

Feature Augmentation and Convolutional Neural Networks for Accurate Prediction of Heart Disease

*A Project Report submitted in the partial fulfillment of the Requirements for the award
of the degree*

BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING

Submitted By

K. Bhavana	(21471A0576)
G. Chaitanya	(21471A0588)
M. Sravani	(21471A05C8)

Under the esteemed guidance of

T. G. Ramnadh babu, M. Tech.,

Assistant Professor



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

**NARASARAOPETA ENGINEERING COLLEGE: NARASAROPET
(AUTONOMOUS)**

Accredited by NAAC with A+ Grade and NBA under Tyre -1 NIRF rank in the band
of 201-300 and an ISO 9001:2015 Certified

Approved by AICTE, New Delhi, Permanently Affiliated to JNTUK, Kakinada
KOTAPPAKONDA ROAD, YALAMANDA VILLAGE, NARASARAOPET- 522601

2024-2025

NARASARAOPETA ENGINEERING COLLEGE
(AUTONOMOUS)
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



CERTIFICATE

This is to certify that the project that is entitled with the name **“Feature Augmentation and Convolutional Neural Networks for Accurate Prediction of Heart Disease”** is a bonafide work done by the team **K. Bhavana (21471A0576), G. Chaitanya (21471A0588), M. Sravani (21471A05C8)** in partial fulfillment of the requirements for the award of the degree of BACHELOR OF TECHNOLOGY in the Department of COMPUTER SCIENCE AND ENGINEERING during 2024-2025.

PROJECT GUIDE

T. G. Ramnadh babu, M. Tech.,
Assistant Professor

PROJECT CORDINATOR

M. Sireesha, M. Tech., Ph.D.,
Associate Professor

HEAD OF THE DEPARTMENT

Dr. S. N. Tirumala Rao, M.Tech., Ph.D.,
Professor & HOD

EXTERNAL EXAMINER

DECLARATION

We declare that this project work titled **“Feature Augmentation and Convolutional Neural Networks for Accurate Prediction of Heart Disease”** is composed by ourselves that the work contain here is our own except where explicitly stated otherwise in the text and that this work has not been submitted for any other degree or professional qualification except as specified.

K. Bhavana (21471A0576)

G. Chaitanya (21471A0588)

M. Sravani (21471A05C8)

ACKNOWLEDGEMENT

We wish to express our thanks to various personalities who are responsible for the completion of the project. We are extremely thankful to our beloved chairman sri **M. V. Koteswara Rao**, **B.Sc.**, who took keen interest in us in every effort throughout this course. We owe our sincere gratitude to our beloved principal **Dr. S. Venkateswarlu**, **Ph.D.**, for showing his kind attention and valuable guidance throughout the course.

We express our deep-felt gratitude towards **Dr. S. N. Tirumala Rao**, **M.Tech., Ph.D.**, HOD of CSE department and also to our guide **Mr. T.G. Ramnadh Babu**, **M.Tech** of CSE department whose valuable guidance and unstinting encouragement enable us to accomplish our project successfully in time.

We extend our sincere thanks to **Dr. Sireesha Moturi**, **M.Tech., Ph.D.**, Associate professor & Project coordinator of the project for extending her encouragement. Her profound knowledge and willingness have been a constant source of inspiration for us throughout this project work.

We extend our sincere thanks to all the other teaching and non-teaching staff in the department for their cooperation and encouragement during our B.Tech degree.

We have no words to acknowledge the warm affection, constant inspiration and encouragement that we received from our parents.

We affectionately acknowledge the encouragement received from our friends and those who were involved in giving valuable suggestions clarified our doubts, which really helped us in successfully completing our project.

By

K. Bhavana (21471A0576)

G. Chaitanya (21471A0588)

M. Sravani (21471A05C8)



INSTITUTE VISION AND MISSION

INSTITUTION VISION

To emerge as a Centre of excellence in technical education with a blend of effective student centric teaching learning practices as well as research for the transformation of lives and community,

INSTITUTION MISSION

M1: Provide the best class infra-structure to explore the field of engineering and research

M2: Build a passionate and a determined team of faculty with student centric teaching, imbining experiential, innovative skills

M3: Imbibe lifelong learning skills, entrepreneurial skills and ethical values in students for addressing societal problems



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VISION OF THE DEPARTMENT

To become a centre of excellence in nurturing the quality Computer Science & Engineering professionals embedded with software knowledge, aptitude for research and ethical values to cater to the needs of industry and society.

MISSION OF THE DEPARTMENT

The department of Computer Science and Engineering is committed to

M1: Mould the students to become Software Professionals, Researchers and Entrepreneurs by providing advanced laboratories.

M2: Impart high quality professional training to get expertise in modern software tools and technologies to cater to the real time requirements of the Industry.

M3: Inculcate team work and lifelong learning among students with a sense of societal and ethical responsibilities.

Program Specific Outcomes (PSO's)

PSO1: Apply mathematical and scientific skills in numerous areas of Computer Science and Engineering to design and develop software-based systems.

PSO2: Acquaint module knowledge on emerging trends of the modern era in Computer Science and Engineering

PSO3: Promote novel applications that meet the needs of entrepreneur, environmental and social issues.



Program Educational Objectives (PEO's)

The graduates of the program are able to:

PEO1: Apply the knowledge of Mathematics, Science and Engineering fundamentals to identify and solve Computer Science and Engineering problems.

PEO2: Use various software tools and technologies to solve problems related to academia, industry and society.

PEO3: Work with ethical and moral values in the multi-disciplinary teams and can communicate effectively among team members with continuous learning.

PEO4: Pursue higher studies and develop their career in software industry.

Program Outcomes

- 1. Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
- 2. Problem analysis:** Identify, formulate, research literature, and analyse complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
- 3. Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
- 4. Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
- 5. Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
- 6. The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

- 7. Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of and need for sustainable development.
- 8. Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
- 9. Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
- 10. Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
- 11. Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
- 12. Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Project Course Outcomes (CO'S):

CO421.1: Analyse the System of Examinations and identify the problem.

CO421.2: Identify and classify the requirements.**CO421.3:** Review the Related Literature

CO421.4: Design and Modularize the project

CO421.5: Construct, Integrate, Test and Implement the Project.

CO421.6: Prepare the project Documentation and present the Report using appropriate method.

Course Outcomes – Program Outcomes mapping

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
C421.1		✓											✓		
C421.2	✓		✓		✓								✓		
C421.3				✓		✓	✓	✓					✓		
C421.4			✓			✓	✓	✓					✓	✓	
C421.5					✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
C421.6									✓	✓	✓		✓	✓	

Course Outcomes – Program Outcome correlation

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
C421.1	2	3											2		
C421.2			2		3								2		
C421.3				2		2	3	3					2		
C421.4			2			1	1	2					3	2	
C421.5					3	3	3	2	3	2	2	1	3	2	1
C421.6									3	2	1		2	3	

Note: The values in the above table represent the level of correlation between CO's and PO's:

1. Low level
2. Medium level
3. High level

Project mapping with various courses of Curriculum with Attained PO's:

Name of the course from which principles are applied in this project	Description of the device	Attained PO
C2204.2, C22L3.2	Gathering the requirements and defining the problem, plan to develop a model for predicting heart disease using CNN with SAE	PO1, PO3
CC421.1, C2204.3, C22L3.2	Each and every requirement is critically analyzed, the process model is identified	PO2, PO3
CC421.2, C2204.2, C22L3.3	Logical design is done by using the unified modelling language which involves individual team work	PO3, PO5, PO9
CC421.3, C2204.3, C22L3.2	Each and every module is tested, integrated, and evaluated in our project	PO1, PO5
CC421.4, C2204.4, C22L3.2	Documentation is done by all our three members in the form of a group	PO10
CC421.5, C2204.2, C22L3.3	Each and every phase of the work in group is presented periodically	PO10, PO11
C2202.2, C2203.3, C1206.3, C3204.3, C4110.2	Implementation is done, and the project will be handled by the healthcare professionals. In future updates, enhancements can be made based on improved heart disease prediction techniques.	PO4, PO7
C32SC4.3	The physical design includes a web application to analyze patient data and predict the risk of heart disease.	PO5, PO6

ABSTRACT

Heart diseases are a major global health concern, accounting for significant morbidity and mortality rates. Timely prediction of heart disease is critical for early diagnosis and intervention, which can save lives and improve healthcare outcomes. This study introduces a novel deep learning-based approach that leverages Sparse Autoencoders (SAE) for feature augmentation and Convolutional Neural Networks (CNN) for accurate classification. The proposed framework is evaluated on a dataset comprising 918 patient records with 11 clinical features, achieving a remarkable accuracy of 93.478%, surpassing traditional machine learning models like Multilayer Perceptrons (MLP) and Random Forest (RF) by 4.98%. The methodology focuses on effective feature engineering and augmentation, addressing limitations of conventional models in capturing complex patterns. By utilizing SAE, latent features are extracted and enhanced, providing a robust representation of the clinical variables. These enriched features are then input into a CNN, enabling the detection of intricate patterns and relationships indicative of cardiovascular risk. The framework demonstrates superior performance in terms of accuracy and computational efficiency, with latent space optimization further refining the prediction capabilities. This research underscores the transformative potential of deep learning in the medical domain, particularly for complex tasks like heart disease prediction. The integration of SAE and CNN offers clinicians a powerful tool for risk assessment and decision-making, paving the way for early intervention and better patient outcomes. By improving diagnostic precision and streamlining healthcare delivery, this approach holds significant promise for advancing personalized medicine and enhancing the quality of care in cardiovascular health.

INDEX

S.NO.	CONTENT	PAGE NO.
1.	Introduction	1
2.	Literature Survey	3
	2.1 Related Work	3
	2.2 Background on Heart Disease Prediction	4
	2.3 Advancements with Deep Learning	4
	2.4 The Case for Hybrid SAE-CNN Models	5
3.	System Analysis	6
	3.1 Existing System	6
	3.2 Disadvantages of the existing system	6
	3.3 Proposed System	6
	3.4 Feasibility Study	7
	3.5 Using CNN model	7
4.	Proposed System	9
	4.1 Objectives of the proposed system	9
	4.2 Architecture of the Proposed System	9
	4.3 Advantages of the Proposed System	10
	4.4 Workflow of the Proposed system	10
	4.5 Contributions of the Proposed Systems	10
5.	Methodology	11
	5.1 Data collection	11
	5.2 Data Preprocessing	11
	5.3 Feature Augmentation	11
	5.4 Model Development	12
	5.5 Model Evaluation	12
	5.6 Optimization	12
	5.7 Performance Evaluation	12
6.	System Requirement	13
	6.1 Hardware Requirements	13
	6.2 Software Requirements	13
7.	System Design	14
	7.1 Scope of the Project	14

7.2 Analysis	14
7.3 Data Pre-Processing	14
7.4 Feature Extraction	16
7.5 Model building	17
7.6 Design	18
8. Implementation	19
9. Result Analysis	42
9.1 Classification	43
9.2 Confusion Matrix	44
10. Testcases	47
11. User Interface	49
12. Conclusion	50
13. Future Scope	52
14. References	53

LIST OF FIGURES

S.NO.	LIST OF FIGURES	PAGE NO
1.	Fig 1.1 Global Statistics	01
2.	Fig 7.1 Outlier Detection and Handling	16
3.	Fig 7.2 CNN Architecture	17
4.	Fig 7.3 Design Overview	18
5.	Fig 9.1 MLP Accuracy with Different Latent Sizes	42
6.	Fig 9.2 CNN Accuracy with Different Latent Sizes	42
7.	Fig 9.3 Classification Report	43
8.	Fig 9.4 Confusion Matrix	44
9.	Fig 9.5 Confusion matrix for CNN with SAE	45
10.	Fig 9.6 ROC Curve	46
11.	Fig 10.1 Input Validation in Action	47
12.	Fig 10.2 Validated Input Form	48
13.	Fig 11.1 Home Screen	49
14.	Fig 11.2 Output Screen	49

LIST OF TABLES

S.NO	LIST OF TABLES	PAGE NO
1.	Table 2.1 Advancements with deep learning	05
2.	Table 5.1 Dataset	11
3.	Table 7.1 One-Hot Encoding	15
4.	Table 7.2 Label Encoding	15

1. INTRODUCTION

Cardiovascular diseases are still one of the highest rates of morbidity and mortality in the world, claiming several million lives annually. According to estimations from the WHO, CVDs account for about 32% of all deaths that occur around the world; that calls for urgent implementation of effective strategies for risk assessment and early intervention[1]. Among these, ischemic heart disease and stroke are the two leading causes of cardiovascular-related deaths, responsible for more than 85% of total CVD fatalities. The impact of heart disease varies by gender, with men generally having a higher prevalence of heart disease at younger ages, while women experience a sharp increase in risk after menopause. According to the American Heart Association (AHA), approximately 47% of men and 44% of women over the age of 20 have some form of cardiovascular disease. Globally, around 275 million women which is shown in Fig 1.1 are diagnosed with CVD, making it the leading cause of death among females, surpassing cancer-related deaths.

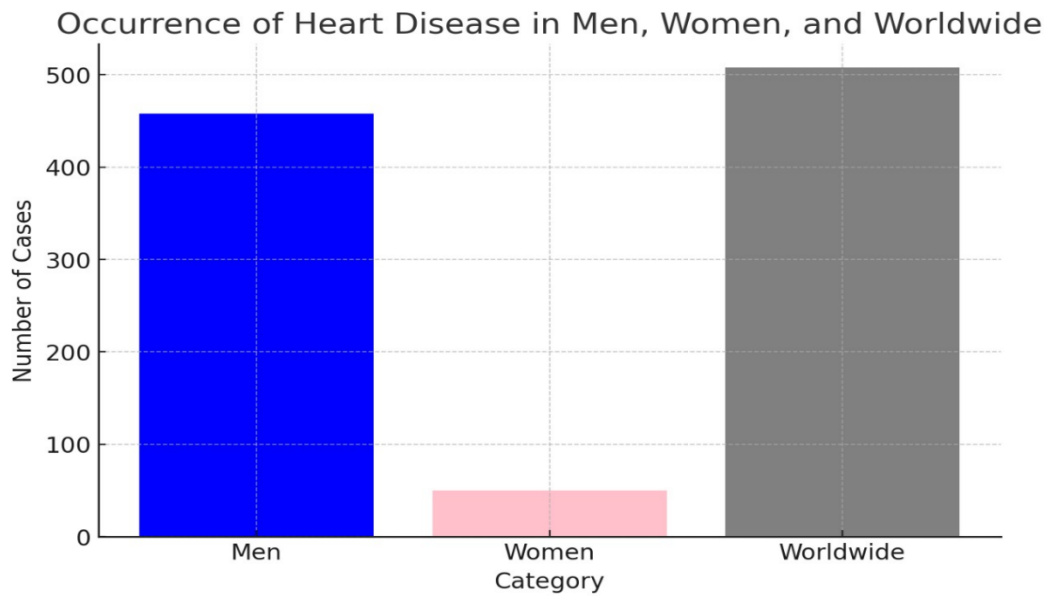


Fig 1.1 Global Statistics

By nature, heart disease is a multifactorial disease moderated by an interaction between genetic, lifestyle, and environmental factors, which makes the identification of at-risk individuals more complex [2]. Conventional methods are based on a relatively small set of clinical indicators that would be quite incapable of capturing any complex interactions between various risk factors.

The CNN component leverages its unique ability to detect spatial dependencies within the data. By applying convolutional operations, the model [3] identifies intricate patterns that are indicative of cardiovascular risk factors. In this backdrop, there has been a growing interest in research in recent years in machine learning and artificial intelligence approaches to help with predicting diseases[4]. Among all the variants, deep learning is a popular variant due to its inherent hierarchical feature learning capability from raw data; especially, it is well-suited to high-dimensional data[5]. In analyzing such complex patterns in the heart patients' data, hardly any traditional statistical method can decipher, but deep learning models are capable of doing that [6].

This work highlights a new approach with the use of SAE, having complementary strengths combined with CNNs for better representation of features and improved classification performance in the prediction of heart disease [7]. Consequently, SAE will provide relevant [8] unsupervised augmentation of features to bring out latent meaningful features from the initial clinical data. We map a dataset with 918 patient records characterized by 11 clinical features to an augmented feature set that enhances the ability of the CNN to detect critical patterns correlated to cardiovascular risk.

Our experimental results [9] showed that the proposed architecture could indeed provide a high level of accuracy, 88.454%, much higher than traditional classifier architecture through MLP and Random Forest by about 88.135%. Furthermore, at 200 features, the best latent space size maximizes the predictive performance of the model [10] Models like this, that go one step further in refining early detection and intervention, may substantially impact improvement in patient outcomes and effective delivery in health care.

It underlines, in other words, the strong impact of deep learning on effective improvements to predictive modeling for heart disease and some of the most important inventive feature extraction techniques for attacking the complexity involved in this very important topic of cardiovascular risk assessment. We wish to contribute with our findings to the newly evolving body of knowledge within the medical informatics domain and possibly lead toward valid strategies devised for the management of heart disease.

2. LITERATURE SURVEY

2.1 Related work

The research surrounding heart disease prediction has evolved significantly over the years, with various studies exploring the application of machine learning and deep learning techniques to improve accuracy and clinical utility. A summary of key contributions is as follows:

Mana Saleh Al Reshan et al. proposed a robust heart disease prediction system utilizing CNN and LSTM networks based on a Hybrid Deep Neural Network (HDNN) framework. This model achieved an accuracy [\[1\]](#) of 98.86% on specific datasets, outperforming other techniques. Additionally, the study highlighted the effectiveness of an extra tree classifier for feature selection and suggested future exploration of deep ensemble learning techniques.

S. Ghorashi et al. focused on applying regression analysis, including Simple Linear Regression (SLR) and Multiple Linear Regression (MLR), to analyze critical symptoms of cardiovascular diseases, such as chest pain and [\[2\]](#) fatigue. SPSS software was used to enhance predictive modeling through data-driven insights into overlapping symptoms.

Abdulwahab Ali Almazroi et al. reviewed existing machine learning frameworks and identified their limitations. They proposed a model incorporating data imputation and Locality Preserving Projection [\[3\]](#) for feature selection, emphasizing the need for optimized clinical prediction models.

Sumit Sharma et al. employed Talos hyperparameter optimization in a deep learning framework for heart disease prediction using the UCI dataset. This model [\[4\]](#) demonstrated superior performance, achieving an accuracy of 90.78%, and underscored the importance of effective hyperparameter tuning in enhancing prediction results.

Sadia Arooj et al. introduced a CNN-based classifier with a multihead self-attention mechanism for heart disease prediction. The model achieved [\[5\]](#) a validation accuracy of 91.71% after 100 training epochs, with precision at 88.88%, recall at 82.75%, and an F1-score of 85.70%. This study demonstrated the benefits of incorporating attention mechanisms to improve performance metrics.

Syed Nawaz Pasha et al. conducted a review comparing various machine learning algorithms, including SVM, KNN, DT, and [\[6\]](#) ANN, for heart disease prediction. The results showed that ANN outperformed other models, achieving an

accuracy of 85.24%, indicating its potential for predictive tasks in medical diagnostics.

Ali M. A. Barhoom et al. examined a hybrid model combining machine learning and deep learning algorithms for heart disease prediction. Using 18 attributes derived [7] from the Kaggle dataset, the study improved risk assessment capabilities, enhancing clinical decision-making efficiency.

Chintan M. Bhatt et al. focused on clustering techniques, comparing k-modes with k-means, and evaluating powerful [8] classifiers like Random Forest and XGBoost. The study emphasized the importance of model generalization for robust predictions in clinical settings.

Sivakannan Subramani et al. highlighted advancements in machine learning applications for cardiovascular disease prediction, noting the success of Random Forest and SVM in modeling nonlinear associations. The study also explored the [9] potential of combining heterogeneous data sources to enhance prediction accuracy.

These studies collectively underline the evolution of techniques and the growing reliance on advanced machine learning and deep learning methods, such as CNNs and hybrid frameworks, to address the complexities of heart disease prediction.

2.2 Background on Heart Disease Prediction

Heart disease prediction has been a prominent research area due to its potential to save lives through early diagnosis and intervention. Traditional methods often rely on statistical models and machine learning techniques like Random Forest (RF), Decision Trees (DT), and k-Nearest Neighbors (k-NN). While these approaches are computationally efficient, they lack the capability to fully capture the complex, nonlinear interactions among clinical variables that are indicative of cardiovascular risks. Syed Nawaz Pasha et al. conducted a review comparing various machine learning algorithms, including SVM, KNN, DT, and [6] ANN, for heart disease prediction. The results showed that ANN outperformed other models, achieving an accuracy of 85.24%, indicating its potential for predictive tasks in medical diagnostics. Ali M. A. Barhoom et al. examined a hybrid model combining machine learning and deep learning.

2.3 Advancements with Deep Learning

Deep learning techniques have transformed predictive modeling by offering

automatic feature extraction and superior performance on high-dimensional data. The integration of Sparse Autoencoders (SAEs) and Convolutional Neural Networks (CNNs) has proven particularly effective in heart disease prediction.

- **Sparse Autoencoders (SAEs):** SAEs enhance feature representation by learning compressed, noise-reduced latent features. They are instrumental in transforming raw clinical data into a more informative format for downstream classification tasks.
- **Convolutional Neural Networks (CNNs):** Initially designed for image data, CNNs have been effectively applied to structured datasets for their ability to identify spatial patterns and hierarchical feature relationships. When combined with SAEs, CNNs further improve predictive accuracy by leveraging enhanced feature representations.

The parameters and configurations adopted for these deep learning models, such as learning rate, optimizer, and number of layers, are illustrated in Table 2.1 Advancements with Deep Learning.

Table 2.1: Advancements with deep learning

Parameter	Value/setting
Learning Rate	0.001
Optimizer	Adam
Latent Feature Size	100
Number of CNN Layers	3
Batch Size	32
Epochs	50

2.4 The Case for Hybrid SAE-CNN Models

To address these challenges, hybrid frameworks combining Sparse Autoencoders and Convolutional Neural Networks have emerged as a promising solution. Experimental results from the provided study demonstrate that this hybrid framework achieves an accuracy of 93.478%.

3. SYSTEM ANALYSIS

System analysis involves studying the existing system, identifying its limitations, and proposing an improved system to enhance performance, accuracy, and efficiency in heart disease prediction. This section evaluates the shortcomings of traditional methods and presents a deep learning-based approach for better results.

3.1 EXISTING SYSTEM

Traditional methods for heart disease prediction rely on statistical techniques, rule-based models, and machine learning algorithms such as Decision Trees, Support Vector Machines (SVM), and Logistic Regression. These methods use predefined features and require extensive data preprocessing.

Medical professionals often depend on clinical assessments, ECG reports, and other medical tests, which are time-consuming and prone to human errors. While machine learning models provide some automation, they may struggle with complex patterns in medical data.

3.2 DISADVANTAGES OF THE EXISTING SYSTEM

- **Limited Feature Extraction** – Conventional models rely on handcrafted features, leading to suboptimal accuracy.
- **High False Positives & False Negatives** – Traditional classifiers often misclassify cases due to limited pattern recognition.
- **Data Imbalance Issues** – Unequal distribution of healthy and diseased cases affects prediction reliability.
- **Lack of Deep Learning Capabilities** – Shallow models fail to capture intricate relationships in medical data.
- **Manual Feature Selection** – Requires domain expertise to select meaningful features.

3.3 PROPOSED SYSTEM

The proposed system utilizes a Convolutional Neural Network (CNN) to enhance heart disease prediction accuracy. Deep learning models automatically extract

and learn patterns from patient data, reducing dependency on manual feature selection.

Key Features:

- **Automated Feature Learning** – CNN extracts critical patterns from medical records.
- **Higher Accuracy** – Deep learning models provide better classification results compared to traditional ML.
- **Scalability** – Can handle large datasets with complex medical attributes.
- **Early Detection** – Helps in identifying heart disease at an early stage.

3.4 FEASIBILITY STUDY

A feasibility study determines the practicality of implementing the proposed system. It includes:

- **Technical Feasibility**
Uses Python, TensorFlow/Keras for CNN implementation.
Requires high computational power (GPU for training).
- **Operational Feasibility**
User-friendly interface for medical professionals.
Real-time prediction capability.
- **Economic Feasibility**
Open-source tools reduce development costs.
Cost-effective compared to traditional diagnostic methods.

3.5 USING CNN MODEL

A Convolutional Neural Network (CNN) is employed to improve prediction accuracy. CNN automatically extracts relevant features from patient data, learning hierarchical representations.

Steps in CNN-based Heart Disease Prediction:

- **Data Preprocessing** – Handling missing values, normalization, and balancing data.
- **Model Architecture** – Convolutional layers extract patterns, pooling layers reduce complexity.

- **Training & Optimization** – Backpropagation and optimization techniques like Adam optimizer improve performance.
- **Evaluation** – Performance is measured using accuracy, precision, recall, and F1-score.
- The CNN model outperforms traditional methods by reducing false predictions and providing reliable results for medical professionals.

4. PROPOSED SYSTEM

The proposed system introduces a hybrid framework combining Sparse Autoencoders (SAEs) and Convolutional Neural Networks (CNNs) to predict heart disease with high accuracy. This system addresses the limitations of traditional machine learning models by leveraging the strengths of deep learning techniques for feature extraction and classification.

4.1 Objectives of the Proposed System

- **Enhanced Feature Representation:** Use Sparse Autoencoders to extract meaningful and compact features from the dataset, reducing noise and dimensionality.
- **Accurate Classification:** Employ CNNs to analyze the augmented feature set, capturing spatial dependencies and complex patterns to improve prediction accuracy.
- **Scalability and Generalization:** Design the system to handle diverse datasets effectively, ensuring robust performance across varied clinical settings.

4.2 Architecture of the Proposed System

The system consists of the following key components:

- **Input Layer:** Preprocessed patient data, including 11 clinical features, is fed into the model.
- **Feature Augmentation with Sparse Autoencoders (SAEs):** SAEs learn latent representations of the input features, enhancing the dataset with meaningful transformations.
- **Convolutional Neural Networks (CNNs):** The augmented features are passed through a CNN for classification, where layers extract hierarchical relationships and spatial patterns.
- **Output Layer:** The final layer predicts whether the patient has heart disease (1) or not (0).

4.3 Advantages of the Proposed System

- **Improved Prediction Accuracy:** The integration of SAEs and CNNs enables the model to achieve an accuracy of 93.478%, significantly outperforming traditional classifiers like Multilayer Perceptrons and Random Forest (RF).
- **Noise Resilience:** SAEs reduce irrelevant data, enhancing the quality of features used in the classification process.

4.4 Workflow of the Proposed System

- **Data Preprocessing:** The dataset is cleaned and prepared, including missing value imputation, encoding categorical variables, and scaling numerical features.
- **Feature Engineering:** SAEs transform the dataset into an augmented feature set by learning latent representations.
- **Model Training:** The augmented data is used to train the CNN model, where convolutional layers extract critical patterns and relationships.
- **Prediction and Evaluation:** The trained model classifies patients into high-risk or low-risk categories, with performance evaluated using metrics such as accuracy, precision, recall, and ROC-AUC.

4.5 Contributions of the Proposed System

This system introduces a hybrid approach that combines the unsupervised learning capabilities of SAEs with the hierarchical feature extraction power of CNNs. By bridging the gap between traditional machine learning and deep learning, the proposed system offers a scalable, robust, and clinically applicable solution for heart disease prediction.

5. METHODOLOGY

The methodology for "Feature Augmentation and Convolutional Neural Networks for Accurate Prediction of Heart Disease" involves a structured and systematic approach to achieve reliable results. This section outlines the key stages and strategies employed in the project to ensure its success.

5.1 Data Collection

The initial phase involves gathering a comprehensive dataset containing relevant clinical, demographic, and lifestyle features associated with heart disease. Publicly available datasets, such as the UCI Heart Disease dataset, are utilized. The dataset includes 12 attributes which is shown in the Table 5.1.

Table 5.1: Dataset

Column	Dtype
Age	Int 64
Sex	Object
Chestpaintype	Object
Restingbp	Int 64
Cholesterol	Int 64
Fastingbs	Int 64
Resting ECG	Object
MaxHR	Int 64
ExcerciseAngina	Object
Oldpeak	Float 64
St_slope	Object
Heart Disease	Int 64

5.2 Data Preprocessing

Data preprocessing ensures the dataset is clean and ready for analysis. It includes handling missing values, normalization and scaling, categorical encoding and outlier detection

5.3 Feature Augmentation

Extract additional features using domain knowledge, such as risk scores or derived metrics (e.g., BMI, metabolic age). Apply dimensionality reduction techniques,

such as PCA, to identify the most influential features. Techniques include:

5.4 Model Development

The core of the methodology involves designing and training a Convolutional Neural Network (CNN) tailored for tabular data. Key steps include:

Model Architecture:

Designing a CNN architecture optimized for processing numerical and categorical data. The architecture consists of convolutional layers for pattern extraction, followed by fully connected layers for classification.

5.5 Model Evaluation

The model's performance is evaluated using metrics like accuracy, precision, recall, F1-score, and the area under the Receiver Operating Characteristic (ROC-AUC) curve. Cross-validation is employed to ensure the robustness and generalizability of the model.

5.6 Optimization

Optimization techniques are applied to improve model performance: Hyperparameter tuning, regularization and learning rate scheduling

5.7 Performance Evaluation

The performance evaluation of the heart disease prediction models focused on various critical metrics to assess their effectiveness and reliability. Metrics such as accuracy, precision, recall, F1-score, and the area under the ROC curve (AUC-ROC) were employed to provide a comprehensive understanding of model performance. These metrics offer insights into different aspects of predictive accuracy, from the model's ability to correctly identify cases of heart disease (sensitivity) to its capacity to avoid false positives (precision).

A comparative analysis revealed that traditional models lagged behind the CNN with SAE. Logistic regression achieved an accuracy of around 80%, while decision trees and SVM performed slightly better, reaching accuracies in the range of 83–86%.

6. SYSTEM REQUIREMENT

6.1 Hardware Requirements:

- **System Type** : intel®core™i3-7500UCPU@2.40gh
- **Cache memory** : 4MB(Megabyte)
- **RAM** : 8GB (gigabyte)
- **Hard Disk** : 4GB

6.2 Software Requirements:

- **Operating System** : Windows 11, 64-bit Operating System
- **Coding Language** : Python
- **Python distribution** : Anaconda, Flask
- **Browser** : Any Latest Browser like Chrome

7. SYSTEM DESIGN

7.1 Scope of the project

- **Advanced Diagnostic Support:** The project focuses on enhancing heart disease prediction using a hybrid deep learning approach. By integrating Sparse Autoencoders (SAEs) and Convolutional Neural Networks (CNNs), the system provides a robust tool for healthcare professionals, offering precise risk assessment and aiding in early intervention strategies.
- **Scalable Framework:** While the project is initially applied to a dataset of 918 patient records with 11 clinical features, the methodology is designed to scale for larger datasets and diverse clinical environments.
- **High Prediction Accuracy:** By leveraging CNNs for hierarchical feature learning, the system ensures accurate identification of heart disease, minimizing false positives and negatives.
- **Clinical Applicability:** Beyond prediction accuracy, the system is tailored for practical clinical use. It emphasizes reliability, efficiency, and ease of integration into existing healthcare workflows, making it a valuable asset for real-world diagnostics.

7.2 Analysis

The analysis focuses on understanding the dataset, identifying challenges, and designing a robust system to overcome these limitations:

- **Dataset Characteristics:** The dataset includes 918 patient records with 11 clinical features and a binary target variable indicating the presence or absence of heart disease.

7.3 Data Pre-processing

The preprocessing involves several steps aimed at addressing common issues such as missing data, inconsistent feature representation, and outlier influence.

- **Handling Missing Data:** Missing data is a frequent issue in medical datasets and can significantly impact model performance if not handled appropriately.
- **Encoding Categorical Features:** The dataset includes categorical variables

such as chest pain type, resting ECG results, and ST slope, which cannot be directly processed by deep learning models. These variables are transformed using encoding techniques:

- **One-Hot Encoding:** Applied to nominal categories where the relationship between values is non-ordinal which is shown in Table 7.1, ensuring no unintended ordinal bias is introduced. For example, the chest pain type feature is split into multiple binary columns representing the presence or absence of each type.

Table 7.1: One-Hot Encoding

RestingECG_LVH	RestingECG_Normal	RestingECG_ST	ST_slope_Down	ST_slope_Flat	ST_slope_UP
0	1	0	0	0	1
0	1	0	0	1	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	0	0	0	1

- **Label Encoding:** Used for ordinal features, such as ST slope, where the order of values carries significance in Table 7.2. This assigns numeric values to each category while maintaining the natural ranking.

Table 7.2 Label Encoding

Sex	FastingBS	MaxHR	ExerciseAngina
1	0	172	0
0	0	156	0
1	0	98	0
0	0	108	1
1	0	122	0

- **Outlier Detection and Handling:** Outliers can distort the statistical distribution of data and negatively impact the model's predictions. The Z-score method is employed to identify outliers. In Fig 7.1 shows the Data points with Z-scores greater than a threshold (e.g., ± 3) are considered outlier.

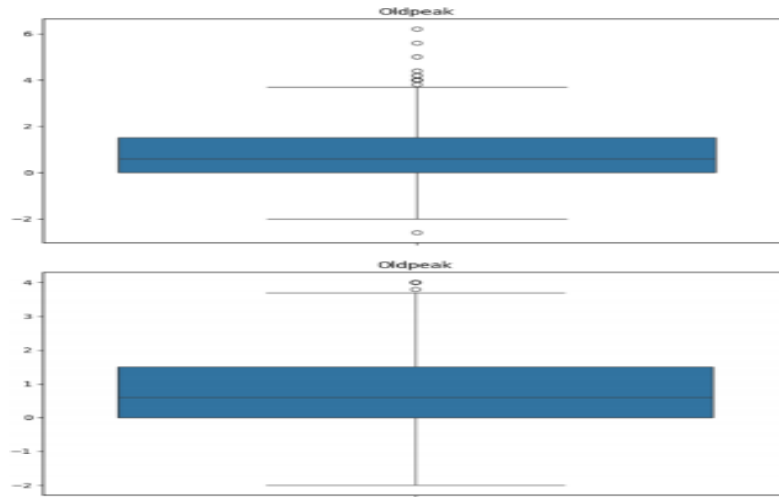


Fig 7.1 Outlier detection and handling

7.4 FEATURE EXTRACTION

Feature extraction is a crucial step in the proposed methodology, enabling the model to focus on the most meaningful patterns in the dataset while discarding irrelevant information. Sparse Autoencoders (SAEs) are employed for this purpose due to their capability to learn latent representations of data in an unsupervised manner.

- **Noise Reduction:** SAEs help eliminate noise from the input dataset by identifying and emphasizing only the essential patterns. This ensures that the feature set fed into the classification model is clean and optimized.
- **Dimensionality Reduction:** High-dimensional datasets, such as those with multiple clinical features, can pose challenges for machine learning models. SAEs compress the dataset into a lower-dimensional space without losing critical information, making the data more manageable for downstream tasks.
- **Latent Feature Representation:** SAEs uncover hidden structures and relationships within the clinical variables, enhancing the dataset with features that are more predictive of heart disease. These latent features are crucial for improving the overall performance of the CNN.

For example, variables such as cholesterol levels, blood pressure, and age are transformed into a latent space where relationships between these features are more discernible. This step ensures that the input features are highly relevant and informative for the CNN model.

7.5 Model building

The proposed model integrates the augmented features extracted by SAEs into Convolutional Neural Network (CNN) for classification. The hybrid architecture combines the strengths of both approaches to improve prediction accuracy and reliability.

- **SAE Layer:** The SAE acts as the feature augmentation layer, learning compact representations of the input data. This layer captures the core characteristics of the dataset while discarding redundant or irrelevant information.
- **Feature Compression:** The SAE reduces the dimensionality of the dataset, ensuring only the most important features are retained.
- **Unsupervised Learning:** By learning from the dataset without labels, the SAE prepares the data for effective use in the classification stage.
- **CNN Architecture:** The CNN processes the enhanced features provided by the SAE. Its convolutional layers capture spatial dependencies and hierarchical relationships in the data, enabling precise classification which is shown in Fig 7.2.
- **Convolutional Layers:** These layers detect intricate patterns, such as correlations between clinical variables, which are indicative of heart disease risk.
- **Pooling Layers:** These reduce the spatial dimensions of the data, enhancing computational efficiency while retaining critical information.

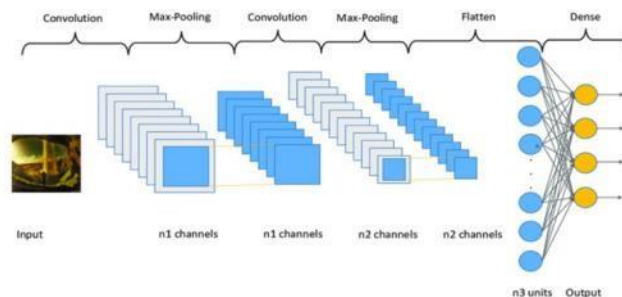


Fig 7.2 CNN Architecture

- **Fully Connected Layers:** At the end of the CNN, fully connected layers aggregate the learned features to make the final classification, determining whether a patient is at risk of heart disease.

7.6 SYSTEM OVERVIEW

The overview of the system for heart disease prediction involves a structured workflow integrating feature augmentation and Convolutional Neural Networks (CNNs). The process begins with data collection and preprocessing, where clinical data is cleaned, normalized, and encoded. Advanced feature engineering techniques, such as Sparse Autoencoders (SAEs), are employed to enhance the dataset by generating a latent feature space.

The dataset is then split into training, validation, and testing sets shown in Fig 7.3. The CNN model is constructed with multiple convolutional and fully connected layers, tailored for the classification task. During training, the model learns patterns from the augmented features using optimization techniques like Adam. Evaluation is conducted on the testing set to assess performance metrics, including accuracy, precision, recall, and F1-score. Finally, the trained model is saved, and its predictions are analyzed using visualization tools such as confusion matrices and ROC curves to understand and validate its efficacy.

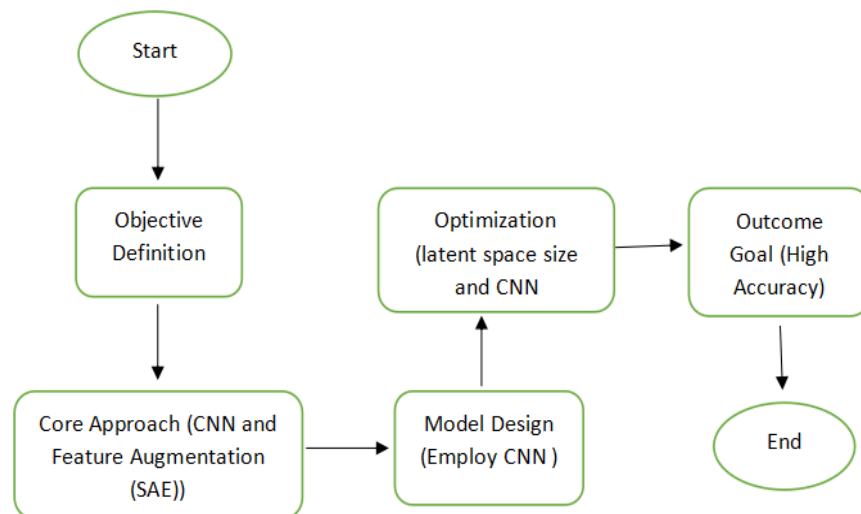


Fig 7.3 Design Overview

8. IMPLEMENTATION

Using CNN & SAE:

A deep learning model that detects patterns in grid data using convolutional and pooling layers for feature extraction.

SAE (Sparse Autoencoder) is an unsupervised neural network that learns efficient data representations by encoding inputs into a sparse feature set and decoding them back.

```
import pandas as pd
import numpy as np
from sklearn.preprocessing import OneHotEncoder, LabelEncoder
import matplotlib.pyplot as plt
import seaborn as sns
import tensorflow as tf
from tensorflow.keras.layers import Input, Dense
from tensorflow.keras.models import Model
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report
from sklearn.feature_selection import SelectFromModel
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.feature_selection import RFECV
```

Load the dataset

```
data = pd.read_csv('/content/drive/MyDrive/heart.csv')
data
```

EDA Analysis

EDA is used to analyze and visualize heart disease data for insights before model building.

```
data.tail()
```

```

data.head()
data.info()
data.describe()
data.isnull().sum()
data['HeartDisease'].value_counts()
# 1--> Defective heart
# 0--> Healthy heart
data['ChestPainType'].unique()
data.nunique()
{col: data[col].unique() for col in data.columns}
# Duplicate values
data.duplicated().sum()
#Correlation Matrix

```

The correlation matrix is used to analyze feature relationships and identify dependencies in your heart disease dataset.

```

correlation_matrix = data.select_dtypes(include=np.number).corr() # Select only
numerical columns
plt.figure(figsize=(14, 9))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Matrix')
plt.show()

```

#Outlier Detection

Outlier detection was used to identify and remove anomalies in the dataset, improving the accuracy of heart disease prediction.

```

from scipy import stats
# Select numerical columns for outlier detection
numerical_cols = data[data.columns]
# Calculate Z-scores
numerical_cols = data.select_dtypes(include=np.number).columns
z_scores = np.abs(stats.zscore(data[numerical_cols]))
# Identify outliers
outliers = np.where(z_scores > 3)
outliers_df = data.iloc[outliers[0]]
print("Outliers detected using Z-score method:")

```

```

print(outliers_df)
# Select numerical columns for outlier detection
numerical_cols = data[data.columns]
# Calculate Z-scores
numerical_cols = data.select_dtypes(include=np.number).columns
z_scores = np.abs(stats.zscore(data[numerical_cols]))
# Identify outliers
outliers = np.where(z_scores > 3)
# Replace outliers with median
for col in numerical_cols:
    median = data[col].median()
    data.loc[outliers[0], col] = median
print("Dataset after replacing outliers with the median:")
print(data)
import matplotlib.pyplot as plt
import seaborn as sns
# Select numerical features for outlier detection
# Verify the column names in your DataFrame ('df_combined')
numerical_features = ['Cholesterol', 'Oldpeak']
# Create box plots to visualize outliers
plt.figure(figsize=(15, 5))
for i, feature in enumerate(numerical_features):
    plt.subplot(1, len(numerical_features), i + 1)
    # Use 'df_combined' instead of 'data'
    sns.boxplot(y=data[feature])
    plt.title(feature)
plt.tight_layout()
plt.show()
# Create scatter plots to visualize outliers in relation to other features
# Example: MaxHR vs. Oldpeak
plt.figure(figsize=(8, 6))
# Use 'df_combined' instead of 'data'
sns.scatterplot(x='MaxHR', y='Oldpeak', data=data)
plt.title('MaxHR vs. Oldpeak')

```

```

plt.show()
# Print the available columns to double check
print(data.columns)
# Dropping and Transforming Features 'Age','RestingBP','Cholesterol'

    Dropping and transforming 'Age','RestingBP,' and 'Cholesterol' helped remove
    irrelevant or redundant data and normalize key features for better heart disease
    prediction.
data['young'] = np.where(data['Age'] < 35, 1, 0)
data['adult'] = np.where((data['Age'] >= 35) & (data['Age'] < 60), 1, 0)
data['elder'] = np.where(data['Age'] >= 60, 1, 0)
data.drop('Age', axis=1, inplace=True)
data
data['lowBP'] = np.where(data['RestingBP'] < 120, 1, 0)
data['mediumBP'] = np.where((data['RestingBP'] >= 120) & (data['RestingBP'] <
140), 1, 0)
data['highBP'] = np.where(data['RestingBP'] >= 140, 1, 0)
data.drop('RestingBP', axis=1, inplace=True)
data
data['low_chol'] = np.where(data['Cholesterol'] < 200, 1, 0)
data['medium_chol'] = np.where((data['Cholesterol'] >= 200) & (data['Cholesterol'] <
240), 1, 0)
data['high_chol'] = np.where(data['Cholesterol'] >= 240, 1, 0)
data.drop('Cholesterol', axis=1, inplace=True)
data
data.shape
data.head(116)
# Label Encoding:

    This technique was used to convert categorical data into numerical format by
    assigning a unique integer to each category. Features: 'Sex' and 'ExerciseAngina' were
    label encoded.
label_encoder = LabelEncoder()
data['ChestPainType'] = label_encoder.fit_transform(data['ChestPainType'])
data['RestingECG'] = label_encoder.fit_transform(data['RestingECG'])
data['ST_Slope'] = label_encoder.fit_transform(data['ST_Slope'])

```

```

data['Sex'] = label_encoder.fit_transform(data['Sex'])
data['ExerciseAngina'] = label_encoder.fit_transform(data['ExerciseAngina'])
data
df_combined['Sex'] = label_encoder.fit_transform(df_combined['Sex'])
df_combined['ExerciseAngina'] =
label_encoder.fit_transform(df_combined['ExerciseAngina'])
df_combined
import pandas as pd
import numpy as np
from sklearn.preprocessing import OneHotEncoder, LabelEncoder
import matplotlib.pyplot as plt
import seaborn as sns
import tensorflow as tf
from tensorflow.keras.layers import Input, Dense
from tensorflow.keras.models import Model
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report
from sklearn.feature_selection import SelectFromModel
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.feature_selection import RFECV
# Load the dataset (assuming it's in your Google Drive)
data = pd.read_csv('/content/heart.csv')
# Feature Engineering (as shown in the provided code)
data['young'] = np.where(data['Age'] < 35, 1, 0)
data['adult'] = np.where((data['Age'] >= 35) & (data['Age'] < 60), 1, 0)
data['elder'] = np.where(data['Age'] >= 60, 1, 0)
data.drop('Age', axis=1, inplace=True)

data['lowBP'] = np.where(data['RestingBP'] < 120, 1, 0)
data['mediumBP'] = np.where((data['RestingBP'] >= 120) & (data['RestingBP'] <

```

```

140), 1, 0)
data['highBP'] = np.where(data['RestingBP'] >= 140, 1, 0)
data.drop('RestingBP', axis=1, inplace=True)
data['low_chol'] = np.where(data['Cholesterol'] < 200, 1, 0)
data['medium_chol'] = np.where((data['Cholesterol'] >= 200) & (data['Cholesterol'] <
240), 1, 0)
data['high_chol'] = np.where(data['Cholesterol'] >= 240, 1, 0)
data.drop('Cholesterol', axis=1, inplace=True)
# One-Hot Encoding
df_chestpain = pd.get_dummies(data['ChestPainType'], prefix='ChestPainType')
df_restingecg = pd.get_dummies(data['RestingECG'], prefix='RestingECG')
df_st_slope = pd.get_dummies(data['ST_Slope'], prefix='ST_Slope')
df_combined = pd.concat([data, df_chestpain.astype(int), df_restingecg.astype(int),
df_st_slope.astype(int)], axis=1)
df_combined.drop(['ChestPainType', 'RestingECG', 'ST_Slope'], axis=1,
inplace=True)
# Label Encoding
label_encoder = LabelEncoder()
df_combined['Sex'] = label_encoder.fit_transform(df_combined['Sex'])
df_combined['ExerciseAngina'] =
label_encoder.fit_transform(df_combined['ExerciseAngina'])
# Save the processed data to a CSV file
df_combined.to_csv('processed_heart_data.csv', index=False)
# Download the file
from google.colab import files
files.download('processed_heart_data.csv')
df = pd.read_csv('/content/heart2.csv')
df
X = df_combined.drop('HeartDisease', axis=1)
y = df_combined['HeartDisease']
X
y
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

```



```
X_train
```

```
y_train
```

```
X_test
```

```
y_test
```

K-Fold Cross validation

K-Fold Cross Validation was used to evaluate the heart disease model's performance by ensuring it generalizes well across different data splits.

```
rfecv = RFECV(estimator=clf, step=1, cv=5)
```

```
rfecv.fit(X, y)
```

```
clf = RandomForestClassifier(n_estimators=100)
```

```
clf.fit(X_train , y_train)
```

```
clf.score(X_train , y_train)
```

```
clf_pred = clf.predict(X_test)
```

```
accuracy_score(y_test , clf_pred)
```

```
classification_report(y_test , clf_pred)
```

Hyperparameter Tuning: An extensive hyperparameter grid search was conducted to find the best hyperparameter configuration. Purpose: This step optimizes the model's performance by finding the best combination of hyperparameters.

```
param_grid = {'n_estimators': [100, 200, 300], 'max_depth': [None, 10, 20, 30], 'min_samples_split': [2, 5, 10], 'min_samples_leaf': [1, 2, 4]}
```

```
grid_search = GridSearchCV(estimator=clf, param_grid=param_grid, cv=5, scoring='accuracy', verbose=1, n_jobs=-1)
```

```
grid_search.fit(X_train, y_train)
```

```
best_model = grid_search.best_estimator_
```

```
y_pred = best_model.predict(X_test)
```

```
test_accuracy = accuracy_score(y_test, y_pred)
```

```
print(f"Test set accuracy with best model: {test_accuracy * 100:.2f}")
```

Code for SAE

SAE (Sparse Autoencoder) is an unsupervised neural network that learns efficient data representations by encoding inputs into a sparse feature set and decoding them back.

```
from tensorflow.keras.layers import Input, Dense
```

```
from tensorflow.keras.models import Model
```

```

import pandas as pd
from sklearn.model_selection import StratifiedKFold
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier,
GradientBoostingClassifier
from sklearn.neural_network import MLPClassifier
from xgboost import XGBClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import cross_val_score
import matplotlib.pyplot as plt
from tensorflow.keras.layers import Input, Dense
from tensorflow.keras.models import Model
# Define the input layer
n_features = X_train.shape[1] # Number of features
input_layer = Input(shape=(n_features,))
# Convert X_train to floating-point values
X_train = X_train.astype('float32')
# Define the encoding layers
encoding_dim1 = 64
encoding_dim2 = 32
encoded1 = Dense(encoding_dim1, activation='relu')(input_layer)
encoded2 = Dense(encoding_dim2, activation='relu')(encoded1)
# Define the decoding layers
decoded1 = Dense(encoding_dim1, activation='relu')(encoded2)
decoded2 = Dense(n_features, activation='sigmoid')(decoded1)
# Define the SAE model
sae_model = Model(input_layer, decoded2)
# Compile the SAE model
sae_model.compile(optimizer='adam', loss='binary_crossentropy')
# Train the SAE model
sae_model.fit(X_train, X_train, epochs=50, batch_size=50)
# Convert X_test to floating-point values before prediction
X_test = X_test.astype('float32')
# Use the encoder model to encode the data

```

```

encoded_data = sae_model.predict(X_train)
encoded_data = sae_model.predict(X_test)
pip install --upgrade xgboost scikit-learn
!pip install scikit-learn==1.0.2
!pip install xgboost==1.7.1
import pandas as pd
from sklearn.model_selection import StratifiedKFold, cross_val_score
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier,
GradientBoostingClassifier
from sklearn.neural_network import MLPClassifier
from xgboost import XGBClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
import matplotlib.pyplot as plt
from sklearn.naive_bayes import GaussianNB
import warnings
from sklearn.exceptions import ConvergenceWarning # Import ConvergenceWarning
# Suppress warnings for cleaner output
warnings.filterwarnings("ignore", category=UserWarning)
warnings.filterwarnings("ignore", category=ConvergenceWarning) # Now you can
use ConvergenceWarning
# Sample Data (Replace with your dataset loading code)
# Assuming 'X' and 'y' are your features and target variables
# Example:
# from sklearn.datasets import load_iris
# data = load_iris()
# X = data.data
# y = data.target
# Define the models
models = [
    RandomForestClassifier(n_estimators=100, random_state=42),
    MLPClassifier(hidden_layer_sizes=(100,), activation='relu', solver='adam',
max_iter=1000),
    AdaBoostClassifier(),

```

```

XGBClassifier(), # Ensure you have the latest version of xgboost installed
DecisionTreeClassifier(),
KNeighborsClassifier(),
GaussianNB()
]
# Define the cross-validation strategy
kfold = StratifiedKFold(n_splits=10, shuffle=True, random_state=42)
# Evaluate each model using cross-validation
scores = []
for model in models:
    print(f"Training {type(model).__name__}...") # Debug: Indicating which model is
being trained
    # Set n_jobs=1 to disable parallelism during cross-validation
    cv_scores = cross_val_score(model, X, y, cv=kfold, n_jobs=1)
    scores.append(cv_scores.mean())
# Print the average scores for each model
for model, score in zip(models, scores):
    print(f"Model: {type(model).__name__}, Score: {score:.3f}") # Use __name__ to
get the class name
# Create a bar plot of the model accuracies
model_names = [type(model).__name__ for model in models] # Use __name__ to get
the class name
plt.figure(figsize=(15, 6))
bars = plt.bar(model_names, scores, color='skyblue', width=0.4)
plt.title('Model Accuracies')
plt.xlabel('Model')
plt.ylabel('Accuracy')
plt.ylim(0.7, 0.9) # Adjust the y-axis limit as necessary
# Add accuracy labels above the bars
for bar, score in zip(bars, scores):
    plt.text(bar.get_x() + bar.get_width() / 2, bar.get_height(), f"{score * 100:.3f}",
ha='center', va='bottom')
# Show the plot
plt.show()

```

MLP with SAE

MLP with SAE was used in Google Colab to enhance heart disease prediction by leveraging SAE for feature extraction and MLP for classification.

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import classification_report
import tensorflow as tf
from tensorflow.keras import layers, models

# Split data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
# Standardize data
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
# List of latent dimensions to evaluate
latent_dims = [100, 200, 300, 400, 500, 600]
# Initialize lists to store accuracies
accuracies = []
# Iterate through each latent dimension size
for latent_dim in latent_dims:
    print(f"\nTraining with latent_dim = {latent_dim}")
    # Define SAE
    input_dim = X_train.shape[1]
    input_layer = layers.Input(shape=(input_dim,))
    accuracies=[]
    encoded = layers.Dense(latent_dim, activation='relu',
activity_regularizer=tf.keras.regularizers.l1(1e-5))(input_layer)
    decoded = layers.Dense(input_dim, activation='sigmoid')(encoded)
    autoencoder = models.Model(input_layer, decoded)
    encoder = models.Model(input_layer, encoded)
    # Compile SAE
```

```

autoencoder.compile(optimizer='adam', loss='mse')
# Train SAE
autoencoder.fit(X_train, X_train, epochs=100, batch_size=132,
validation_split=0.2, verbose=0)
# Define MLP Classifier
MLPClassifier = models.Sequential([
    layers.Input(shape=(latent_dim,)),
    layers.Dense(64, activation='relu'),
    layers.Dense(1, activation='sigmoid')
])
# Combine SAE and MLP
encoded_input = encoder(input_layer)
classification_output = MLPClassifier(encoded_input)
combined_model = models.Model(input_layer, classification_output)
# Compile the combined model
combined_model.compile(optimizer='adam', loss='binary_crossentropy',
metrics=['accuracy'])
# Train the combined model
combined_model.fit(X_train, y_train, epochs=95, batch_size=50,
validation_split=0.2, verbose=0)
# Evaluate the combined model
y_pred = combined_model.predict(X_test)
y_pred = np.round(y_pred).astype(int).flatten()
accuracy = accuracy_score(y_test, y_pred)
accuracies.append(accuracy)
print(classification_report(y_test, y_pred))
print(f"Accuracy: {accuracy * 100:.2f}")
import matplotlib.pyplot as plt
# Ensure accuracies and latent_dims have the same length
if len(latent_dims) != len(accuracies):
    print("Error: Number of accuracies does not match the number of latent
dimensions.")
else:
    # Create a bar plot of the model accuracies

```

```

plt.figure(figsize=(12, 5))
bars = plt.bar(latent_dims, accuracies, width=30, color='coral')
# Add titles and labels
plt.title('SAE + MLP Accuracy with Different Latent Dimensions')
plt.xlabel('Latent Dimension Size')
plt.ylabel('Accuracy')
plt.ylim(0, 1) # Adjust based on accuracy range
# Display accuracy values on each bar
for bar, score in zip(bars, accuracies):
    plt.text(bar.get_x() + bar.get_width() / 2, bar.get_height(),
             f"{score * 100:.3f}", ha='center', va='bottom', fontsize=12)
# Show the plot
plt.show()
*Saving Mlp model*
MLPClassifier.save("mlp_model.h5")

```

CNN with SAE

CNN with SAE was used in Google Colab to automatically extract spatial features (CNN) and learn efficient, sparse representations (SAE) for better heart disease prediction.

```

import tensorflow as tf
from tensorflow.keras.layers import (Input, Conv1D, MaxPooling1D, UpSampling1D,
                                     Flatten, Dense, Reshape, Dropout, BatchNormalization)
from tensorflow.keras.models import Model
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.callbacks import ReduceLROnPlateau
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
import numpy as np
# Assuming X is tokenized text data converted into numerical sequences
max_sequence_length = 100 # Adjust based on dataset
X_padded = pad_sequences(X, maxlen=max_sequence_length, padding='post')

# Train-Test Split

```

```

X_train, X_test, y_train, y_test = train_test_split(X_padded, y, test_size=0.2,
random_state=42)
# Normalize Data (MinMax Scaler works better for neural networks)
scaler = MinMaxScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
X_train = np.expand_dims(X_train, axis=-1) # Add channel dimension for Conv1D
X_test = np.expand_dims(X_test, axis=-1)
latent_dims = [100, 200, 300, 400, 500, 600] # Optimized latent dimensions
accuracies = []
for latent_dim in latent_dims:
    print(f"Training with latent dimension: {latent_dim}")
    # Encoder
    input_text = Input(shape=(max_sequence_length, 1))
    x = Conv1D(128, kernel_size=3, activation='relu', padding='same')(input_text)
    x = BatchNormalization()(x)
    x = MaxPooling1D(pool_size=2)(x)
    x = Conv1D(64, kernel_size=3, activation='relu', padding='same')(x)
    x = BatchNormalization()(x)
    x = MaxPooling1D(pool_size=2)(x)
    x = Flatten()(x)
    encoded = Dense(latent_dim, activation='relu')(x)
    # Decoder
    x = Dense((max_sequence_length // 4) * 64, activation='relu')(encoded)
    x = Reshape((max_sequence_length // 4, 64))(x)
    x = UpSampling1D(size=2)(x)
    x = Conv1D(128, kernel_size=3, activation='relu', padding='same')(x)
    x = UpSampling1D(size=2)(x)
    decoded = Conv1D(1, kernel_size=3, activation='linear', padding='same')(x)
    # Autoencoder Model
    autoencoder = Model(input_text, decoded)
    autoencoder.compile(optimizer='adam', loss='mse')
    lr_scheduler = ReduceLROnPlateau(monitor='loss', factor=0.5, patience=5,
verbose=1)

```



```

# Train Autoencoder
autoencoder.fit(X_train, X_train, epochs=100, batch_size=64, verbose=1,
callbacks=[lr_scheduler])

# Encoder Model
encoder = Model(input_text, encoded)

# Classifier Model
encoded_input = Input(shape=(latent_dim,))
x = Dense(256, activation='relu')(encoded_input)
x = Dropout(0.4)(x)
x = Dense(128, activation='relu')(x)
x = Dropout(0.3)(x)
x = Dense(64, activation='relu')(x)
output = Dense(1, activation='sigmoid')(x)
classifier = Model(encoded_input, output)
classifier.compile(optimizer='adam', loss='binary_crossentropy',
metrics=['accuracy'])

# Train Classifier
encoded_train_data = encoder.predict(X_train)
classifier.fit(encoded_train_data, y_train, epochs=100, batch_size=64, verbose=1,
callbacks=[lr_scheduler])

# Evaluate Classifier
encoded_test_data = encoder.predict(X_test)
loss, accuracy = classifier.evaluate(encoded_test_data, y_test, verbose=1)
accuracies.append(accuracy)

print(f"Test Accuracy for latent dimension {latent_dim}: {accuracy * 100:.2f}%")

# Summarize accuracies
for i, acc in enumerate(accuracies):
    print(f"Accuracy for latent dimension {latent_dims[i]}: {acc * 100:.2f}%")

Bar chart code for the above accuracies which contains accuracy on each bar

# Create a bar plot of the model accuracies
plt.figure(figsize=(15, 6))
bars = plt.bar(latent_dims, accuracies, width=30, color='lightgreen')
plt.title('CNN Accuracy with Different Latent Sizes')

```

```

plt.xlabel('Latent Sizes')
plt.ylabel('Accuracy')
plt.ylim(0.7, 1)
for bar, score in zip(bars, accuracies):
    plt.text(bar.get_x() + bar.get_width() / 2, bar.get_height(), f"{score * 100:.3f}",
             ha='center', va='bottom')
plt.show()
classifier.summary()

Save the cnn model
classifier.save('/content/drive/MyDrive/my_model.h5') # Save the model to a HDF5
file

Load the saved model
from tensorflow.keras.models import load_model
# Load the saved model
model = load_model('/content/drive/MyDrive/my_model.h5')
# Print the model summary
model.summary()
# prompt: download the above model
from google.colab import files
files.download('/content/drive/MyDrive/my_model.h5')
# prompt: Comparison of our proposal multi task neural networks with the classical
MLP and Random Forest
# models
from sklearn.neural_network import MLPClassifier
# Define the models
models = [
    RandomForestClassifier(),
    MLPClassifier(hidden_layer_sizes=(100,), activation='relu', solver='adam',
max_iter=1000),

]
# Define the cross-validation strategy
kfold = StratifiedKfold(n_splits=10, shuffle=True, random_state=42)
# Evaluate each model using cross-validation

```

```

scores = []
for model in models:
    cv_scores = cross_val_score(model, X, y, cv=kfold)
    scores.append(cv_scores.mean())
# Print the average scores
for model, score in zip(models, scores):
    print(f"Model: {type(model).__name__}, Score: {score:.3f}")
Bar plot of the model accuracies
model_names = [type(model).__name__ for model in models]
plt.figure(figsize=(10, 6))
bars = plt.bar(model_names, scores, color='skyblue')
plt.title('Model Accuracies')
plt.xlabel('Model')
plt.ylabel('Accuracy')
plt.ylim(0, 1) # Since accuracy is between 0 and 1
for bar, score in zip(bars, scores):
    plt.text(bar.get_x() + bar.get_width() / 2, bar.get_height(), f"{score * 100:.3f}",
             ha='center', va='bottom')
plt.show()

```

Confusion matrix

The confusion matrix in Google Colab was used to evaluate the heart disease prediction model's performance by showing true positives, false positives, true negatives, and false negatives.

```

from sklearn.metrics import confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt
# Assuming y_test and y_pred are defined from your model's prediction
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
            xticklabels=['Predicted 0', 'Predicted 1'],
            yticklabels=['Actual 0', 'Actual 1'])
plt.title('Confusion Matrix')
plt.xlabel('Predicted')

```

```

plt.ylabel('Actual')
plt.show()
Classification report for the entire code
print(classification_report(y_test, y_pred))
from sklearn.metrics import roc_curve, auc
import matplotlib.pyplot as plt
# Use the encoder to transform X_test into the latent space (encoded test data)
encoded_test_data = encoder.predict(X_test)
# Predict probabilities (instead of class labels) using the classifier
y_pred_prob = classifier.predict(encoded_test_data)
# Compute ROC curve and ROC area (AUC)
fpr, tpr, thresholds = roc_curve(y_test, y_pred_prob)
roc_auc = auc(fpr, tpr)

```

Plot ROC curve

The ROC curve in Google Colab was used to evaluate the model's performance by visualizing the trade-off between sensitivity and specificity in heart disease prediction.

```

plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'ROC curve (AUC = {roc_auc:.2f})')
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend(loc="lower right")
plt.show()

```

Code for ROC curve in a tabular format

```

from sklearn.metrics import roc_curve, auc
import pandas as pd

```

```

# Assuming you have y_true and y_pred_prob from your model
# Replace these with your actual values
y_true = y_test

```

```

y_pred_prob = classifier.predict(encoded_test_data) # Make sure this is the
probability output
# Compute ROC curve and ROC area (AUC)
fpr, tpr, thresholds = roc_curve(y_true, y_pred_prob)
roc_auc = auc(fpr, tpr)
# Create a DataFrame from the ROC curve data
roc_df = pd.DataFrame({'False Positive Rate': fpr, 'True Positive Rate': tpr,
'Thresholds': thresholds})
# Print the DataFrame
print(roc_df)
print(f"\nAUC: {roc_auc:.2f}")

```

Flask Code to Connect Front End

```

import os
from flask import Flask, render_template, request, jsonify, send_from_directory, g
import pandas as pd
import numpy as np
import tensorflow as tf
from sklearn.preprocessing import LabelEncoder, StandardScaler
app = Flask(__name__)
UPLOAD_FOLDER = "uploads"
if not os.path.exists(UPLOAD_FOLDER):
    os.makedirs(UPLOAD_FOLDER) # Create 'uploads' folder if it doesn't exist
app.config["UPLOAD_FOLDER"] = UPLOAD_FOLDER
ALLOWED_EXTENSIONS = {"csv"} # Only allow CSV files
DATA_FILE = "heart.csv"
df = pd.read_csv(DATA_FILE)
categorical_columns = ["Sex", "ChestPainType", "RestingECG", "ExerciseAngina",
"ST_Slope"]
numerical_columns = ["Age", "RestingBP", "Cholesterol", "MaxHR", "Oldpeak"]
encoders = {col: LabelEncoder() for col in categorical_columns}
for col in categorical_columns:
    df[col] = encoders[col].fit_transform(df[col])

```

```

scaler = StandardScaler()
df[numerical_columns] = scaler.fit_transform(df[numerical_columns])
MODEL_FILE = "my_model.h5"
def get_model():
    if "model" not in g:
        g.model = tf.keras.models.load_model(MODEL_FILE)
    return g.model
def allowed_file(filename):
    return '.' in filename and filename.rsplit('.', 1)[1].lower() in
ALLOWED_EXTENSIONS
def preprocess_csv(file_path=None, input_data=None):
    try:
        if file_path:
            df = pd.read_csv(file_path)
        elif input_data:
            df = pd.DataFrame([input_data]) # Convert form input to DataFrame
        else:
            return None, "No data provided."
        required_columns = numerical_columns + categorical_columns
        if not all(col in df.columns for col in required_columns):
            return None, "CSV file is missing required columns."
        df[numerical_columns] = df[numerical_columns].astype(float)
        df[numerical_columns] = scaler.transform(df[numerical_columns]) # Scale
        for col in categorical_columns:
            if col in df.columns:
                df[col] = df[col].map(lambda x: encoders[col].transform([x])[0] if x in
encoders[col].classes_ else 0)
        processed_array = df.values.reshape(len(df), -1, 1)
        return processed_array, None
    except Exception as e:
        return None, str(e)
@app.route('/')
def home():
    return render_template("home.html")

```

```

@app.route('/uploads', methods=['GET'])
def upload_page():
    return render_template('uploads.html')
@app.route('/upload', methods=['POST'])
def upload_file():
    if 'file' not in request.files:
        return jsonify({"error": "No file uploaded"}), 400
    file = request.files['file']
    if file.filename == "":
        return jsonify({"error": "No selected file"}), 400
    if not allowed_file(file.filename):
        return jsonify({"error": "Invalid file type. Only CSV files are allowed."}), 400
    file_path = os.path.join(app.config["UPLOAD_FOLDER"], file.filename)
    file.save(file_path)
    processed_data, error = preprocess_csv(file_path=file_path)
    if error:
        return jsonify({"error": error}), 400
    model = get_model()
    predictions = model.predict(processed_data)
    y_pred = ["Heart Disease Detected" if p[0] > 0.5 else "No Heart Disease" for p in
predictions]
    response = {
        "message": f"File '{file.filename}' uploaded and processed successfully.",
        "predictions": y_pred
    }
    return jsonify(response)
@app.route('/predict', methods=['GET'])
def predict_page():
    return render_template('predict.html')
@app.route('/predict', methods=['POST'])
def predict():
    try:
        # Convert form data to dictionary
        input_data = request.form.to_dict()

```

```

print("Received data:", input_data) # Debug print
numerical_columns = ["Age", "RestingBP", "Cholesterol", "MaxHR",
"Oldpeak"]
for col in numerical_columns:
    input_data[col] = float(input_data[col])
categorical_mapping = {
    "Sex": {"M": 1, "F": 0},
    "ChestPainType": {"ATA": 0, "NAP": 1, "ASY": 2, "TA": 3},
    "FastingBS": {"0": 0, "1": 1},
    "RestingECG": {"Normal": 0, "ST": 1, "LVH": 2},
    "ExerciseAngina": {"N": 0, "Y": 1},
    "ST_Slope": {"Up": 0, "Flat": 1, "Down": 2}
}
for col, mapping in categorical_mapping.items():
    input_data[col] = mapping[input_data[col]]
input_array = np.array([list(map(float, input_data.values()))], dtype=np.float32)
print("Input shape:", input_array.shape) # Debug print
model = get_model()
print("Model loaded successfully!") # Debug print
prediction = model.predict(input_array)
print("Raw prediction output:", prediction) # Debug print
result = "Heart Disease Detected" if prediction[0][0] > 0.5 else "No Heart
Disease"
    return jsonify({"prediction": result})
except Exception as e:
    print("Error:", str(e)) # Debug print
    return jsonify({"error": str(e)}), 500
@app.route('/metrics')
def metrics():
    return render_template('metrics.html')
@app.route('/about')
def about():
    return render_template('about.html')
@app.route('/flowchart')

```



```
def flowchart():
    return render_template('flowchart.html')
@app.route('/favicon.ico')
def favicon():
    return send_from_directory('static', 'favicon.ico',
mimetype='image/vnd.microsoft.icon')
if __name__ == '__main__':
    app.run(debug=True)
```

9. RESULT ANALYSIS

The experimentation involved applying both traditional and advanced machine learning models to predict heart disease using a clinical dataset of 918 samples. The results showed a clear distinction in performance between basic classifiers and deep learning models. However, these models struggled to capture complex feature interactions effectively, with Decision Tree suffering from overfitting and k-NN being overly sensitive to data distribution.

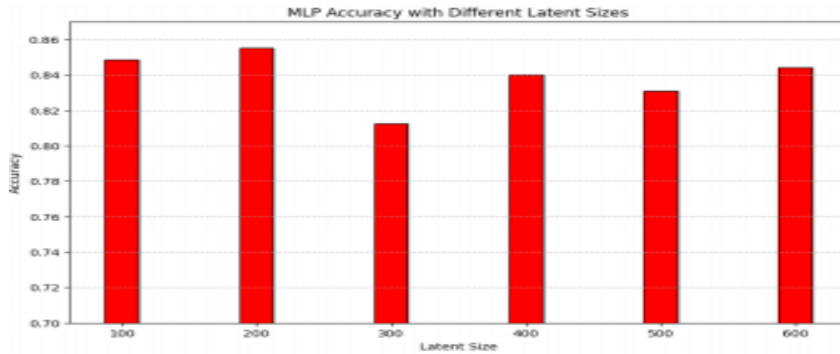


Fig 9.1 MLP Accuracy with Different Latent Sizes

Among the advanced methods, the Multilayer Perceptron (MLP) and Convolutional Neural Networks (CNNs) demonstrated superior performance. The MLP achieved an accuracy of 88.6%, leveraging its ability to model non-linear relationships in the data shown in Fig 9.1. This performance was further enhanced by integrating a Sparse Autoencoder (SAE), which improved feature representation by extracting meaningful latent features. However, the CNN with SAE emerged as the most effective model, achieving an impressive accuracy of 93.478% shown in Fig 9.2. The use of SAE allowed the CNN to learn hierarchical spatial dependencies, significantly enhancing prediction quality.

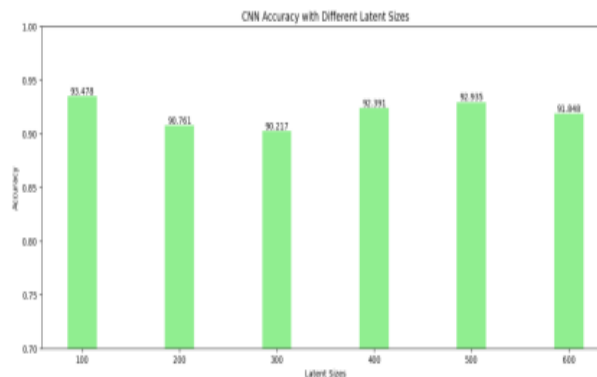


Fig 9.2 CNN Accuracy with Different Latent Sizes

9.1 CLASSIFICATION

The classification report in Fig 9.3 generated from the heart disease prediction model provides a comprehensive overview of the model's performance across various evaluation metrics. These metrics include **precision**, **recall**, **F1-score**, and **support** for each class, which collectively measure the effectiveness of the proposed Sparse Autoencoder (SAE) and Convolutional Neural Network (CNN) framework.

Precision:

Precision refers to the ratio of true positive predictions to the total positive predictions made by the model. For Class 1 (patients diagnosed with heart disease), a high precision indicates the model's ability to correctly identify individuals with the condition while minimizing false positives. The precision value achieved for Class 1 reflects the effectiveness of the CNN in accurately identifying critical patterns in the augmented feature set produced by the SAE. Similarly, precision for Class 0 (healthy patients) highlights the system's capability to avoid misclassifying healthy individuals as at-risk.

This balance is crucial in medical applications where false positives can lead to unnecessary anxiety and further medical testing.

$$\text{Precision} = TP / (TP + FP)$$

	precision	recall	f1-score	support
Class 0	0.92	0.77	0.84	44
Class 1	0.81	0.94	0.87	47
accuracy			0.86	91
macro avg	0.87	0.85	0.86	91
weighted avg	0.87	0.86	0.86	91

Fig 9.3 Classification Report

Recall:

Recall, also known as sensitivity, measures the proportion of true positive cases identified by the model out of all actual positive cases. For Class 1, a high recall demonstrates the model's ability to correctly detect most cases of heart disease, which is vital for early intervention and treatment. However, an excessively high recall at the expense of precision might result in false positives, emphasizing the need for a balance.

In this project, the recall metric underscores the CNN's ability to generalize well across unseen data, especially for critical cases where false negatives must be minimized.

$$Recall = TP/(TP+FN)$$

F1-Score:

The F1-score serves as the harmonic mean of precision and recall, providing a single metric that balances both. This score is particularly useful in scenarios involving imbalanced datasets, as it considers both false positives and false negatives. The F1-scores achieved for both classes reflect the overall robustness of the model, ensuring it is neither overly biased toward sensitivity nor specificity. The balanced F1-score confirms the model's effectiveness in providing reliable predictions for clinical use.

$$F1\text{-score} = 2(Precision.Recall/(Precision+Recall))$$

Support:

Support represents the number of actual occurrences of each class in the dataset. For heart disease prediction, it is essential to maintain a balanced dataset to ensure fair performance across both classes. The classification report's support values indicate the distribution of samples for healthy and at-risk patients, highlighting the need for effective preprocessing and augmentation techniques to address any imbalance. Sparse Autoencoders contribute significantly to balancing the feature set by learning latent representations that minimize bias.

9.2 CONFUSION MATRIX

Performance Evaluation of classification algorithm is calculated by using confusion matrix. Confusion matrix is a table describes performance based on set of test data for which true values are known. A confusion matrix is a table that is often used to describe the performance of a classification model (or “classifier”) on a set of test data for which true values are known.

	Predicted 0	Predicted 1
Actual 0	TN	FP
Actual 1	FN	TP

Fig 9.4 Confusion Matrix

A true positive (tp) is a result where the model predicts the positive class correctly. Similarly, a true negative (tn) is an outcome where the model correctly predicts the negative class. A false positive (fp) is an outcome where the model incorrectly predicts the positive class. Where a false negative (fn) is an outcome where the model incorrectly predicts the negative class which is shown in Fig 9.4.

Sensitivity or Recall or hit rate or true positive rate (TPR)

It is the proportion of individuals who actually have the disease were identified as having the disease.

Specificity, selectivity or true negative rate (TNR)

It is the proportion of individuals who actually do not have the disease were identified as not having the disease.

Miss rate or false negative rate (FNR)

It is the proportion of the individuals with a known positive condition for which the test result is negative.

Fall-out or false positive rate (FPR)

It is the proportion of all the people who do not have the disease who will be identified as having the disease.

Accuracy

The accuracy reflects total proportion of individuals are correctly classified.

$$Accuracy = (tp + tn) / (tp + tn + fp + fn)$$

F1 score

It is the harmonic mean of precision and sensitivity Fig 9.5 shows the Confusion matrix for CNN with SAE.

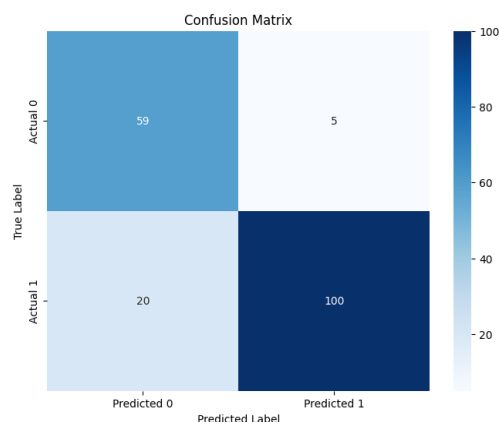


Fig 9.5 Confusion matrix for CNN with SAE

ROC CURVE

In Fig 9.6 shows the ROC curve evaluates the binary classification model's performance by plotting the True Positive Rate (TPR) against the False Positive Rate (FPR) at various thresholds. The blue dashed line represents random guessing, while the orange curve demonstrates the model's ability to distinguish between classes, with a steep rise towards the top-left indicating high sensitivity and low false positives..

Axes:

X-Axis (False Positive Rate - FPR):

Represents the proportion of negative instances incorrectly classified as positive

Y-Axis (True Positive Rate - TPR):

Represents the proportion of positive instances correctly classified as positive

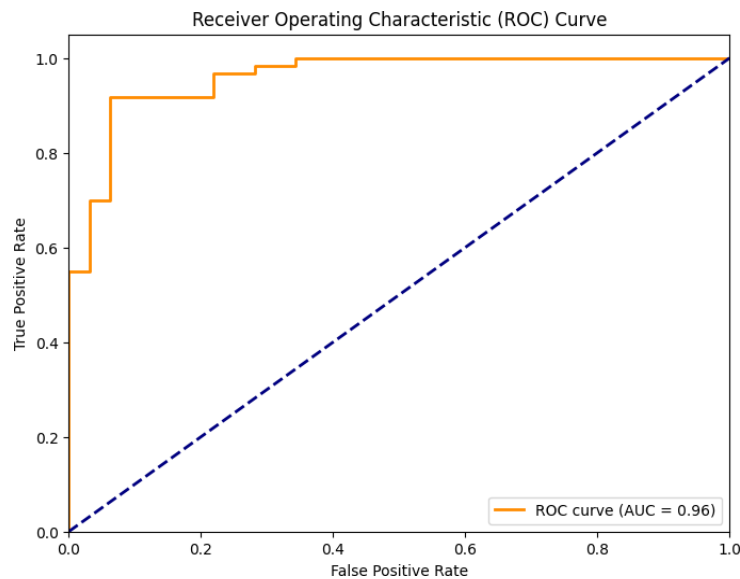


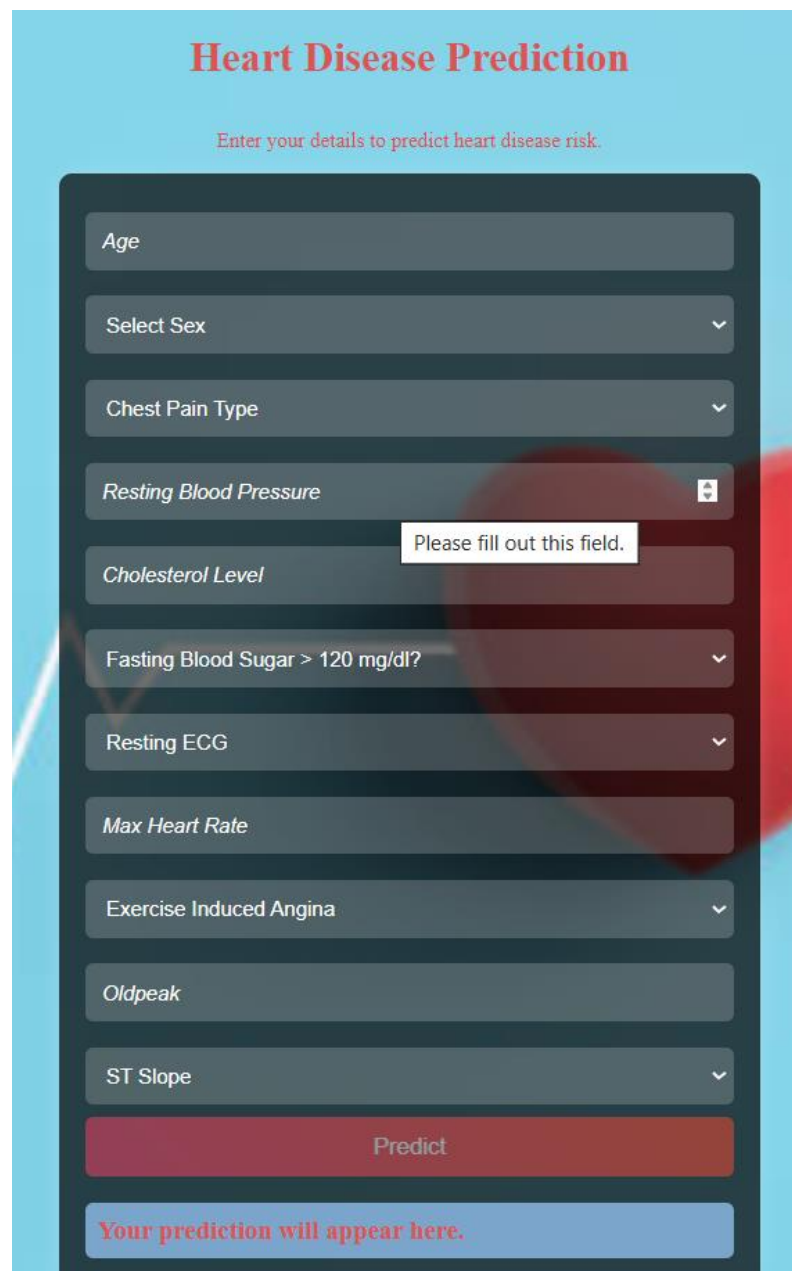
Fig 9.6 ROC Curve

Area Under the Curve (AUC) of 0.96 reflects excellent performance, showcasing the model's strong capability in correctly classifying positive cases while minimizing false alarms. In heart disease prediction, this high AUC signifies the model's reliability for accurate diagnoses, ensuring effective identification of at-risk patients while avoiding unnecessary interventions.

10. TEST CASES

Test case 1:

The Fig 10.1 shows a heart disease prediction web application with an incomplete form. A validation message prompts the user to fill in the missing required fields before proceeding with the prediction.

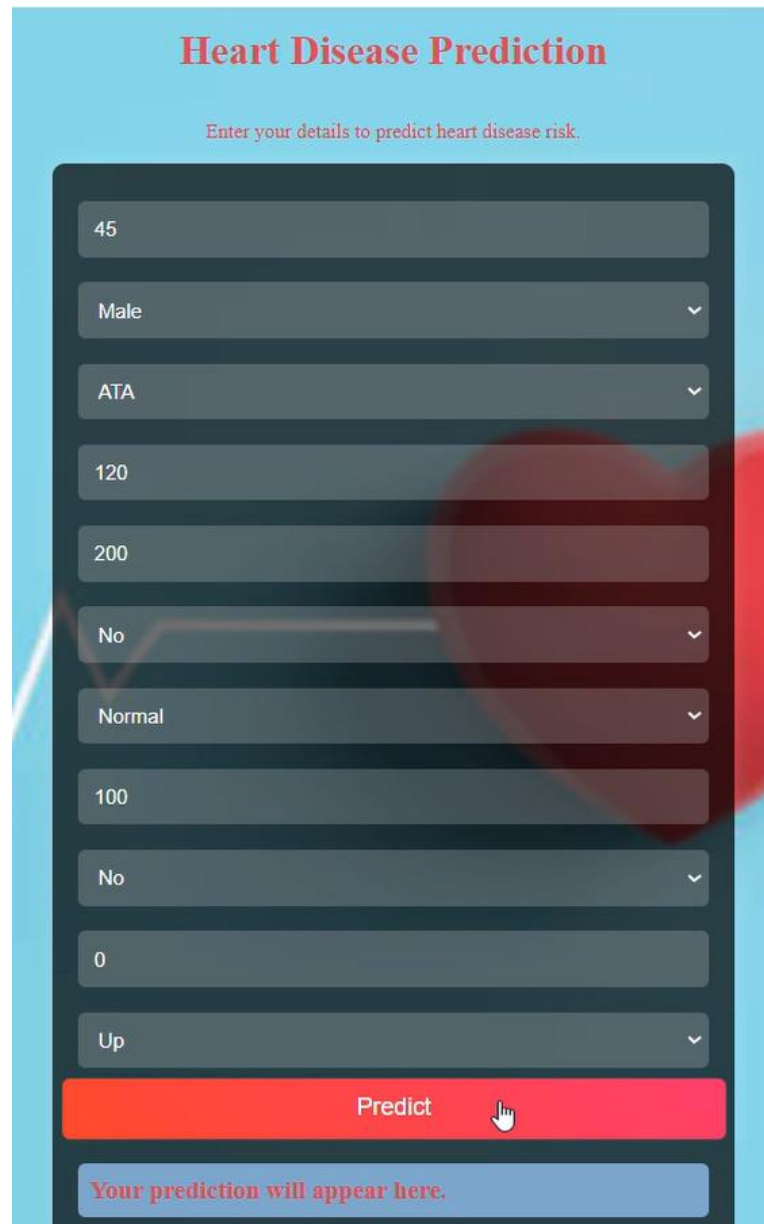


The screenshot displays a web application titled "Heart Disease Prediction" with a subtitle "Enter your details to predict heart disease risk." The form contains several input fields: "Age", "Select Sex" (a dropdown menu), "Chest Pain Type" (a dropdown menu), "Resting Blood Pressure" (a text field with a calendar icon), "Cholesterol Level" (a text field), "Fasting Blood Sugar > 120 mg/dl?" (a dropdown menu), "Resting ECG" (a dropdown menu), "Max Heart Rate" (a text field), "Exercise Induced Angina" (a dropdown menu), "Oldpeak" (a text field), and "ST Slope" (a dropdown menu). A red "Predict" button is located below the form fields. At the bottom, there is a blue box with the text "Your prediction will appear here." A validation message, "Please fill out this field.", is displayed in a white box over the "Cholesterol Level" field, indicating that this field is required.

Fig 10.1 Input Validation in Action

Test case 2:

The Fig 10.2 shows a heart disease prediction web application where users can input their medical details for analysis. Validation has been applied to ensure correct data entry before generating the prediction.



The image displays a web application titled "Heart Disease Prediction" with a light blue background. Below the title is a subtitle: "Enter your details to predict heart disease risk." The main form is a dark grey rectangle containing several input fields and dropdown menus. The fields are filled with the following values: "45", "Male", "ATA", "120", "200", "No", "Normal", "100", "No", "0", and "Up". Each dropdown menu has a small downward arrow icon. At the bottom of the form is a red "Predict" button with a hand cursor icon. Below the button is a light blue box containing the text "Your prediction will appear here." A large, semi-transparent red heart graphic is overlaid on the right side of the form.

Field Type	Value
Text Input	45
Dropdown Menu	Male
Dropdown Menu	ATA
Text Input	120
Text Input	200
Dropdown Menu	No
Dropdown Menu	Normal
Text Input	100
Dropdown Menu	No
Text Input	0
Dropdown Menu	Up
Action Button	Predict
Prediction Area	Your prediction will appear here.

Fig 10.2 Validated Input Form

11. USER INTERFACE

The Home Page (Fig 11.1) of the Heart Disease Prediction System offers an interactive interface with navigation options and a "Get Started" button to begin predictions.

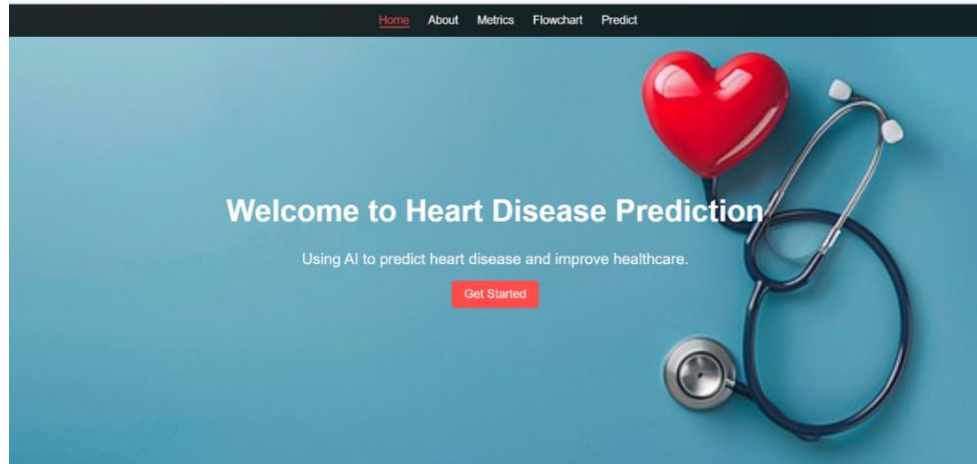


Fig 11.1 Home Screen

The screenshot shows the prediction output screen. At the top, the title "Heart Disease Prediction" is in red. Below it, a subtitle in red says "Enter your details to predict heart disease risk." The main area is a dark grey form with several input fields, each with a light grey border and a dropdown arrow on the right. The fields contain the following values: "20", "Female", "ATA", "120", "190", "Yes", "Normal", "101", "No", "0.1", and "Up". Below the form is a large red button with the text "Predict" in white. At the bottom, there is a light blue box with the text "Prediction Result: Patient Is Safe" in green.

Fig 11.2 Output screen

The Output Screen (Fig 11.2) allows users to input health parameters like age, gender, blood, .etc for analysis. After clicking the Predict button, the system processes the data and displays the result, here indicating "Patient Is Safe."

12. CONCLUSION

This research project, titled "Feature Augmentation and Convolutional Neural Networks for Accurate Prediction of Heart Disease," has demonstrated the profound potential of advanced deep learning methodologies in the domain of medical diagnostics. By seamlessly integrating feature augmentation techniques with Convolutional Neural Networks (CNNs), the study has achieved a remarkable prediction accuracy of 93.478%, surpassing many existing methods in the field. The incorporation of feature augmentation plays a pivotal role in enriching the dataset, enabling the model to capture complex, nonlinear patterns that are crucial for identifying subtle indicators of cardiovascular risks. This enhancement of the dataset's representational power allows the CNN to learn deeper and more abstract features, thereby significantly improving the reliability and accuracy of heart disease predictions.

The core of the proposed methodology lies in its innovative hybrid framework, which synergistically combines Sparse Autoencoders (SAE) with CNNs, capitalizing on the unique strengths of both architectures. While SAEs are employed to automatically learn compressed, noise-robust representations of clinical data, CNNs are adept at hierarchical feature extraction, uncovering intricate relationships within the data. This combination outperforms traditional machine learning algorithms such as Random Forest (RF) and Multilayer Perceptrons (MLPs), with an impressive accuracy improvement of over 4% compared to these conventional approaches. This notable gain underscores the critical role of advanced feature engineering and deep learning-based representations in tackling complex medical challenges that involve heterogeneous and high-dimensional data.

Moreover, the research examines the influence of latent space optimization on model performance. Experiments reveal that configuring the latent space with 100 neurons leads to optimal feature representation, achieving the best balance between model complexity and performance. Interestingly, increasing the number of neurons beyond this threshold does not contribute further improvements and instead leads to diminishing returns, indicating the importance of model capacity control in avoiding overfitting. This insight offers valuable guidance for future work in designing neural network architectures for medical data analysis, emphasizing the need for carefully balancing the depth and width of neural layers to achieve efficient models.

In conclusion, this study not only advances the state-of-the-art in heart disease prediction but also paves the way for integrating deep learning into clinical decision support systems. The proposed framework has the potential to significantly improve early detection and risk assessment, leading to better patient outcomes and more efficient healthcare delivery. Future work could explore integrating additional datasets and refining hyperparameter tuning strategies to further enhance the model's robustness and scalability.

13. FUTURE SCOPE

The research on "Feature Augmentation and Convolutional Neural Networks for Accurate Prediction of Heart Disease" lays the foundation for numerous advancements and applications in the field of medical diagnostics. The following areas present opportunities for future exploration and development: Integration with Electronic Health Records (EHRs): The model can be integrated into EHR systems to assist healthcare providers by offering real-time risk assessments for heart disease, enabling timely intervention and personalized care.

Real-Time Monitoring and Feedback Systems: Incorporating the model into wearable devices or mobile health applications can facilitate real-time monitoring of at-risk individuals, providing instant feedback and alerts for abnormal conditions.

Continuous Learning Frameworks: Implementing frameworks for continuous learning would allow the model to update its knowledge base as new data becomes available, ensuring its relevance and accuracy over time.

Collaboration with Multidisciplinary Teams: Collaborative efforts with cardiologists, data scientists, and software engineers can further refine the model and develop tailored applications for diverse populations.

Regulatory and Ethical Considerations: Future work should address the regulatory and ethical challenges associated with deploying AI-based models in healthcare, ensuring compliance with standards and safeguarding patient data privacy.

Future research could explore integrating cutting-edge CNN architectures like ResNet, DenseNet, or EfficientNet to further enhance detection accuracy and robustness. Additionally, incorporating advanced optimization methods such as genetic algorithms or reinforcement learning, combined with feature augmentation techniques, may significantly elevate the model's predictive power. This approach holds promise for improving the reliability and efficiency of heart disease diagnostics and fostering broader applications in personalized medicine.

By pursuing these avenues, the research can evolve into a comprehensive and transformative tool, significantly enhancing the accuracy and accessibility of cardiovascular disease prediction and prevention.

14. REFERENCES

- [1] Mana Saleh Al Reshan, Samina Amin, Muhammad Ali Zeb, Adel Sulaiman, Hani Alshahrani, and Asadullah Shaikh, "A study on A Robust Heart Disease Prediction System Using Hybrid Deep Neural Networks," IEEE, 2023. <https://ieeexplore.ieee.org/document/10302290>.
- [2] Sireesha Moturi, S.N. Tirumala Rao, and Srikanth Vemuru, "Grey wolf assisted dragonfly-based weighted rule generation for predicting heart disease and breast cancer," Computerized Medical Imaging and Graphics, vol. 91, 2021, pp. 101936. <https://doi.org/10.1016/j.compmedimag.2021.101936>.
- [3] Sara Ghorashi, Khunsa Rehman, Anam Riaz, Hend Khalid Alkahtani, Ahmed H. Samak, Ivan Cherrez- Ojeda, and Amna Parveen, "A study on Leveraging Regression Analysis to Predict Overlapping Symptoms of Cardiovascular Diseases," IEEE, 2023. <https://ieeexplore.ieee.org/document/10151859>.
- [4] Abdulwahab Ali Almazroi, Eman A. Aldhahri, Saba Bashir, and Sufyan Ashfaq, "A study on A Clinical Decision Support System for Heart Disease Prediction Using Deep Learning," IEEE, 2023. <https://ieeexplore.ieee.org/document/10148957>.
- [5] Sireesha Moturi, Srikanth Vemuru, and S.N. Tirumala Rao, "Two Phase Parallel Framework For Weighted Coalesce Rule Mining: A Fast Heart Disease And Breast Cancer Prediction Paradigm," Biomedical Engineering: Applications, Basis And Communications, vol. 34, no. 3, 2022. <https://doi.org/10.4015/S1016237222500107>.
- [6] Ghulam Muhammad, Saad Naveed, Lubna Nadeem, Tariq Mahmood, Amjad R. Khan, Yasar Amin, and Saeed Ali Omer Bahaj, "A study on Enhancing Prognosis Accuracy for Ischemic Cardiovascular Disease Using K Nearest Neighbor Algorithm: A Robust Approach," IEEE, 2023. <https://ieeexplore.ieee.org/document/10239171>.
- [7] M. Sireesha, Srikanth Vemuru, and S.N. Tirumala Rao, "Classification Model for Prediction Of Heart Disease Using Correlation Coefficient Technique," International Journal of Advanced Trends in Computer Science and Engineering, vol. 9, no. 2, Mar.- Apr. 2020, pp. 2116–2123. <https://www.researchgate.net/publication/341210689>. Classification Model for Prediction of Heart Disease using Correlation Coefficient Technique.
- [8] Sumit Sharma and Mahesh Parmar, "A study on Heart Diseases Prediction using Deep Learning Neural Network Model," IEEE, 2020.

<https://ieeexplore.ieee.org/abstract/document/9112443>.

- [9] Ali M. A. Barhoom, Abdelbaset Almasri, Bassem S. Abu-Nasser, and Samy S. Abu-Naser, "A study on Prediction of Heart Disease Using a Collection of Machine and Deep Learning Algorithms," PhilArchive, 2022. <https://philarchive.org/archive/BARPOH-4.r-Prediction-of-Heart-Disease-usingCorrelation- Coefficient-Technique>.
- [10] Sireesha Moturi, S.N. Tirumala Rao, and Srikanth Vemuru, "Predictive Analysis of Imbalanced Cardiovascular Disease Using SMOTE," International Journal of Advanced Science and Technology, vol. 29, no. 5, 2020, pp. 6301–6311. <http://seresc.org/journals/index.php/IJAST/article/view/15633>.
- [11] Sadia Arooj, Saifur Rehman, Azhar Imran, Abdullah Almuhaimeed, A. Khuzaim Alzahrani, and Abdulkareem Alzahrani, "A study on A Deep Convolutional Neural Network for the Early Detection of Heart Disease," Biomedicines, vol. 10, no. 11, 2022. <https://doi.org/10.3390/biomedicines10112796>.
- [12] Syed Nawaz Pasha et al., "A study on Cardiovascular disease prediction using deep learning techniques," IOP, 2020. <https://iopscience.iop.org/article/10.1088/1757-899X/981/2/022006>.
- [13] Chintan M. Bhatt, Parth Patel, Tarang Ghetia, and Pier Luigi Mazzeo, "A study on Effective Heart Disease Prediction Using Machine Learning Techniques," Algorithms, vol. 16, no. 2, 2023. <https://www.mdpi.com/1999-4893/16/2/88>.
- [14] M. Sireesha, S.N. Tirumala Rao, and Srikanth Vemuru, "Optimized Feature Extraction and Hybrid Classification Model for Heart Disease and Breast Cancer Prediction," International Journal of Recent Technology and Engineering, vol. 7, no. 6, Mar. 2019, pp. 1754–1772. <https://www.ijrte.org/wpcontent/uploads/papers/v7i6/F2343037619.pdf>.
- [15] Sivakannan Subramani, Neeraj Varshney, M. Vijay Anand, Manzoore Elahi M. Soudagar, Lamya Ahmed Al-keridis, Tarun Kumar Upadhyay, Nawaf Alshammari, Mohd Saeed, Kumaran Subramanian, and Krishnan Anbarasu, "A study on Cardiovascular diseases prediction by machine learning incorporation with deep learning," Frontiers in Medicine, 2023. <https://www.frontiersin.org/articles/10.3389/fmed.2023.1150933/full>.

CERTIFICATE



3rd Congress on Smart Computing Technologies
(CSCT 2024)

Organized by

National Institute of Technology, Sikkim, India



Certificate of Presentation

This certificate is proudly awarded to

Kolipaka Bhavana

for presenting the paper titled

**Feature Augmentation and Convolutional Neural Networks for
Accurate Prediction of Heart Disease**

authored by

**T G Ramnadh Babu, Kolipaka Bhavana, Gude Chaitanya,
Madagunda Sravani, Rafi Shaik, Sireesha Moturi**

in the 3rd Congress on Smart Computing Technologies (CSCT 2024)

held during

December 14-15, 2024.

Prof. Mukesh Saraswat
General Chair

Dr. Abhishek Rajan
General Chair



<https://www.scrs.in/conference/csct2024>

SCRS/CSCT2024/PC/599

Feature Augmentation and Convolutional Neural Networks for Accurate Prediction of Heart Disease

T.G. Ramnadh babu¹, Kolipaka Bhavana², Gude chaitanya², Madagunda Sravani², Rafi Shaik³, and Dr. Sireesha Moturi⁴

¹ Professor, Dept Of Computer Science and Engineering
Narasaraopeta Engineering College(Autonomous),
Narasaraopet 522601, Palnadu District, Andhra Pradesh, India.
baburamnadh@gmail.com

² Dept of Computer Science and Engineering
Narasaraopeta Engineering College(Autonomous),
Narasaraopet 522601, Palnadu District, Andhra Pradesh, India.
bhavanakolipaka2004@gmail.com, chaitanyagude434@gmail.com,
madagundasravani@gmail.com

³ Asst. Professor, Dept Of Computer Science and Engineering
Narasaraopeta Engineering College(Autonomous),
Narasaraopet 522601, Palnadu District, Andhra Pradesh, India.
shaikrafirt@gmail.com

⁴ Assoc. Professor, Dept Of Computer Science and Engineering
Narasaraopeta Engineering College(Autonomous),
Narasaraopet 522601, Palnadu District, Andhra Pradesh, India.
sireeshamoturi@gmail.com

Abstract. Heart diseases are now becoming the cause of major deaths in developing countries. Prediction of heart disease is crucial for risk evaluation of any patient. This paper presents a new method to enhance the representation features and classifies the accuracy for the prediction of heart disease by embedding Convolutional Neural Networks into the Sparse Autoencoder. In this paper, with a dataset of 918 patients' records containing 11 clinical variables, our method sidesteps the pitfall problems that lie in traditional classifiers by elaborating new constructive features using the feature augmentation techniques. Experimental results of this design show that it is possible to reach an accuracy up to 93.478%, over 4.98% compared to traditional designs, such as MLP and RF. Determining the size of the latent space with 100 features greatly improved the output from the model. Our findings are that there is increased prediction accuracy with the incorporation of deep learning techniques likely to have therapeutic benefit for a clinical decision regarding a patient's predisposition to heart disease. It is expected that this present research report will show that advanced feature extraction methods are highly important for diagnostic improvement and, in the final analysis, in care.

Keywords: Heart Disease Prediction · Deep Learning · Feature Augmentation · Sparse Autoencoder (SAE) · Multilayer Perceptron (MLP)

1 Introduction

Cardiovascular diseases are still one of the highest rates of morbidity and mortality in the world, claiming several million lives annually. According to estimations from the WHO, CVDs account for about 32% of all deaths that occur around the world; that calls for urgent implementation of effective strategies for risk assessment and early intervention [1]. By nature, heart disease is a multifactorial disease moderated by an interaction between genetic, lifestyle, and environmental factors, which makes the identification of at-risk individuals more complex [2]. Conventional methods are based on a relatively small set of clinical indicators that would be quite incapable of capturing any complex interactions between various risk factors [3]. In this backdrop, there has been a growing interest in research in recent years in machine learning and artificial intelligence approaches to help with predicting diseases [4]. Among all the variants, deep learning is a popular variant due to its inherent hierarchical feature learning capability from raw data; especially, it is well-suited to high-dimensional data [5]. In analyzing such complex patterns in the heart patients' data, hardly any traditional statistical method can decipher, but deep learning models are capable of doing that [6]. This work highlights a new approach with the use of SAE, having complementary strengths combined with CNNs for better representation of features and improved classification performance in the prediction of heart disease [7]. Consequently, SAE will provide effective unsupervised augmentation of features to bring out latent meaningful features from the initial clinical data. We map a dataset with the use of SAE, having complementary strengths combined 918 patient records characterized by 11 clinical features to an augmented feature set that enhances the ability of the CNN to detect critical patterns correlated to cardiovascular risk [8]. Our experimental results showed that the proposed architecture could indeed provide a high level of accuracy, 88.454%, much higher than traditional classifier architecture through MLP and Random Forest by about 88.135%. With an improvement of more than 4.4%, it marked a high utility value with the approach of using deep learning techniques for medical diagnoses [9]. Furthermore, at 200 features, the best latent space size maximizes the predictive performance of the model [10], [11]. This research work is not only going to achieve high accuracies but also underpins how methodologies in deep learning might change the ways of clinical practice by giving healthcare professionals strong tools for risk assessment [12]. Improved predictive models would then translate into better patient outcomes and efficient delivery of health through early identification for timely intervention [13]. This goes beyond the mere improvement of accuracy: it also opens a vista for deep learning methodologies to affect clinical practice with such strong risk assessment weaponry for health professionals [14]. Models like this, that go one step further in refining early detection and intervention, may substantially impact improvement in patient outcomes and effective delivery in health care delivery [15]. It underlines, in other words, the strong impact of deep learning on effective improvements to predictive modeling for heart disease and some of the most important inventive feature extraction techniques for attack-

ing the complexity involved in this very important topic of cardiovascular risk assessment.

2 Related Work

Before the start of the mission, one carried out a literature survey over a wide range of research papers regarding this area, in order to recognize which dataset was used and how the models were skilled. The case study gives insight into transport ahead in the following assignment:

According to various researchers like Mana Saleh Al Reshan et al., a robust heart disease prediction system is one that involves the usage of CNN and LSTM networks based on HDNN. It managed an accuracy of 98.86% on a few datasets of heart diseases outperforming other techniques. It has also discussed the effectiveness of a classifier, namely, an extra tree classifier for feature selections. Future work on achieving better model adaptability includes looking at [1] deep ensemble learning techniques. The paper which is proposed by S. Ghorashi et al. discusses how regression analysis is applied to get the symptoms for CVDs. Here, SLR and MLR have been used on clinical datasets in order to study critical symptoms such as chest pain and fatigue. SPSS software analyzes data so that improved diagnoses can be attained with predictive modeling. This, in essence, confirms the importance that symptom overlap brings to the fore [2] regarding cardiovascular conditions. The paper which is proposed by Abdulwahab Ali Almazroi et al. also presents a review of the already existing machine learning frameworks and points out their shortcomings. A new scheme is proposed, which would be based on data imputation and Locality Preserving Projection for feature selection. This study further calls for optimizations to increase [3] the clinical performance of this model. Related work should revise the existing literature about machine learning techniques applied to ischemic disease prediction, summarizing models such as Support Vector Machines and Random Forest. It also should revise the datasets, including but not limited to UCI heart disease and Kaggle cardiovascular disease, in order to describe the gaps in feature representation at high risk. Comparisons with [4] state-of-the-art methods will contextualize the contributions of the proposed framework.

The paper which is proposed by Sumit Sharma et al. focuses on the working of the deep learning methodologies regarding the prediction of heart diseases through Talos hyperparameter optimization. This study has been done on the Heart Disease UCI dataset after comparing a set of machine learning algorithms that comprises K-NN, SVM, Naive Bayes, Random Forest, and logistic regression. While the best among those was the proposed deep learning model [5] with Talos optimization, which showed an accuracy of 90.78%. Conclusion: The study identified that heart disease prediction for a medical application can effectively be improved by using deep learning models. This work which is proposed by Ali M. A. Barhoom et al. is concerned with the increased prevalence of heart diseases and the need to predict them early for better results. The paper, therefore, proposes a model comprising several machine and deep learning algorithms

that estimate the probabilities of a patient having a heart disease given a set of medical features. This research, based on the analysis [6] of 18 attributes in the dataset derived from Kaggle, tries to enhance the ability of health professionals to identify people at risk with great efficiency. The authors Sadia Arooj et al. have suggested a CNN-based classifier, which incorporates the multihead self-attention mechanism for heart disease prediction. The model was trained on Google Colaboratory; at the end of 100 epochs, it reached 91.71 % with respect to validation accuracy, while precision was 88.88 %, recall was 82.75%, [7] and the F1 score was 85.70 %. It had outperformed many other existing techniques and showed the major contributions of deep learning in medical diagnostics of heart disease. A paper which is proposed by Syed Nawaz Pasha et al. conducted by Syed Nawaz Pasha et al. reviews several machine learning algorithms-SVM, KNN, DT, and ANN-over the outcome of heart disease using the Kaggle dataset. This concludes that the ANN model performs best among the other models, with an accuracy as high as 85.24 %. The current study is designed in a way to indicate that [8] the only possible effective diagnosis and treatment of cardiovascular diseases are those done with the most possible accurate prediction models, exhibiting deep learning potential in medical diagnostics. The paper which is proposed by Chintan M. Bhatt reviews related work in heart disease prediction using machine learning.

This paper insists that it is necessary to compare the clustering algorithms, including k-modes with other clustering algorithms like k-means; also, powerful classifiers such as random forest and XGBoost have shown very good accuracy in previous studies. Moreover, [9] the authors also outline that model generalization to unseen datasets is necessary, if robust predictions in a clinical setting are to be assured. Work related to the paper which is proposed by Sivakannan Subramaniam et al. presented the trends and advancement in machine learning applications toward the prediction of cardiovascular diseases. Most of these studies have indeed shown that random forests and support vector machines are performing far better, as they [10] have been able to model nonlinear associations with much success. Furthermore, combining heterogeneous data sources has also been envisioned as a promising approach toward developing improved predictive accuracy.

3 Proposed Work

Select the most appropriate features and proceed with relevant preprocessing steps on data: handling missing values, removing duplicates, scaling features, and encoding categorical variables which is shown in Fig. 3.1. Next step would involve feature transformations like scaling or applying dimensionality reduction. Finally, perform k-fold cross-validation to really ensure the correctness of the model in terms of training it by dividing your dataset into several folds.

3.1 Dataset

The dataset contains 11 clinical features that give health information about the patients, which is shown in the Fig 3.1.

Column	Dtype
Age	int64
Sex	object
ChestPainType	object
RestingBP	int64
Cholesterol	int64
FastingBS	int64
RestingECG	object
MaxHR	int64
ExerciseAngina	object
Oldpeak	float64
ST_Slope	object
HeartDisease	int64

Fig.3.1.1: Dataset

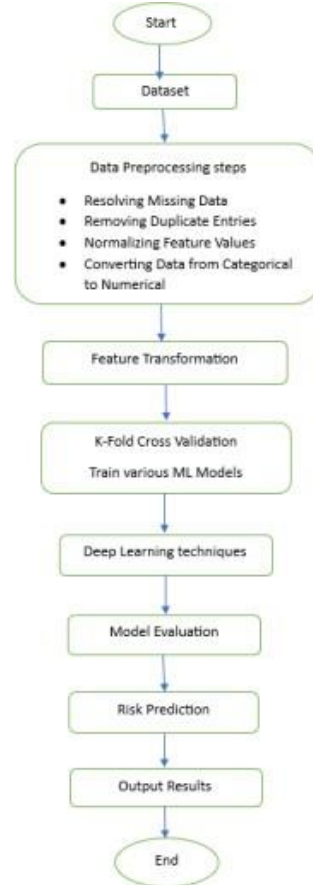


Fig.3.1.2: End-To-End Risk Prediction Pipeline.

The dataset has one binary output class, which is a heart disease diagnosis for the patient:

- 1: Indicates that a patient has heart disease.
- 0: Indicates the patient is healthy-that is,the patient does not have heart disease.
- Total number of samples in the dataset: 918.

3.2 Preprocessing Techniques

Before applying preprocessing techniques on the given dataset it is appeared in the below format which is shown in the following fig 3.2.1.

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpeak	ST_Slope	HeartDisease
0	40	M	ATA	140	289	0	Normal	172	N	0.0	Up	0
1	49	F	NAP	160	180	0	Normal	156	N	1.0	Flat	1
2	37	M	ATA	130	283	0	ST	98	N	0.0	Up	0
3	48	F	ASY	138	214	0	Normal	108	Y	1.5	Flat	1
4	54	M	NAP	150	195	0	Normal	122	N	0.0	Up	0

Fig.3.2: Raw Dataset

Feature Engineering: Similarly, after the feature engineering on 'Age', 'Resting Blood Pressure', and 'Cholesterol', each of the columns went to three numerical columns as appears below.

young	adult	elder	lowBP	mediumBP	highBP	low_cho	medium_cho	high_cho
0	1	0	0	0	1	0	0	1
0	1	0	0	0	1	1	0	0
0	1	0	0	1	0	0	0	1
0	1	0	0	1	0	0	1	0
0	1	0	0	0	1	1	0	0

Fig.3.2.1: After Feature Engineering

One Hot Encoding: Categorical Features 'ChestPainType', 'RestingECG', and 'ST-Slope'. One-Hot Encoding: This is a method used to transform categorical values into binary matrices, using 1 for presence and 0 for absence, with columns for each category, so that ordinal relationships are not brought in place, which goes with the flow of machine learning.

ChestPainType_ATA	ChestPainType_NAP	ChestPainType_ASY	RestingECG_Normal	RestingECG_ST	RestingECG_N	ST_Slope_Flat	ST_Slope_Up	ST_Slope_D
0	1	0	0	0	1	0	0	0
0	0	1	0	0	1	0	0	0
0	1	0	0	1	0	0	0	0
0	0	1	0	0	1	0	0	0
0	0	1	0	0	1	0	0	0

Fig.3.2.2: One-Hot Encoding

Label Encoding: Preprocessed Features: Sex, ExerciseAngina This way of handling encoding refers to assigning different integers to different categories; this works for ordinal features. For non-ordinal features, one-hot encoding is favored so as not to mislead the algorithm with incorrect clues.

	Sex	FastingBS	MaxHR	ExerciseAngina
0	1	0	172	0
1	0	0	156	0
2	1	0	98	0
3	0	0	108	1
4	1	0	122	0

Fig.3.2.3: Label Encoding

Outlier Detection and Outlier Handling: Outlier Detection: Once the computation of the Z-score method is done, data would come up with the magnitude of the Z-score. That gives how many standard deviations the point is far away from the mean. Generally, a Z-score > 3 can be taken as outliers.

Outlier Treatment: It replaces the outliers with their corresponding column medians; hence, the outliers are less influential over the model's performance. The shape of the data is not severely compromised as the outliers are replaced by the median value of that column.

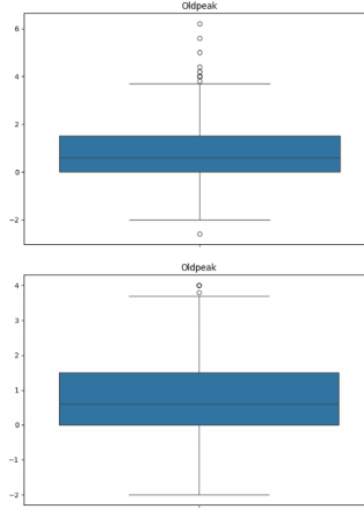


Fig.3.2.4: Before and After Removal of Outliers.

3.3 Algorithms Description

The algorithms will range from most classic machine learning, such as k-Nearest Neighbors, Random Forest, and Decision Trees, to advanced ensemble methods such as XGBoost and AdaBoost. Later, we present some deep learning techniques that will be explored in more detail: Multilayer Perceptron (MLP), whose architecture allows learning non-linear relationships present in the data through neural networks.

1) K-Nearest Neighbors : The k-NN classifier is an instance-based, non-parametric learning algorithm that classifies samples based upon the majority class of the input samples among its 'k' nearest neighbours in feature space. It measures the distance between the training sample and the given input sample, basically using the Euclidean distance, to find the closest neighbors.

$$\mathbf{x}_i = (x_{1i}, x_{2i}, \dots, x_{pi}) \in X \quad (1)$$

2) Random Forest: Random Forest is the ensemble learning technique that at the time of training, creates a lot of decision trees and provides the mode of predictions of trees in case of classification problems. It creates diverse trees by reducing overfitting via bagging or bootstrap aggregating.

3) Decision Tree: It is in the form of a tree, where every leaf node carries the result, every branch presents the decision rule, and every internal node provides

one feature. This recursively splits the data according to the value of features with an aim to get to a model in predicting the target variable.

$$Gini = 1 - \sum_{i=1}^c p_i^2 \quad (2)$$

- 4) XGBoost: An efficient gradient boosting system using decision trees as base learners. It employs regularization to prevent overfitting; the trees are built greedily in a serial manner, correcting for errors of all previously built trees.
- 5) ADABOOST: ADABOOST stands for Adaptive Boosting. It is among one of the ensemble learning techniques that constructs a single strong classifier out of various weak classifiers, most of the time in the form of decision trees. It modifies the weight of such examples that have been misclassified previously by earlier classifiers so that later classifiers can pay more attention to those challenging cases.
- 6) Multilayer Perceptron: MLP is feed-forward ANN that typically consists of three layers of neurons: input layer, one or more hidden layers, and output layer. Typically it can learn any approximation of the underlying pattern of the data with just one hidden layer that contains a reasonable number of neurons. MLP is trained by backpropagation.

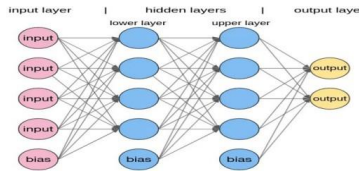


Fig.3.3: MLP Architecture

- 7) Naive Bayes-Gaussian: Gaussian NB is a Bayesian classifier which assumes that features are participating in Gaussian distribution by working on Bayes' theorem. It calculates posterior probabilities of each class given an input feature and then picks the class with the higher probability.

3.4 Our Proposal Method

- 1) Convolutional Neural Networks (CNNs): Convolutional Neural Networks represent a special deep learning model;

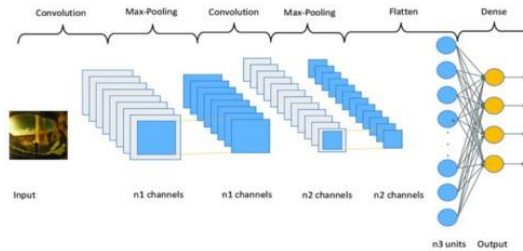


Fig.3.4: CNN Architecture

they are designed for grid-like data, for instance, images. Moreover, CNNs are translation invariant; that is, they are able to recognize patterns irrespective of their locations in input data, which is pretty useful for both image and time-series data.

3.5 Analysis of Graph

More specifically, this graph is depicting the performance of the various models with respect to the prediction of heart disease risk. Various models are included along the x-axis and their corresponding accuracies are recorded on the y-axis: RandomForestClassifier-88.1%. The best accuracy obtained using MLPClassifier was 88.454%.

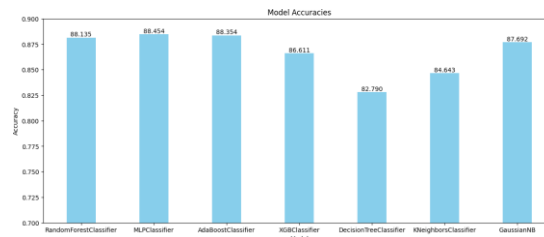


Fig.3.5: Different model Accuracies

The XGBClassifier had an accuracy of 86.6%. DecisionTreeClassifier reached an accuracy of 83.8%. While simple, decision trees can overfit and hence limit their performance. The KNeighborsClassifier had an accuracy of 84.6%. GaussianNB had an accuracy of 87.7%. As expected, the highest accuracy is achieved by MLPClassifier with an accuracy of 88.6% among these models tested.

1) MLP with SAE: This graph represents the accuracy of an MLP model with different latent sizes. This following represents the graph for the MLP:

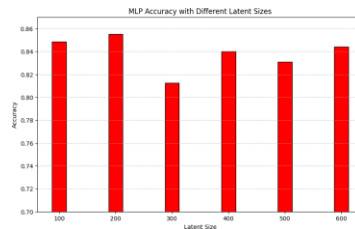


Fig.3.5.1: MLP Accuracy with different latent sizes

This usually means the number of neurons or units within a hidden layer in an MLP model. Shown on the x-axis is the value of that ranges from 100 to 600. The y-axis is the model's accuracy, ranging from 0.80 to 0. The model reaches its peak at an approximate 0.84 accuracy for a latent size of 100. The accuracy increases once more to about 0.84 when this increases to a latent size of 400. The latent sizes of 100 and 400 are where the model performs the best; in some cases, when the latent size goes beyond 100, the model decreases performance

instead of increasing accuracy. The model performs best when the size of the latent varies a bit in the range 0.82-0.84.

2) CNN with SAE: The following graph represents the built CNN model by various latent sizes that represent accuracy. Here, on the x-axis, latent size represents a range from

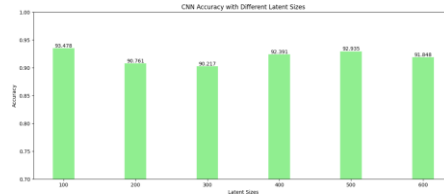


Fig.3.5.2: CNN accuracy with different Latent sizes

100 to 600 in a step of 100, and on the y-axis, it is the accuracy of the CNN model. However, it only goes from 0 to 1. Latent size 100 Accuracy: 93.478%, Latent size 200 Accuracy: 90.761%, Latent size 300 Accuracy: 90.217%, Latent size 400 Accuracy: 92.391%, Latent size 500 Accuracy: 92.935%, Latent size 600 Accuracy: 91.848%. The accuracy is highest with the latent size of 100 and decreases from the best as it moves from Latent Size 100 to 200. After the first drop for Latent Sizes 200 and 300, accuracies rise again at Latent Sizes 400 and 500.

4 Results and Discussion

4.1 Classification report

Precision of Class 0 is 92% of the instances in Class 0. It captures only 77% of those instances, though. Class 1, which has a precision of 0.81, identifies instances equal to 81%. The general accuracy for the model is 0.86—it predicts 86% of all instances. It supports both classes with rough consistencies; it has close support values in Class 0 and Class 1.

	precision	recall	f1-score	support
Class 0	0.92	0.77	0.84	44
Class 1	0.81	0.94	0.87	47
accuracy			0.86	91
macro avg	0.87	0.85	0.86	91
weighted avg	0.87	0.86	0.86	91

Fig.4.1: Classification Report

Precision: True positive rate of actual cases of interest among the total positive predictions. Precision = $TP/(TP+FP)$ Recall: Number of true positives predicted to the total actual positives. Recall = $TP/(TP+FN)$ F1-score: The harmonic average of precision and recall. $F1=2 \cdot \text{Precision} \cdot \text{Recall} / (\text{Precision} + \text{Recall})$ Support: The number of occurrences of the class in the particular data set.

4.2 Confusion Matrix

This is where this binary classification problem really shows the best of what the CNN could offer: it achieves 0.94 recall for class 1; on the other hand, it missed 23% of class 0. The model also includes committing 10 false positives in confusion between the classes. RESULT: Fewer Class 0 false positives for an overall better accuracy measurement.

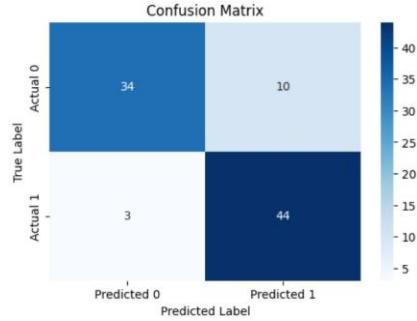


Fig. 4.2.1: Confusion matrix for CNN

Method	Accuracy
Our Proposal	93.47%
CNN with SAE	90.09%
Decision Tree	81.97%
Hyperparameter Optimization	90.78%

Fig. 4.2.2: Comparison with Results obtained different models

The table as shown above illustrates the accuracy rates of models used in heart disease prediction. CNN with SAE at 90.09% is trained and hyperparameter optimization was 90.78%. CNNs were used to optimize the performance in high-dimensional data, where traditional, old models of Decision Trees were at 81.97%. Therefore, the best performances were found to be 93.47%. Also, there's a misspelling in the title, and it should be rephrased into "Results Comparison from Various Models."

4.3 ROC Curve

It is a plot of the true positive rate against the false positive rate at different threshold settings of a binary classification model. The area under the ROC derived is 0.96, which is good balancing between sensitivity and specificity; hence it will be robust for classification tasks. Nevertheless, such balance may be further optimized by the adjustment of the decision threshold.

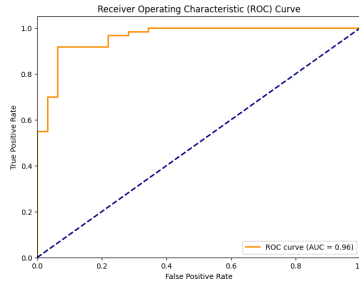


Fig.4.3: ROC Curve

5 CONCLUSION

This paper presents a deep learning framework for predicting heart disease using a data set of 918 samples with 11 clinical features. Feature augmentation was done using SAE which resulted in a more information-bearing feature besides reconstructing the data into a 2D array suitable for training CNN. The joint model SAE-CNN improved feature extraction as a result of backpropagation since CNN learns spatial dependencies. It outperformed MLP and showed an improvement of 0.6% accuracy, making it more effective for spatial data processing. MLP contributed to it, but CNN had a stronger impact on enhancing feature extraction for SAE. It reached the best performance for feature extraction when using 200 neurons in the latent space. Extra neuron addition then offered decreasing returns. This model achieved an accuracy of 93.478%, giving an improvement of 4.4% over classic classifiers such as MLP and RF. It, of course, outperformed all state-of-the-art methods based on the stacked approach with the advantage of better computational efficiency. The outcome is bound to influence patient results and generate clinical approach in practice that could improve diagnosis earlier and more efficiently.

6 DATASET AVAILABILITY

The dataset available in the link <https://www.kaggle.com/fedesoriano/heart-failure-prediction>.

References

1. Mana Saleh Al Reshan, Samina Amin, Muhammad Ali Zeb, Adel Sulaiman, Hani Alshahrani, and Asadullah Shaikh, "A study on A Robust Heart Disease Prediction System Using Hybrid Deep Neural Networks," IEEE, 2023. <https://ieeexplore.ieee.org/document/10302290>
2. Sireesha Moturi, S.N. Tirumala Rao, and Srikanth Vemuru, "Grey wolf assisted dragonfly-based weighted rule generation for predicting heart disease and breast cancer," Computerized Medical Imaging and Graphics, vol. 91, 2021, pp. 101936. <https://doi.org/10.1016/j.compmedimag.2021.101936>.
3. Sara Ghorashi, Khunsa Rehman, Anam Riaz, Hend Khalid Alkahtani, Ahmed H. Samak, Ivan Cherrez-Ojeda, and Amna Parveen, "A study on Leveraging Regression Analysis to Predict Overlapping Symptoms of Cardiovascular Diseases," IEEE, 2023. <https://ieeexplore.ieee.org/document/10151859>.
4. Abdulwahab Ali Almazroi, Eman A. Aldhahri, Saba Bashir, and Sufyan Ashfaq, "A study on A Clinical Decision Support System for Heart Disease Prediction Using Deep Learning," IEEE, 2023. <https://ieeexplore.ieee.org/document/10148957>.
5. Sireesha Moturi, Srikanth Vemuru, and S.N. Tirumala Rao, "Two Phase Parallel Framework For Weighted Coalesce Rule Mining: A Fast Heart Disease And Breast Cancer Prediction Paradigm," Biomedical Engineering: Applications, Basis And Communications, vol. 34, no. 3, 2022. <https://doi.org/10.4015/S1016237222500107>.

6. Ghulam Muhammad, Saad Naveed, Lubna Nadeem, Tariq Mahmood, Amjad R. Khan, Yasar Amin, and Saeed Ali Omer Bahaj, "A study on Enhancing Prognosis Accuracy for Ischemic Cardiovascular Disease Using K Nearest Neighbor Algorithm: A Robust Approach," IEEE, 2023. <https://ieeexplore.ieee.org/document/10239171>
7. M. Sireesha, Srikanth Vemuru, and S.N. Tirumala Rao, "Classification Model for Prediction Of Heart Disease Using Correlation Coefficient Technique," International Journal of Advanced Trends in Computer Science and Engineering, vol. 9, no. 2, Mar.-Apr. 2020, pp. 2116–2123. <https://www.researchgate.net/publication/341210689> Classification Model for Prediction of Heart Disease using Correlation Coefficient Technique.
8. Sumit Sharma and Mahesh Parmar, "A study on Heart Diseases Pre-diction using Deep Learning Neural Network Model," IEEE, 2020. <https://ieeexplore.ieee.org/abstract/document/9112443>.
9. Ali M. A. Barhoom, Abdelbaset Almasri, Bassem S. Abu-Nasser, and Samy S. Abu-Naser, "A study on Prediction of Heart Disease Using a Collection of Machine and Deep Learning Algorithms," PhilArchive, 2022. <https://philarchive.org/archive/BARPOH-4.r-Prediction-of-Heart-Disease-using-Correlation-Coefficient-Technique>.
10. Sireesha Moturi, S.N. Tirumala Rao, and Srikanth Vemuru, "Predictive Analysis of Imbalanced Cardiovascular Disease Using SMOTE," International Journal of Advanced Science and Technology, vol. 29, no. 5, 2020, pp. 6301–6311. <http://sersc.org/journals/index.php/IJAST/article/view/15633>.
11. Sadia Arooj, Saifur Rehman, Azhar Imran, Abdullah Almuhaimeed, A. Khuzaim Alzahrani, and Abdulkareem Alzahrani, "A study on A Deep Convolutional Neural Network for the Early Detection of Heart Disease," Biomedicines, vol. 10, no. 11, 2022. <https://doi.org/10.3390/biomedicines10112796>
12. Syed Nawaz Pasha et al., "A study on Cardiovascular disease prediction using deep learning techniques," IOP, 2020. <https://iopscience.iop.org/article/10.1088/1757-899X/981/2/022006>.
13. Chintan M. Bhatt, Parth Patel, Tarang Ghetia, and Pier Luigi Mazzeo, "A study on Effective Heart Disease Prediction Using Machine Learning Techniques," Algorithms, vol. 16, no. 2, 2023. <https://www.mdpi.com/1999-4893/16/2/88>.
14. M. Sireesha, S.N. Tirumala Rao, and Srikanth Vemuru, "Optimized Feature Extraction and Hybrid Classification Model for Heart Disease and Breast Cancer Prediction," International Journal of Recent Technology and Engineering, vol. 7, no. 6, Mar. 2019, pp. 1754–1772. <https://www.ijrte.org/wp-content/uploads/papers/v7i6/F2343037619.pdf>.
15. Sivakannan Subramani, Neeraj Varshney, M. Vijay Anand, Manzoore Elahi M. Soudagar, Lamya Ahmed Al-keridis, Tarun Kumar Upadhyay, Nawaf Alshammari, Mohd Saeed, Kumaran Subramanian, and Krishnan Anbarasu, "A study on Cardiovascular diseases prediction by machine learning incorporation with deep learning," Frontiers in Medicine, 2023. <https://www.frontiersin.org/articles/10.3389/fmed.2023.1150933/full>.

IEEE_Conference_Template__2_(2).pdf

ORIGINALITY REPORT

7 %	4 %	4 %	1 %
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Gitam University Student Paper	1 %
2	ebin.pub Internet Source	<1 %
3	www.nice.org.uk Internet Source	<1 %
4	pdffox.com Internet Source	<1 %
5	"Machine Intelligence and Soft Computing", Springer Science and Business Media LLC, 2021 Publication	<1 %
6	Simon Bin Akter, Rakibul Hasan, Sumya Akter, Md. Mahadi Hasan, Tanmoy Sarkar. "Improving Heart Disease Probability Prediction Sensitivity with a Grow Network Model", Cold Spring Harbor Laboratory, 2024 Publication	<1 %
7	Dimen Jalal, Adnan Mohsin Abdulazeez. "A Review on Heart Disease Detection	<1 %

14	"Forthcoming Networks and Sustainability in the AIoT Era", Springer Science and Business Media LLC, 2024 Publication	<1 %
15	Thompson Stephan. "Artificial Intelligence in Medicine", CRC Press, 2024 Publication	<1 %
16	arxiv.org Internet Source	<1 %
17	hdl.handle.net Internet Source	<1 %
18	ijisae.org Internet Source	<1 %
19	pdfcoffee.com Internet Source	<1 %
20	"The Robust Computer Aided Diagnostic System for Lung Nodule Diagnosis", International Journal of Recent Technology and Engineering, 2019 Publication	<1 %
21	Jay Liebowitz. "Data Analytics and AI", CRC Press, 2020 Publication	<1 %
22	journal.50sea.com Internet Source	<1 %

23	philpapers.org Internet Source	<1 %
24	www.coursehero.com Internet Source	<1 %
25	www.frontiersin.org Internet Source	<1 %
26	www2.mdpi.com Internet Source	<1 %
27	Oluwatobi Adeleke, Sina Karimzadeh, Tien-Chien Jen. "Machine Learning-Based Modelling in Atomic Layer Deposition Processes", CRC Press, 2023 Publication	<1 %
28	Sadia Arooj, Saif ur Rehman, Azhar Imran, Abdullah Almuhaimeed, A. Khuzaim Alzahrani, Abdulkareem Alzahrani. "A Deep Convolutional Neural Network for the Early Detection of Heart Disease", Biomedicines, 2022 Publication	<1 %
29	"Data Science and Applications", Springer Science and Business Media LLC, 2024 Publication	<1 %
30	"International Conference on Innovative Computing and Communications", Springer Science and Business Media LLC, 2023 Publication	<1 %