(rri_signal, kernel_size=3)
ampl_tm, ampl_siganl = rpeaks / float(fs), signal[rpeaks]
hr = 60 / rri_signal
# Remove physiologically impossible HR signal
if np.all(np.logical_and(hr >= hr_min, hr <= hr_max)):
# Save extracted signal
X.append([(rri_tm, rri_signal), (ampl_tm, ampl_siganl)])
y.append(0. if labels[j] == 'N' else 1.)
groups.append(name)
return X, y, groups
if __name__ == "__main__":
apnea_ecg = {}
names = [
"a01", "a02", "a03", "a04", "a05", "a06", "a07", "a08", "a09", "a10",
"a11", "a12", "a13", "a14", "a15", "a16", "a17", "a18", "a19", "a20",
"b01", "b02", "b03", "b04", "b05",
"c01", "c02", "c03", "c04", "c05", "c06", "c07", "c08", "c09", "c10"
]
o_train = []
y_train = []
groups_train = []
print('Training...')
with ProcessPoolExecutor(max_workers=num_worker) as executor:
task_list = []
for i in range(len(names)):
labels = wfdb.rdann(os.path.join(base_dir, names[i]), extension="apn").symbol
task_list.append(executor.submit(worker, names[i], labels))
for task in as_completed(task_list):
X, y, groups = task.result()
o_train.extend(X)
y_train.extend(y)
groups_train.extend(groups)
print()
answers = {}
with open(os.path.join(base_dir, "event-2-answers"), "r") as f:
for answer in f.read().split("\n\n"):
```

```
answers[answer[:3]] = list("".join(answer.split()[2::2]))
names = [
"x01", "x02", "x03", "x04", "x05", "x06", "x07", "x08", "x09", "x10",
"x11", "x12", "x13", "x14", "x15", "x16", "x17", "x18", "x19", "x20",
"x21", "x22", "x23", "x24", "x25", "x26", "x27", "x28", "x29", "x30",
"x31", "x32", "x33", "x34", "x35"
]
o_test = []
y_test = []
groups_test = []
print("Testing...")
with ProcessPoolExecutor(max_workers=num_worker) as executor:
task_list = []
for i in range(len(names)):
labels = answers[names[i]]
task_list.append(executor.submit(worker, names[i], labels))
for task in as_completed(task_list):
X, y, groups = task.result()
o_test.extend(X)
y_test.extend(y)
groups_test.extend(groups)
apnea_ecg = dict(o_train=o_train, y_train=y_train, groups_train=groups_train,
o_test=o_test, y_test=y_test,
groups_test=groups_test)
with open(os.path.join(base_dir, "apnea-ecg.pkl"), "wb") as f:
pickle.dump(apnea_ecg, f, protocol=2)
print("\nok!")
# Train Model with LeNet
"""NOTES: Batch data is different each time in keras, which result in slight differences in
results."""
import pickle
import keras
import matplotlib.pyplot as plt
import numpy as np
import os
import os
from keras.callbacks import LearningRateScheduler
from keras.layers import Conv1D, Dense, Dropout, Flatten, MaxPooling1D, Input
from keras.models import Model
from keras.regularizers import l2
from scipy.interpolate import splev, splrep
from scipy.interpolate import splev, splrep
import pandas as pd
base_dir = DATA_DIR
ir = 3 # interpolate interval
before = 2
after = 2
```

```
# normalize
scaler = lambda arr: (arr - np.min(arr)) / (np.max(arr) - np.min(arr))
def load_data():
tm = np.arange(0, (before + 1 + after) * 60, step=1 / float(ir))
with open(os.path.join(DATA_DIR, "apnea-ecg.pkl"), 'rb') as f: # read preprocessing result
apnea_ecg = pickle.load(f)
x_train = []
o_train, y_train = apnea_ecg["o_train"], apnea_ecg["y_train"]
groups_train = apnea_ecg["groups_train"]
for i in range(len(o_train)):
(rri_tm, rri_signal), (ampl_tm, ampl_siganl) = o_train[i]
# Curve interpolation
rri_interp_signal = splev(tm, splrep(rri_tm, scaler(rri_signal), k=3), ext=1)
ampl_interp_signal = splev(tm, splrep(ampl_tm, scaler(ampl_siganl), k=3), ext=1)
x_train.append([rri_interp_signal, ampl_interp_signal])
x_train = np.array(x_train, dtype="float32").transpose((0, 2, 1)) # convert to numpy format
y_train = np.array(y_train, dtype="float32")
x_test = []
o_test, y_test = apnea_ecg["o_test"], apnea_ecg["y_test"]
groups_test = apnea_ecg["groups_test"]
for i in range(len(o_test)):
(rri_tm, rri_signal), (ampl_tm, ampl_siganl) = o_test[i]
# Curve interpolation
rri_interp_signal = splev(tm, splrep(rri_tm, scaler(rri_signal), k=3), ext=1)
ampl_interp_signal = splev(tm, splrep(ampl_tm, scaler(ampl_siganl), k=3), ext=1)
x_test.append([rri_interp_signal, ampl_interp_signal])
x_test = np.array(x_test, dtype="float32").transpose((0, 2, 1))
y_test = np.array(y_test, dtype="float32")
return x_train, y_train, groups_train, x_test, y_test, groups_test
def create_model(input_shape, weight=1e-3):
"""Create a Modified LeNet-5 model"""
inputs = Input(shape=input_shape)
# Conv1
x = Conv1D(32, kernel_size=5, strides=2, padding="valid", activation="relu",
kernel_initializer="he_normal",
kernel_regularizer=l2(weight), bias_regularizer=l2(weight))(inputs)
x = MaxPooling1D(pool_size=3)(x)
# Conv3
x = Conv1D(64, kernel_size=5, strides=2, padding="valid", activation="relu",
kernel_initializer="he_normal",
kernel_regularizer=l2(1e-3), bias_regularizer=l2(weight))(x)
x = MaxPooling1D(pool_size=3)(x)
x = Dropout(0.8)(x) # Avoid overfitting
# FC6
x = Flatten()(x)
x = Dense(32, activation="relu")(x)
outputs = Dense(2, activation="softmax")(x)
```

```
model = Model(inputs=inputs, outputs=outputs)
return model
def lr_schedule(epoch, lr):
if epoch > 70 and \
(epoch - 1) % 10 == 0:
lr *= 0.1
print("Learning rate: ", lr)
return lr
def plot(history):
"""Plot performance curve"""
fig, axes = plt.subplots(1, 2, figsize=(10, 4))
axes[0].plot(history["loss"], "r-", history["val_loss"], "b-", linewidth=0.5)
axes[0].set_title("Loss")
axes[1].plot(history["accuracy"], "r-", history["val_accuracy"], "b-", linewidth=0.5)
axes[1].set_title("Accuracy")
fig.tight_layout()
fig.show()
x_train, y_train, groups_train, x_test, y_test, groups_test = load_data()
y_train = keras.utils.to_categorical(y_train, num_classes=2) # Convert to two categories
y_test = keras.utils.to_categorical(y_test, num_classes=2)
print("train num:", len(y_train))
print("test num:", len(y_test))
model = create_model(input_shape=x_train.shape[1:])
model.summary()
from keras.utils import plot_model
plot_model(model, "model.png") # Plot model
# model = (model, gpus=2) # Multi-gpu acceleration (optional)
model.compile(optimizer="adam", loss="categorical_crossentropy", metrics=['accuracy'])
lr_scheduler = LearningRateScheduler(lr_schedule) # Dynamic adjustment learning rate
history = model.fit(x_train, y_train, batch_size=128, epochs=100, validation_data=(x_test,
y_test),
callbacks=[lr_scheduler])
model.save(os.path.join(MODEL_DIR, "model-new.final.h5")) # Save training model
loss, accuracy = model.evaluate(x_test, y_test) # test the model
print("Test loss: ", loss)
print("Accuracy: ", accuracy)
# save prediction score
y_score = model.predict(x_test)
output = pd.DataFrame({"y_true": y_test[:, 1], "y_score": y_score[:, 1], "subject":
groups_test})
output.to_csv(os.path.join(OUT_DIR, "LeNet.csv"), index=False)
plot(history.history)
# Test with Saved/ Trained Model
import pickle
import numpy as np
import os
from keras.models import load_model
```

```
from scipy.interpolate import splev, splrep
from sklearn.metrics import confusion_matrix
base_dir = DATA_DIR
ir = 3
before = 2
after = 2
# normalize
scaler = lambda arr: (arr - np.min(arr)) / (np.max(arr) - np.min(arr))
def load_data():
tm = np.arange(0, (before + 1 + after) * 60, step=1 / float(ir))
with open(os.path.join(base_dir, "apnea-ecg.pkl"), 'rb') as f:
apnea_ecg = pickle.load(f)
x_train = []
o_train, y_train = apnea_ecg["o_train"], apnea_ecg["y_train"]
groups_train = apnea_ecg["groups_train"]
for i in range(len(o_train)):
(rri_tm, rri_signal), (ampl_tm, ampl_siganl) = o_train[i]
rri_interp_signal = splev(tm, splrep(rri_tm, scaler(rri_signal), k=3), ext=1)
ampl_interp_signal = splev(tm, splrep(ampl_tm, scaler(ampl_siganl), k=3), ext=1)
x_train.append([rri_interp_signal, ampl_interp_signal])
x_train = np.array(x_train, dtype="float32").transpose((0, 2, 1))
y_train = np.array(y_train, dtype="float32")
x_test = []
o_test, y_test = apnea_ecg["o_test"], apnea_ecg["y_test"]
groups_test = apnea_ecg["groups_test"]
for i in range(len(o_test)):
(rri_tm, rri_signal), (ampl_tm, ampl_siganl) = o_test[i]
rri_interp_signal = splev(tm, splrep(rri_tm, scaler(rri_signal), k=3), ext=1)
ampl_interp_signal = splev(tm, splrep(ampl_tm, scaler(ampl_siganl), k=3), ext=1)
x_test.append([rri_interp_signal, ampl_interp_signal])
x_test = np.array(x_test, dtype="float32").transpose((0, 2, 1))
y_test = np.array(y_test, dtype="float32")
return (x_train, y_train, groups_train), (x_test, y_test, groups_test)
(x_train, y_train, groups_train), (x_test, y_test, groups_test) = load_data()
model = load_model(os.path.join(MODEL_DIR, "model-new.final.h5"))
model.summary()
print("training:")
y_true, y_pred = y_train, np.argmax(model.predict(x_train, batch_size=1024, verbose=1),
axis=-1)
C = confusion_matrix(y_true, y_pred, labels=(1, 0))
TP, TN, FP, FN = C[0, 0], C[1, 1], C[1, 0], C[0, 1]
acc, sn, sp = 1. * (TP + TN) / (TP + TN + FP + FN), 1. * TP / (TP + FN), 1. * TN / (TN + FP)
print("acc: {}, sn: {}, sp: {}".format(acc, sn, sp))
print("testing:")
y_true, y_pred = y_test, np.argmax(model.predict(x_test, batch_size=1024, verbose=1),
axis=-
1)
```

```
C = confusion_matrix(y_true, y_pred, labels=(1, 0))
TP, TN, FP, FN = C[0, 0], C[1, 1], C[1, 0], C[0, 1]
acc, sn, sp = 1. * (TP + TN) / (TP + TN + FP + FN), 1. * TP / (TP + FN), 1. * TN / (TN + FP)
print("acc: {}, sn: {}, sp: {}".format(acc, sn, sp))

import numpy as np
import os
from sklearn.metrics import confusion_matrix, classification_report
import matplotlib.pyplot as plt
import seaborn as sns


(x_train, y_train, groups_train), (x_test, y_test, groups_test) = load_data()
model = load_model(os.path.join(MODEL_DIR, "model-new.final.h5"))
model.summary()

print("Training:")
y_true_train, y_pred_train = y_train, np.argmax(model.predict(x_train, batch_size=1024,
verbose=1), axis=-1)

print("Testing:")
y_true_test, y_pred_test = y_test, np.argmax(model.predict(x_test, batch_size=1024,
verbose=1), axis=-1)

cm_train = confusion_matrix(y_true_train, y_pred_train, labels=(1, 0))
cm_test = confusion_matrix(y_true_test, y_pred_test, labels=(1, 0))

# Visualize confusion matrices
plt.figure(figsize=(10, 5))

plt.subplot(1, 2, 1)
sns.heatmap(cm_train, annot=True, fmt='d', cmap='Blues', cbar=False)
plt.xlabel('Predicted')
plt.ylabel('True')
plt.title('Confusion Matrix - Training Data')

plt.subplot(1, 2, 2)
sns.heatmap(cm_test, annot=True, fmt='d', cmap='Blues', cbar=False)
plt.xlabel('Predicted')
plt.ylabel('True')
plt.title('Confusion Matrix - Testing Data')

plt.tight_layout()
plt.show()

# Model evaluation
print("Training Report:")
```

```
print(classification_report(y_true_train, y_pred_train))

print("Testing Report:")
print(classification_report(y_true_test, y_pred_test))
```