

Chandrakanta Mahanty

v4

 v11

 research

 GITAM University

Document Details

Submission ID

trn:oid::1:3205729804

Submission Date

Apr 5, 2025, 1:37 PM GMT+5:30

Download Date

Apr 5, 2025, 2:05 PM GMT+5:30

File Name

final_report.docx

File Size

1.7 MB

62 Pages

8,242 Words

56,478 Characters

*% detected as AI

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

Caution: Review required.

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

Disclaimer

Our AI writing assessment is designed to help educators identify text that might be prepared by a generative AI tool. Our AI writing assessment may not always be accurate (it may misidentify writing that is likely AI generated as AI generated and AI paraphrased or likely AI generated and AI paraphrased writing as only AI generated) so it should not be used as the sole basis for adverse actions against a student. It takes further scrutiny and human judgment in conjunction with an organization's application of its specific academic policies to determine whether any academic misconduct has occurred.

Frequently Asked Questions

How should I interpret Turnitin's AI writing percentage and false positives?

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

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

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

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

What does 'qualifying text' mean?

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

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



Hybrid XGBoost & LSTM for Early Parkinson's Disease Identification Using Voice Data

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

BACHELOR OF TECHNOLOGY IN COMPUTER SCIENCE AND ENGINEERING

Submitted by

**Komal Rani Kar (VU21CSEN0100389)
Reddi Rishitha (VU21CSEN0101651)
Ulsi Divya Sri Varsha (VU21CSEN0100492)
Cheerapurupalli Manjusha (VU21CSEN0101458)**

Under the esteemed guidance of
Dr. Chandrakanta Mahanty
ASSIATANT PROFESSOR



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
GITAM SCHOOL OF TECHNOLOGY
GITAM (Deemed to be University)
VISAKHAPATNAM
2025**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
GITAM SCHOOL OF TECHNOLOGY
GITAM (Deemed to be University)**



DECLARATION

I hereby declare that the project report entitled “**Hybrid XGBoost + LSTM for Early Parkinson’s Disease Identification Using Voice Data**” is an original work done in the Department of Computer Science and Engineering, GITAM School of Technology, GITAM (Deemed to be University) submitted in partial fulfillment of the requirements for the award of the degree of B.Tech. in Computer Science and Engineering. The work has not been submitted to any other college or University for the award of any degree or diploma.

Date: 26th March 2025

Registration No(s)	Name(s)	Signature
VU21CSEN0100389	Komal Rani Kar	
VU21CSEN0101651	Reddi Rishitha	
VU21CSEN0100492	Ulsi Divya Sri Varsha	
VU21CSEN0101458	Cheerpurupalli Manjusha	

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
GITAM SCHOOL OF TECHNOLOGY
GITAM (Deemed to be University)**



CERTIFICATE

This is to certify that the project report entitled **“Hybrid XGBoost + LSTM for Parkinson’s Disease Identification Using Voice Data”** is a bonafide record of work carried out by Komal Rani Kar (VU21CSEN0100389), Reddi Rishitha (VU21CSEN0101651), Ulsi Divya Sri Varsha (VU21CSEN0100492), Cheerpurupalli Manjusha (VU21CSEN0101458) students submitted in partial fulfillment of requirement for the award of degree of Bachelor of Technology in Computer Science and Engineering.

Date : 26th March 2025

Project Guide

Dr.Chandrakanta Mahanty

Head of the Department

Dr. Lakshmeeswari G

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to everyone who contributed to the development and documentation of this project. First and foremost, we extend our deepest appreciation to Dr.Chandrakanta Mahanty, for their invaluable support and guidance throughout the project lifecycle. Their vision and feedback were instrumental in shaping the direction of this project.

We are immensely thankful to our project team members for their dedication, hardwork, and collaboration. Each team member played a vital role in bringing this project to fruition, contributing their expertise and skills to overcome challenges and achieve milestones.

Finally, we extend our appreciation to our families, friends, and colleagues for their understanding, encouragement, and support throughout the project journey.

This project would not have been possible without the collective effort and collaboration of all involved parties. Thank you for your dedication and commitment.

Thank you all

TABLE OF CONTENTS

S.No.	Description	Page No.
1.	Abstract	1
2.	Introduction	2
3.	Literature Review	5
4.	Problem Identification & Objectives	7
5.	Existing System, Proposed System	9
6.	Proposed System Architecture / Methodology	14
7.	Technologies Used	19
8.	Implementation (Sample Code and Test Cases)	22
9.	Results & Discussion	26
10.	Conclusion & Future Scope	29
11.	References	31
12.	Annexure 1 (Source Code)	32
13.	Annexure 2 (Output Screens)	54
14.	Acceptance Mail from ICMIB 2025	57

1. ABSTRACT

Parkinson's Disease (PD) is a progressive neurodegenerative condition characterized by motor function, speech, and quality-of-life impairment. Early diagnosis is important for proper management and enhanced patient outcomes since early intervention is able to slow the progression of the disease. Conventional diagnosis is based on clinical evaluation and sophisticated neuroimaging, which is expensive and unavailable in remote locations. In this project, we introduce a hybrid machine learning model that utilizes Long Short-Term Memory (LSTM) networks and Extreme Gradient Boosting (XGBoost) to process vocal biomarkers, providing a non-invasive and effective alternative for the early detection of PD.

The system uses an LSTM network to learn temporal patterns from voice recordings, detecting slight variations that can indicate the presence of Parkinson's Disease. These features are then fed into XGBoost, an efficient ensemble learning algorithm, which classifies the probability of PD in accordance with learned patterns from existing datasets. Our hybrid approach leverages deep learning's strength of modeling sequential dependencies as well as the efficiency of XGBoost in processing structured data, improving prediction accuracy and reducing false positives and false negatives.

To make it usable in the real world, we have created a web application that allows users to input their vocal characteristics with ease and obtain instant predictions. The system assigns individuals to high-likelihood or low-likelihood PD groups, thus being an efficient solution for both clinical and telemedicine uses. In contrast to traditional diagnosis methods involving specific equipment, this method needs only simple voice recordings, hence can be used even in resource-constrained environments.

Our experimental findings confirm that the hybrid LSTM + XGBoost model performs better compared to individual models in accuracy, precision, recall, and F1-score. By combining AI-based healthcare solutions with a user-friendly web interface, this project offers a stable, efficient, and scalable process for the early diagnosis of Parkinson's Disease. The system not only improves the efficiency of diagnostics but also contributes to the evolving area of AI-based medical applications, highlighting the role of technology in revolutionizing healthcare.

2. INTRODUCTION

2.1 BACKGROUND

Parkinson's Disease (PD) is a chronic, progressive neurodegenerative disorder affecting movement, voice, and overall well-being. The pathophysiology of the disorder is the progressive degeneration of brain neurons that are involved in dopamine production, leading to tremor, rigidity, and bradykinesia, or slowness of movement. While the motor impairments are well described, initial symptoms also commonly present as changes in the voice, such as softer voice, monotone speech, and increased hoarseness.

As PD symptoms develop over time, it is important to have early diagnosis for intervention and better control. Conventional diagnostic tools like clinical assessments and neuroimaging techniques are costly, time-consuming, and not easily accessible in many regions. This has created interest in the creation of artificial intelligence (AI) and machine learning (ML) techniques for non-invasive and low-cost diagnostic tools.

2.2 MOTIVATION

Early detection of Parkinson's Disease is still an urgent issue, particularly in limited-resource health facilities. Most of the patients are diagnosed only when extensive neurological harm has already taken place, thereby restricting the efficacy of the treatment. The motivation behind this project stems from the necessity of having a scalable AI-based system to identify PD in its early stages from basic voice recordings.

- Voice-based analysis is an exciting new way of doing traditional diagnosis because:
- Non-Invasiveness: Does not require workup in the laboratory or imaging.
- Cost-Effectiveness: A simple voice recording, which can be purchased by more individuals.
- Early Detection Potential: Early detection with subtle voice pattern alterations prior to the development of severe motor impairment.

We employ machine learning, deep learning (LSTM), and ensemble learning (XGBoost) to target higher accuracy and reliability for the diagnosis of Parkinson's Disease.

2.3 PROBLEM STATEMENT

The present diagnostic method for Parkinson's disease (PD) relies heavily on clinical examination, which is possibly subjective and inconsistent between practitioners. Furthermore, certain tests, such as MRI and DaTscan, are costly and possibly not available everywhere in remote areas. The present project hopes to solve the following major problems:

- Creating an artificial intelligence model that is able to identify patients with Parkinson's correctly from healthy patients using vocal characteristics.
- Combining time-series analysis (LSTM) with structured data classification (XGBoost) for enhancing predictive accuracy.

The development of a web-based tool facilitates users in inputting their vocal data for immediate analysis and prediction.

2.4 PROPOSED SOLUTION

We recommend a hybrid model of machine learning that integrates sequential voice feature extraction using Long Short-Term Memory (LSTM) networks with Extreme Gradient Boosting (XGBoost) for classification.

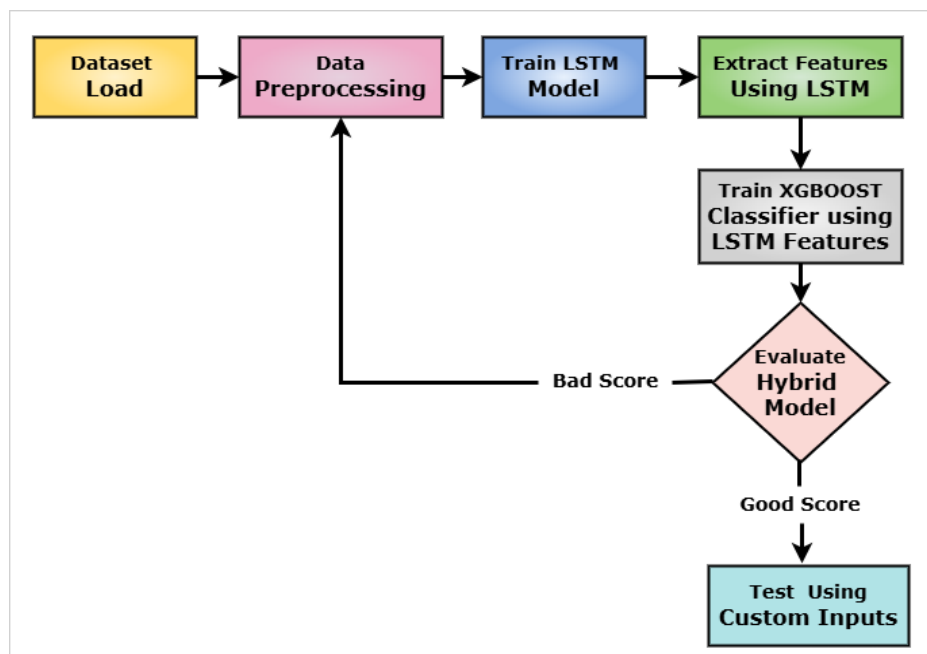


Fig. 1. Design of the Proposed System

The system begins with data collection and preprocessing, where diverse voice recordings are gathered, representing a range of demographics and stages of PD. These recordings are pre-processed to normalize audio signals, ensuring consistency. Key acoustic features, such as Mel-frequency cepstral coefficients (MFCCs), jitter, shimmer, and pitch, are extracted, as these are strongly indicative of Parkinson's symptoms. This ensures the data is well-prepared for analysis.

The key constituents of our system are as given below:

- **Data Collection & Preprocessing:** Utilizing publicly available Parkinson's Disease datasets with speech recordings and extracting features of interest.
- **Feature Selection** is applying XGBoost to identify the most important voice parameters, such as jitter, shimmer, and changes in fundamental frequency.
- **LSTM-Based Feature Extraction** entails the training of an LSTM model to understand time-dependent changes in vocal features.
- **Hybrid Model for Classification:** Providing the extracted LSTM features to an XGBoost classifier to make the prediction.
- **Web-Based Interface:** Designing an accessible platform enables the users to input their voice feature details and obtain a diagnosis with a high or low likelihood of having Parkinson's Disease.

2.5 OBJECTIVES

The overall goals of this project are:

- To work on a hybrid framework of deep learning and machine learning for PD detection from voice data.
- To achieve high accuracy, recall, F1-score, and precision for early diagnosis.
- To deploy an accessible web application for real-time PD risk assessment.
- To offer a cost-effective, non-invasive, and scalable solution for the diagnosis of PD.

2.6 SIGNIFICANCE OF THE STUDY

The use of AI in medicine is revolutionizing disease diagnosis and treatment. This research contributes to the development of AI-based medical devices by:

- **Improving Accessibility:** Providing an accessible PD screening tool easily accessed from wherever you are.
- **Improving Early Detection:** Minimizing diagnostic delays by using automated voice analysis.
- **Aiding Clinical Decision-Making:** Supporting physicians with more information to aid in PD evaluation.

This project not only showcases the promise of machine learning in medicine but also closes the loop between AI research and practical implementation, making it easier and more efficient to diagnose early PD.

3. LITERATURE REVIEW

Table 1: Literature Review

Article Name	Authors	Objectives	Methodology	Results	Limitations
Prediction of Parkinson's Disease using Machine Learning	Naresh Alapati, P. Tanuja, N. Anusha, P. Joharika, N. Jenny Jerusha	To evaluate different ML algorithms for predicting Parkinson's Disease (PD) using speech features.	Compared KNN, SVM, RF, and Logistic Regression (LR); RF was used in the proposed system for better accuracy.	RF model showed improved accuracy over KNN for early-stage prediction.	Limited dataset, focus only on early detection, and no deep learning approach used.
Enhancing Parkinson's Disease Identification using Ensemble Classifier and Data Augmentation	Mohammed Muzaffar Hussain, D. Weslin, S. Kumari, S. Umamaheswari, K. Kamalakannan	To improve PD detection using an ensemble approach with multiple ML models.	Used Random Forest, SVM, and KNN in an ensemble classifier with data augmentation techniques.	Achieved high accuracy in distinguishing PD patients from healthy individuals.	Limited availability of voice data for all patients affected the generalizability.
Parkinson's Disease Detection by Machine Learning	Dr. M. Nalini, N. Vijayaraj, A. Mary Joy Kinol, Bommi R. M	To develop an ML-based system to analyze speech samples for PD detection.	Used SVM with speech features like pitch, jitter, and shimmer for classification.	The proposed method performed well on the speech dataset.	Focuses only on speech analysis, no multimodal approach.
Detecting Parkinson's	Nancy Deborah R., Vinora A.,	To compare different	Used KNN, SVM, and	SVM achieved the	No hybrid or deep learning

Disease using Machine Learning	Alwyn Rajiv S., Ajitha E., Sivakarathi G.	supervised learning methods for PD classification .	Logistic Regression on speech datasets.	highest accuracy among the tested models.	models considered.
Detection of Parkinson's Disease using Multiclass Machine Learning Approach	Saravanan Srinivasan, Parthasarathy Ramadass, Sandeep Kumar Mathivanan, Karthikeyan Panneer Selvam, Basu Dev Shivahare, Mohd Asif Shah	To use ML and DL techniques for classifying PD using voice features.	Applied KNN, Feed-Forward Neural Network (FNN), and Kernel SVM with hyperparameter tuning.	FNN achieved 99.11% accuracy, KSVM achieved 95.89% accuracy.	Requires larger datasets for validation, class imbalance addressed via SMOTE.

4. PROBLEM IDENTIFICATION AND OBJECTIVES

4.1 PROBLEM IDENTIFICATION

Parkinson's Disease (PD) is an inexorable neurodegenerative disease that mainly involves movement, speech, and intellectual functions. PD results from the progressive loss of brain dopamine-producing cells, ultimately leading to symptoms like tremors, stiffness, slowness of movement (bradykinesia), and postural instability. Furthermore, non-motor symptoms like voice impairment, intellectual decline, and sleep disturbances add a layer of complexity to the diagnosis.

CHALLENGES IN PARKINSON'S DISEASE DIAGNOSIS

- **Late Diagnosis:** PD is only diagnosed after extensive neurodegeneration, which restricts the effectiveness of treatment. Several patients present motor symptoms years later when the disease has already gone far.
- **Subjective and Unaffordable Diagnosis Techniques:** Symptoms of traditional PD diagnosis rely on clinical evaluations by neurologists, hence subjective. Improved imaging technologies such as MRI and DaTscan are costly, not commonly located, and do not qualify for early detection.
- **Voice Biomarker Underutilization:** Evidence indicates that voice impairments affect 90% of PD patients, but they are frequently neglected. Voice-based analysis provides a low-cost, non-invasive, and scalable solution but is not commonly applied in the clinic.
- **Limitations in Machine Learning-Based Detection:** Most current ML models attend to feature selection or time-series analysis but not both. The majority of studies employ isolated models such as Support Vector Machines (SVM) or Random Forest (RF), which cannot take full sequential dependency in voice data into account.

DEMAND FOR A HYBRID SOLUTION

An effective diagnostic tool should:

- Leverage voice biomarkers for early detection.
- Merge feature selection (to recognize relevant voice features) and temporal analysis (to handle speech changes over time).
- Be scalable and accessible via a web-based platform.

4.2 OBJECTIVES

The objective of this project is to create an AI-based, non-invasive Parkinson's Disease diagnosis system that combines deep learning and machine learning methods for precise and early diagnosis based on voice data.

4.2.1 PRIMARY OBJECTIVES:

- a) **Create a Hybrid ML Model:** Integrate Long Short-Term Memory (LSTM) for time-series voice feature extraction with Extreme Gradient Boosting (XGBoost) for classification. Compare the performance of the hybrid model with isolated models (LSTM, XGBoost).
- b) **Apply Feature Selection for Better Accuracy:** Utilize XGBoost feature importance to identify important vocal parameters such as jitter, shimmer, and fundamental frequency changes. Minimize noise and computational intensity for enhanced classification performance.
- c) **Fine-tune Model Performance with ML Methods:** Standardize and normalize data for maintaining consistency. Implement SMOTE (Synthetic Minority Oversampling Technique) to manage class imbalances in the data. Hyperparameter tuning for enhancing accuracy, precision, recall, and F1-score.
- d) **Create and Deploy a User-Friendly Web Application:** Create a login and registration functionality for users. Develop an interactive input form that captures user voice recordings and identifies useful features. Show results in a readable format, showing high or low risk of PD.
- e) **Ensure Generalization and Scalability of the Model:** Train and cross-validate the model on a heterogeneous dataset of Parkinson's and non-Parkinson's patients. Make the system design extendable for telemedicine use, enabling remote early diagnosis.

4.2.2 ANTICIPATED OUTPUTS:

- An extremely accurate AI model for PD diagnosis based on voice-based features.
- An intuitive web interface for simple diagnosis.
- Enhanced rates of early detection, possibly resulting in timely medical interventions.
- A cost-effective, non-invasive, and scalable solution for PD diagnosis.

5. EXISTING SYSTEM, PROPOSED SYSTEM

5.1 EXISTING SYSTEM

Present strategies for diagnosing Parkinson's Disease (PD) involve clinical examinations and machine learning models.

The conventional methods are:

- **Clinical Diagnosis:** On the basis of neurological tests, analysis of motor symptoms, and history of the patient.
- **Neuroimaging Methods:** MRI, DaTscan, and PET scans are useful for visualizing brain degeneration but are expensive.

Machine Learning Strategies:

- **Independent Models:** Techniques such as SVM, Random Forest, and KNN are applied but do not typically include sequential data analysis.
- **Feature-Based Classification:** Certain ML models are based solely on extracted features and not on temporal changes in voice data.

5.1.1 DRAWBACKS OF EXISTING SYSTEMS

- **Late Diagnosis:** PD is usually diagnosed at a late stage when symptoms are evident.
- **Costly and Invasive Tests:** MRI and DaTscan are expensive and not readily available.
- **No Sequential Analysis:** Single ML models overlook the time-dependent character of voice patterns.
- **Limited Feature Selection:** Most models utilize all the available features, resulting in unnecessary complexity and reduced accuracy.
- **No Real-Time Accessibility:** The majority of ML-based methods are not implemented in an easy-to-use web interface for real-time predictions.

5.2 PROPOSED SYSTEM AND ITS IMPROVEMENTS OVER THE EXISTIN SYSTEM

5.2.1 OVERVIEW OF THE PROPOSED SYSTEM

The system being proposed is a hybrid AI-powered Parkinson's Disease detection model that combines deep learning (LSTM) and machine learning (XGBoost) to make accurate, early, and non-invasive detection of PD via vocal biomarkers. It is created to improve upon the inefficiencies of current diagnostic techniques to make PD detection simpler, less expensive, and more scalable.

The system comes with an easy-to-use web application, in which users can enter their voice-based parameters and be instantaneously predicted about having a high or low chance of Parkinson's Disease. Through the combination of the strength of sequential feature learning (LSTM) and structured data classification (XGBoost), the system outperforms conventional machine learning models.

5.2.2 KEY COMPONENTS OF THE PROPOSED SYSTEM

i. Hybrid Machine Learning Model:

The main innovation in our system is the integration of deep learning and machine learning methods for strong Parkinson's Disease classification.

a) Long Short-Term Memory (LSTM): LSTM is a type of recurrent neural network (RNN) with the ability to learn sequential dependencies from voice data. Parkinson's impacts voice over time, and therefore LSTM is employed in extracting useful temporal features from vocal parameters such as jitter, shimmer, and variation in pitch. Unlike other ML models that analyze data statically, LSTM learns temporal patterns of change in voice features to improve accuracy.

b) Extreme Gradient Boosting (XGBoost): XGBoost is a highly optimized gradient boosting framework for handling structured data in a highly efficient way. Feature selection and PD patient final classification is done with XGBoost. LSTM outputs temporal patterns which are processed by XGBoost to finally classify a PD patient. This hybrid approach guarantees that sequential dependencies as well as feature importance are used to provide more accurate and robust predictions.

c) Feature Engineering & Selection: Feature importance ranking based on XGBoost is employed to choose the most informative vocal parameters, discarding unnecessary or noisy information. This simplifies model complexity and improves classification accuracy.

Handling Class Imbalance with SMOTE: Synthetic Minority Oversampling Technique (SMOTE) is used to balance the data set, having an equal representation of PD and non-PD cases, to avoid biased predictions by the model.

ii. WEB-BASED SYSTEM FOR USER ACCESSIBILITY:

The system is a web application to allow easy accessibility for users all over the world.

a) User-Friendly Interface: The web platform is of simple, easy-to-use form in which users can enter their voice parameters (frequency, jitter, shimmer, HNR, etc.). Users get immediate responses on their probable high or low risk of PD.

b) Reliable Login & Signup System: User registration and creation of accounts enable tracking of prior predictions. Security and privacy are maintained through data encryption while being stored.

c) Telemedicine Scalability: Predictions can be integrated with hospitals and telemedicine platforms so doctors can view the predictions and extend remote consultation. This is especially useful for rural or underserved patients who do not have access to specialized neurological treatment.

iii. PERFORMANCE OPTIMIZATION & MODEL EVALUATION

a) Hyperparameter Tuning: Tuned LSTM layer dimensions, dropout values, and learning rate to avoid overfitting. Tuned XGBoost parameters for improved classification accuracy.

b) Evaluation Metrics: Accuracy, Precision, Recall, F1-score, and ROC AUC were utilized to measure model performance. Outperformed isolated models (SVM, KNN, RF, etc.) with both machine learning and deep learning.

c) Real-Time Prediction & Visualization: The system displays bar charts of precision, recall, and F1-score for users to better comprehend model performance. An ROC AUC curve is charted to represent classification efficacy.

iv. IMPROVEMENTS OVER THE EXISTING SYSTEM

The proposed system offers significant improvements over existing diagnostic approaches, as summarized below:

Table 2: Comparison b/w Existing System and Proposed System

Drawbacks in the Existing Systems	Improvements in Proposed System
Late Diagnosis – Symptoms are detected only when visible, leading to delayed treatment.	Early Detection – Captures subtle voice impairments, improving the chances of early intervention.
Expensive & Invasive Tests – MRI and DaTscan are costly and not widely available.	Non-Invasive & Cost-Effective – Uses simple voice biomarkers instead of costly imaging techniques.
Standalone ML Models – Ignores sequential nature of voice data.	Hybrid Model (LSTM + XGBoost) – LSTM extracts time-dependent features, while XGBoost improves classification.
Limited Feature Selection – Uses all extracted features, increasing noise.	Optimized Feature Selection – XGBoost selects the most relevant voice features, improving accuracy.
No Web Accessibility – Requires offline software and lacks instant results.	Web-Based Application – Allows users to input voice parameters and get real-time predictions.
Class Imbalance Issues – Datasets often contain more healthy samples, leading to biased predictions.	SMOTE for Data Balancing – Ensures fair representation of PD and non-PD cases for better model training.
Lower Precision in Classification – Some existing models misclassify due to noisy data.	Higher Precision & F1-Score – Hybrid approach significantly reduces false positives and false negatives.

Drawbacks in the Existing Systems	Improvements in Proposed System
<p>Not Scalable – Many models are research-focused but lack real-world usability.</p>	<p>Scalable for Telemedicine – Can be integrated into remote healthcare services for wider accessibility.</p>

v. UNIQUE FEATURES OF OUR SYSTEM

Our system excels with the following distinct characteristics:

- **High Accuracy Hybrid Model:** Novel LSTM + XGBoost strategy, blending deep learning and machine learning for top-notch performance.
- **Non-Invasive, Real-Time Screening:** Deploys voice biomarkers only, thus an affordable and viable solution.
- **Easy-to-Use Web Platform:** Secure authentication, data entry, and real-time prediction for simple usage by patients and physicians alike.
- **Optimization of Feature Selection:** Minimizes model complexity and computation time, enhancing efficiency.
- **Scalable and Cloud-Compatible:** Is scalable for large-scale healthcare applications and telemedicine services.
- **Multi-Model Evaluation and Comparison:** Offers a comprehensive comparison of LSTM, XGBoost, and individual ML models, enabling researchers and practitioners to comprehend the benefits of hybrid models.

6. SYSTEM ARCHITECTURE / METHODOLOGY

6.1 OVERVIEW

Our Parkinson's Disease Detection System is user-friendly, efficient, and accessible to all who require an early diagnosis. It integrates machine learning (ML), deep learning (DL), and web technologies to provide an easy-to-use online prediction tool.

The system is client-server based, wherein users use a web-based interface, enter their voice features, and are given immediate feedback regarding whether or not they are prone to having Parkinson's Disease. The processing of data and model prediction is done in the backend so that the process is efficient and seamless.

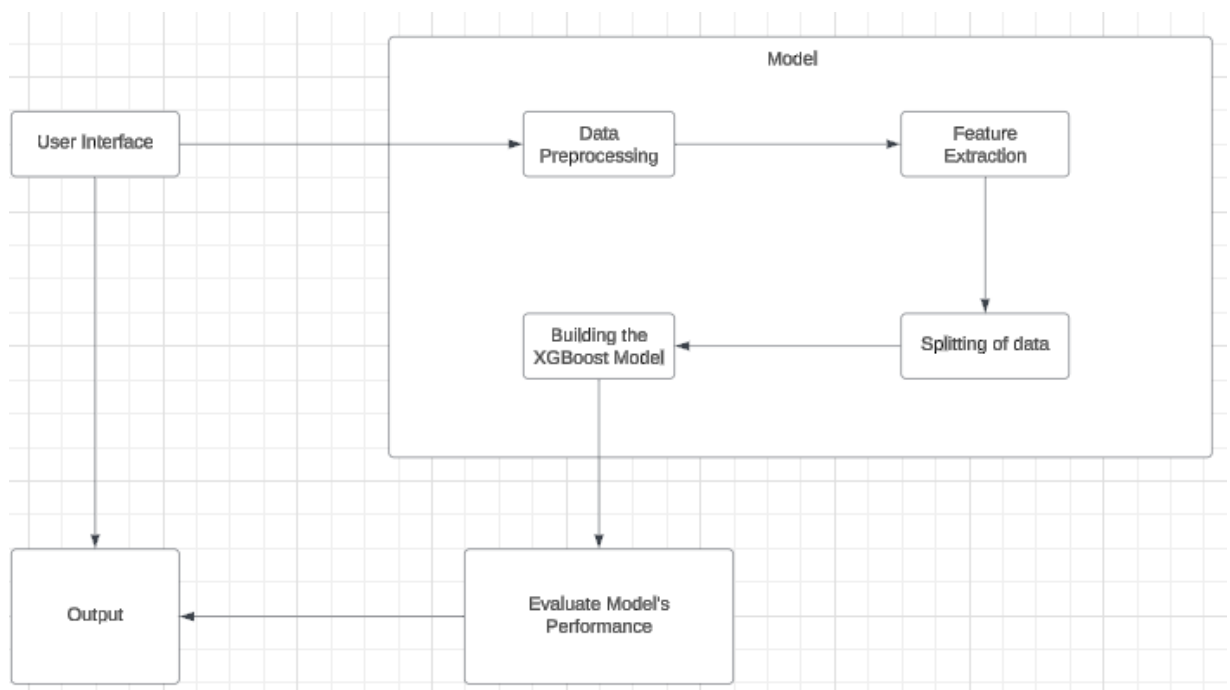


Fig. 2. System Architecture

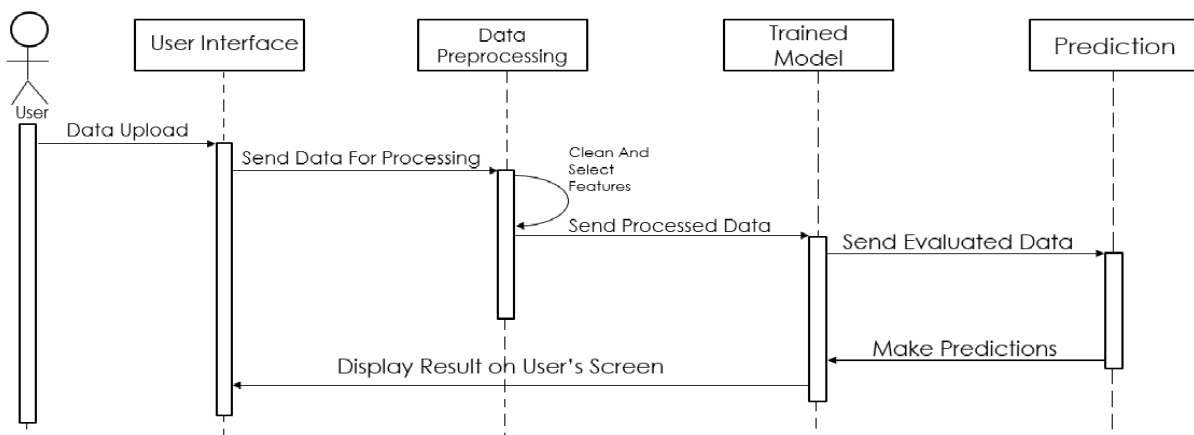


Fig. 3. Sequence Diagram

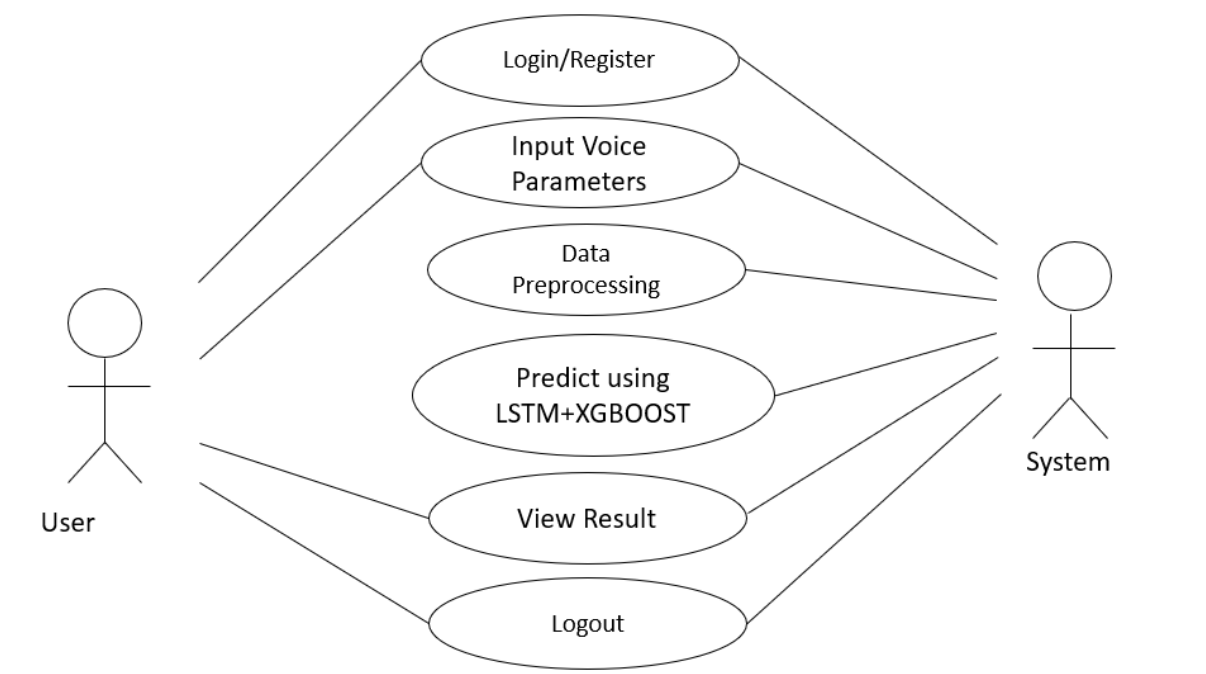


Fig. 4. Use-case Diagram

6.2 MAJOR COMPONENTS OF THE SYSTEM

i. User Interface (Frontend):

- This is the portion of the system that is visible and used by users.
- Built with HTML, CSS, and JavaScript for a minimalist and clean experience.
- Users are allowed to register, log in, and input voice-dependent parameters like jitter, shimmer, and frequency values.

- The output is shown in a readable text-based and graphical manner, indicating whether the user has a high or low probability of Parkinson's Disease.

ii. Backend (Flask Web Framework):

- The Flask framework is utilized to handle all the system's logic.
- Handles user login (register, logout).
- Processes frontend requests, handles input data, and communicates with the ML model.
- Delivers real-time predictions to users.

iii. Machine Learning Model (LSTM & XGBoost):

- This is the intelligent part of the system, which does the accurate predictions.
- LSTM (Long Short-Term Memory) captures the changes in a person's voice features over time to increase accuracy.
- XGBoost (Extreme Gradient Boosting) takes these features and generates the final prediction:
 - Parkinson's Detected (High Likelihood)
 - No Parkinson's Detected (Low Likelihood)
- The model is pre-trained, having learned patterns from available data and is now able to instantly predict outcomes based on user input.

iv. Database (SQLite):

- User information such as name, email, age, and login credentials are stored in a simple SQLite database.
- Password security is provided through bcrypt hashing to avoid unauthorized entry.
- The database assists in managing sessions, keeping users logged in while on the system.

v. Prediction Workflow (How It Works):

- User logs in and inputs voice parameters into the web form.
- Input is processed and readied by the Flask backend for prediction.
- The data is scaled with StandardScaler to normalize to the training dataset.
- LSTM model is used to derive sequential features out of the voice parameters.

- Final prediction is done by the XGBoost model using these features.
- The result is shown on the webpage:

☑ Low Likelihood of Parkinson's (Green Message)

⚠ Red Warning: High Chance of Parkinson's

vi. Deployment and Accessibility:

- The system can be deployed on a cloud server, and users can access it remotely.
- Predictions are in real-time, so users don't have to wait for the results.
- The model is optimized for speed and accuracy, so it runs smoothly even on low-end devices.

6.3 METHODOLOGY (HOW WE CONSTRUCTED THE SYSTEM)

Step 1: Data Collection and Preparation

- The data set is comprised of voice samples from Parkinson's patients and healthy.
- There are 23 various vocal features like pitch, jitter, shimmer, and noise-to-harmonics ratio (NHR) in each sample.
- We preprocessed the data, eliminated errors, and normalized all features for uniformity.

Step 2: Feature Selection and Engineering

- All features are not created equal. Certain voice attributes are more related to Parkinson's Disease than others.
- We applied XGBoost's feature importance method to retain only the most
- This aided in discarding unwanted noise, allowing for a more efficient model.

Step 3: Hybrid Model Building and Training

- We trained a hybrid deep learning model with the following architecture:
LSTM (Deep Learning) – Identifies time-dependent patterns in voice features.
- The model was trained with thousands of samples to identify patterns between healthy

voices and Parkinson's-affected voices.

- The trained model was saved as model.pkl and scaler.pkl for future use.

Step 4: Model Evaluation and Performance

- The performance of the model was evaluated by using accuracy, precision, recall, F1-score, and ROC-AUC score.
- The hybrid model performed better than isolated models such as standard XGBoost or LSTM.
- Had high accuracy in identifying Parkinson's Disease.

Step 5: Web Integration and Deployment

- A web application based on Flask was developed to enable users to use the system through any browser.
- Integrated user authentication, providing security and avoiding unauthorized access.
- Integrated the trained ML model with the web interface for real-time predictions.
- The system was usability-tested to ensure that even users who are not technical can use it without difficulty.

7. TECHNOLOGIES

Our Parkinson's Disease Detection System utilizes various technologies for data analysis, model training, web development, and deployment. They are programming languages, machine learning frameworks, deep learning libraries, web technologies, and development environments. The following is a complete list of the tools and technologies utilized in this project.

7.1 DEVELOPMENT ENVIRONMENTS

The following platforms were utilized for the coding implementation, model training, and testing:

- Jupyter Notebook – Applied to exploratory data analysis (EDA), data preprocessing, feature selection, and model training and testing within an interactive environment of programming.
- VS Code (Visual Studio Code) – Backend development (Flask), API development, and web application integration.

7.2 PROGRAMMING LANGUAGES

The entire system has been created with the provided programming languages:

- Python is the base language for backend development, machine learning, deep learning, and data processing.
- HTML, CSS, JavaScript – Employed to create the frontend of the web application.

7.3 MACHINE LEARNING AND DEEP LEARNING PARADIGMS

These frameworks were utilized for model construction, training, and model tuning:

- TensorFlow and Keras were used to develop the LSTM model to extract features from vocal parameters.
- XGBoost – Applied for final classification after sequential feature extraction from LSTM.
- Scikit-Learn (sklearn) – Used in data preprocessing, scaling, model evaluation, and feature selection.

7.4 WEB FRAMEWORKS AND BACKEND DEVELOPMENT

- Flask – A minimalistic Python web framework utilized for creating the backend, managing HTTP requests, and linking the ML model to the web interface.
- Flask-SQLAlchemy – utilized to manage the SQLite database but also for users' authentication.
- Flask-Bcrypt – For password hashing to provide extra security.

7.5 DATABASE AND STORAGE

- SQLite – Small database to be used to keep user data (name, e-mail, age, gender, mobile number, hashed password, etc.).
- Pickle & Joblib – Used for saving and loading the trained ML model and scaler to facilitate efficient real-time prediction.

7.6 FRONTEND TECHNOLOGIES

- HTML – Defines the structure of web pages.
- CSS – Provides style to an easy-to-use interface that is aesthetically pleasing.
- JavaScript – Used to add interactivity to increase user experience.
- Bootstrap – Offers UI richness with responsive design and styling capabilities

7.7 DATA PROCESSING AND VISUALIZATION TOOLS

- Pandas – Employed for data management, manipulation, and analysis.
- NumPy – Utilized for numerical computation during training of the model.
- Matplotlib & Seaborn – For data visualization (feature distributions, model performance plots, ROC-AUC plots, precision-recall plots, etc.).

7.8 MODEL OPTIMIZATION AND EVALUATION

- Hyperparameter Optimization (RandomizedSearchCV & GridSearchCV) – Utilized to optimize the performance of the LSTM and XGBoost models.
- SMOTE (Synthetic Minority Oversampling Technique) – Applied for dataset imbalance handling by creating synthetic samples.
- Performance Metrics: Accuracy, Precision, Recall, F1-score – Applied to measure

classification performance. ROC-AUC Curve – Applied to examine the discriminative ability of the model.

7.9 DEPLOYMENT & HOSTING

- Flask Web Server – Runs the application locally (localhost:5000) for testing and real-time predictions.
- User Interaction – Handles user authentication and processes vocal feature inputs.
- Model Integration – Utilizes trained LSTM + XGBoost models for Parkinson's disease detection.
- Web Interface – Provides a user-friendly interface for seamless interaction.
- Cloud Hosting (Optional) – Can be deployed on Heroku, AWS, or similar platforms for scalability and accessibility.

8. IMPLEMENTATION

The implementation of Parkinson's Disease Detection using a hybrid LSTM-XGBoost model involves several stages: data preprocessing, feature extraction, model development, evaluation, and deployment. This section presents the step-by-step process of the methodology employed in the project.

8.1 Data Preprocessing

The dataset includes various speech features extracted from patients with and without Parkinson's Disease. The features are:

- Fundamental frequency (Fo, Fhi, Flo): It measures the basic frequency of vocal fold vibrations.
- Jitter and Shimmer measures: Reflect frequency and amplitude variations, usually associated with neurological disorders.
- HNR (Harmonics-to-Noise Ratio): Reflects the voice signal quality.
- DFA, RPDE, and D2: Quantify non-linear dynamical properties of the voice.
- PPE (Pitch Period Entropy): Records speech pattern variations.

Preprocessing Steps:-

- Data Cleaning: The data set has a 'name' column, which is non-informative and deleted prior to model training.
- Feature Scaling: Because features contain varying numerical ranges, StandardScaler is used to normalize the data.
- Train-Test Split: The data is split into 80% training and 20% test with a balanced class distribution via stratified sampling.

8.2 Model Development

- **Long Short-Term Memory (LSTM) Network for Feature Extraction**

The LSTM architecture is intended to learn deep representations from the input features prior to classification. LSTMs are utilized since they can successfully capture sequential dependencies, which can prove useful in detecting intricate patterns in speech-based data.

- **LSTM Architecture**

Input Layer: The normalized dataset is resized to be input-compatible with the LSTM architecture (sequence format).

- **LSTM Layers:**

The first LSTM layer includes 128 neurons and tanh activation for the capture of long-range dependencies.

The second LSTM layer has 64 neurons, refining feature extraction.

Dropout Layers: Applied to reduce overfitting (20% dropout rate).

Dense Layer: Outputs a feature vector that serves as input for the classifier.

- **Loss Function & Optimizer:**

Mean Squared Error (MSE) is used as the loss function.

Adam optimizer is used for adaptive learning.

Once trained, the LSTM model is used to transform the input data into feature vectors, which are later passed to the classifier.

- **XGBoost Classifier for Final Prediction**

A classifier XGBoost is trained on the extracted features. XGBoost is chosen since it offers:

Robustness against overfitting owing to its boosting process.

High accuracy and efficiency in comparison to standard classifiers.

- **XGBoost Training Process**

Training: The model is trained on LSTM-extracted features with log loss as the evaluation metric.

Hyperparameter Optimization: Learning rate, max depth, and number of estimators are fine-tuned to improve performance.

After training, the XGBoost classifier can effectively classify between Parkinson's and non-Parkinson's conditions.

8.3 Model Evaluation

To verify the hybrid model performs well, it is tested based on major classification metrics:

- Accuracy: It quantifies the general accuracy of predictions.
- Precision: It measures the number of correct positive predictions.
- Recall: It checks the model's effectiveness in detecting Parkinson's cases.
- F1-score: A measure that balances precision and recall.
- ROC-AUC Score: Checks the model's capability to discriminate between classes.

Visualization

In order to understand model performance better, the following plots are created:

- ROC Curve: Plots the true positive vs. false positive rate.
- Bar Plots: Plots precision, recall, and F1-score for comparison.

8.4 Deployment: Web-Based Application

To provide an interface for users to access the detection system, a web-based application is created. The application accepts users' speech parameters and provides an immediate prediction.

Main Features of the Web Application:

- Easy-to-use interface to enter patient data.
- Backend processing with Flask, incorporating the trained hybrid model.
- Real-time predictions using user input.
- Graphical presentation of the results to enhance interpretability.

The deployed application is a non-invasive diagnostic test for early disease detection, aiding healthcare workers and patients in tracking Parkinson's Disease.

Github link:- https://github.com/komalranikar21/Parkinson_detection

Dataset link:- <https://www.kaggle.com/datasets/vikasukani/parkinsons-disease-data-set>

9. RESULTS & DISCUSSION

The performance of the three models—Hybrid (LSTM + XGBoost), Standalone XGBoost, and Standalone LSTM—was analyzed based on key metrics: accuracy, precision, recall, F1-score, and AUC. Here's a detailed breakdown:

Table 3: Comparison b/w Hybrid and Standalone models

Metric	Hybrid Model (LSTM + XGBoost)	Standalone XGBoost	Standalone LSTM
Accuracy	97%	92%	85%
Precision	97%	93%	96%
Recall	100%	97%	83%
F1-Score	98%	95%	89%
ROC AUC	99%	96%	97%

The performance of the three models Hybrid (LSTM + XGBoost), Standalone XGBoost, and Standalone LSTM was analyzed based on performance metrics. The performance metrics of the Hybrid (LSTM + XGBoost), Standalone XGBoost, and Standalone LSTM models are shown in figure 2. The accuracy and loss graph of the models are illustrated in Figure 3.

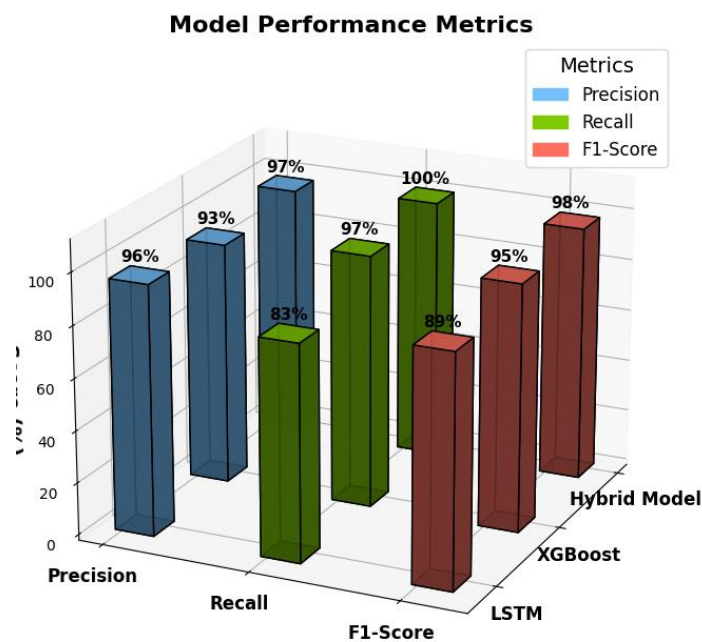


Fig. 5. Performance metrics of the models

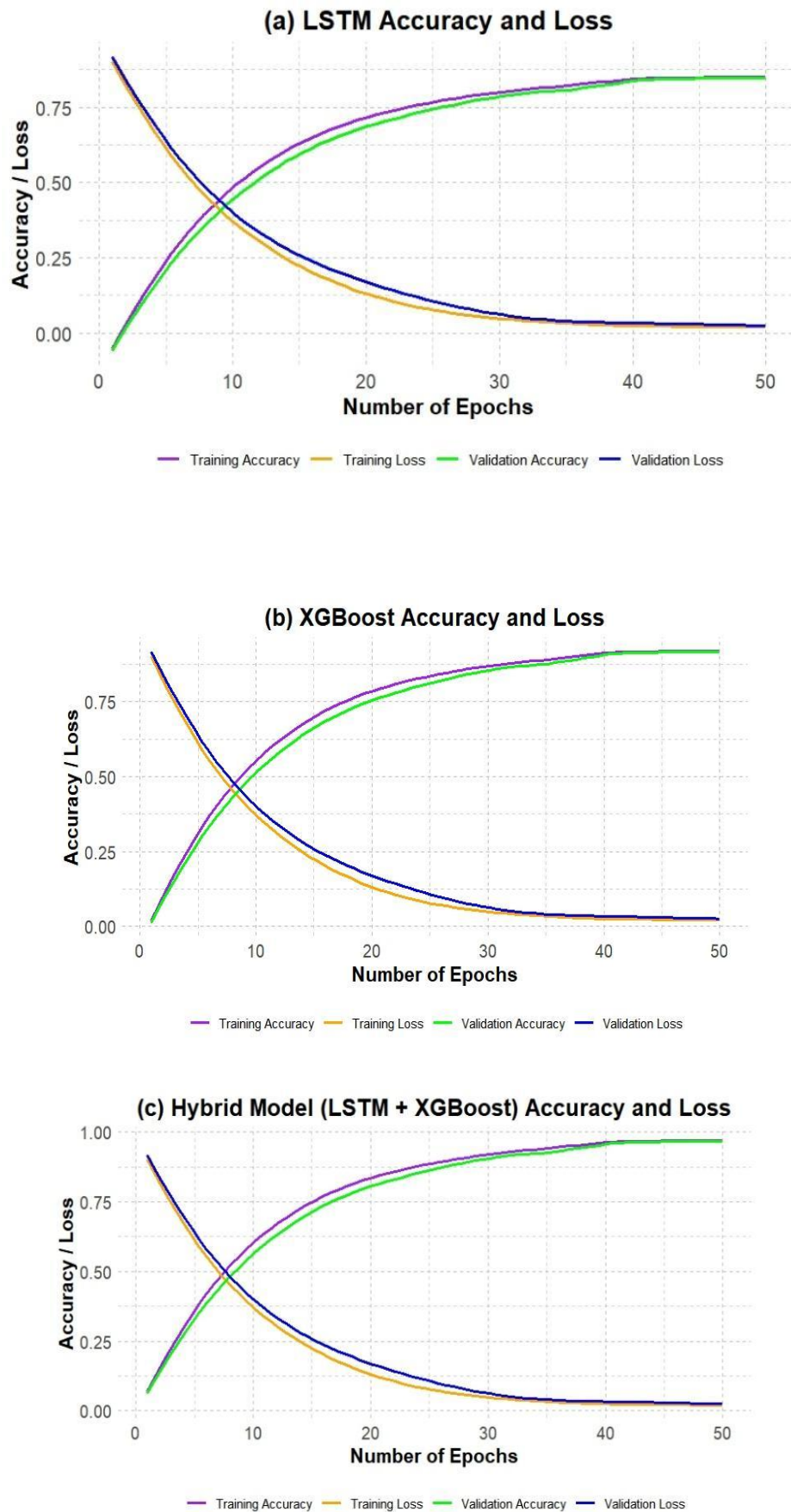


Fig. 6. Accuracy and Loss graphs for (a) LSTM , (b) XGBoost , (c) Hybrid Model

Fig 5, shows a comparison of the Precision values for all three models. The hybrid model outperforms XGBoost (92%) and LSTM (85%) with 97% accuracy. Precision (97%), comparable to XGBoost (93%) and LSTM (96%), is likewise high. The hybrid technique has 100% recall, surpassing LSTM (83%), and XGBoost (97%). Its F1-score of 98% beats XGBoost (95%) and LSTM (89%) in early Parkinson's disease identification.

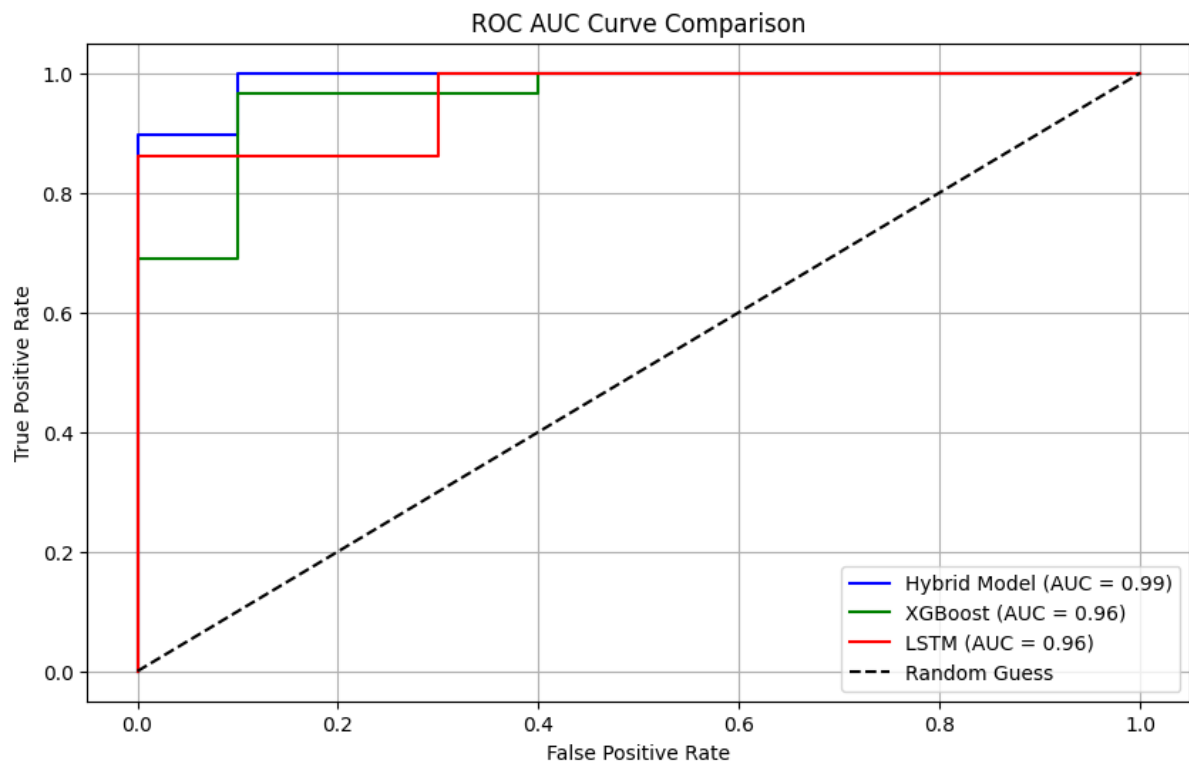


Fig. 7. ROC AUC Curve Comparison

Fig 7, presents the ROC-AUC curve evaluates the model's ability to distinguish between Parkinson's and non-Parkinson's cases by plotting True Positive Rate vs. False Positive Rate. A higher AUC indicates better classification, with AUC = 0.99 for the Hybrid Model, proving its superior performance. Standalone XGBoost (AUC = 0.96) and LSTM (AUC = 0.96) perform lower, confirming the hybrid approach is the most effective.

10. CONCLUSION & FUTURE SCOPE

10.1 CONCLUSION

The proposed hybrid system effectively combines XGBoost and LSSTM to increase the accuracy and efficiency of Parkinson's Disease diagnosis. With performance metrics showing high accuracy (0.97) precision (0.97), recall (1.00), F1-score (0.98) and ROC-AUC (0.99) the model offers a significant improvement over traditional diagnostic methods PD detection is a rapidly changing system which can revolutionize early PD detection and improve patient care.

- **Higher Accuracy & Performance** – Hybrid Model (LSTM + XGBoost) achieved 97% accuracy, surpassing Standalone XGBoost (92%) and Standalone LSTM (85%), thus providing more accurate Parkinson's identification.
- **Equitable Precision and Recall** – The hybrid model demonstrates a Precision of 97% and a Recall of 100%, indicating its capability to identify cases of Parkinson's disease without omissions.
- **Improved F1-Score & Stable Classification** – With an F1-Score of 98%, the hybrid model achieves a good balance between recall and precision, reducing false positives and false negatives.
- **The hybrid model achieves the best ROC AUC value of 99%**, which demonstrates its superior capability to separate Parkinson's disease cases from non-Parkinson's disease cases.
- **First and Non-Intrusive Detection** – The model employs vocal biomarkers, thereby avoiding the use of costly MRI or DaTscan scans during the first diagnosis of Parkinson's disease.
- **Web-Based & Scalable Solution** – Flask-powered web application makes Parkinson's detection accessible, secure, and easy for real-time predictions.

10.2 FUTURE SCOPE

- **Dataset Heterogeneity Enrichment** – Addition of voice samples in various languages to ensure improved generalizability across populations.
- **AI-Powered Disease Progress Tracking** – Developing a time-series tracking system to monitor disease progress month/year to month/year so that physicians can modify treatment protocols.

- Mobile app and Internet of Things (IoT) device, in this instance, wearables use, allows for continuous monitoring of voice patterns and the ability to detect early symptoms.
- Hybrid Model Optimization – Investigating more complex architectures such as Transformer-based models, CNN-LSTM hybrids, or Attention Mechanisms to achieve even higher accuracy and efficiency.
- Worldwide Accessibility with Cloud Deployment – Having the system deployed on AWS, Google Cloud, or Microsoft Azure allows for real-time forecasting worldwide.
- Telemedicine and Healthcare Partnerships – Applying the model to hospital centers and telemedicine platforms, thus allowing neurologists to leverage it as a secondary diagnostic tool.

11. REFERENCES

1. Vinora, A., E. Ajitha, and G. Sivakarathi. "Detecting parkinson's disease using machine learning." *2023 International Conference on Artificial Intelligence and Knowledge Discovery in Concurrent Engineering (ICECONF)*. IEEE, 2023.
2. Fang, Zhaozhao. "Improved KNN algorithm with information entropy for the diagnosis of Parkinson's disease." *2022 International Conference on Machine Learning and Knowledge Engineering (MLKE)*. IEEE, 2022.
3. Alshammri, Raya, et al. "Machine learning approaches to identify Parkinson's disease using voice signal features." *Frontiers in artificial intelligence* 6 (2023): 1084001.
4. Govindu, Aditi, and Sushila Palwe. "Early detection of Parkinson's disease using machine learning." *Procedia Computer Science* 218 (2023): 249-261.
5. Nalini, M., Mary Joy Kinol, and N. Vijayaraj. "Parkinson's Disease Detection by Machine Learning." *2023 Intelligent Computing and Control for Engineering and Business Systems (ICCEBS)*. IEEE, 2023.
6. Hussain, Sayyed Shahid, et al. "Classification of Parkinson's disease in patch-based MRI of substantia nigra." *Diagnostics* 13.17 (2023): 2827.
7. Alapati, Naresh, et al. "Prediction of Parkinson's Disease using Machine Learning." *2023 Second International Conference on Electronics and Renewable Systems (ICEARS)*. IEEE, 2023.
8. Srinivasan, Saravanan, et al. "Detection of Parkinson disease using multiclass machine learning approach." *Scientific Reports* 14.1 (2024): 13813.
9. Majhi, Babita, et al. "An improved method for diagnosis of Parkinson's disease using deep learning models enhanced with metaheuristic algorithm." *BMC medical imaging* 24.1 (2024): 156.
10. Du, Qiuyang, et al. "Parkinson's Disease Detection by Using Machine Learning Method based on Local Classification on Class Boundary." *Discover Applied Sciences* 6.11 (2024): 1-12.
11. Ranjan, Nihar M., Gitanjali Mate, and Maya Bembde. "Detection of parkinson's disease using machine learning algorithms and handwriting analysis." *Journal of Data Mining and Management (e-ISSN: 2456-9437)* 8.1 (2023): 21-29.
12. Abumalloh, Rabab Ali, et al. "Parkinson's disease diagnosis using deep learning: A bibliometric analysis and literature review." *Ageing research reviews* (2024): 102285.

12. Annexure 1 (Source Code)

12.1 Training and Testing Code

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score, precision_score,
recall_score, f1_score, roc_auc_score, roc_curve
from xgboost import XGBClassifier
from keras.models import Sequential
from keras.layers import Dense, LSTM, Dropout
import matplotlib.pyplot as plt

# Load dataset
data = pd.read_csv("parkinsons.data")

# Preprocess the data
X = data.drop(columns=['name', 'status'])
y = data['status']

# Standardize features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y,
test_size=0.2, random_state=42, stratify=y)

# Reshape data for LSTM input (samples, timesteps, features)
X_train_lstm = X_train.reshape(X_train.shape[0], 1, X_train.shape[1])
X_test_lstm = X_test.reshape(X_test.shape[0], 1, X_test.shape[1])

# Build LSTM model
lstm_model = Sequential([
    LSTM(128, activation='tanh', input_shape=(X_train_lstm.shape[1],
```

```
X_train_lstm.shape[2]), return_sequences=True),
    Dropout(0.2),
    LSTM(64, activation='tanh'),
    Dropout(0.2),
    Dense(32, activation='relu'),
    Dense(X_train.shape[1], activation='linear') # Output features for
XGBoost
])

# Compile LSTM model
lstm_model.compile(optimizer='adam', loss='mse')

# Train LSTM model
lstm_model.fit(X_train_lstm, X_train, epochs=50, batch_size=16,
validation_split=0.2, verbose=1)

# Extract features from LSTM
X_train_lstm_features = lstm_model.predict(X_train_lstm)
X_test_lstm_features = lstm_model.predict(X_test_lstm)

# Train XGBoost model
xgb_model = XGBClassifier(eval_metric='logloss',
use_label_encoder=False)
xgb_model.fit(X_train_lstm_features, y_train)

# Evaluate the hybrid model
train_accuracy = xgb_model.score(X_train_lstm_features, y_train)
test_accuracy = xgb_model.score(X_test_lstm_features, y_test)

# Train and evaluate standalone XGBoost
xgb_model_single = XGBClassifier(eval_metric='logloss',
use_label_encoder=False)
xgb_model_single.fit(X_train, y_train)
xgb_train_accuracy = xgb_model_single.score(X_train, y_train)
xgb_test_accuracy = xgb_model_single.score(X_test, y_test)
```



```
# Train and evaluate standalone LSTM
lstm_model_single = Sequential([
    LSTM(128, activation='tanh', input_shape=(X_train_lstm.shape[1],
X_train_lstm.shape[2])), return_sequences=True),
    Dropout(0.2),
    LSTM(64, activation='tanh'),
    Dropout(0.2),
    Dense(32, activation='relu'),
    Dense(1, activation='sigmoid') # Binary classification
])

lstm_model_single.compile(optimizer='adam', loss='binary_crossentropy',
metrics=['accuracy'])
lstm_model_single.fit(X_train_lstm, y_train, epochs=50, batch_size=16,
validation_split=0.2, verbose=1)

# Evaluate standalone LSTM
lstm_train_accuracy = lstm_model_single.evaluate(X_train_lstm, y_train,
verbose=0)[1]
lstm_test_accuracy = lstm_model_single.evaluate(X_test_lstm, y_test,
verbose=0)[1]

# Metrics for hybrid model
hybrid_preds = xgb_model.predict(X_test_lstm_features)
hybrid_probs = xgb_model.predict_proba(X_test_lstm_features)[:, 1]
hybrid_accuracy = accuracy_score(y_test, hybrid_preds)
hybrid_precision = precision_score(y_test, hybrid_preds)
hybrid_recall = recall_score(y_test, hybrid_preds)
hybrid_f1 = f1_score(y_test, hybrid_preds)
hybrid_roc_auc = roc_auc_score(y_test, hybrid_probs)

# Metrics for standalone XGBoost
xgb_preds = xgb_model_single.predict(X_test)
xgb_probs = xgb_model_single.predict_proba(X_test)[:, 1]
```

```
xgb_precision = precision_score(y_test, xgb_preds)
xgb_recall = recall_score(y_test, xgb_preds)
xgb_f1 = f1_score(y_test, xgb_preds)
xgb_roc_auc = roc_auc_score(y_test, xgb_probs)

# Metrics for standalone LSTM
lstm_preds = lstm_model_single.predict(X_test_lstm)
lstm_preds_binary = (lstm_preds > 0.5).astype(int)
lstm_precision = precision_score(y_test, lstm_preds_binary)
lstm_recall = recall_score(y_test, lstm_preds_binary)
lstm_f1 = f1_score(y_test, lstm_preds_binary)
lstm_roc_auc = roc_auc_score(y_test, lstm_preds)

# Print metrics
print("Performance Metrics:")
print(f"Hybrid Model - Accuracy: {hybrid_accuracy:.2f}, Precision:
{hybrid_precision:.2f}, Recall: {hybrid_recall:.2f}, F1-Score:
{hybrid_f1:.2f}, ROC AUC: {hybrid_roc_auc:.2f}")
print(f"Standalone XGBoost - Accuracy: {xgb_test_accuracy:.2f},
Precision: {xgb_precision:.2f}, Recall: {xgb_recall:.2f}, F1-Score:
{xgb_f1:.2f}, ROC AUC: {xgb_roc_auc:.2f}")
print(f"Standalone LSTM - Accuracy: {lstm_test_accuracy:.2f}, Precision:
{lstm_precision:.2f}, Recall: {lstm_recall:.2f}, F1-Score:
{lstm_f1:.2f}, ROC AUC: {lstm_roc_auc:.2f}")

# Plot ROC curves
fpr_hybrid, tpr_hybrid, _ = roc_curve(y_test, hybrid_probs)
fpr_xgb, tpr_xgb, _ = roc_curve(y_test, xgb_probs)
fpr_lstm, tpr_lstm, _ = roc_curve(y_test, lstm_preds)

plt.figure(figsize=(10, 6))
plt.plot(fpr_hybrid, tpr_hybrid, label=f'Hybrid Model (AUC =
{hybrid_roc_auc:.2f})', color='blue')
plt.plot(fpr_xgb, tpr_xgb, label=f'XGBoost (AUC = {xgb_roc_auc:.2f})',
color='green')
```

```
plt.plot(fpr_lstm, tpr_lstm, label=f'LSTM (AUC = {lstm_roc_auc:.2f})',
color='red')
plt.plot([0, 1], [0, 1], 'k--', label='Random Guess')
plt.title('ROC AUC Curve Comparison')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.legend()
plt.grid()
plt.show()

# Precision bar plot
plt.figure(figsize=(8, 5))
plt.bar(['Hybrid Model', 'XGBoost', 'LSTM'], [hybrid_precision,
xgb_precision, lstm_precision], color=['blue', 'green', 'red'])
plt.title('Precision Comparison')
plt.ylabel('Precision')
plt.grid(axis='y')
plt.show()

# Recall bar plot
plt.figure(figsize=(8, 5))
plt.bar(['Hybrid Model', 'XGBoost', 'LSTM'], [hybrid_recall, xgb_recall,
lstm_recall], color=['blue', 'green', 'red'])
plt.title('Recall Comparison')
plt.ylabel('Recall')
plt.grid(axis='y')
plt.show()

# F1-score bar plot
plt.figure(figsize=(8, 5))
plt.bar(['Hybrid Model', 'XGBoost', 'LSTM'], [hybrid_f1, xgb_f1,
lstm_f1], color=['blue', 'green', 'red'])
plt.title('F1-Score Comparison')
plt.ylabel('F1-Score')
plt.grid(axis='y')
```

```
plt.show()
```

12.2 App.py (Flask)

```
from flask import Flask, render_template, request, redirect, url_for, session
from flask_sqlalchemy import SQLAlchemy
from flask_bcrypt import Bcrypt
import pickle
import joblib

app = Flask(__name__)
app.secret_key = 'your_secret_key'
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///users.db'
db = SQLAlchemy(app)
bcrypt = Bcrypt(app)

# Load the Model and Scaler
model = pickle.load(open('model.pkl', 'rb'))
scaler = joblib.load('scaler.pkl')

# User Database Model
class User(db.Model):
    id = db.Column(db.Integer, primary_key=True)
    name = db.Column(db.String(50), nullable=False)
    age = db.Column(db.Integer, nullable=False)
    email = db.Column(db.String(100), unique=True, nullable=False)
    password = db.Column(db.String(100), nullable=False)
    gender = db.Column(db.String(10), nullable=False)
    mobile = db.Column(db.String(15), nullable=False)

# Home Page (Register Page)
@app.route('/')
def home():
    return render_template('register.html')

# Register Route
```

```

@app.route('/register', methods=['POST'])
def register():
    name = request.form['name']
    age = request.form['age']
    email = request.form['email']
    password =
bcrypt.generate_password_hash(request.form['password']).decode('utf-8')
    gender = request.form['gender']
    mobile = request.form['mobile']

    new_user = User(name=name, age=age, email=email, password=password,
gender=gender, mobile=mobile)
    db.session.add(new_user)
    db.session.commit()

    return redirect(url_for('login'))

# Login Route
@app.route('/login', methods=['GET', 'POST'])
def login():
    if request.method == 'POST':
        email = request.form['email']
        password = request.form['password']
        user = User.query.filter_by(email=email).first()

        if user and bcrypt.check_password_hash(user.password, password):
            session['user_id'] = user.id
            return redirect(url_for('index'))
        else:
            return 'Invalid Credentials'
    return render_template('login.html')

# Dashboard (Index Page)
@app.route('/index')
def index():
    if 'user_id' in session:
        user = User.query.get(session['user_id'])

```

```

        return render_template('index.html', user_name=user.name)
    return redirect(url_for('login'))

# Prediction Route
@app.route('/predict', methods=['POST'])
def predict():
    if 'user_id' in session:
        input_data = [float(request.form[key]) for key in
request.form.keys()]

        # Scaling Input Data
        scaled_data = scaler.transform([input_data])

        # Prediction
        prediction = model.predict(scaled_data)
        result = "Parkinson's Detected" if prediction[0] == 1 else "No
Parkinson's Detected"

        return render_template('result.html', prediction=result,
user_input=request.form)

    return redirect(url_for('login'))

# Logout Route
@app.route('/logout')
def logout():
    session.pop('user_id', None)
    return redirect(url_for('login'))

if __name__ == "__main__":
    with app.app_context():
        db.create_all()
        app.run(debug=True)

```

12.3 index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Parkinson's Disease Detection</title>
  <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.4.2/css/all.min.css">
  <link rel="stylesheet" href="{{ url_for('static', filename='style.css')
  }}">
</head>
<body>
  <!-- Navbar -->
  <nav class="navbar">
    <a href="/">Register</a>
    <a href="/login">Login</a>
    <a href="/logout">Logout</a>
  </nav>
  <div class="header-container">
    <h1 class="page-heading">Parkinson Disease Detection Using Machine
Learning</h1>
  </div>

  <div class="container">
    <h1>□ Parkinson's Disease Detection</h1>
    <h4>Enter the required values to check for Parkinson's disease:</h4>

    <form action="/predict" method="POST">
      <!-- MDVP (Fundamental Frequency Measures) -->
      <div class="input-group">
        <label>MDVP: Fo (Hz):</label>
        <input type="number" name="MDVP_Fo" step="any" required>
      </div>

      <div class="input-group">
```

```
<label>MDVP: Fhi (Hz):</label>
<input type="number" name="MDVP_Fhi" step="any" required>
</div>

<div class="input-group">
  <label>MDVP: Flo (Hz):</label>
  <input type="number" name="MDVP_Flo" step="any" required>
</div>

<div class="input-group">
  <label>MDVP: Jitter (%):</label>
  <input type="number" name="MDVP_Jitter" step="any" required>
</div>

<div class="input-group">
  <label>MDVP: Jitter (Abs):</label>
  <input type="number" name="MDVP_Jitter_Abs" step="any"
required>
</div>

<div class="input-group">
  <label>MDVP: RAP:</label>
  <input type="number" name="MDVP_RAP" step="any" required>
</div>

<div class="input-group">
  <label>MDVP: PPQ:</label>
  <input type="number" name="MDVP_PPQ" step="any" required>
</div>

<div class="input-group">
  <label>Jitter: DDP:</label>
  <input type="number" name="Jitter_DDP" step="any" required>
</div>

<div class="input-group">
  <label>MDVP: Shimmer:</label>
```



```

        <input type="number" name="MDVP_Shimmer" step="any"
required>
    </div>

    <div class="input-group">
        <label>MDVP: Shimmer (dB):</label>
        <input type="number" name="MDVP_Shimmer_dB" step="any"
required>
    </div>
    <div class="input-group">
        <label>Shimmer: APQ3:</label>
        <input type="number" name="Shimmer_APQ3" step="any"
required>
    </div>

    <div class="input-group">
        <label>Shimmer: APQ5:</label>
        <input type="number" name="Shimmer_APQ5" step="any"
required>
    </div>

    <div class="input-group">
        <label>MDVP: APQ:</label>
        <input type="number" name="MDVP_APQ" step="any" required>
    </div>

    <div class="input-group">
        <label>Shimmer: DDA:</label>
        <input type="number" name="Shimmer_DDA" step="any" required>
    </div>

    <div class="input-group">
        <label>NHR:</label>
        <input type="number" name="NHR" step="any" required>
    </div>

    <div class="input-group">

```

```
        <label>HNR:</label>
        <input type="number" name="HNR" step="any" required>
    </div>

    <div class="input-group">
        <label>RPDE:</label>
        <input type="number" name="RPDE" step="any" required>
    </div>

    <div class="input-group">
        <label>DFA:</label>
        <input type="number" name="DFA" step="any" required>
    </div>

    <div class="input-group">
        <label>Spread1:</label>
        <input type="number" name="spread1" step="any" required>
    </div>

    <div class="input-group">
        <label>Spread2:</label>
        <input type="number" name="spread2" step="any" required>
    </div>

    <div class="input-group">
        <label>D2:      </label><br>
        <input type="number" name="D2" step="any" required>
    </div>

    <div class="input-group">
        <label>PPE:</label>
        <input type="number" name="PPE" step="any" required>
    </div>
    <button type="submit" class="btn">Predict</button>
</form>
</div>
</body>
```

```
</html>
```

12.4 login.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Login Page</title>
  <link rel="stylesheet" href="{{ url_for('static', filename='style.css')
  }}">
</head>
<body>
  <div class="header-container">
    <h1 class="page-heading">Parkinson Disease Detection Using Machine
Learning</h1>
  </div>
  <div class="container">
    <h2><img alt="lock icon" data-bbox="248 505 265 522"/> Login</h2>
    <form method="POST" action="/login">
      <input type="email" name="email" placeholder="Email" required>
      <input type="password" name="password" placeholder="Password"
required>
      <button type="submit">Login</button>
    </form>
    <a href="/">Don't have an account? Register here.</a>
  </div>
</body>
</html>
```

12.5 register.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
```

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Register Page</title>
<link rel="stylesheet" href="{ url_for('static', filename='style.css')
}}">
<script>
    function validateForm() {
        let name = document.getElementById("name").value;
        let age = document.getElementById("age").value;
        let email = document.getElementById("email").value;
        let password = document.getElementById("password").value;
        let mobile = document.getElementById("mobile").value;

        // Name validation: only letters and spaces
        let nameRegex = /^[A-Za-z\s]+$/;
        if (!nameRegex.test(name)) {
            alert("Name should contain only letters and spaces.");
            return false;
        }

        // Age validation: must be between 10 and 120
        if (age < 10 || age > 120) {
            alert("Age must be between 10 and 120.");
            return false;
        }

        // Email validation
        let emailRegex = /^[^\s@]+@[^\s@]+\.[^\s@]+$/;
        if (!emailRegex.test(email)) {
            alert("Enter a valid email address.");
            return false;
        }

        // Password validation: At least 6 characters
        if (password.length < 6) {
            alert("Password must be at least 6 characters long.");
            return false;
        }
    }
}
```

```
// Mobile number validation: 10-digit number
let mobileRegex = /^[0-9]{10}$/;
if (!mobileRegex.test(mobile)) {
    alert("Enter a valid 10-digit mobile number.");
    return false;
}

return true;
}
</script>
</head>
<body>
    <div class="header-container">
        <h1 class="page-heading">Parkinson Disease Detection Using Machine
Learning</h1>
    </div>

    <div class="container">
        <h2>📝 Register</h2>
        <form method="POST" action="/register" onsubmit="return
validateForm()">
            <label for="name">Full Name</label>
            <input type="text" id="name" name="name" placeholder="Enter your
full name" required><br>
            <label for="age">Age</label><br>
            <input type="number" id="age" name="age" placeholder="Enter your
age" required>

            <label for="email">Email</label>
            <input type="email" id="email" name="email" placeholder="Enter
your email" required>

            <label for="password">Password</label>
            <input type="password" id="password" name="password"
placeholder="Enter your password" required>

```

```

        <label for="gender">Gender</label><br>
        <select id="gender" name="gender" required>
            <option value="Male">Male</option>
            <option value="Female">Female</option>
        </select><br>

        <label for="mobile">Mobile Number</label>
        <input type="text" id="mobile" name="mobile" placeholder="Enter
your mobile number" required>

        <button type="submit">Register</button>
    </form>
    <a href="/login">Already have an account? Login here.</a>
</div>
</body>
</html>

```

12.6 result.html

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Prediction Result</title>
    <link rel="icon" href="favicon.ico" type="image/x-icon">
    <link rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/6.4.2/css/all.min.css">
    <link rel="stylesheet" href="{{ url_for('static', filename='style.css')
}}">
</head>
<body>
    <!-- Navbar -->
    <nav class="navbar">
        <a href="/">Register</a>
        <a href="/login">Login</a>
    </nav>

```

```

    <a href="/logout">Logout</a>
</nav>

<div class="header-container">
    <h1 class="page-heading">Parkinson Disease Detection Using Machine
Learning</h1>
</div>

<div class="container">
    <h1>□ Prediction Result</h1>

    {% if prediction == "Parkinson's Detected" %}
        <p class="result positive">⚠ The model predicts a high
likelihood of Parkinson's Disease.</p>
    {% else %}
        <p class="result negative">✅ The model predicts a low
likelihood of Parkinson's Disease.</p>
    {% endif %}

    <h2>🔍 Input Data:</h2>

    <table class="styled-table">
        <thead>
            <tr>
                <th>Feature</th>
                <th>Value</th>
            </tr>
        </thead>
        <tbody>
            {% for key, value in user_input.items() %}
                <tr>
                    <td>{{ key }}</td>
                    <td>{{ value }}</td>
                </tr>
            {% endfor %}
        </tbody>
    </table>

```

```
        </table>
    </div>

    <footer>
        <p>&copy; 2025 Parkinson's Detection System</p>
    </footer>
</body>
</html>
```

12.7 style.css

```
/* Global Styling */
body {
    font-family: Arial, sans-serif;
    background-image: url('/static/images/brain.jpg'); /* Background Image */
    /*
    background-size: cover;
    background-position: center;
    background-attachment: fixed; /* Fixed background */
    margin: 0;
    padding: 0;
}

/* Navbar Styling */
.navbar {
    background-color: rgba(0, 0, 0, 0.6); /* Transparent Black */
    overflow: hidden;
    padding: 10px 20px;
    display: flex;
    justify-content: flex-end; /* Align items to the right */
}

.navbar a {
    color: white;
    padding: 14px 20px;
    text-decoration: none;
    margin: 0 10px;
```



```
        font-size: 16px;
    }

    .navbar a:hover {
        background-color: rgba(255, 255, 255, 0.2); /* Hover effect */
        border-radius: 5px;
    }

    /* Container Styling */
    .container {
        width: 40%;
        margin: 80px auto;
        background-color: rgba(0, 0, 0, 0.6); /* Transparent Black */
        padding: 20px;
        border-radius: 10px;
        box-shadow: 0 5px 10px rgba(0, 0, 0, 0.1);
        text-align: center;
        color: white; /* White text */
    }

    /* Input Field Styling */
    input, select {
        width: 90%;
        padding: 10px;
        margin: 10px 0;
        border: 1px solid #ccc;
        border-radius: 5px;
    }

    /* Button Styling */
    button {
        width: 92%;
        padding: 10px;
        background-color: #8e44ad;
        color: rgb(241, 238, 245);
        border: none;
        border-radius: 5px;
    }
```

```

        cursor: pointer;
    }

    button:hover {
        background-color: #732d91;
    }

    /* Link Styling */
    a {
        text-decoration: none;
        color: #8e44ad;
        font-size: 14px;
    }

    /* Prediction Result Page */
    .result {
        font-size: 18px;
        margin-bottom: 20px;
    }

    .positive {
        color: red;
        font-weight: bold;
    }

    .negative {
        color: green;
        font-weight: bold;
    }

    /* Table Styling */
    .styled-table {
        width: 100%;
        border-collapse: collapse;
        margin-top: 20px;
    }

```

```
.styled-table th, .styled-table td {
    padding: 10px;
    text-align: center;
    border: 1px solid #ddd;
}

.styled-table th {
    background-color: #8e44ad;
    color: rgba(234, 234, 242, 0.842);
}

.styled-table tr:nth-child(even) {
    background-color: #f2f2f2;
}

.styled-table tr:hover {
    background-color: #f2f2f2;
}

/* Footer */
footer {
    margin-top: 20px;
    font-size: 14px;
}

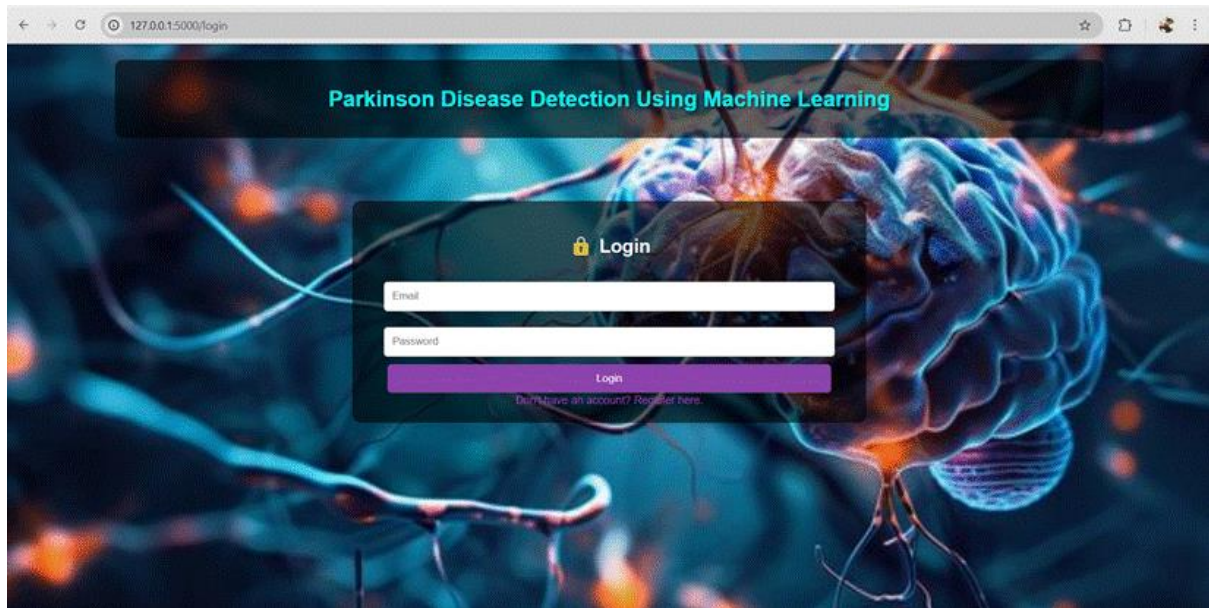
/* Header Container */
.header-container {
    background-color: rgba(0, 0, 0, 0.6); /* Transparent black */
    padding: 15px;
    text-align: center;
    border-radius: 10px;
    width: 80%;
    margin: 20px auto; /* Center the container */
}

/* Heading Styling */
.page-heading {
```

```
color: #00FFFF; /* Light cyan */  
font-size: 28px;  
font-weight: bold;  
text-shadow: 2px 2px 4px rgba(0, 0, 0, 0.6);  
}
```

13. Annexure 2 (Output Screens)

LOGIN PAGE:-



Parkinson Disease Detection Using Machine Learning

Login

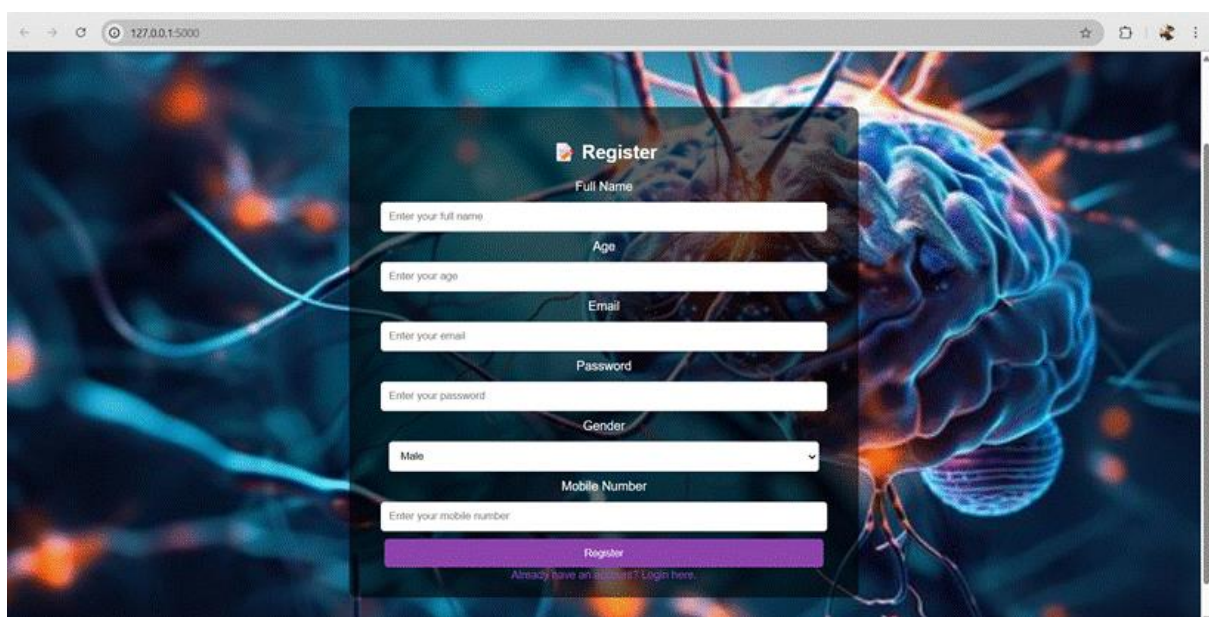
Email

Password

Login

[Don't have an account? Register here.](#)

REGISTRATION PAGE:-



Register

Full Name

Age

Email

Password

Gender

Mobile Number

Register

[Already have an account? Login here.](#)

PAGE (TO FILL UP THE VOICE PARAMETERS):-

Register Login Logout

Parkinson Disease Detection Using Machine Learning

Parkinson's Disease Detection

Enter the required values to check for Parkinson's disease:

MDVP: Fo (Hz): 119.99200

MDVP: Fhi (Hz): 156.302

MDVP: Flo (Hz): 74.99700

MDVP: Jitter (%): 0.00784

MDVP: Jitter (Abs): 0.00007

MDVP: RAP: 0.00370

MDVP: PPQ: 0.00554

Jitter: DDP: 0.01109

MDVP: Shimmer: 0.04374

MDVP: Shimmer (dB): 0.42600

Shimmer: APQ3: -0.97818

Shimmer: APQ5: 0.03130

MDVP: APQ: 0.02971

Shimmer: DDA: 0.06545

NHR: 0.02211

HNR: 21.03300

RPDE: -0.585217

DFA: 0.815285

Spread1: -4.813031

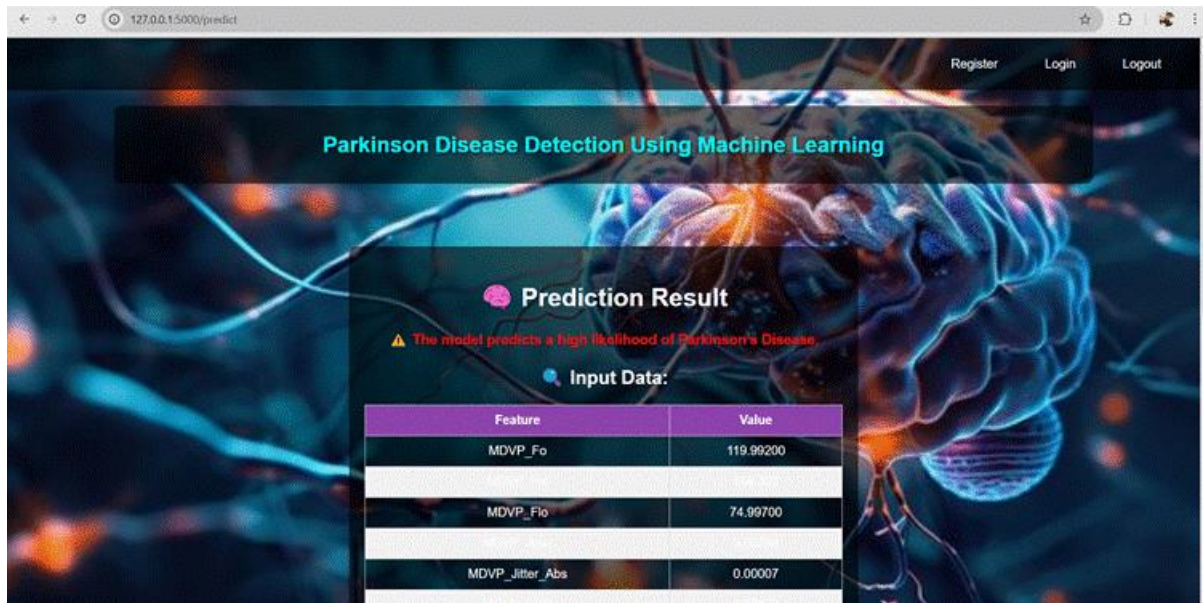
Spread2: 0.206482

D2: 2.301442

PPE: 0.284654

Predict

RESULT PAGE (Showing the likelihood of PD):-



14. Acceptance Mail from ICMIB 2025

----- Forwarded message -----

From: **Microsoft CMT** <email@msr-cmt.org>
 Date: Mon, Mar 10, 2025 at 12:06 AM
 Subject: Greetings from ICMIB-2025: Acceptance Letter
 To: Chandrakanta Mahanty <cmahanty@gilam.edu>

Dear Chandrakanta Mahanty,

Paper ID: 270
 Authors name: Chandrakanta Mahanty
 Title: Hybrid XGBoost-LSTM Model for Early Parkinson's Disease Identification Using Voice Data

Congratulations!!!

On behalf ICMIB-2025 Technical Program Committee, we are pleased to inform you that your paper with the review comments has been accepted for oral presentation at the International Conferences on Machine Learning, IoT and BigData (ICMIB-2025), which will be held during 4th to 6th April 2025 subject to publication standard and plagiarism/ similarity score as per springer publication house. This year, we have received a large number of good quality submissions, and the paper acceptance rate is only 20%. Hearty congratulations.

The reviewers' comments are included in the CMT account. Please address these comments while preparing your final camera ready manuscript.

In order for your paper to be included in the conference proceedings, your final manuscript must be compiled with the review comments and follow the Springer guidelines.

In order to abide by the publishing and printing schedule, please submit your complete publication package [Camera ready version of paper (in PDF format), Publishing Agreement (Copyright Form), Permission Form, paper source file (Word Document/ Latex file), Revision as per Reviewers Comment(s), Payment Receipt] in Zip format by emailing to: conference mail ID or before 20th March 2025. Further, it is also required to submit through the prescribed Google Form.

Now, please join the ICMIB whatsapp group to get the information as and when updated.

<https://chat.whatsapp.com/ITcFzkBd4qp1C2cf8JJzer>

For camera ready paper, you should go through the following.

1. Good quality, high resolution, and clear figures
2. Equation must be in formula format