VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"Jnana Sangama", Belagavi- 590018

PROJECT REPORT

ON

"HYBRID MACHINE LEARNING AND DEEP LEARNING FRAMEWORK FOR AUTISM DIAGNOSIS"

Submitted in partial fulfilment of the requirements for the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE & ENGINEERING

Under the Guidance of

Mr. Harish S B.E., M.Tech.,MIE
Assistant Professor
Department of Computer Science & Engineering
Adichunchanagiri Institute of Technology
Chikkamagaluru

Submitted By

SONALI M N (4AI21CS099) **SINDHU A M (4AI21CS097)**
SUBHIKSHA H R (4AI21CS103) **SANNIDHI G S (4AI20CS085)**

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
ADICHUNCHANAGIRI INSTITUTE OF TECHNOLOGY
(Affiliated to V.T.U., Accredited by NAAC)
CHIKKAMAGALURU-577102, KARNATAKA
2024- 2025

ADICHUNCHANAGIRI INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi) Chikkamagaluru, Karnataka, India-577102.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the Project work (21CSP76) entitled "**HYBRID MACHINE LEARNING AND DEEP LEARNING FRAMEWORK FOR AUTISM DIAGNOSIS**" is a bonafide work carried out by **Sonali M N (4AI21CS099)**, **Subhiksha H R (4AI21CS0103), Sindhu A M (4AI21CS097)**, **Sannidhi G S (4AI21CS085)**) in partial fulfilment for the award of Degree of Bachelor of Engineering in Computer Science and Engineering of the Visvesvaraya Technological University, Belagavi during the academic year **2024-2025**. It is certified that all corrections and suggestions indicated for Internal Assessment have been incorporated in the report deposited in the department library. The project report has been approved, as it satisfies the academic requirements in respect of Project Work prescribed for the said Degree.

Under the Guidance of
Mr. Harish S **B.E., M.Tech., MIE**
Assistant Professor
Dept. of CS&E
A.I.T Chikkamagaluru

Signature of the Project Coordinator
Mrs. Mithuna B. N **B.E., M.Tech**
Assistant Professor
Dept. of CS&E
A.I.T, Chikkamagaluru

Signature of the Project Coordinator
Mrs. Shruthi G.K **B.E., M.Tech**
Assistant Professor
Dept. of CS&E
A.I.T, Chikkamagaluru

Signature of the HOD
Dr. Pushpa Ravikumar **B.E., M.Tech., Ph.D**
Professor & Head
Dept. of CS&E
A.I.T, Chikkamagaluru

Signature of the Principal
Dr. C.T Jayadeva **B.E., M.Tech., Ph.D**
Principal
A.I.T, Chikkamagaluru

External Examiners **Signature with date**

ADICHUNCHANAGIRI INSTITUTE OF TECHNOLOGY

(Affiliated to Visvesvaraya Technological University, Belagavi)

Chikkamagaluru, Karnataka, India – 577102.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

APPROVAL

The Project Report ((21CSP76)) entitled **"HYBRID MACHINE LEARNING AND DEEP LEARNING FRAMEWORK FOR AUTISM DIAGNOSIS** "is here by approved as a credible study of engineering subject carried out and presented in a satisfactory manner for acceptance as a pre-requiestee to the degree of **BACHELOR OF ENGINEERING in COMPUTER SCIENCE AND ENGINEERING** during academic year 2024-25.

Submitted By:

SONALI M N (4AI21CS099)
SUBHIKSHA H R (4AI21CS0103)
SINDHU A M (4AI21CS097)
SANNIDHI G S (4AI21CS085)

Signature of the Guide
Mr. Harish S **B.E.,M.Tech.,MIE**
Assistant Professor
Dept. of CS&E
A.I.T Chikkamagaluru

Signature of the Project Coordinator
Mrs. Mithuna B.N **B.E.,M.Tech**
Assistant Professor
Dept. of CS&E
A.I.T Chikkamagaluru

Signature of the Project Coordinator
Mrs. Shruthi G K **B.E.,M.Tech.,**
Assistant Professor
Dept.of CS&E
A.I.T Chikkamagaluru

Signature of the HOD
Dr. Pushpa Ravikumar **B.E.,M.Tech.,Ph.D**
Professor and Head
Dept.of CS&E
A.I.T Chikkamagaluru

ABSTRACT

The project on autism prediction aims to leverage machine learning techniques for behavioural analysis and convolutional neural networks (CNNs) for image classification to improve the accuracy of early autism diagnosis. Autism Spectrum Disorder (ASD) is a complex neurodevelopmental disorder, and its early detection plays a critical role in providing timely interventions. Behavioural analysis using machine learning can help in identifying patterns in the behaviour of children, such as social communication difficulties and repetitive behaviours, which are characteristic of autism.

The combination of machine learning for behavioural analysis and CNN-based image classification presents a promising avenue for enhancing autism diagnosis, providing a tool for healthcare professionals to identify autism more accurately and at an earlier stage. This project could significantly aid in the development of automated diagnostic tools that can assist in early intervention, ultimately improving the quality of life for individuals with autism and their families.

ACKNOWLEDGEMENTS

We express our humble Pranamas to his holiness **Parama Poojya Jagadguru Padmabushana Sri Sri Sri Dr. Balagangadharanatha Mahaswamiji** and **Parama Poojya Jagadguru Sri Sri Sri Dr. Nirmalanandanatha Mahaswamiji** and alo to **Sri Sri Gunanatha Swamiji** Sringeri Branch Chikkamagaluru who have showered their blessings on us for framing our career successfully.

We are deeply indebted to our honorable Director **Dr. C K Subbaraya** for creating the right kind of care and ambience.

We are Thankful to our beloved Principal **Dr. C T Jayadeva** for inspiring us to achieve greater endeavors in all aspects of Learning.

We express our deepest gratitude to **Dr. Pushpa Ravikumar**, Professor and Head of the Department of Computer Science & Engineering for her valuable guidance, suggestions and constant encouragement without which success of our Project work would have been difficult.

We are Grateful to our project coordinators **Mrs. Mithuna B.N and Mrs. Shruthi G.K** Assistant Professor, Department of Computer Science and Engineering for their excellent guidance, constant encouragement, support and constructive suggestions. We are thankful to our guide **Mr. Harish S** Assistant Professor, Department of Computer Science and Engineering,Adichunchanagiri Institute of Technology, Chikkamagaluru for his inspiration and lively correspondence for carrying our project work.

We would like to thank our beloved parents for their support, encouragement and blessings. And last but not least, I would be pleased to express our heartfelt thanks to all teaching and non-teaching staff of Computer Science and Engineering Department and our friends who have rendered their help, motivation and support.

SONALI M N	**(4AI21CS099)**
SUBHIKSHA H R	**(4AI21CS103)**
SINDHU A M	**(4AI21CS097)**
SANNIDHI G S	**(4AI21CS085)**

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SNAPSHOTS

Chapter 1

INTRODUCTION

1.1 Introduction

Autism Spectrum Disorder (ASD) is a complex neurodevelopmental condition that affects individuals in various ways, primarily impacting their social interactions, communication skills, and behavior patterns. Early diagnosis of autism is crucial for timely intervention and effective treatment, as early behavioral therapy can significantly improve outcomes for individuals with autism. However, diagnosing autism can be challenging due to the variability in symptoms and the subjective nature of traditional diagnostic methods. This project aims to address this challenge by combining machine learning techniques for behavioral analysis and convolutional neural networks (CNNs) for image classification, providing a more accurate and reliable approach to autism prediction.

By using behavioral data, such as observations of social interaction, communication, and repetitive behaviors, machine learning models can be trained to identify characteristics indicative of autism. These models can assist clinicians in making more informed decisions and offer a more objective method of behavioral analysis, helping in the early identification of autism. This approach could improve diagnostic accuracy, reduce diagnostic delays, and minimize human bias in the process.

In addition to behavioral analysis, this project integrates CNN-based image classification for autism detection through brain imaging. Research has suggested that certain brain structure and function anomalies may be associated with autism, and CNNs are particularly well-suited for analyzing complex image data, such as MRI scans, to identify these patterns. By training CNN models on a large dataset of brain images, this project aims to develop an automated tool that can classify individuals as either autistic or non-autistic based on their brain scans. This dual approach of behavioral analysis combined with CNN-based image classification offers a comprehensive method to predict autism, which could ultimately aid in early diagnosis and more effective intervention strategies.

1.1.1 Overview of the Project

Autism Spectrum Disorder (ASD) is a neurodevelopmental disorder that affects a significant portion of the population, and its early diagnosis is vital for effective intervention and treatment. Early intervention can drastically improve the outcomes for individuals with autism, including improvements in social skills, communication, and behavioral development.

However, current diagnostic practices are subjective, based primarily on behavioral observations, which can vary significantly between individuals and clinicians. Additionally, autism presents with a broad spectrum of symptoms. In light of these challenges, the project aims to develop an automated prediction model for autism using machine learning techniques for behavioral analysis and convolutional neural networks (CNNs) for image classification, focusing on both behavioral data and medical imaging, such as brain scans.

These datasets are then processed and fed into machine learning algorithms such as decision trees, support vector machines (SVM), or random forests to identify potential indicators of autism. The goal is to develop a model that can accurately differentiate between children with and without autism, enabling faster and more objective assessments.

For this part of the project, MRI scan datasets of individuals diagnosed with autism and healthy controls are used to train a deep learning model. The CNN model is designed to learn spatial hierarchies of features within the brain images to classify them into categories such as "autistic" or "non-autistic." By processing these complex medical images, the model aims to uncover subtle differences in brain structure and activity that may be indicative of autism

The overall objective of this project is to develop a comprehensive tool for early autism prediction that combines behavioral analysis and image classification, offering healthcare professionals an automated and objective way to assess autism risk. The model's performance will be evaluated based on several metrics, including accuracy, sensitivity, specificity, and precision, to ensure it can reliably detect autism in different age groups and populations. This project aims to not only enhance the diagnostic process for autism but also contribute to the broader field of machine learning in healthcare by exploring the potential of combining behavioral data and medical imaging for predictive modeling.

1.1.2 Data Preprocessing

Data preprocessing is an essential and foundational step in the process of building accurate predictive models for autism diagnosis. Whether using machine learning techniques for behavioral analysis or deep learning models like convolutional neural networks (CNNs) for image classification, preprocessing ensures that the data used is clean, relevant, and in a suitable format for the models to learn from. In the context of autism prediction, data preprocessing involves cleaning and transforming both behavioral data (such as questionnaires, test results, and clinical observations) and medical image data (such as MRI scans).

Data Cleaning

Data cleaning refers to the process of addressing issues with raw data, such as missing values, duplicates, and inconsistencies, to ensure that the dataset is reliable and accurate for use in machine learning and deep learning models. In the autism prediction project, data cleaning is a crucial step for both behavioral and image datasets, ensuring that the models can be trained on high-quality data that will lead to better performance. Below are the key steps involved in cleaning both types of data:

1. **Remove Duplicate or Irrelevant Observations**: Duplicates in the dataset, especially in behavioral data, can lead to overfitting and inaccurate model performance. These duplicates should be identified and removed. In the case of medical imaging data, irrelevant images or non-pertinent scans must be filtered out to focus only on those that are related to autism diagnosis.
2. **Fix Structural Errors**: Structural errors in the dataset can occur due to inconsistencies in the way data is recorded or formatted. For instance, if behavioral data collected from different sources (questionnaires, tests, etc.) is not in a consistent format, it needs to be standardized. Similarly, for MRI scan images, ensuring that all images are of the same resolution and orientation is essential for effective processing and analysis.
3. **Filter Unwanted Outliers**: Outliers in both behavioral data and image data can significantly affect the model's performance by skewing the results. For example, an unusually high or low score on a behavioral test, which is not representative of the broader population, can lead to misleading conclusions. For image data, pixel anomalies or corrupted images can distort the analysis.
4. **Handle Missing Data**: Missing data is common in many real-world datasets. Various methods for handling missing data include imputation (replacing missing values with estimates based on other data points), or in some cases, removing records with excessive missing data. For medical image data, missing or incomplete image sets may need to be addressed by removing the associated images or by filling in missing metadata, if applicable.

5. **Validate the Data**: After cleaning, the dataset must be validated to ensure that it is accurate, consistent, and free from errors. This involves cross-checking the data with external sources or expert validation to confirm the integrity of the dataset.

Data Transformation

Once the data is cleaned, the next step is to transform it into a format suitable for model training. In the case of behavioral data, this may involve normalizing or scaling numerical values, encoding categorical features (e.g., gender or autism diagnosis status), and possibly engineering new features to enhance model performance. For image data, transformations such as resizing images to a uniform dimension, normalization of pixel values, and augmentation techniques (rotations, flips, etc.) may be employed to increase the diversity of the dataset and improve the robustness of the CNN model.

Data Preprocessing in Autism Prediction

Data preprocessing involves a series of techniques that prepare raw data for analysis by machine learning and deep learning models. The purpose of data preprocessing is to clean, transform, and structure the data in such a way that models can use it effectively to make accurate predictions. In the case of autism prediction, both behavioral and image data require careful handling to ensure that they are ready for analysis.

Data Cleaning and Outlier Detection

Data cleaning is the first step in preparing the dataset for machine learning. Missing data can occur for various reasons, such as incomplete responses in questionnaires or tests, or errors during data collection. One common approach to handling missing data is imputation, where missing values are replaced with the mean, median, or mode of the available data. In cases where there is excessive missing data, the affected records may need to be removed entirely.

Normalization and Scaling

Once data is cleaned, the next step in preprocessing is normalization and scaling.For example, the age of a child in years and the number of hours spent engaging in social activities may have vastly different ranges, making it difficult for the model to interpret the relationships between these variables. Normalization transforms each feature so that it falls within a similar range, such as between 0 and 1, or to a standard normal distribution with a mean of 0 and a standard deviation of 1.

In image data, normalization is also essential. MRI scans typically contain pixel values ranging from 0 to 255, representing different intensities of light. Normalizing the pixel values to a range between 0 and 1 helps the model to process the images more efficiently. CNNs perform better when the input data is scaled to a consistent range, as this reduces the variance in pixel intensity values and accelerates the convergence of the training process.

Encoding Categorical Variables

In the case of behavioral data, encoding categorical variables is necessary to ensure that they can be used by machine learning models. Many behavioral datasets include categorical features, such as gender, autism diagnosis status, or family history of autism.

Image Preprocessing

In addition to behavioral data, this project also uses medical images, specifically MRI scans, to predict autism.First, image resizing is applied to ensure that all images are of the same dimension, as neural networks require consistent input sizes. Common sizes for MRI images include 224x224 or 256x256 pixels, but the dimensions may vary depending on the specific CNN architecture being used.

Another key aspect of image preprocessing is image augmentation, which involves creating modified versions of the original images by applying transformations such as rotation, flipping, cropping, and zooming. Image augmentation increases the diversity of the training dataset, which helps prevent overfitting and improves the model’s generalization ability

Feature Engineering

Feature engineering is the process of creating new features from existing data to improve the performance of machine learning models. In behavioral data, feature engineering can involve combining multiple features or transforming them to better represent the underlying patterns. For example, features related to social behavior, communication skills, and sensory sensitivity might be combined into a single composite score that captures a child’s overall social communication abilities.

1.2 Motivation

Autism Spectrum Disorder (ASD) affects millions of individuals worldwide, with early diagnosis and intervention being crucial for improving long-term outcomes. The motivation behind this project is to overcome these limitations by leveraging machine learning and deep learning techniques to develop a more objective, data-driven approach for autism prediction, providing clinicians with a tool to make more reliable and early diagnoses.

Early detection of autism is critical because it allows for timely intervention, which can greatly improve the quality of life for individuals with autism. Research has shown that early interventions focused on enhancing social communication, behavior, and learning skills can lead to better developmental outcomes.This early identification can facilitate prompt intervention, which is crucial for improving developmental trajectories.

The long-term vision of this project extends beyond the development of an accurate diagnostic tool. By providing a more reliable and scalable way to predict autism, the project has the potential to reduce the diagnostic delay, lower the costs of evaluation, and minimize the dependence on highly specialized expertise. This can lead to more accessible healthcare, particularly in underserved regions where autism specialists may be scarce.

1.3 Problem Statement

Input : The primary inputs for this project are two distinct datasets: behavioral data and medical imaging data.

1. **Behavioral Data:** This consists of a wide range of observational and self-reported information gathered through questionnaires, structured behavioral assessments, and clinical evaluations. Behavioral data includes information such as social communication skills, repetitive behaviors, language development, sensory processing patterns, and responses to stimuli. These datasets are usually obtained from parents, caregivers, and clinicians.
2. **Medical Imaging Data (MRI Scans):** In addition to behavioral data, medical imaging, particularly MRI scans, is a crucial input. These images provide detailed insights into brain structure and function, which may reveal biomarkers associated with autism. MRI datasets can include brain scans of individuals diagnosed with autism as well as healthy control groups, highlighting structural and functional differences in the brain

Process : The process involves several stages of data preparation, machine learning model training, and prediction using behavioral and medical imaging data.

1. **Data Preprocessing:** The first step in the process is the preprocessing of both behavioral and imaging data to make them usable for machine learning models.

   - **Behavioral Data Preprocessing**: This includes handling missing values, normalizing data, encoding categorical features, and detecting and removing outliers. Behavioral data is cleaned, normalized, and transformed into numerical features for the machine learning algorithms.
   - **Medical Imaging Data Preprocessing**: For MRI scans, preprocessing involves resizing images to a uniform dimension, normalization of pixel values, and applying data augmentation techniques to enhance the diversity of the dataset and prevent overfitting. These preprocessing steps ensure that the images are suitable for input into convolutional neural networks (CNNs), which will learn to recognize patterns in brain structure that correlate with autism.
2. **Model Training**: After preprocessing, the next step is the application of machine learning and deep learning algorithms to train models.
   - **Behavioral Prediction Model**: For the behavioral data, machine learning algorithms such as decision trees, support vector machines (SVM), or random forests can be applied. These models are trained to recognize patterns in the behavioral data that are indicative of autism. Feature engineering techniques might be used to extract the most relevant behavioral traits for predicting autism.
   - **Image Classification with CNNs**: For the MRI image data, deep learning models, specifically CNNs, are used to analyze the spatial patterns within the brain scans. CNNs are well-suited for image classification tasks and can automatically detect features such as brain structure anomalies, abnormal connectivity between brain regions, or differences in brain size, all of which may be linked to autism. CNNs are trained on labeled MRI datasets to classify images into categories such as "autistic" and "non-autistic."
3. **Model Evaluation**: Once the models have been trained, their performance is evaluated using various metrics like accuracy, sensitivity, specificity, and precision. These metrics help assess how well the models predict autism, both on behavioral and imaging data. Cross-validation techniques are also employed to validate the models' ability to generalize to new, unseen data.
4. **Hybrid Model**: The final step in the process is the integration of the behavioral and image classification models. Since both types of data offer valuable insights into autism, the

process involves combining the predictions made by the machine learning model based on behavioral data with the predictions made by the CNN model using MRI scan images.

Output : The output of this project is a comprehensive and automated autism prediction system capable of providing reliable predictions based on both behavioral and medical imaging data.

1. **Behavioral Prediction**: The output of the behavioral prediction model is a classification of whether an individual shows signs of autism based on behavioral data. This is typically presented as a binary classification (e.g., "Autistic" or "Non-Autistic"), which can assist clinicians in early identification. Additionally, the output may include an explanation of the key behavioral traits that contributed to the prediction, aiding in clinical decision-making.
2. **Image Classification**: The output of the CNN-based image classification model is a prediction based on MRI scan images. The model classifies the brain images into categories such as "Autistic" or "Non-Autistic," depending on the structural patterns identified by the neural network. This provides a neurological perspective on autism and supports clinicians by offering an additional layer of diagnostic information.
3. **Hybrid Output**: The final output is a combined prediction that integrates both behavioral and image-based analysis. This hybrid output leverages the strengths of both data sources to increase the accuracy of autism prediction. The result is a single, unified prediction that informs clinicians about the likelihood of autism in a patient, incorporating both the behavioral signs and the brain imaging features.
4. **Visualization and Interpretation**: In addition to the predictions, the system can provide visualizations of the behavioral and imaging data that contributed to the output. This may include graphs or heatmaps for the behavioral model, and segmented brain images for the CNN model, showing which areas of the brain were important for classification. These visual aids help clinicians interpret the model's decision and provide a more comprehensive understanding of the prediction.

1.4 Scope of the Project

The scope of this project revolves around the development of an advanced autism prediction system utilizing machine learning (ML) and deep learning (DL) techniques. This system aims to enhance the accuracy and efficiency of autism diagnosis by integrating both behavioral data and medical imaging data, such as MRI scans, to provide a comprehensive and data-driven approach.

The primary focus of the project will be on data collection, preprocessing, and model development. The scope of the preprocessing phase will include handling issues like missing values, inconsistencies, outliers, and irrelevant data in the behavioral dataset, as well as addressing image quality issues in the MRI data.

Furthermore, the scope of this project includes the development of an intuitive user interface (UI) for healthcare professionals, allowing them to easily interact with the prediction system. The system will be designed to accept both behavioral data and MRI images as input, and generate predictions along with relevant visualizations. This user-friendly interface will allow clinicians to input patient data and receive quick, data-driven predictions about the likelihood of autism.

In conclusion, the scope of this project is broad, covering everything from data collection and preprocessing to model development, evaluation, and deployment. The integration of both behavioral and medical imaging data, coupled with advanced machine learning and deep learning techniques, has the potential to significantly improve autism prediction and early detection. By providing a reliable, data-driven tool for clinicians, the project aims to facilitate earlier diagnosis and intervention, ultimately enhancing the quality of life for individuals affected by autism spectrum disorder.

1.5 Objectives

- To develop a machine learning model that predicts autism based on behavioral data.
- To implement a convolutional neural network (CNN) for autism classification using MRI brain scans.
- To combine behavioral data and medical imaging data to create a hybrid prediction model.
- To preprocess and clean behavioral and MRI scan data for accurate model training.
- To evaluate and validate the developed models for reliable and early autism diagnosis.

1.6 Review of Literature

[1] Volume-based analysis of the amygdala and hippocampal subfields in infants at risk of ASD

- **Authors**: Guannan Li et al.
- **Year of Publication**: Not mentioned

- **Method**: This study proposed a deep learning approach, specifically the dilated-dense U-Net architecture, to segment the amygdala and hippocampal subfields in infants at risk of autism spectrum disorder (ASD) at around 24 months of age. The model automates the segmentation process, which typically requires expert manual intervention.
- **Pros**:
    - The use of a deep-learning approach, particularly the dilated-dense U-Net, is an advanced method that effectively handles the segmentation of brain subfields, which are crucial in understanding the neurological markers associated with ASD.
    - Accurate segmentation of the amygdala and hippocampal regions could provide meaningful insights into the developmental anomalies linked with autism risk.
- **Cons**:
    - One of the challenges faced in this study is the low tissue contrast in the imaging data, which can affect the accuracy of the segmentation algorithm. These difficulties are common in neuroimaging studies, especially when dealing with infant brains or early developmental stages.

[2] Visual Communication Learning for Children with Autism

- **Authors**: Fitrilia Susanti et al.
- **Year of Publication**: Not mentioned
- **Method**: This study developed a digital media-based learning system using the Picture Exchange Communication System (PECS) to assist children with autism in improving their communication skills. PECS is a system where children use pictures to express their thoughts or needs, making it easier for non-verbal children to communicate.
- **Pros**:
    - This system focuses on improving communication through visual learning, which aligns with the cognitive strengths many children with autism possess, as they tend to respond better to visual stimuli than to verbal communication.
    - The use of digital media can make the learning process more engaging and accessible to children, which may facilitate better interaction and expression.

- **Cons**:
  - While PECS has proven effective, it is primarily limited to non-verbal communication. It may not be suitable for children with verbal communication abilities, thus limiting its overall applicability for all children with autism.

[3] Diagnostic Challenges in Autism Detection

- **Authors**: Leslie Mertz et al.
- **Year of Publication**: Not mentioned
- **Method**: This paper highlights the diagnostic challenges in autism, especially in the United States, where the average age of diagnosis is 4.3 years, significantly later than the ideal window for early intervention (around age two). The study emphasizes the role of trained clinicians in identifying ASD and discusses the shortage of qualified professionals available to diagnose milder cases.
- **Pros**:
  - The emphasis on early diagnosis is critical, as research shows that early intervention can dramatically improve outcomes for children with autism.
  - The paper raises awareness of the significant challenges in diagnosing autism at an early stage, which could inspire policy changes or initiatives to train more professionals in the field.
- **Cons**:
  - The primary challenge highlighted is the delay in diagnosis due to a shortage of skilled professionals. While the paper brings attention to this issue, it does not propose a direct solution to address the shortage, which could limit the practical impact of the findings.

[4] Classification Techniques for ASD Diagnosis

- **Authors**: Osman et al.
- **Year of Publication**: Not mentioned
- **Method**: This study used machine learning techniques, specifically K Nearest Neighbor (KNN) and Linear Discriminant Analysis (LDA), on a dataset of children aged 4-11 years to classify autism. The data was split into 70% for training and 30% for testing, and the LDA model achieved 90.8% accuracy, while the KNN model achieved 88.5% accuracy.

- **Pros**:
  - High classification accuracy with LDA (90.8%) and KNN (88.5%) demonstrates the potential of machine learning in diagnosing autism. This result is promising and could serve as a foundation for creating automated diagnostic tools.
- **Cons**:
  - The limited size of the dataset used for testing may not provide a comprehensive evaluation of the models' effectiveness, especially in diverse populations or different clinical settings.

1.7 Organization of the Report

The report is organized into the chapters as follows:

Chapter 1- Introduction: The chapter presents a brief description about Polycystic ovary syndrome and its detection.

Chapter 2 - System requirement specification: The chapter 2 presents the specific requirement, software and hardware requirements interfaces used. It also presents a brief summary about the chapter.

Chapter 3 - High level Design: The chapter 3 presents the Design consideration made, system architecture of proposed system, specification of the proposed system using use case diagram. It also describes module specification, data flow diagram for every module and state chart for the proposed method. it presents a brief summary about the chapter.

Chapter 4 - Detail design: The chapter 4 briefs about the structural chart diagram and detail functionality and description of each module.

Chapter 5 – Conclusion: The chapter 5 describes the Conclusion of the project.

1.8 Summary

The first chapter gives the brief introduction about the importance of detection of autism spectrum using Machine learning and Deep learning techniques. The motivation of project is discussed in section 1.2 Section 1.3 presents the problem statement of the project. Scope of the project is described in the section 1.4 and objectives are presented in the section 1.5. Finally, section 1.6 elaborates the reviews of literature, the important papers referred. The papers we have reviewed here gives us a glimpse of how accuracy of machine learning systems directly depends on the size of the training dataset-the larger the training dataset, then better the accuracy of the trained model.

Chapter 2

SYSTEM REQUIREMENT SPECIFICATION

The System Requirement Specification (SRS) for the autism prediction project outlines the necessary hardware and software components required to build and deploy the machine learning and deep learning models. The system needs a powerful computing environment with a multi-core processor (Intel i5 or higher) and a minimum of 16GB RAM for efficient data processing and model training. A dedicated GPU (such as NVIDIA GTX 1080 or higher) is essential for accelerating the training of the convolutional neural network (CNN) for image classification. The system should support Python 3.6+ and relevant machine learning libraries, including TensorFlow, Keras, Scikit-learn, and OpenCV for image preprocessing. For data storage, a high-capacity hard drive (1TB or more) is recommended to handle large datasets, including behavioral data and MRI scans.

Additionally, the system must be equipped with necessary data acquisition tools for collecting behavioral data (questionnaires, clinical reports) and access to MRI scan datasets. The operating system should be Windows, Linux, or macOS, with internet access for downloading libraries and datasets, and cloud services (e.g., AWS, Google Cloud) may be used for distributed computing if required.

2.1 Specific Requirements

Python

Programming language used to design the proposed method is Python. Python is a high-level programming language with dynamic semantics. It is an interpreted language i.e. interpreter executes the code line by line at a time, thus makes debugging easy Python Imaging Library (PIL) is one of the popular libraries used for image processing. PIL can be used to display image, create thumbnails, resize, rotation, convert between file formats, contrast enhancement, filter and apply other digital image processing techniques etc.

OpenCV – Python Tool

OpenCV-Python is a library of Python bindings designed to solve computer vision problems. Visual information is the most important type of information perceived, processed and interpreted by the human brain. Image processing is a method to perform some operations on an image, in order to extract some useful information from it. An image is nothing more than a two dimensional matrix (3-D in case of colored images) which is defined by the mathematical function f(x,y) where x and y are the two co-ordinates horizontally and vertically. The value of f(x,y) at any point is gives the pixel value at that point of an image.

TensorFlow

TensorFlow is an open-source machine learning framework developed by the Google Brain team, designed to facilitate the development and deployment of machine learning models. Known for its flexibility, scalability, and extensive community support, TensorFlow has become a cornerstone in the field of deep learning. One of its notable features is its computational graph paradigm, where operations are represented as nodes in a graph, allowing for efficient execution on CPUs, GPUs. This flexibility enables researchers and developers to seamlessly transition from prototyping models on a personal machine to scaling them up for production environments.

Scikit learn

Scikit learn a popular open-source machine learning library for Python, serves as a versatile and user-friendly tool for a wide range of machine learning tasks. Developed on the principles of simplicity and effectiveness, scikit-learn provides a consistent interface for various algorithms, making it accessible to both novice and experienced practitioners.

2.2 Hardware Requirements

- Processor : Intel Core i3 Processor
- RAM : 4 GB
- Speed : 2.2GHZ

2.3 Software Requirements

- Programming Language: Python
- Front End : HTML, CSS.
- Code Editor : VS Code
- Operating System : Windows 7 and above.

2.4 Functional Requirements

Useful requirements describe the product's internal activities: that is, the technical subtitles, monitoring and handling of data and other specific functionality demonstrating how to satisfy the use cases. They are upheld by non-utilitarian prerequisites the force the plan or execution of imperatives.

- System should process the data.
- System should segment ovary Ultrasound scan image.
- System should detect Follicles.

2.5 Non-Functional Requirements

- Usability: System should be user friendly.
- Reliability: The system should be reliable.
- Performance: The system should not take excess time in detecting polycystic ovarian syndrome.
- Supportability: System should be easily updatable for future enhancement.

2.6 Summary

The chapter 2 considers all the system requirements which is required to develop this proposed system. The specific requirements for this project have been explained in section 2.1. The hardware requirements for this project have been explained in section 2.2 The backend software is clearly explained in section 2.3. The functional and non-functional requirements are explained in the section 2.4 and 2.5 respectively.

In essence, the summary of the Software Requirements Specification encapsulates the essential components of the document, offering a high-level understanding of the software's purpose, functionality, and the criteria that will define its success. This document serves as a crucial reference for both the development team and throughout the software development life cycle.

Chapter 3

HIGH LEVEL DESIGN

High-level design (HLD) explains the architecture that would be used for developing a software product. The architecture diagram provides an overview of an entire system, identifying the main components that would be developed for the product and their interfaces. The HLD uses possibly non-technical to mildly technical terms that should be understandable to the administrators of the system. In contrast low level design further exposes the logical detailed design of each of these elements for programmers.

High level design is the design which is used to design the software related requirements.In this chapter complete system design is generated and shows how the modules, sub modules and the flow of the data between them are done and integrated. It is very simple phase that shows the implementation process. The errors done here will be modified in the coming processes.

3.1 Design Considerations

The design of the autism prediction system involves several critical decisions to ensure that the final solution is both efficient and effective. These design considerations focus on key areas such as data handling, system architecture, model selection, user interface, and overall scalability. Each aspect must be carefully planned to meet the specific requirements of the project and ensure that the system can provide accurate, timely, and reliable predictions for autism spectrum disorder (ASD) diagnosis.

1. Data Collection and Preprocessing

The design of the system must account for the complexity and variability of the data sources. Two main types of data will be used in the system- behavioral data and neuroimaging data (specifically MRI scans).

Both data types present unique challenges and considerations for preprocessing:

Behavioral Data: This data will include information from questionnaires, clinical observations, and caregiver inputs. Behavioral data often suffer from missing values, inconsistencies, and outliers. A key design decision will be to use robust preprocessing techniques like data imputation for missing values and anomaly detection methods (e.g., Z-score or IQR) to identify and handle outliers. Additionally, the system must normalize numerical data and encode categorical features to ensure that the data is in a format suitable for machine learning models.

Neuroimaging Data (MRI Scans): The preprocessing of MRI scans requires techniques such as resizing, normalization, and data augmentation. Design considerations for image preprocessing will ensure that the images are compatible with deep learning models, particularly Convolutional Neural Networks (CNNs), which will be used for image classification. MRI scans often vary in quality and resolution, so applying standardization and augmentation methods like rotation, flipping, and zooming will help improve model robustness and reduce overfitting.

2. Model Selection and Integration

A significant design consideration is the choice of models for both behavioral data analysis and image classification.

The decision will depend on the type of data and the expected output:

Machine Learning Models for Behavioral Data: Given that behavioral data is often structured, traditional machine learning models like Decision Trees, Random Forests, and Support Vector Machines (SVMs) will be considered. These models are well-suited for handling numerical and categorical data, providing interpretability and efficiency. For multi-class classification tasks, these models can be trained on different behavioral features to predict the likelihood of autism based on individual behavioral indicators.

Convolutional Neural Networks (CNN) for MRI Image Classification: For the image classification task, CNNs will be used to classify MRI scans. CNNs are particularly effective in capturing spatial hierarchies in image data, making them ideal for medical image analysis. However, CNNs require substantial computational resources and large datasets to achieve optimal performance. To overcome potential limitations, design considerations include using pre-trained networks with transfer learning to leverage existing knowledge and improve model accuracy with a smaller dataset.

Hybrid Model: One of the most critical design choices is integrating the machine learning model for behavioral data with the deep learning model for MRI scans into a hybrid model. This integrated approach is expected to provide a more comprehensive prediction by combining the strengths of both data types. The hybrid model should be able to handle inputs from both behavioral data and MRI images and produce a unified output (i.e., autism prediction).

3. System Architecture

The system architecture must be designed to handle large volumes of data and support multiple concurrent users, particularly in clinical or research settings. The architecture should be modular and scalable to allow for easy updates and integration with new data sources or machine learning models.

Key architectural considerations include :

Backend Architecture: The backend will consist of a server that handles data storage, model training, and predictions. It will be responsible for processing raw behavioral data, preprocessing MRI scans, and invoking machine learning models. Cloud-based computing resources may be employed to provide scalable computing power for training the models.

Data Storage: The system will require secure and efficient data storage to handle both structured (behavioral data) and unstructured (MRI images) data. A relational database will store the behavioral data, while a cloud storage service will be used for storing MRI images.

Model Deployment: After training the machine learning and deep learning models, the deployment process should allow for seamless integration of the models into the system. The system will need a model-serving infrastructure, such as TensorFlow Serving or FastAPI, that can handle requests for predictions in real-time.

4. User Interface Design

The user interface (UI) must be designed to accommodate healthcare professionals and researchers who will interact with the system. It needs to be intuitive, user-friendly, and capable of presenting complex data and model outputs in an understandable way.

Design considerations for the UI include:

Data Input Forms: The UI should provide easy-to-use forms for entering behavioral data (e.g., questionnaires, observations) and uploading MRI images. These forms should be designed to minimize user input errors and ensure data consistency.

Results Visualization: The system should present model predictions and performance metrics clearly. This may include displaying the autism diagnosis (positive or negative) along with a confidence score. Visualizations, such as confusion matrices, ROC curves, and accuracy graphs, should be included to help users understand the model's performance.

User Roles and Permissions: The system may support different user roles, such as administrators, healthcare professionals, and researchers, with varying levels of access to features.

5. Scalability and Future Enhancements

Scalability is a key consideration in the design, especially as the system may need to handle increasing amounts of data as more patients are processed or new datasets become available.

Key design elements for scalability include:

Cloud Integration: Leveraging cloud platforms allows the system to scale based on demand, enabling the use of distributed computing for large-scale model training and data processing.

Model Retraining: As new data becomes available, the system should allow for periodic retraining of models to incorporate fresh insights and improve accuracy. The design should include version control for models, so the system can track and deploy updates seamlessly.

3.2 System Architecture

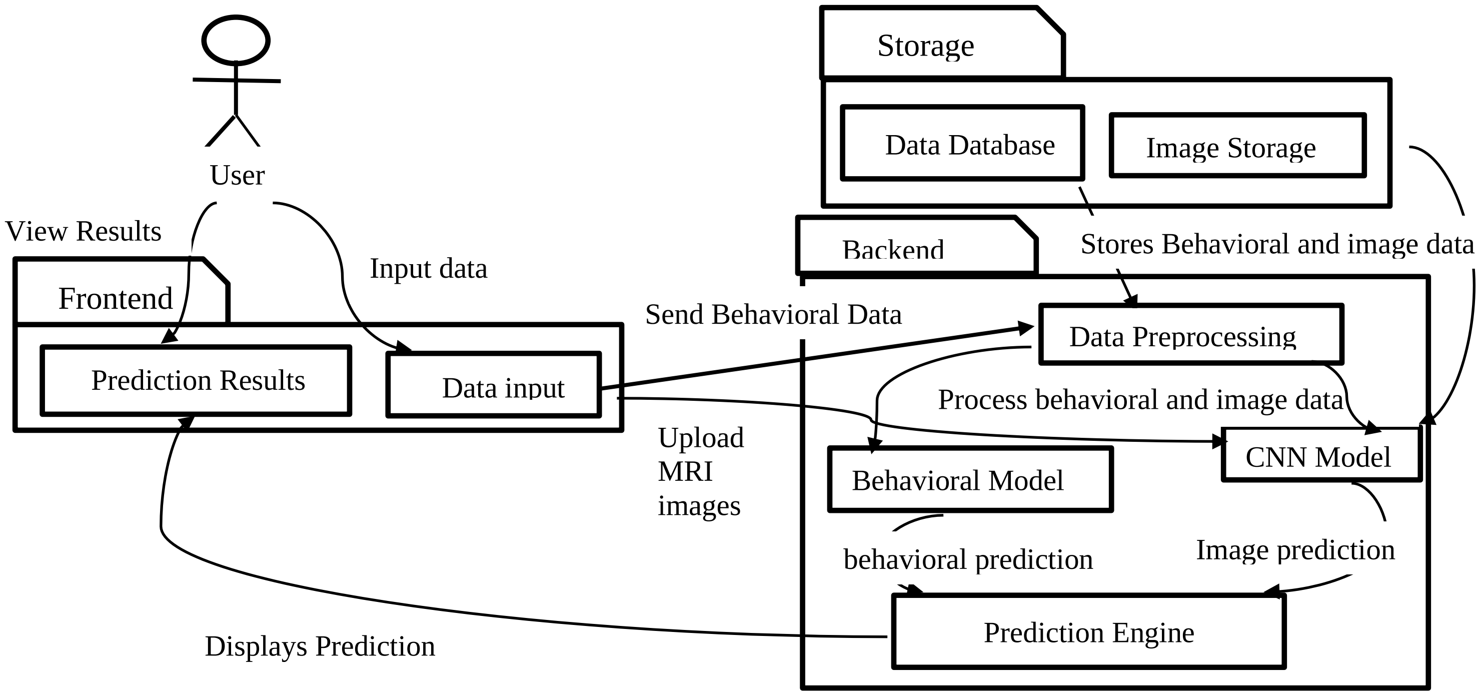


Figure 3.1 : Architecture Diagram of Autism Prediction System

1. User (Actor)

- **Role**: The user interacts with the system. This could be a clinician, researcher, or a caregiver who inputs data and views predictions.
- **Interactions**:
  - The user enters behavioral data through the Data Input Form.
  - The user views the results in the Prediction Results section.

2. Frontend (User Interface)

- **Data Input Form**:
  - **Purpose**: The form where the user enters various data, such as behavioral information or uploads MRI images of the subject (patient).

    - **Interaction**:
        - It sends the behavioral data to the Data Preprocessing module in the backend.
        - If MRI images are uploaded, it sends them to the CNN Model for processing.
- **Prediction Results**:
    - **Purpose**: This component displays the final predictions generated by the backend system.
    - **Interaction**:
        - It receives the final predictions from the Prediction Engine and shows them to the user.

3. Backend (Processing)

- **Data Preprocessing**:
    - **Purpose**: This module handles data cleaning, normalization, and transformation tasks to ensure that both the behavioral data and images are in the correct format for the models.
    - **Interaction**:
        - It processes the behavioral data and sends it to the Behavioral Model.
        - It also processes image data and sends it to the CNN Model.
- **Behavioral Model**:
    - **Purpose**: This could involve using models such as Decision Trees, Random Forests, or Support Vector Machines (SVMs).
    - **Interaction**:
        - It receives preprocessed behavioral data and sends predictions to the Prediction Engine.
        -
- **CNN Model (Convolutional Neural Network)**:
    - **Purpose**: This deep learning model processes MRI images to predict the likelihood of autism based on image features. CNNs are especially effective in extracting spatial patterns from images, making them ideal for medical image classification tasks.

  - **Interaction**:
    - It receives preprocessed MRI images and sends predictions to the Prediction Engine.
- **Prediction Engine**:
  - **Purpose**: This module combines the results from both the Behavioral Model (machine learning) and the CNN Model (deep learning) to generate a final prediction (e.g., whether the subject is predicted to have autism or not).
  - **Interaction**:
    - It receives predictions from both models and integrates them to provide a comprehensive result to the Prediction Results section of the frontend.

4. Storage

- **Data Database**:
  - **Purpose**: Stores behavioral data used for training the machine learning model. This data could include behavioral observations, questionnaires, or assessments related to autism.
  - **Interaction**:
    - The Data Preprocessing module retrieves or stores data in the Data Database for further processing.
- **Image Storage**:
  - **Purpose**: Stores MRI images that are used as input for the CNN Model.
  - **Interaction**:
    - The CNN Model retrieves images from the Image Storage for analysis.

Flow of Data and Predictions:

1. **User Input**: The user enters behavioral data and optionally uploads MRI images using the Data Input Form.
2. **Data Preprocessing**: The input data is processed (e.g., cleaned and normalized) by the Data Preprocessing module.
3. **Model Processing**:
   - The behavioral data is passed to the Behavioral Model for machine learning-based prediction.
   - The MRI images are passed to the CNN Model for deep learning-based prediction.

4. **Prediction Integration**: The predictions from both the Behavioral Model and CNN Model are passed to the Prediction Engine, which combines the results into a final prediction (e.g., autism risk or diagnosis).
5. **Display Results**: The final prediction is shown to the user in the Prediction Results section of the frontend.

Coloring and Design Considerations:

- **Color Scheme**: The use of blue for classes and components, with distinct areas for frontend, backend, and storage, helps visually distinguish between the different sections of the system.
- **Organization**: The flow of data is easy to follow, from user input to the backend processing and finally to the results display.

The research was carried out in four phases

1. User (Actor)

- **Description**: The User represents the person interacting with the system. In this case, the user could be a healthcare professional (such as a doctor, therapist, or clinician), a researcher, or a caregiver.
- **Responsibilities**:
  - **Data Input**: The user inputs behavioral data and uploads MRI images, which are necessary for the prediction of autism.
  - **Viewing Results**: The user views the final prediction or diagnosis result, which indicates the likelihood of autism in the subject.

2. Frontend

The Frontend represents the user interface of the system. It includes the components that allow the user to input data and view results.

Data Input Form

- **Description**: This is the component where the user enters relevant data about the subject (such as age, behavior, questionnaires, etc.). The form can also allow for the uploading of MRI images.
- **Responsibilities**:
  - **Data Entry**: It enables the user to input behavioral data related to autism, which may include observations of behavior, developmental milestones, and test results.

  - **Image Upload**: The user can upload MRI images of the subject that will be processed by the CNN Model for image-based predictions.
  - **Data Transmission**: Once the user enters the data, the Data Input Form sends the entered data to the Data Preprocessing module for further processing.

Prediction Results

- **Description**: This component displays the final predictions or results of the autism diagnosis. It takes input from the Prediction Engine and shows the outcome to the user.
- **Responsibilities**:
  - **Display Results**: After the backend processes the data, the prediction results (whether the subject is likely to have autism or not) are displayed here.
  - **User Feedback**: The system could provide additional insights or recommendations to the user based on the prediction results, such as further testing or interventions.

3. Backend

The Backend represents the core processing layer of the system. It handles the preprocessing of data, analysis through machine learning models, and combines the outputs from different models to generate the final prediction.

Data Preprocessing

- **Description**: This component is responsible for cleaning and transforming the raw input data (both behavioral data and MRI images) into a format suitable for analysis.
- **Responsibilities**:
  - **Data Cleaning**: It addresses any missing data, inconsistencies, or anomalies (such as invalid values or incomplete records) in the behavioral data.
  - **Data Normalization**: Ensures that numerical data is standardized or normalized so that it can be used effectively by machine learning models.
  - **Feature Extraction**: In the case of image data, preprocessing could involve extracting features or resizing images to make them compatible with the CNN model.
  - **Data Transmission**: Once preprocessed, the data is sent to both the Behavioral Model and the CNN Model for further analysis.

Behavioral Model

- **Description**: This model analyzes the behavioral data to predict the likelihood of autism. It uses machine learning algorithms to identify patterns in the data that are associated with autism spectrum disorder (ASD).
- **Responsibilities**:
  - **Behavioral Analysis**: The model looks for signs of autism in the input data, such as behavioral patterns, social interactions, communication skills, and cognitive abilities.
  - **Machine Learning**: It could use techniques like Decision Trees, Random Forest, Support Vector Machines (SVM), or Logistic Regression to classify whether a subject is at risk of autism based on the behavioral features.
  - **Prediction**: After analyzing the data, the Behavioral Model sends its prediction to the Prediction Engine, which combines it with the results from the CNN Model to provide a final decision.

CNN Model (Convolutional Neural Network)

- **Description**: This deep learning model is designed to process MRI images of the subject's brain. CNNs are particularly effective at extracting spatial features from images, making them ideal for medical imaging tasks like detecting autism-related changes in brain structures.
- **Responsibilities**:
  - **Image Analysis**: The CNN analyzes MRI images to detect patterns or anomalies in brain regions (e.g., the amygdala, hippocampus) that might be associated with autism.
  - **Feature Extraction**: The CNN extracts important features from the images, such as specific shapes, textures, or structures that are indicative of autism.
  - **Prediction**: The model outputs a prediction based on the analysis, which is sent to the Prediction Engine to be combined with the results from the Behavioral Model.

Prediction Engine

- **Description**: The Prediction Engine is the central component that takes the predictions from both the Behavioral Model and the CNN Model and integrates them to generate a final prediction.

- **Responsibilities**:
    - **Integrating Predictions**: It combines the results from both models to provide a final output, such as a probability score for autism or a binary classification (autism or not).
    - **Final Decision**: The Prediction Engine makes the final decision about whether the subject is predicted to have autism based on the combined outputs of the two models.
    - **Results Transmission**: It sends the final results to the Prediction Results component for the user to view.

4. Storage

The Storage layer is responsible for storing data, both behavioral and image-based, in a way that it can be accessed, retrieved, and used by the backend components.

Data Database

- **Description**: This is where the behavioral data (such as test scores, behavioral assessments, questionnaire responses) is stored.
- **Responsibilities**:
    - **Data Storage**: It securely stores large volumes of behavioral data for different subjects.
    - **Data Retrieval**: The Data Preprocessing component retrieves data from this database for preprocessing before sending it to the models for analysis.

Image Storage

- **Description**: This is where the MRI images of the subjects are stored. These images are analyzed by the CNN Model.
- **Responsibilities**:
    - **Image Storage**: It stores MRI images in a structured format, making them accessible for processing.
    - **Image Retrieval**: The CNN Model retrieves the images from the storage for analysis during the prediction phase.

Summary of Interactions:

- The **User** interacts with the system by providing behavioral data and uploading MRI images through the Data Input Form.

- The Data Preprocessing component cleans and transforms the data before passing it on to the Behavioral Model and CNN Model.
- The Behavioral Model uses machine learning algorithms to analyze the behavioral data, while the CNN Model uses deep learning techniques to process the MRI images.
- The predictions from both models are sent to the Prediction Engine, which integrates them into a final output.
- The final prediction is displayed to the user through the Prediction Results interface.
- All data, including behavioral and image data, is stored in the Data Database and Image Storage for retrieval and further analysis.

The research was carried out in four phases:

1. Data Collection with Synthetization
2. Training and Testing Dataset
3. Developing the Prediction Model
4. Evaluation of the Prediction Model.

1. Data Collection with Synthetization:

This initial phase involved the collection of relevant data for the PCOS detection project. The dataset comprised clinical information related to PCOS symptoms, demographic details, and medical images. This phase aimed to create a robust dataset that could effectively train and evaluate the subsequent prediction models.

2. Training and Testing Dataset:

The collected and synthesized dataset was then divided into training and testing subsets. The training dataset served as the foundation for training machine learning models, enabling them to learn patterns and relationships within the data. The testing dataset, distinct from the training data, was reserved for evaluating the models' performance.

3. Developing the Prediction Model:

With the datasets prepared, the focus shifted to developing the prediction model. Two main components were created: the Numerical Model, which focused on symptom-based and parameter-based predictions, and the Image-Based Model, utilizing deep learning techniques forthe analysis of medical images. An Integration Layer was introduced to combine the outputs of these models, creating a multi-modal framework.

4. Evaluation of the Prediction Model:

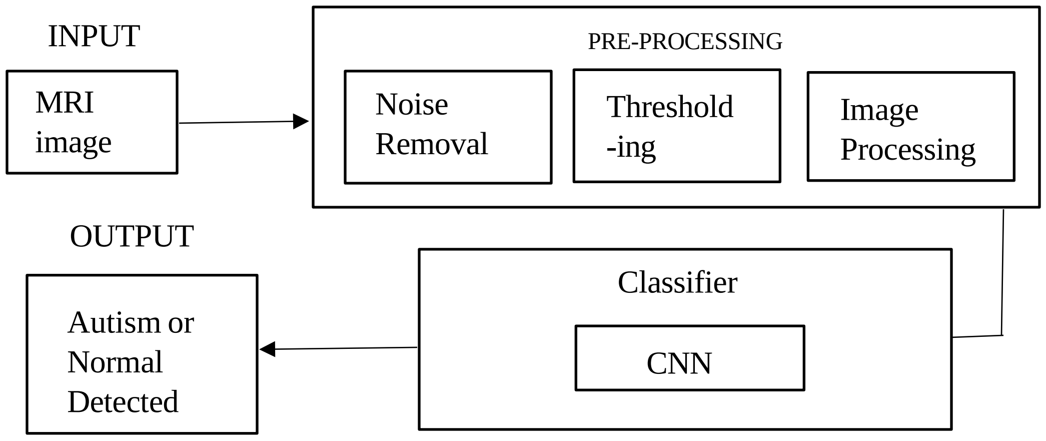


Figure 3.2: Architectural Diagram of the Proposed System using CNN

This figure 3.2 shows the architecture diagram of the proposed system using CNN.The final phase involved the rigorous evaluation of the developed prediction models. Performance metrics such as accuracy, precision, recall, and F1 score were employed to assess the models' effectiveness. This critical evaluation phase provided insights into the reliability and accuracy of the PCOS detection framework, guiding potential refinements and improvements for future iterations.

The proposed system has the following steps for detection of disease using image dataset.

1. Noise Removal
2. Thresholding
3. Image Sharpening
4. Feature Extraction and Classification

3.2.1 Noise Removal

Noise removal is the process of removing or reducing the noise from the image. The noise removal algorithms reduce or remove the visibility of noise by smoothing the entire image leaving areas near contrast boundaries. Here we are making use of Median Filter which is a Noise Removal Technique.

3.2.2 Median Filtering

The median filter is a non-linear digital filtering technique, often used to remove noise from an image or signal. Here 0‟s are appended at the edged and corners to the matrixwhich is the representation of the grey scale image. Then for every3*3 matrix, arrange elements in ascending order, then find median/middle element of those 9 elements.

3.2.3 Thresholding

Thresholding is a type of image segmentation, where we change the pixels of an image to make the image easier to analyze. Most frequently, we use thresholding as a wayto select areas of interest of an image, while ignoring the parts we are not concerned with.We use Basic Global Thresholding.

3.2.4 Image Sharpening

Image sharpening refers to any enhancement technique that highlights edges and fine details in an image. Increasing yields, a more sharpened image. Image sharpening is done by adding to the original image a signal proportional to a high-pass filtered version ofthe image.

3.2.5 High-Pass Filtering

A high-pass filter can be used to make an image appear sharper. The filters emphasize fine details in the image.

3.2.6 Feature Extraction and Classification

Feature extraction is a process of dimensionality reduction by which an initial set of raw data is reduced to more manageable groups for processing. In the terminology of machine learning, classification is considered an instance of supervised learning, i.e.,learning where a training set of correctly identified observations is available.

The corresponding unsupervised procedure is known as clustering, and involves grouping data into categories based on some measure of inherent similarity or distance.

3.2.7 Convolutional Neural Networks

In deep learning, a convolutional neural network (CNN, or ConvNet) is a class of deep neural network most commonly applied to analyzing visual imagery. They are also known as shift invariant or space invariant artificial neural networks (SIANN), basedon their shared-weights architecture and translation invariance characteristics. They have applications in mage recommender systems, image classification, medical image analysis, natural language processing, brain-computer interfaces, and financial time series.

CNNs are regularized versions of multi-layer perceptrons. Multilayer perceptrons usually mean fully connected networks, that is, each neuron in one layer is connected to allneurons in the next layer. The "fully-connectedness" of these networks makes them prone to overfitting data.CNNs take a different approach towards regularization: they take advantage of the hierarchical pattern in data and assemble more complex patterns using smaller and simpler patterns.

3.3 Specification using Use Case Diagram

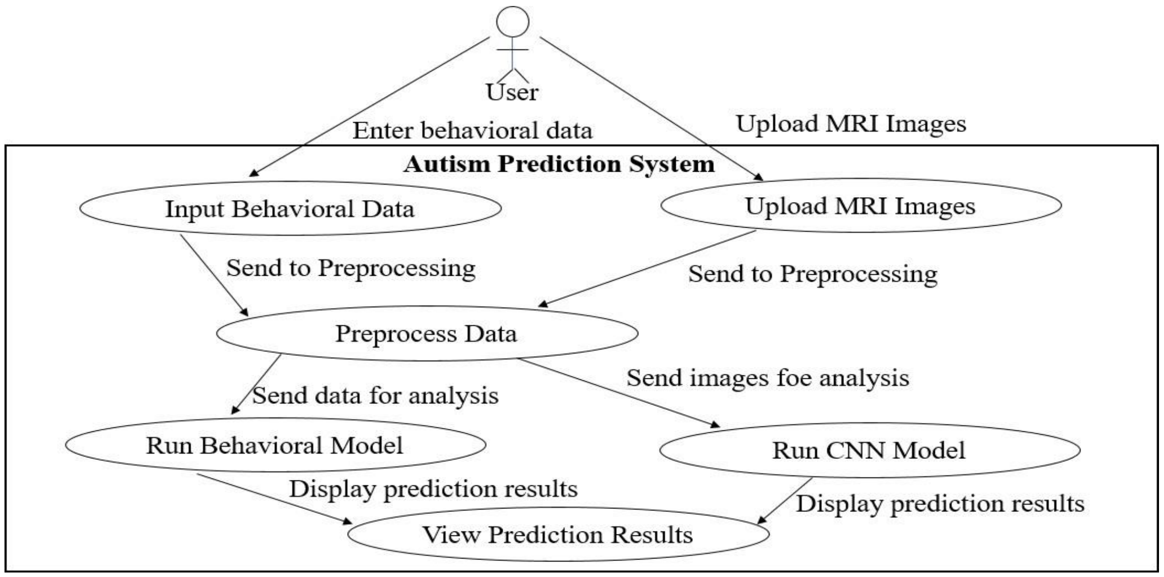


Figure 3.3 : Use Case Diagram of Autism Prediction System

Explanation of the Use Case Diagram:

1. **User (Actor)**:
    - The user (which could be a clinician, researcher, or caregiver) interacts with the system to enter data and view results.
2. **Use Cases**:
    - **Input Behavioral Data (UC1)**: The user enters the behavioral data through a form.
    - **Upload MRI Images (UC2)**: The user uploads MRI images for analysis.
    - **Preprocess Data (UC3)**: The system processes both the behavioral data and MRI images.
    - **Run Behavioral Model (UC4)**: The system runs the machine learning-based behavioral analysis to predict autism.
    - **Run CNN Model (UC5)**: The system processes MRI images using the CNN model to predict autism based on brain imaging.
    - **View Prediction Results (UC6)**: The user views the final autism prediction results, which combine both the behavioral and image-based analyses.

3.4 Module Specification

The Autism Prediction System is composed of several interconnected modules that work together to enable seamless data input, processing, prediction, and result presentation. Each module in the system is designed to perform specific tasks, ensuring that the system can predict autism based on both behavioral data and medical imaging.

3.4.1 User Interface (UI) Module

Purpose:

This module serves as the front-end interface where users interact with the system. It collects input data, allows for image uploads, and presents the final prediction results. The interface is designed to be user-friendly and intuitive, ensuring that clinicians or other users can easily navigate the system.

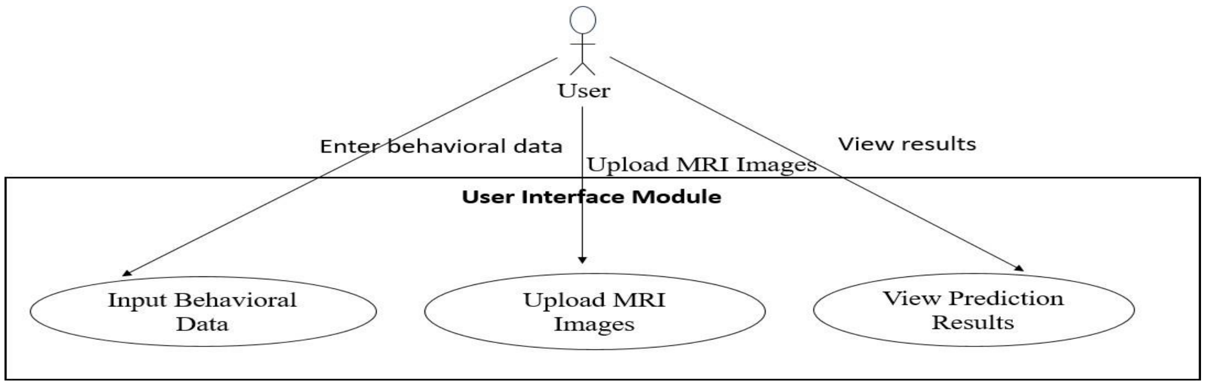


Figure 3.4 : Use Case Diagram of User Interface Module

Components:

- **Data Input Form**: Provides fields to enter behavioral data such as developmental history, communication skills, social interaction behaviors, and other relevant clinical information.
- **Image Upload Section**: Allows the user to upload MRI images for processing by the CNN model.
- **Result Display**: Shows the output of the system’s prediction, including the likelihood of autism and possibly other insights (e.g., severity).

Functionalities:

- Collects and sends input data to the **Data Preprocessing** module.
- Provides feedback to the user about data validation errors or upload issues.

3.4 2. Data Preprocessing Module

Purpose:

This module prepares the raw behavioral and image data to be fed into the machine learning and deep learning models. It performs essential tasks such as cleaning, normalizing, and transforming the data.

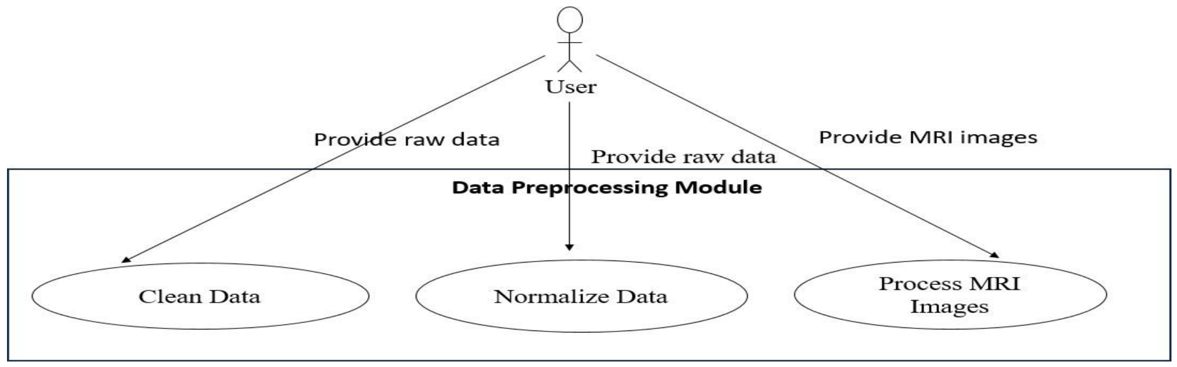


Figure 3.5 : Use Case Diagram of Data Preprocessing Module

Components:

- **Data Cleaning**: Handles missing values, removes irrelevant or duplicate entries, and addresses any structural errors in the input data.
- **Data Normalization**: Scales numerical data to ensure that the models receive inputs in a consistent format
- **Image Preprocessing**: For the MRI images, this could include resizing, cropping, and augmenting the images to ensure they are ready for CNN analysis.

Functionalities:

- Cleans and normalizes the behavioral data and ensures that no crucial information is lost.
- Processes MRI images to a standard format compatible with CNN input requirements.
- Converts the cleaned and preprocessed data into formats suitable for machine learning and deep learning models.

3.4.3. Behavioral Model Module (Machine Learning)

Purpose:

This module applies machine learning algorithms to analyze behavioral data for autism prediction. It utilizes historical data and known diagnostic criteria to detect patterns that might indicate autism spectrum disorder (ASD).

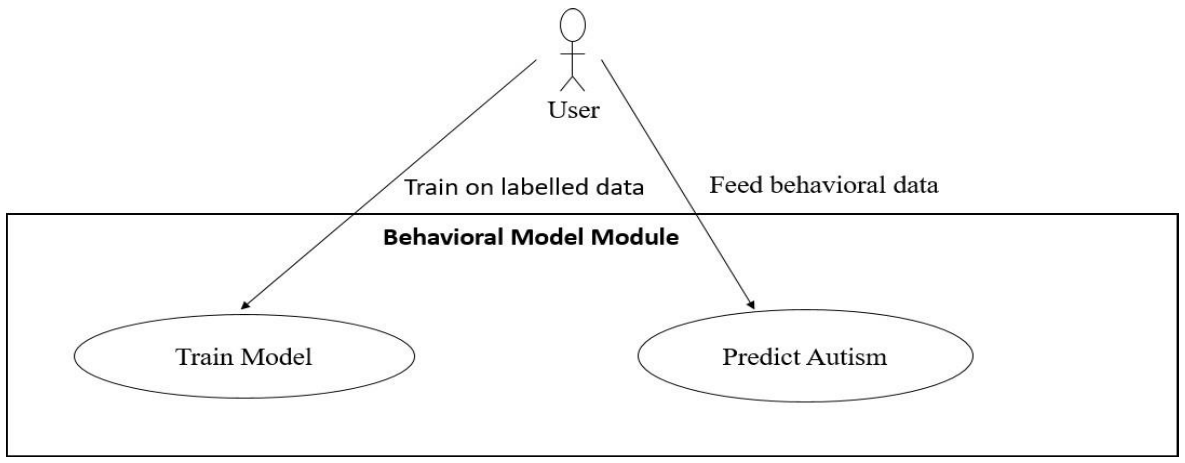


Figure 3.6 : Use Case Diagram of Behavioral Model Module

Components:

- **Data Analysis**: Implements algorithms such as Random Forest, Support Vector Machines (SVM), Logistic Regression, or Decision Trees to analyze behavioral data.
- **Feature Extraction**: Selects relevant features (e.g., age, language development, social behavior) to improve the model's predictive accuracy.
- **Training & Evaluation**: The model is trained on a labeled dataset containing both positive and negative autism diagnoses, followed by evaluation using test data to assess the model's accuracy and reliability.

Functionalities:

- Receives preprocessed behavioral data and applies appropriate machine learning models to predict autism.
- Returns prediction results (e.g., probability of autism diagnosis).
- Can be periodically retrained to improve prediction accuracy using new data.

3.4.4. CNN Model Module (Deep Learning)

Purpose:

The CNN module processes MRI images to detect abnormalities or features related to autism. It uses deep learning techniques to automatically extract relevant features from medical images that could indicate autism-related patterns in brain structure.

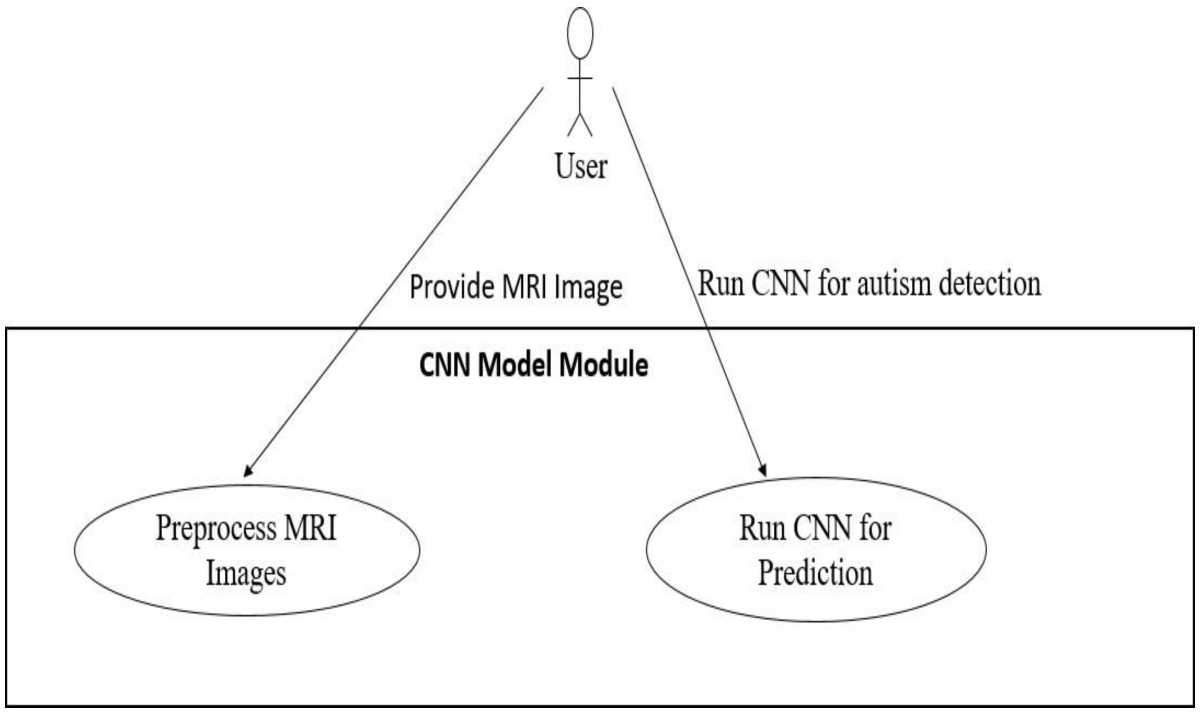


Figure 3.7 : Use Case Diagram of CNN Model Module

Components:

- **Convolutional Layers**: These layers extract spatial hierarchies from MRI images, learning patterns such as brain region anomalies that may be indicative of autism.
- **Pooling Layers**: These layers reduce the dimensionality of the input image, keeping only the most relevant features for classification.
- **Fully Connected Layers**: These layers perform classification by interpreting the features extracted from the images to output a prediction.

Functionalities:

- Processes MRI images and extracts relevant features for autism prediction.
- Classifies the images to determine if they show patterns associated with autism.
- Returns the prediction to the **Prediction Engine** for final decision-making.

3.4.5. Prediction Engine Module

Purpose:

The Prediction Engine is the core module responsible for combining the results from the Behavioral Model and the CNN Model to generate a final autism prediction. It integrates both data sources (behavioral data and MRI images) to provide a comprehensive prediction.

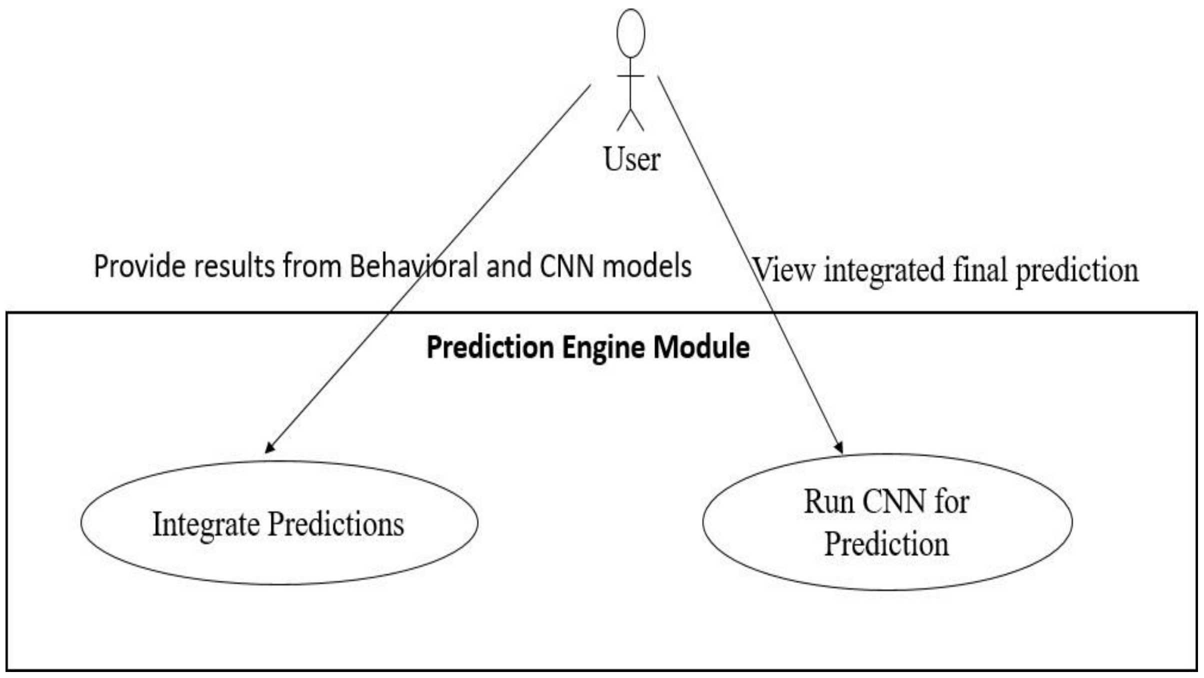

User
Provide results from Behavioral and CNN models
View integrated final prediction
Prediction Engine Module
Integrate Predictions
Run CNN for Prediction

Figure 3.8 : Use Case Diagram of Prediction Engine Module

Components:

- **Data Fusion**: Combines the prediction results from both the behavioral model and the CNN model to arrive at a final autism diagnosis.
- **Decision Making**: Based on the integrated data, the module decides whether the subject is predicted to have autism or not.
- **Result Output**: The final decision is passed to the **Prediction Results** module, which displays the outcome to the user.

Functionalities:

- Integrates the predictions from the machine learning (behavioral) and deep learning (CNN) models.
- Uses a decision-making algorithm to provide a final prediction based on both input sources.
- Outputs the final autism diagnosis result to the **User Interface** for display.

3.4.6. Storage Module

Purpose:

This module is responsible for storing and managing all input data (both behavioral and MRI images) as well as output predictions. It ensures that the data is securely stored and can be easily retrieved for future use, analysis, or auditing.

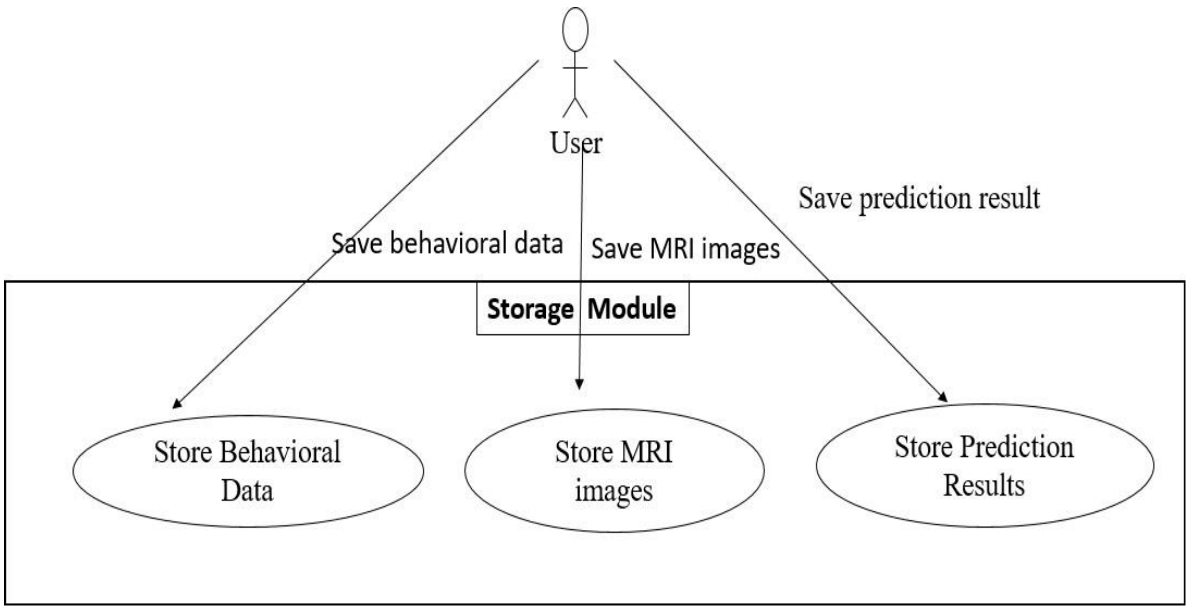


Figure 3.9 : Use Case Diagram of Storage Module

Components:

- **Data Database**: Stores behavioral data input by the user, such as responses to questionnaires, age, and other relevant clinical data.
- **Image Storage**: Stores MRI images uploaded by the user for processing by the CNN model.
- **Results Database**: Stores the results of the autism prediction, which can be accessed later for review or historical purposes.

Functionalities:

- Provides secure storage for both input data (behavioral and image) and the results of the autism prediction.
- Allows for data retrieval, ensuring that users can access past records or predictions.
- Ensures compliance with relevant data privacy and security regulations (e.g., HIPAA or GDPR) for storing medical data.

3.4.7 Authentication and Authorization Module

3.4.7.1 Purpose:

This module ensures that only authorized users can access the system and perform certain actions, such as entering patient data, uploading medical images, or viewing prediction results. It maintains security by requiring user authentication and defining roles (e.g., admin, clinician, caregiver).

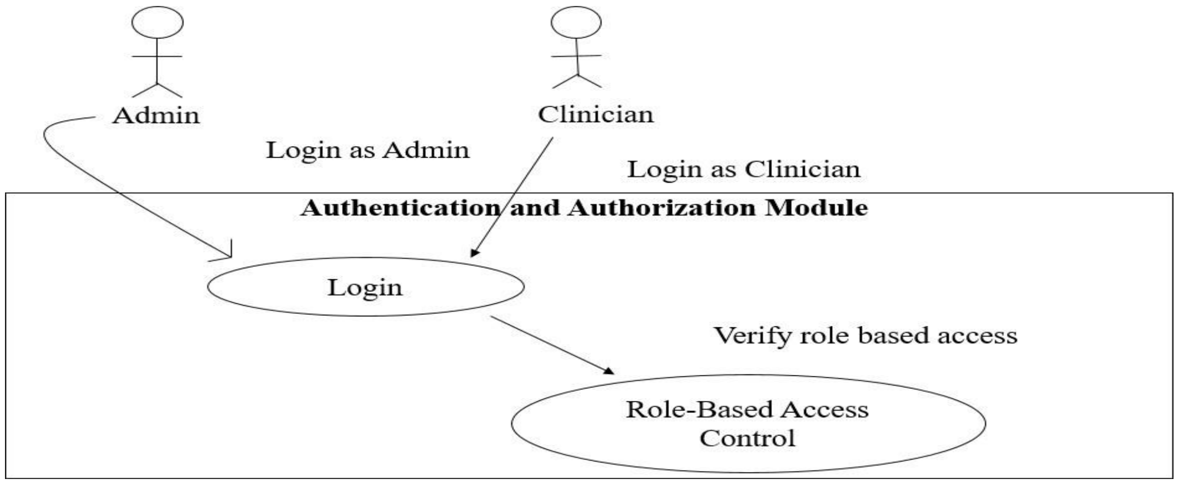


Figure 3.10 : Use Case Diagram of Authentication and Authorization Module

Components:

- **Login**: Requires users to authenticate using credentials (e.g., username and password).
- **Role-Based Access Control (RBAC)**: Defines different levels of access depending on the user's role, restricting access to sensitive data and functionalities.
- **Audit Logs**: Tracks user actions within the system for security and compliance purposes.

Functionalities:

- Ensures that only authenticated users can access the system.
- Enforces role-based access control, ensuring that users only have access to data and functionalities appropriate for their role.

3.5 Data Flow Diagram

As the name specifies so to the meaning of the words, it is the process which is explained in detail like how the data flows between the different processes. The figure 3.11 depicts the flow diagram and is composed of the input, process and the output. In most of the times it is the initial step for designing any of the systems to be implemented.

3.5.1 Data Flow Diagram of Pre-processing Module

Input image through camera is captured and it can be used to store as dataset for training or as input image to detect the disease. The image is captured and stored in any supported format specified by the device.

As shown in the figure 3.12 initially the set of captured images are stored in a temporary file in OpenCV. The storage is linked to the file set account from which the datais accessed.

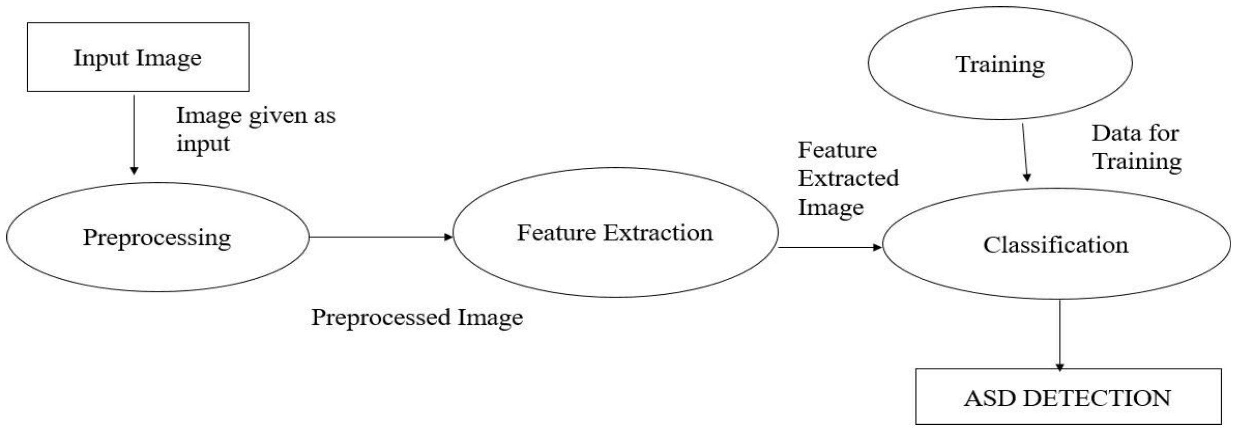


Figure 3.11: Data Flow Diagram of ASD Detection.

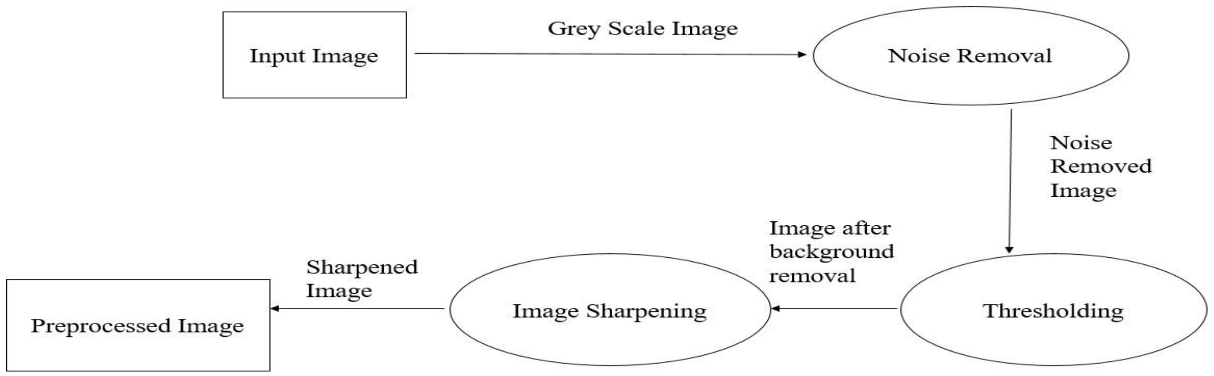


Figure 3.12: Data Flow Diagram for Pre-Processing Module.

The figure 3.11 shows data flow diagram of ASD detection and figure 3.12 shows data flow diagram for pre-processing module.The Pre-processing is required on every image to enhance the functionality of image processing. Captured images are in the RGB format. The pixel values and the dimensionality of the captured images is very high. As images are matrices and mathematical operations are performed on images are the mathematical operations on matrices. So, we convert the RGB image into gray image. Then we carry out Noise Removalfollowed by Thresholding; the last step is Image Sharpening after which we obtain the preprocessed Image.

3.5.2 Data Flow Diagram of Classification using CNN

In deep learning, a convolutional neural network (CNN, or ConvNet) is a class of deep neural network most commonly applied to analyzing visual imagery. They are also known as shift invariant or space invariant artificial neural networks (SIANN), based on their shared-weights architecture and translation invariance characteristics

When CNN is used for classification, we don’t have to do feature extraction. Feature Extraction will also be carried out by CNN. We feed the preprocessed image directly to CNN classifier to obtain the type of disease if present. Figure 3.13 shows the Data Flow Diagram of Classification using CNN.The figure 3.13 shows the data flow diagram of classification using CNN.

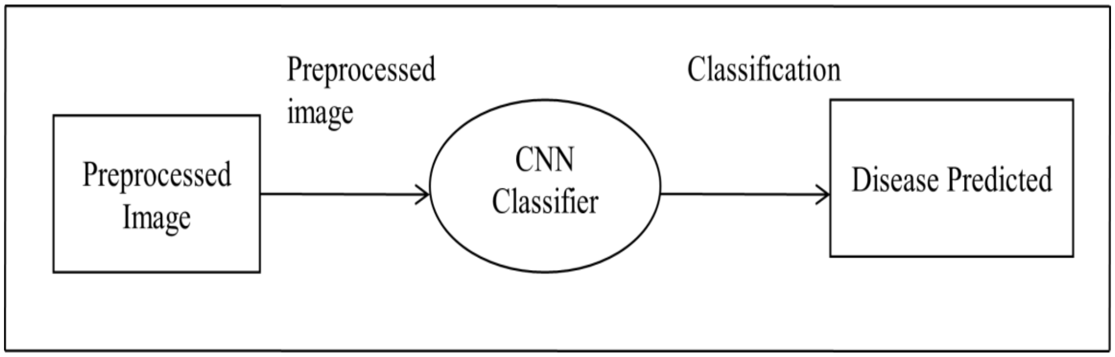


Figure 3.13: Data Flow Diagram of Classification using CNN

3.6 State Chart Diagram for proposed system

A state chart diagram is also named as state diagram. It is popular one among five UML diagrams and it is used to model the dynamic nature of the system. State chart definesvarious states of an object during its lifetime. The state chart diagram is a composition of finite number of states and the functionalities describes the functioning of each module in the system. It is a graph where each state is a directed edge and representedby node and these directed edges represents transition between states.

Each state name must be a unique name. Initial state is arrived on creation of object, entry of the final state infers destruction of the object. Starting state is denoted by the solid circle and ending state is symbolized by bull eye symbol.

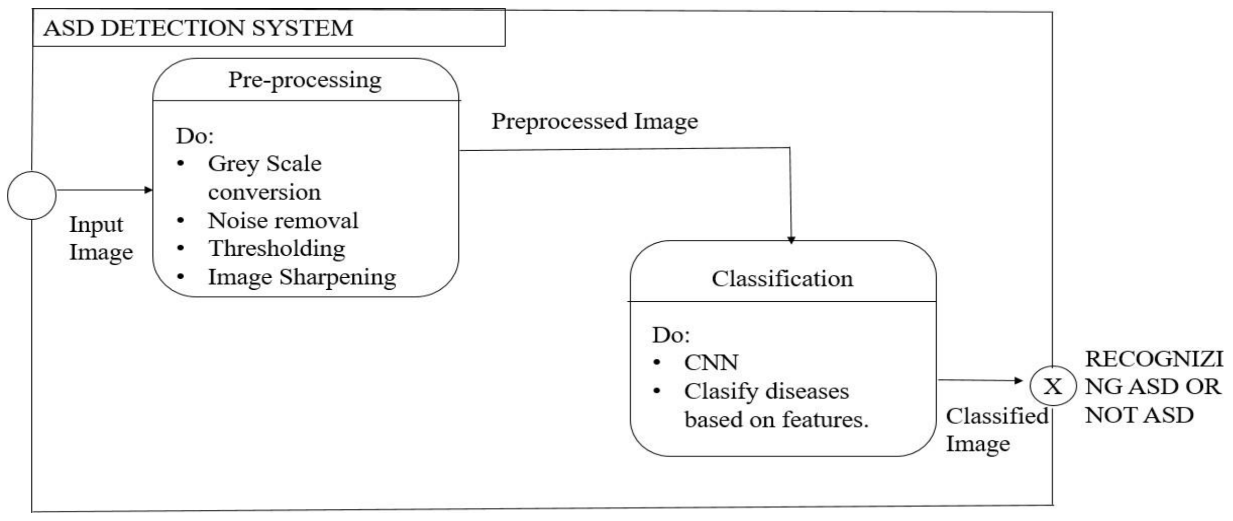


Figure 3.14: State Chart Diagram of Model using CNN.

The figure 3.14 shows the state chart diagram of ASD detection using CNN. The process starts with the solid circle, the first state is reading the image with ultrasound scanas input and the second state is pre-processing to convert RGB to grey and then Noise Removal is used where we remove unwanted variation in pixel, followed by Thresholding, then at last Image sharpening is done.Next in the third state, Classification is carried out here we use Convolutional Neural Network for image classification we make use of all four layers convolution, pooling layer fully connected layer and ReLu layer it classify image based on feature. In the last as the result ASD or Not ASD is displayed.

3.7 Summary

In third chapter, high level design of the proposed method is discussed. Section 3.1 presents the design considerations for the project. Section 3.2 discusses the systemarchitecture of proposed system. The next Section 3.3 describes use case diagram. Section 3.4 describes module specification for all the modules. The data flow diagram for thesystem is explained in section 3.5.

Chapter 4

DETAIL DESIGN

4.1 Support Vector Machine (SVM)

Support Vector Machine is one of the most popular machine learning algorithms, which is used for classification and regression. The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category. The Figure 4.1 illustrates the scatter point and hyperplane and figure 4.2 shows the classification of two classes.

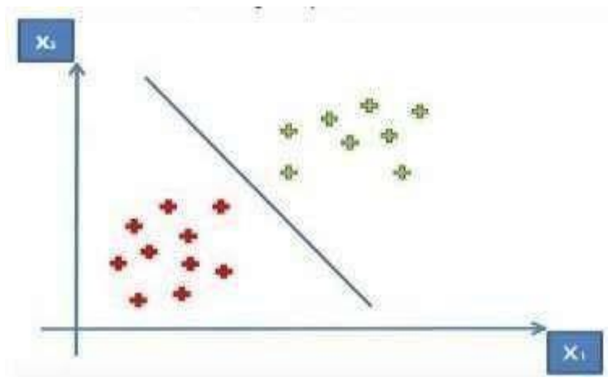


Figure 4.1: Scatter Points and Hyperplane

• This project has a dataset that has two tags (red and blue).

• The dataset has two features x1 and x2.

• We want a classifier that can classify the pair (x1, x2) of coordinates in either red or blue.

• It is 2-d space so by just using a straight line, we can easily separate these two classes.

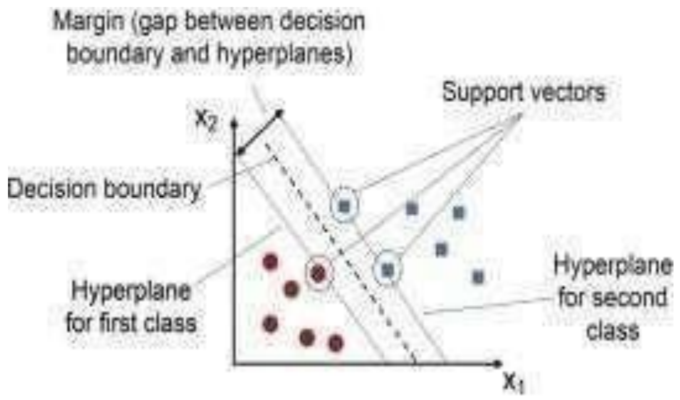

Margin (gap between decision boundary and hyperplanes)
Support vectors
X₂
Decision boundary
Hyperplane for first class
Hyperplane for second class
X₁

Figure 4.2 Illustrates the Classification of Two Classes

Consider two features in the dataset

Table 4.1: SCS and RBS

SCS	RBS
2	7
8	3
7	2
2	9
6	3
9	2
7	4
3	7
5	9
2	8

Plot each of the above value to get the graph as shown in Figure 4.3.

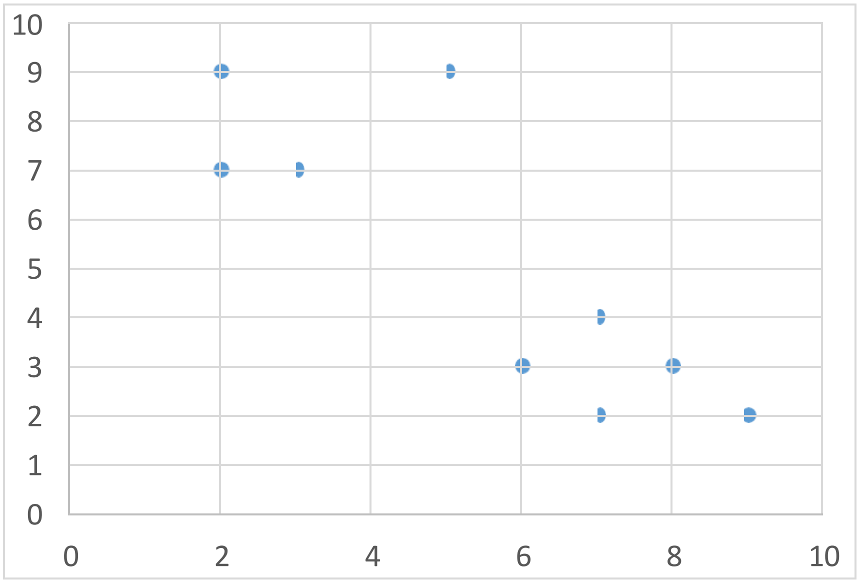


Figure 4.3: Clustered Points Obtained from the Data

The SVM algorithm helps to find the best line or decision boundary. This best boundary or region is called as a hyperplane. SVM algorithm finds the closest point of the lines from both the classes.These points are called support vectors. The distance between the vectors and the hyperplane is called as margin. The goal of SVM is to maximize this margin. The hyperplane with maximum margin is called the optimal hyperplane. To draw a hyperplane, support vectors are required.

Drawing of hyperplane:

The Normal class data points are :

{(8,3),(7,2),(6,3),(9,2),(7,4)}

The Austism class data points are :

{(2,7),(2,9),(3,7),(5,9),(2,8)}

It can be observed that the support vectors are : (8,3),(7,2),(2,9).

The augmented vector can be obtained by adding the bias as given below:

$$S1= \begin{bmatrix} 8 \\ 3 \\ 1 \end{bmatrix} \qquad S2= \begin{bmatrix} 7 \\ 2 \\ 1 \end{bmatrix} \qquad S3= \begin{bmatrix} 2 \\ 9 \\ 1 \end{bmatrix}$$

From these, a set of three equations can be obtained based on these three support vectors as follows:

$$\begin{aligned} \alpha_1 \tilde{s_1} \cdot \tilde{s_1} + \alpha_2 \tilde{s_2} \cdot \tilde{s_1} + \alpha_3 \tilde{s_3} \cdot \tilde{s_1} &= -1 \\ \alpha_1 \tilde{s_1} \cdot \tilde{s_2} + \alpha_2 \tilde{s_2} \cdot \tilde{s_2} + \alpha_3 \tilde{s_3} \cdot \tilde{s_2} &= +1 \\ \alpha_1 \tilde{s_1} \cdot \tilde{s_3} + \alpha_2 \tilde{s_2} \cdot \tilde{s_3} + \alpha_3 \tilde{s_3} \cdot \tilde{s_3} &= +1 \end{aligned}$$

where $\alpha \rightarrow$ Lagrange multiplier

Now to find α_1, α_2 and α_3 substitute S_1, S_2, S_3 in the above equation.

$$a1 \begin{bmatrix} 64 \\ 9 \\ 1 \end{bmatrix} + a2 \begin{bmatrix} 56 \\ 6 \\ 1 \end{bmatrix} + a3 \begin{bmatrix} 16 \\ 27 \\ 1 \end{bmatrix} = -1$$

$$a1 \begin{bmatrix} 56 \\ 6 \\ 1 \end{bmatrix} + a2 \begin{bmatrix} 49 \\ 4 \\ 1 \end{bmatrix} + a3 \begin{bmatrix} 14 \\ 18 \\ 1 \end{bmatrix} = -1$$

$$a1 \begin{bmatrix} 16 \\ 27 \\ 1 \end{bmatrix} + a2 \begin{bmatrix} 14 \\ 18 \\ 1 \end{bmatrix} + a3 \begin{bmatrix} 4 \\ 8 \\ 1 \end{bmatrix} = 1$$

The Equations obtained after solving the matrix are:

$74\alpha_1+63\alpha_2+44\alpha_3=-1$

$63\alpha_1+54\alpha_2+33\alpha_3=-1$

$44\alpha_1+33\alpha_2+86\alpha_3=1$

$\alpha_1 = 0.7916$ $\alpha_2= -0.9166$ $\alpha_3= -0.0416$

$W = \alpha_1 s_1+\alpha_2 s_2+\alpha_3 s_3$

$$W=(0.7916)* \begin{bmatrix} 8 \\ 3 \\ 1 \end{bmatrix} +(-0.9166)* \begin{bmatrix} 7 \\ 2 \\ 1 \end{bmatrix} +(-0.0416)* \begin{bmatrix} 2 \\ 9 \\ 1 \end{bmatrix} = \begin{bmatrix} -0.1666 \\ 0.1672 \\ -0.1666 \end{bmatrix}$$

Weights : [-0.1666,0.1672] **Bias** : -0.1666

$x = (b - w_2y) / (w_1)$ $y = (b - w_1x) / (w_2)$

Put y=0 Put x=0

$x=b/(w_1)$ $y=b/(w_2)$

$x=(-0.1666)/(-0.1666)$ $y=(-0.1666)/(0.1672)$

$x=1$ $y=-0.9964$

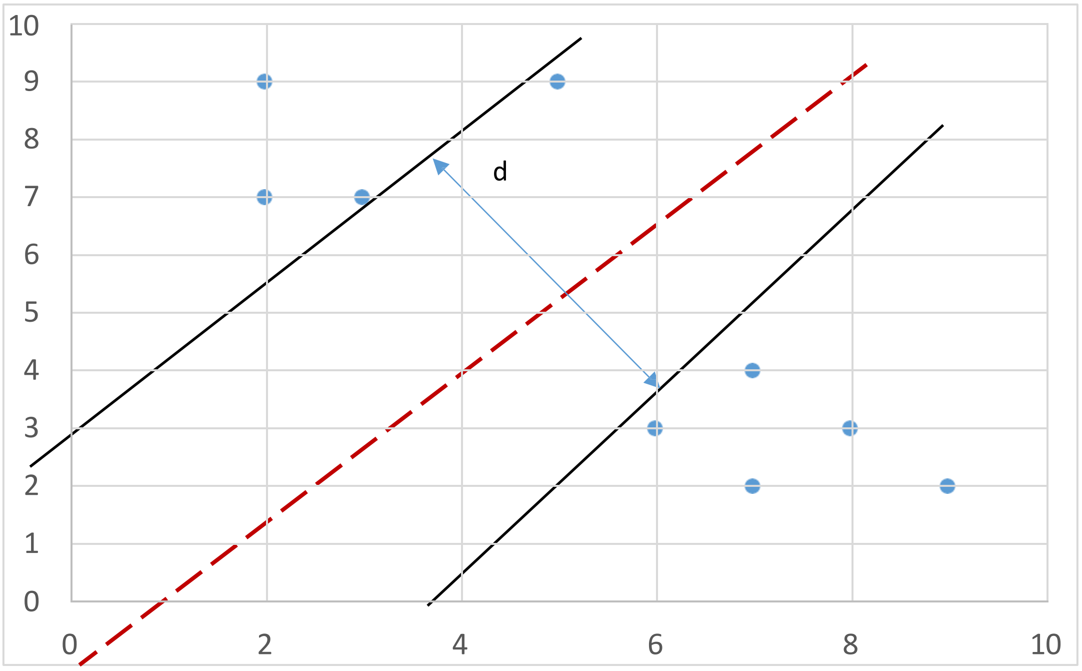


Figure 4.4: Distance between Two Margins

Point 1 = (1,0)

Point 2 = (0,-0.9964)

Distance can be calculated as,

$d = \sqrt{((y_2 - y_1)^2 + (x_2 - x_1)^2)}$

$= \sqrt{((3-9)^2 + (6-5)^2)}$

$= 8.6023$

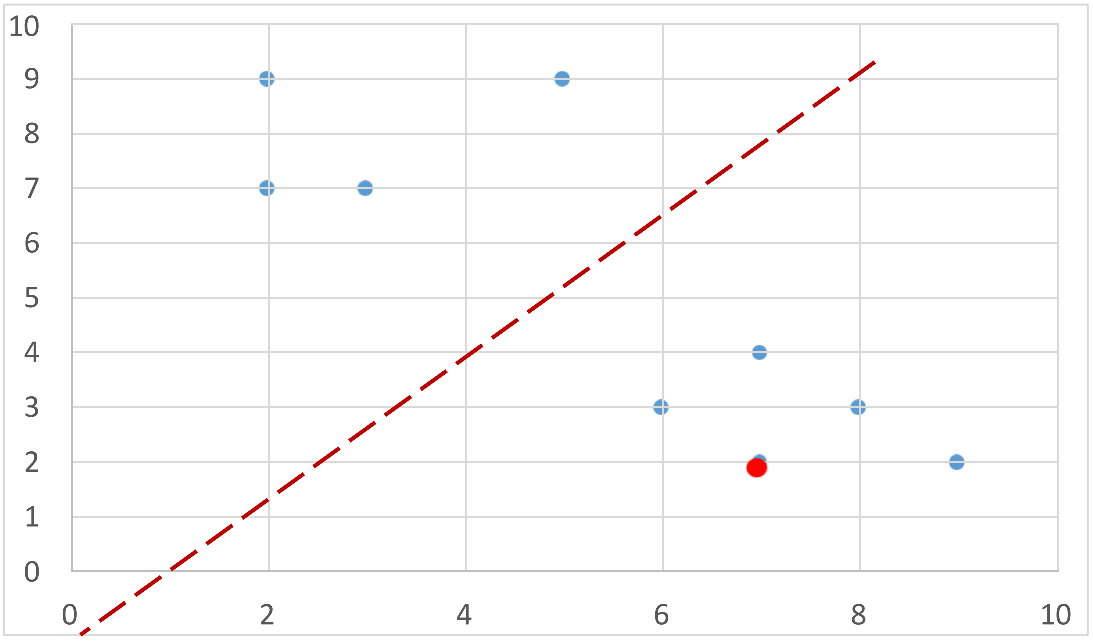


Figure 4.5: Example of Normal Patient's Result

Consider the point (7,2)

$w_1x + w_2y \ - b < 0$

= -0.1666(7) + 0.1672(2) – (-0.1666)

= -0.6652

-0.6652 < 0

Hence, Patient is Normal

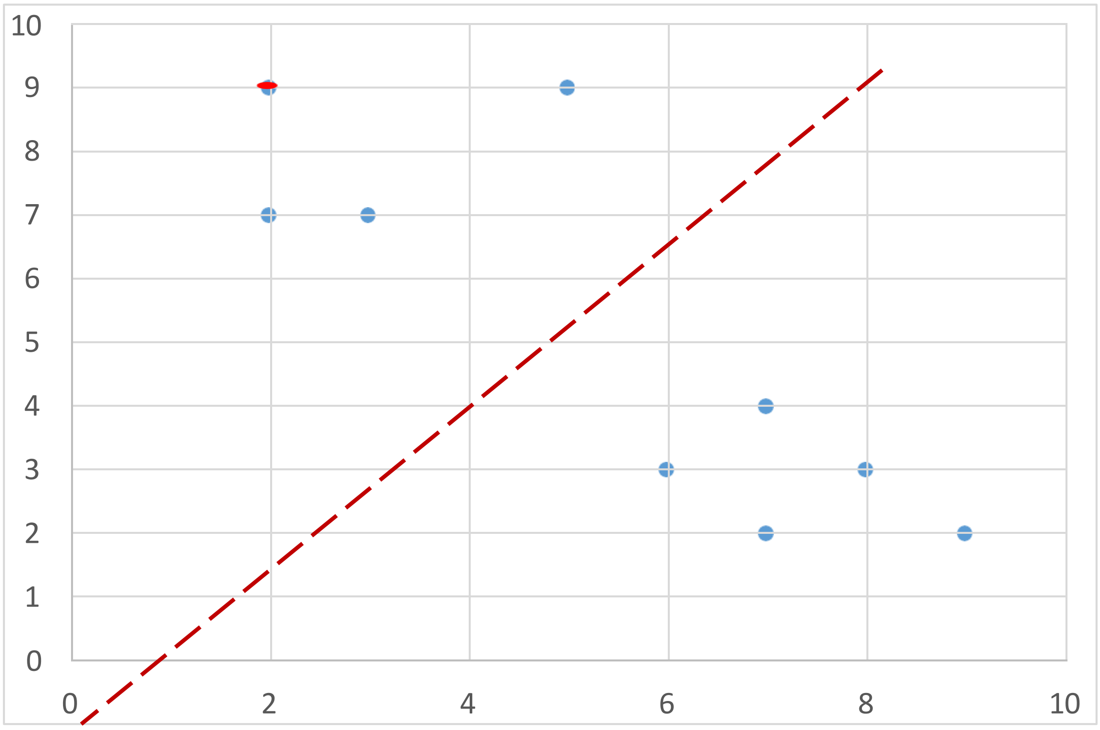


Figure 4.6: Example of ASD Positive Patient's Result

Consider the point(2,9)

$w_1x + w_2y \ - b < 0$

= -0.1666(2) + 0.1672(9) – (-0.1666)

= 1.3382

1.3382> 0

Hence, Patient has autism

4.2 Convolutional Neural Network (CNN)

4.2.1 Image acquisition

The image is obtained from the patient's ultrasound scan report. These Images are considered for undergoing this process.

4.2.2 Noise Removal

Noise removal algorithm is the process of removing or reducing the noise from the image. It removes the visibility of noise by smoothing the entire image leaving areas near contrast boundaries. Here the grayscale image which was obtained in the previous step is given as input. Here we are making use of Median Filter which is a Noise Removal Technique.

Median Filtering:

The median filter is a non-linear digital filtering technique, often used to remove noise from an image or signal. Here 0's are appended at the edges and corners to the matrix which is their presentation of the grey scale image. Then for every3*3 matrix, arrange elements in ascending order, then find median/middle element of those 9 elements, and write that median value to that particular pixel position.The figure 4.7 shows the noise filtering using median filter.

The Original grayscale matrix:

12	63	0	3	1
38	106	161	156	78
96	84	109	197	94
135	187	199	233	157
86	194	151	112	81

Append 0s at edges and corners:

0	0	0	0	0	0	0
0	12	63	0	3	1	0
0	38	106	161	156	78	0
0	96	84	109	197	94	0
0	135	187	199	233	157	0
0	86	194	151	112	81	0
0	0	0	0	0	0	0

The enhanced matrix:

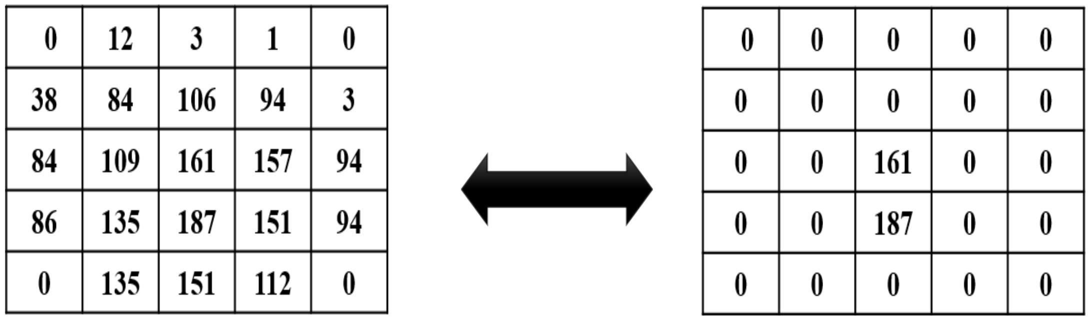
0	12	3	1	0
38	84	106	94	3
84	109	161	157	94
86	135	187	151	94
0	135	151	112	0

Figure 4.7 : Noise Filtering using Median Filter

4.2.3 Basic Global Thresholding

Thresholding is a type of image segmentation, where we change the pixels of an image to make the image easier to analyze.

The main goal of thresholding is to simplify image data by converting it into a binary format, distinguishing between foreground and background. A (i, j) is greater than or equal to the threshold T, retain it. Else, replace the value by 0. Here, the value of T can be manipulated in the frontend, to suit the varying needs of different images. We use trial and error method here to obtain threshold value which may be best suited for us. The figure 4.8 shows the thresholding using basic global thresholding.



0	12	3	1	0
38	84	106	94	3
84	109	161	157	94
86	135	187	151	94
0	135	151	112	0

0	0	0	0	0
0	0	0	0	0
0	0	161	0	0
0	0	187	0	0
0	0	0	0	0

T=161

Figure 4.8 : Thresholding using Basic Global Thresholding

4.2.4 Image sharpening

Image sharpening refers to any enhancement technique that highlights edges and fine details in an image, increasing yields a more sharpened image.

High-Pass Filtering:

A high-pass filter can be used to make an image appear sharper. These filters emphasize fine details in the image. Here the output from the thresholding is given as input. We are making use of a filter, first we append the nearest values to pixels at the boundary pixels. The Figure 4.11 depicts Image Sharpening using High-Pass Filter.

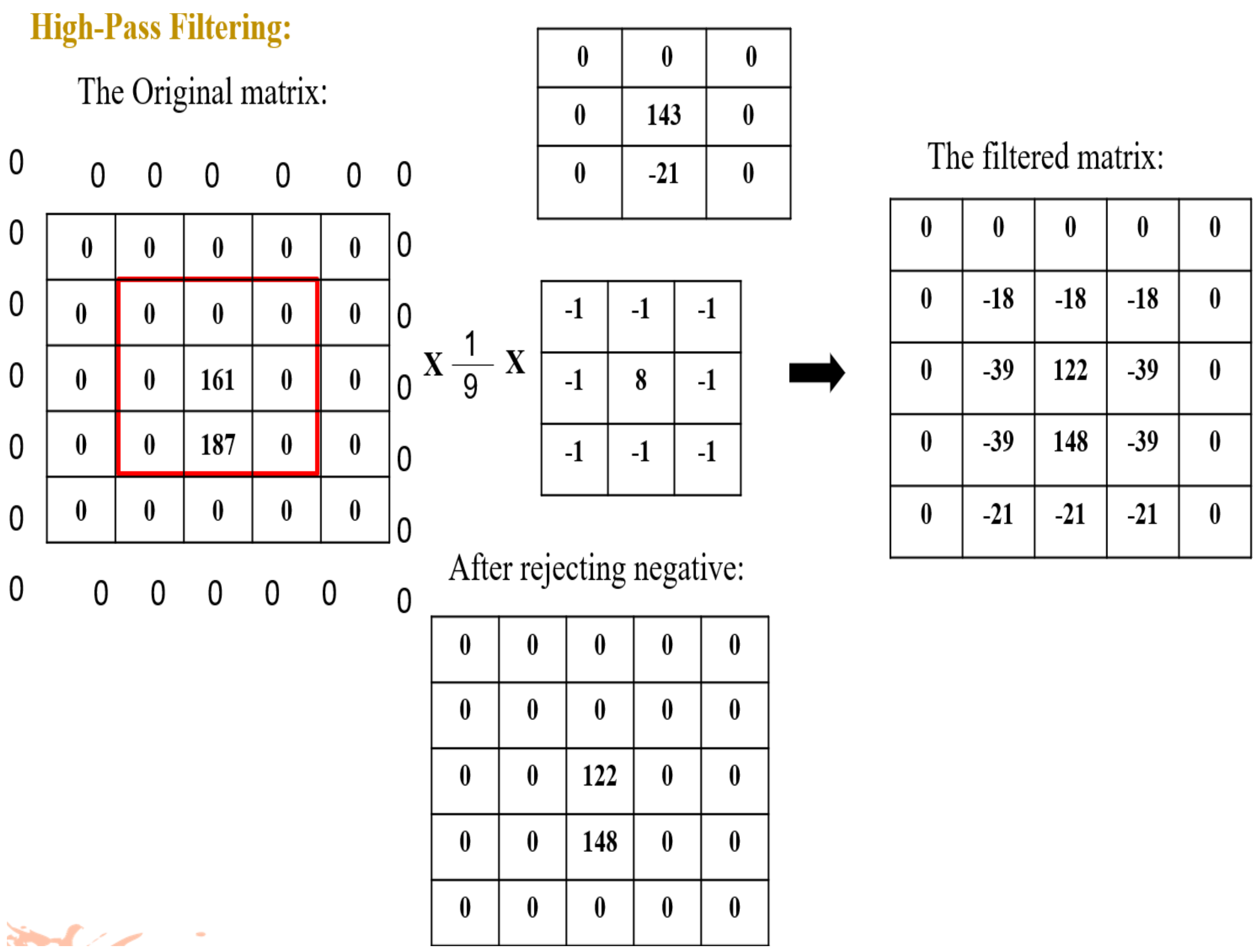


Figure 4.9 : Image Sharpening using High-Pass Filter

The figure 4.9 shows the image sharpening using high-pass filter.We multiply the elements of the 3*3 input matrix with the filter matrix, this can be represented as A (1,1) *B (1,1), in this way all the elements in the 3*3 are multiplied and their sum is divided by 9, which gives the value for the particular pixel position. In the same way the values of all the pixel positions are calculated.

4.2.5 Classification using Convolutional Neural Networks

In CNN, we take the output from the high-pass filter as input, leaving out feature extraction, as CNN is a classifier which simply has a feature extracting process of its own, using convolution, rectification and pooling as the 3 sub-modules, which work in iterations to give out a final comparison matrix.The figure 4.10 shows the fully connected layer and output layer.

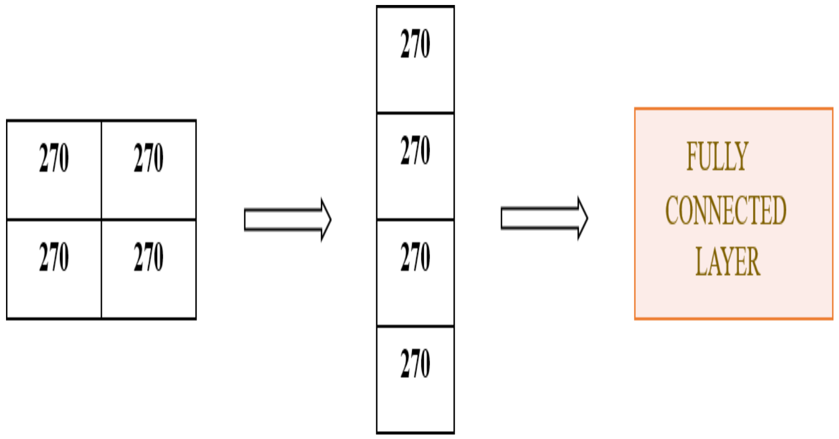


Figure 4.10 : Fully Connected Layer and Output Layer

4.2.6 Typical CNN Architecture

CNN architecture is inspired by the organization and functionality of the visual cortex and designed to mimic the connectivity pattern of neurons within the human brain. The neurons within a CNN are split into a three-dimensional structure, with each set of neurons analyzing a small region or feature of the image. CNNs use the predictions from the layers to produce a final output that presents a vector of probability scores to represent the likelihood that a specific feature belongs to a certain class.

A CNN is composed of several kinds of layers:

4.2.6.1 **Convolutional layer**-Creates a feature map to predict the class probabilities for each feature by applying a filter that scans the whole image, few pixels at a time.

4.2.6.2 **Pooling layer (down sampling)**-Scales down the amount of information the convolutional layer generated for each feature and maintains the most essential information (The process of the convolutional and pooling layers usually repeats several times).

4.2.6.3 **Fully connected layer**- “Flattens” the outputs generated by previous layers to turn them into a single vector that can be used as an input for the next layer. Applies weights over the input generated by the feature analysis to predict an accurate label.

4.2.6.4 **Output layer**-Generates the final probabilities to determine a class for the image.

The Figure 4.13 represents the Typical CNN Architecture.

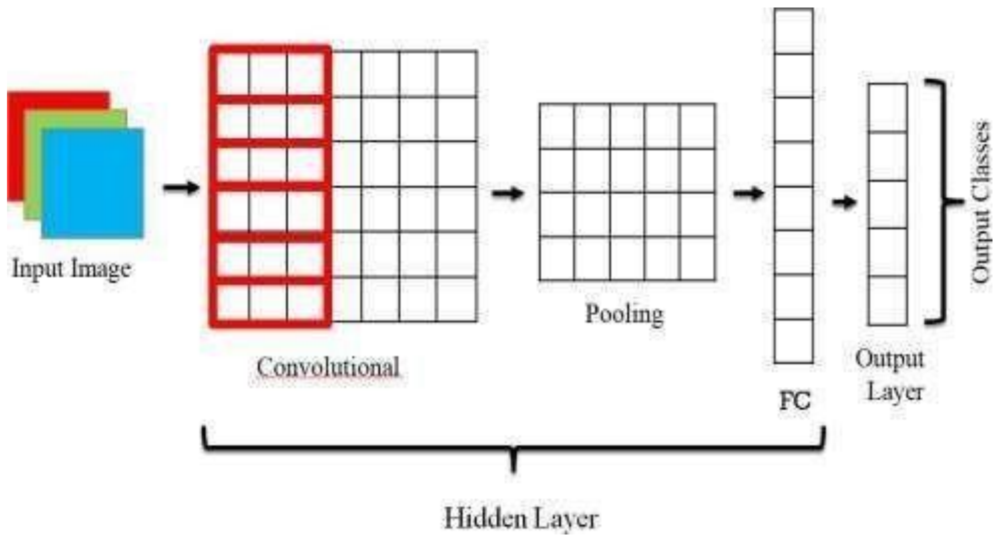
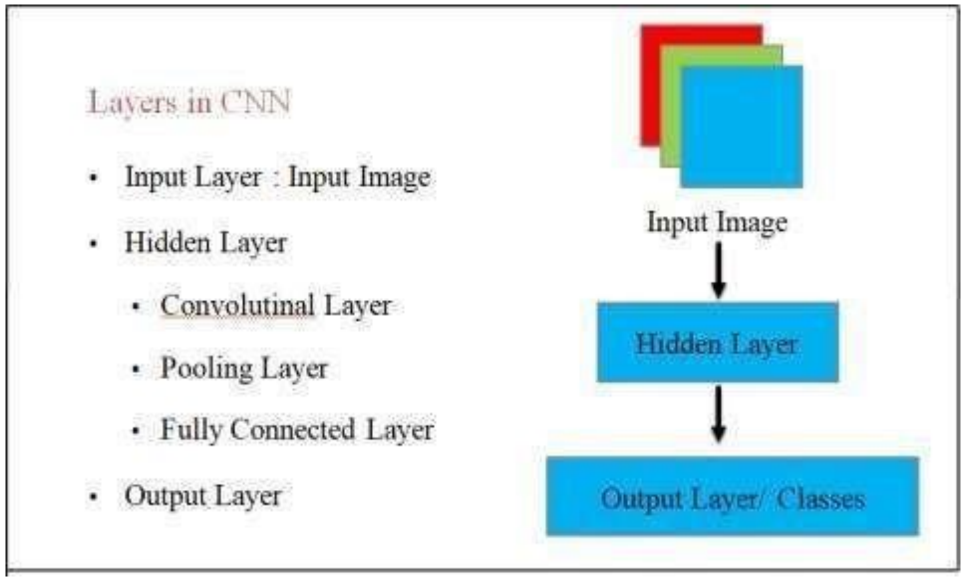


Figure 4.11: Typical CNN Architecture.

The Figure 4.14 represents the Layers in CNN.


Layers in CNN

- Input Layer : Input Image
- Hidden Layer
    - Convolutinal Layer
    - Pooling Layer
    - Fully Connected Layer
- Output Layer

Figure 4.12 : Layers in CNN

Convolutional Layer

Convolutional Layer is the first step in CNN, here 3*3 part of the given matrix which was obtained from High-pass filter is given as input. That 3*3 matrix is multiplied with the filter matrix for the corresponding position and their sum is written in the particular position. This is shown in the below figure. This output is given to pooling layer where the matrix is further reduced. Figure 4.15 shows the Convolutional Layer.

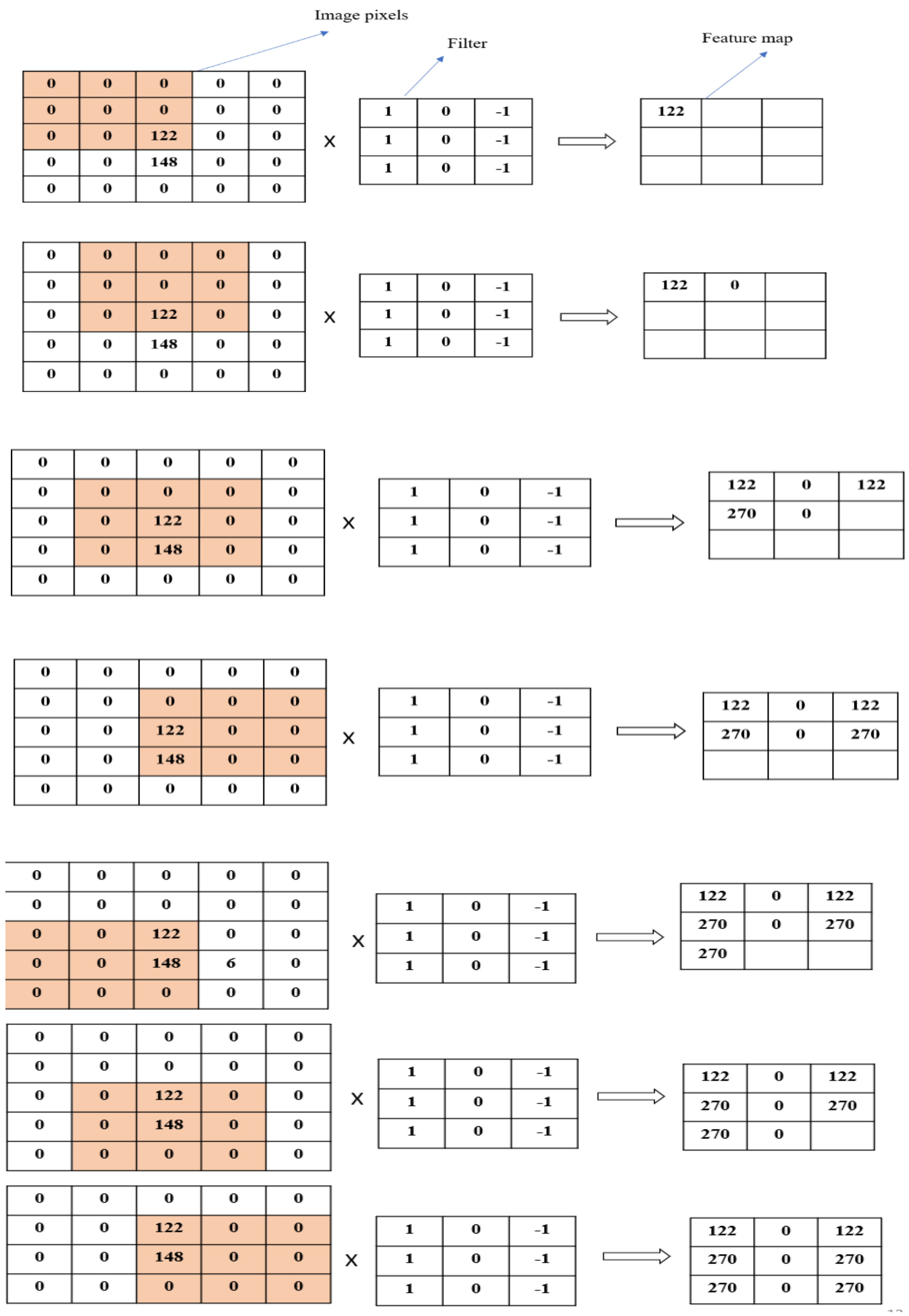


Image pixels
Filter
Feature map

122	**0**	**122**
270	**0**	**270**
270	**0**	**270**

Figure 4.13: Convolutional Layer.

The figure 4.13 shows the Convolutional layer.Convolution is followed by the rectification of negative values to 0s, before the pooling layer. Here, it is not demonstratable, as all values are positive. In fact, multiple iterations of both are needed before pooling.

Pooling Layer

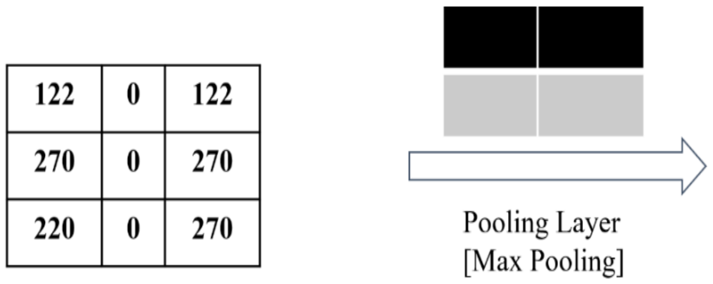


Figure 4.14 : Pooling Layer

The figure 4.14 shows the pooling layer.In Pooling layer 3*3 matrix is reduced to 2*2 matrix, this is done by selecting the maximum of the particular 2*2 matrix for the particular position. Figure 4.16 shows the Pooling Layer.

Fully connected layer and Output Layer

The output of the pooling layer is flattened and this flattened matrix is fed into the Fully Connected Layer. In the fully connected layer there are many layers, Input layer, Hidden layer and Output layers are parts of it. Then this output is fed into the classifier, in this case SoftMax Activation Function is used to classify the image into PCOS or Non PCOS. Figure 4.17 shows the Fully connected layer and Output Layer.The figure 4.15 shows the fully connected layer and output layer.

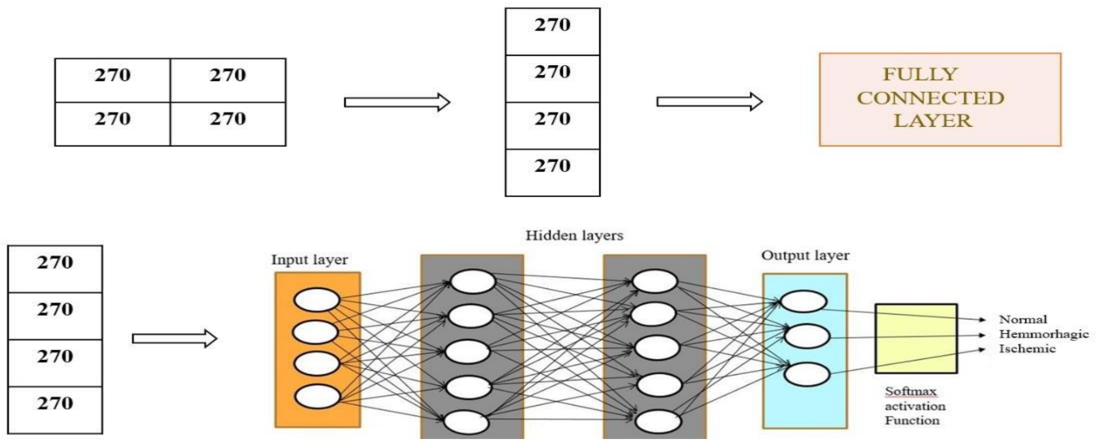


Figure 4.15 : Fully Connected Layer and Output Layer.

SOFTMAX ACTIVATION FUNCTION

The SoftMax activation function is a nonlinear function that takes a vector of real numbers, makes the function useful for tasks such as multi-class classification, where the output of the network needs to be a probability distribution over a set of possible classes.The figure 4.16 shows the softmax activation function.

Formula: $$\text{Softmax } fi(x) = \frac{e^{ai}}{\sum_{k=1}^{n} e^{ak}}$$

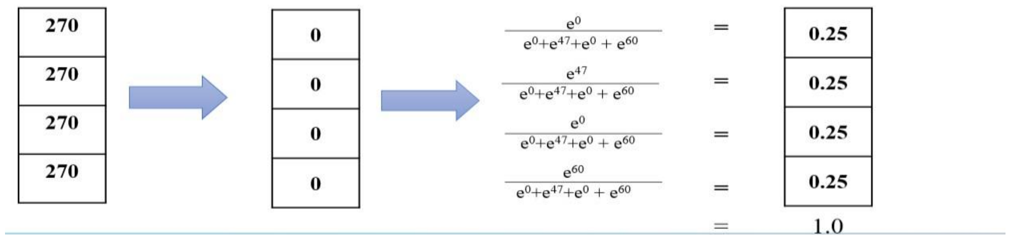

270
270
270
270
0
0
0
0
e^0 / e^0+e^47+e^0 + e^60 = 0.25
e^47 / e^0+e^47+e^0 + e^60 = 0.25
e^0 / e^0+e^47+e^0 + e^60 = 0.25
e^60 / e^0+e^47+e^0 + e^60 = 0.25
= 1.0

Figure 4.16 : Softmax Activation Function

Output of CNN: It gives prediction of autism(present or not).

4.3 Summary

The fourth chapter describes the board work for the proposed system. Various algorithms used are explained in detail. Support Vector Machine is illustrated in section 4.1. The classification of image using CNN is explained in section 4.2.

Chapter 5

IMPLEMENTATION

5.1 Implementation Requirements

To implement Detection using Convolutional Neural Network, software's used are

- Language used to code the project is Python.
- Operating System-Windows 10.

5.2 Visual Studio Code

Visual Studio Code is a freeware source-code editor made by Microsoft for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality.

Visual Studio Code is a source-code editor that can be used with a variety of programming languages, including Java, JavaScript, Go, Node.js, Python and C++. It is based on the Electron framework,[19] which is used to develop Node.js Web applications that run on the Blink layout engine. Visual Studio Code employs the same editor component (codenamed "Monaco") used in Azure DevOps (formerly called Visual Studio Online and Visual Studio Team Services).

5.3 Programming Language Used

Programming language used to design the proposed method is Python. Python is a high-level programming language with dynamic semantics. It is an interpreted language i.e. interpreter executes the code line by line at a time, thus makes debugging easy Python Imaging Library (PIL) is one of the popular libraries used for image processing. PIL can be used to display image, create thumbnails, resize, rotation, convert between file formats, contrast enhancement, filter and apply other digital image processing techniques etc.

Python is often used as a support language for software developers, for build control and management, testing, and in many other ways. Python is designed by Guido van Rossum. It is very easy for user to learn this language because of its simpler coding.

It provides an easy environment to furnish computation, programming visualization. Python supports modules and packages, which encourages program modularity and code reuse. It has various built-in commands and functions which will allow the user to perform functional programming.

5.4 Key Features of Python

• Python is an interpreted language i.e. interpreter executes the code line by line at a time, thus makes debugging easy.

• Python is more expressive language, since it is more understandable and readable.

• To design and solve problems it offers an interactive atmosphere.

• Python has a large and broad library and provides rich set of module and functions for rapid application development.

• Built in graphics for conception of data and tools is also supported. integrated with languages like C, C++, JAVA etc. can be easilyIt

5.5 OpenCV-Python Tool

OpenCV-Python is a library of Python bindings designed to solve computer vision problems. Visual information is the most important type of information perceived, processed and interpreted by the human brain. Image processing is a method to perform some operations on an image, in order to extract some useful information from it. An image is nothing more than a two dimensional matrix (3-D in case of colored images) which is defined by the mathematical function f(x,y) where x and y are the two co- ordinates horizontally and vertically. The value of f(x,y) at any point is gives the pixel value at that point of an image, the pixel value describes how bright that pixel is, and/or what color it should be. In image processing we can also perform image acquisition, storage, image enhancement.

5.6 Flask

Flask (source code) is a Python web framework built with a small core and easy- to extend philosophy. Flask is a lightweight WSGI web application framework. It is designed to make getting started quick and easy, with the ability to scale up to complex applications. It began as a simple wrapper around Werkzeug and Jinja and has become one of the most popular Python web application frameworks. Flask offers suggestions, but doesn't enforce any dependencies or project layout. It is up to the developer to choose the tools and libraries they want to use. There are many extensions provided by the community that make adding new functionality easy.

5.7 Google Colab

Colaboratory, or “Colab” for short, is a product from Google Research. Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education. More technically, Colab is a hosted Jupyter notebook service that requires no setup to use, while providing free access to computing resources including GPUs. Colab resources are not guaranteed and not unlimited, and the usage limits sometimes fluctuate. This is necessary for Colab to be able to provide resources for free. Jupyter is the open source project on which Colab is based.

5.8 Packages

Packages are the namespaces which consists of multiple packages and module themselves. Each package in Python is a directory which must contain a special file called _init_py. This file can be empty, and it indicates that the directory it contains is a Python package, so it can be imported in the same way that the module can be imported. As our application program grows larger in size with a lot of modules, one can place the similar modules in one package and different modules in different packages. This makes a program easy to manage and conceptually clear. We can import the modules from packages using the dot (.) operator.

1. CSV

In this work, to import or export spread sheets and databases for its use in the Python interpreter, the CSV module, or Comma Separated Values format is used. These CSV files are used to store a large number of variables or data and the CSV module is a built-in function that allows Python to parse these types of files. The text inside a CSV file is organized in the form of rows, and each row consists of the columns, all separated by commas, indicates the separate cells in CSV file. There is no standard for CSV modules hence, these modules makes use of “dialects” to support parsing using different parameters. The CSV module includes all the necessary built-in functions, csv.reader and csv.writer are the most commonly used built-in functions in Python. These functions are given below: It will return a reader object which will iterate over lines in the given csvfile. csvfile can be any object which supports the iterator protocol and returns a string each time its next() method is called – file objects and list objects are both suitable. csv.reader (csvfile, dialect=‟excel‟, **fmtparams)

If csvfile is a file object, it must be opened with the „b‟ flag on platforms where that makes a difference. An optional dialect parameter can be given which is used to define a set of parameters specific to a particular CSV dialect. It may be an instance of a subclass of the Dialect class or one of the strings returned by the list_dialects() function. The other optional fmtparams keyword arguments can be given to override individual formatting parameters in the current dialect. Each row read from the csv file is returned as a list of strings. No automatic data type conversion is performed.

csv.writer (csvfile, dialect="excel", **fmtparams)

It returns a writer object responsible for converting the user"s data into delimited strings on the given file-like object. csvfile can be any object with a write() method. If csvfile is a file object, it must be opened with the „b" flag on platforms where that makes a difference. An optional dialect parameter can be given which is used to define a set of dialect.

It may be an instance of a subclass of the Dialect class or one of the strings returned by the list_dialects() function. The other optional fmtparams keyword arguments can be given to override individual formatting parameters in the current dialect. Floats are stringified with repr() before being written. All other non-string data are stringified with str() before being written.

2. Tensorflow

Tensorflow is a free and open source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural network, it is used for both research and production at Google. Tensorflow was developed by the Google Brain team for internal

Google use. It was released under the Apache License 2.0 on November 9, 2015. Tensorflow is Google Brain"s second-generation system. Version 1.0.0 was released on February 11,2017. While the reference implementation runs in single devices, Tensorflow can run on multiple CPUs and GPUs. Tensorflow is available on 64- bit Linux, macOS, Windows and mobile computing platforms including Android and ios. Its flexible architecture allows for the easy deployment of computation across a variety of platforms (CPUs, GPUs and TPUs), and from desktops to clusters to servers to mobile and edge devices.

3. Keras

Keras is an open source neural- network library written in Python. It is capable of running on top of Tensorflow, Microsoft Cognitive Toolkit, R, Theano or PlaidML. It is designed to enable fast experimentation with deep neural networks. It focuses in being user-friendly, modular, and extensible CUDA. Keras contains numerous implementations of commonly used neural- network building blocks such as layers, objectives, activation function, optimizers, and a host of tools to make working with image and text data easier to simplify the code necessary for writing deep neural network code. The code is hosted on GitHub, and commonly support forums include the

GitHub issues page, and a Slack channel. In addition to Standard neural networks, Keras has support for convolutional and recurrent neural networks.

4. NumPy

NumPy is a python library used for working with arrays. It also has functions for working in domain of linear algebra, fourier transform, and matrices. NumPy was created in 2005 by Travis Oliphant. It is an open source project and you can use it freely. NumPy stands for Numerical Python. In Python we have lists that serve the purpose of arrays, but they are slow to process. NumPy aims to provide an array object that is up to 50x faster than traditional Python lists. The array object in NumPy is called ndarray, it provides a lot of supporting functions that make working with ndarray very easy. Arrays are very frequently used in data science, where speed and resources are very important.

5. Matplotlib

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib produces publication-quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shell, web application servers, and various graphical user interface toolkits. Matplotlib. pyplot is a collection of command style functions that make matplotlib work like MATLAB. Each pyplot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc. Some people embed matplotlib into graphical user interfaces like wxpython or pygtk to build rich applications.

5.9 Pseudocodes For Preprocessing Techniques

5.9.1 Pseudocode for Picture Uploading

This code snippet represents a process for handling input images, likely within a web application designed to analyse medical images for indications of Autism Spectrum Diagnosis (ASD). In the provided steps, the user is prompted to upload an image. Upon uploading, the code reads the image from the dataset using the OpenCV library (cv2.imread(args[“first”])). The name of the image is then displayed on the webpage (Display the name of the image on the webpage). Additionally, a dialog box appears, showing the name of the uploaded image (Shows the name of the image in the dialog box).

Input image is obtained from the dataset which represents a healthy or PCOS patient.

Step 1: Click on upload

Step 2: Read the image from dataset imageA=cv2.imread(args[“first”])

Step 3: Display the name of the image on the webpage

Step 4: Shows the name of the image in the dialog box

5.9.2 Pseudocode for thresholding

This code snippet is a simple algorithm designed to process an image and segment it into regions of interest, typically used in applications like obstacle detection in robotics or image processing tasks. The code iterates over each pixel in a 480x640 image. For each pixel,it checks if its intensity value in the image is less than a threshold value, T. If it is below the threshold, the pixel is changed to black, indicating an obstacle, and included in the obstacle set. Otherwise, it's changed to white, indicating free space.

Step 1: for i:=1 to 480*640 pixels

Step 2: if new_im(i)<T

Step 3: Change to black, and consider as obstacle set

new_im(i)=0

Step 4: Change to white, and consider as free space else set
new_im(i)=1

5.9.3 Pseudocode for high pass filtering

A high-pass filter can be used to make an image appear sharper. These filtersemphasize fine details in the image- which is exactly the opposite of low pass filters. High pass filtering works in exactly the same way as a low pass filter except that it uses a differentconvolution kernel. Unfortunately, while low-pass filtering smooths out noise, high-pass onlydoes the opposite by amplifying noise. However, if the original image is not too noisy- this isavoided to an extent.

Step1: Displays the high contrast version of original image.

Step2: Make a copy of the original image.

Initialize color[ ][ ] retvāl copy(original)

```
Step3: Blur the copy of original image.
set color[ ][ ]blurred̄  gaussianBlur(original)
Step4: Subtract the blur image from original image.
        Color[ ][ ]highpass̄  difference(original,blurred)
Step5: Add the unsharp mask to original image to get sharpened image.
Step6: Run a loop for the entire rows and columns
for row=0 to original.length
repeat until
for col=0 to original[row].length
repeat until
initialize color origColororiginal[row][col]
initialize contrast ColorhighContrast[row][col]
set color difference
contrastColor - origColor
end for
end for
```

5.10 Pseudocode for CNN Architecture

```
verify_dir = 'static/images'
    IMG_SIZE = 50
    LR = 1e-3
    MODEL_NAME = 'ASD_DETECTION1-{}-
{}.model'.format(LR, '2conv-basic')
  ##
    MODEL_NAME='keras_model.h5'
    def process_verify_data():
      verifying_data = []
      for img in os.listdir(verify_dir):
        path = os.path.join(verify_dir, img)
        img_num = img.split('.')[0]
```

```
                img = cv2.imread(path, cv2.IMREAD_COLOR)
                img = cv2.resize(img, (IMG_SIZE, IMG_SIZE))
                verifying_data.append([np.array(img), img_num])
                np.save('verify_data.npy', verifying_data)
            return verifying_data
        verify_data = process_verify_data()
        #verify_data = np.load('verify_data.npy')
        tf.compat.v1.reset_default_graph()
        #tf.reset_default_graph()
                convnet = input_data(shape=[None,
          IMG_SIZE,IMG_SIZE, 3], name='input')
        convnet = conv_2d(convnet, 32, 3, activation='relu')
        convnet = max_pool_2d(convnet, 3)
        convnet = conv_2d(convnet, 64, 3, activation='relu')
        convnet = max_pool_2d(convnet, 3)
convnet = conv_2d(convnet, 128, 3, activation='relu')
convnet = max_pool_2d(convnet, 3)
convnet = conv_2d(convnet, 32, 3, activation='relu')
convnet = max_pool_2d(convnet, 3)
convnet = conv_2d(convnet, 64, 3, activation='relu')
convnet = max_pool_2d(convnet, 3)
convnet = fully_connected(convnet, 1024, activation='relu')
convnet = dropout(convnet, 0.8)
convnet = fully_connected(convnet, 2, activation='softmax')
convnet = regression(convnet, optimizer='adam', learning_rate=LR,
loss='categorical_crossentropy',      name='targets')
model = tflearn.DNN(convnet, tensorboard_dir='log')
if os.path.exists('{}.meta'.format(MODEL_NAME)):
    model.load(MODEL_NAME)
    print('model loaded!')
```

```
  fig = plt.figure()str_label=" "
  accuracy=""
           for num, data in enumerate(verify_data):
      img_num = data[1]
      img_data = data[0]
      y = fig.add_subplot(3, 4, num + 1)
      orig = img_data
      data = img_data.reshape(IMG_SIZE, IMG_SIZE, 3)
      # model_out = model.predict([data])[0]
      model_out =
      model.predict([data])[0]
      print(model_out)
      print('model {}'.format(np.argmax(model_out)))
 if np.argmax(model_out) == 0:
     str_label = "Normal"
     print("The predicted image of the Normal is with a accuracy of {} %".
     format(model_out[0]*100))
     accuracy="The predicted image of the Normal is with a accuracy of {:.2f}%".
     format(model_out[0]*100)
elif np.argmax(model_out) == 1:
      str_label  = "ASD"
      print("The predicted image of the ASD is with a accuracy of {} %".
     format(model_out[1]*100))
      accuracy="The predicted image of the ASD is with a accuracy of {:.2f}%".
      format(model_out[1]*100)
  return
     render_template('userlog.html',status=str_label,accuracy=accuracy,
     ImageDisplay="http://127.0.0.1:5000/static/images/"+fileName,
     ImageDisplay1="http://127.0.0.1:5000/static/gray.jpg",
     ImageDisplay2="http://127.0.0.1:5000/static/edges.jpg",
     ImageDisplay3="http://127.0.0.1:5000/static/threshold.jpg")
    return render_template('userlog.html')
    if__name__== "_main ":
      app.run(debug=True)
```

The provided Python script encapsulates a comprehensive ASD (Autism Spectrum Diagnosis) detection system integrated into a Flask web application. The system employs convolutional neural networks (CNNs) for image analysis, aiming to discern between normal and ASD-affected ovaries. Here's a breakdown of its key components and functionality:

The script begins by setting up essential parameters such as the directory containing verification images (verify_dir), image size for processing (IMG_SIZE), learning rate (LR), and the name of the model file (MODEL_NAME).A crucial function, process_verify_data(), is defined to preprocess the verification data. This function iterates through the images in the specified directory, resizes them to the designated dimensions, and prepares them for further analysis. The processed data is then stored in a NumPy array for efficient handling.

Subsequently, the script constructs the CNN model using TensorFlow and TFLearn. The model architecture comprises multiple convolutional layers, followed by max-pooling layers for feature extraction. Fully connected layers are incorporated for high-level feature learning, along with dropout regularization to mitigate overfitting. The model is compiled with appropriate activation functions and loss functions tailored for binary classification (Normal or ASD.

5.11 Summary

This chapter describes the implementation of the Autism Spectrum detection using CNN architectures. Implementation requirement is deliberated in Section 5.1, Section 5.2 briefs about the programming language selected. Further sections describe the platforms that help in implemention. Section 5.9 describes the pseudocode for preprocessing techniques and Section 5.10 describes the pseudocode for CNN Architecture.

Chapter 6

SYSTEM TESTING

Testing is the significant phase in the process or application development life cycle.Testing is final phase where the application is tested for the expected outcomes. Testing of the system is done to identify the faults or prerequisite missing. Throughout the testing phase, the procedure will be implemented with the group of test circumstances and the outcome of a procedure for the test case will be appraised to identify whether the program is executing as projected. Some of the significant aim of the system testing is,

- To confirm the superiority of the project.
- To discover and eradicate any residual errors from prior stages.
- To authenticate the software as a result to the original problem.
- To offer effective consistency of the system.

6.1 Testing of the Autism Prediction System

Testing is a critical phase in the development lifecycle of the Autism Prediction System. The objective of the testing process is to identify any defects in the system, validate the functionalities, ensure reliability, and improve the performance of the system.

1. Unit Testing

Unit testing involves testing individual components or functions of the system in isolation to verify that each part of the program performs as expected. The main aim is to ensure that each module of the system-be it data preprocessing, model training, prediction generation, or result visualization-works independently.

Tests include:

- **Data Preprocessing**: Verify the cleaning and normalization of behavioral data, as well as the preprocessing of MRI images.
    - Example: Test the cleaning algorithm by introducing known inconsistencies or missing values in the dataset and verifying if the system appropriately handles them.
- **Model Training**: Ensure that machine learning models and CNN architectures are trained properly.

  - Example: Check if the training function converges within an acceptable number of epochs and doesn't overfit the data.
- **Prediction Output**: Confirm that the prediction generation function correctly outputs autism predictions based on the input data.
  - Example: Test the prediction function with known input samples and check if the output matches the expected result.

Unit tests are conducted using automated testing frameworks such as **JUnit** or **pytest**. Each component is tested with various valid and invalid input cases to verify that it handles different scenarios correctly.

2. Integration Testing

Integration testing verifies the interactions between different modules of the system. After unit testing, integration tests are conducted to ensure that the modules work together as intended.

Tests include:

- **Behavioral Data and Model Integration**: Ensure that the preprocessed behavioral data integrates correctly with the machine learning model and generates accurate predictions.
  - Example: Check that the preprocessed behavioral data correctly feeds into the trained model and generates output without errors.
- **MRI Image and CNN Integration**: Test if the preprocessed MRI images can be passed to the CNN model for prediction.
  - Example: Validate that the input images are correctly resized and normalized, and if the CNN produces predictions as expected.
- **Data Storage and Retrieval**: Test that the data storage system correctly stores both behavioral data and MRI images and can retrieve them efficiently when required.
  - Example: Check if the system can save and retrieve large datasets like MRI images and prediction results without performance degradation.

This phase ensures that multiple modules interact correctly with each other, and no unexpected errors arise due to module interdependencies.

3. System Testing

System testing ensures that the entire Autism Prediction System works as a whole and meets the specified requirements. It evaluates the system’s end-to-end functionality, covering all the features and user scenarios.

Tests include:

- **End-to-End Workflow**: Test the entire workflow from user input (e.g., entering behavioral data and uploading MRI images) to prediction generation and result display.
    - Example: Simulate a clinician’s use case where they enter patient data, upload MRI images, and view the autism prediction results to confirm the system functions correctly across all stages.
- **User Interface (UI) Testing**: Test the UI for usability, responsiveness, and correctness. Ensure that the system is easy to navigate and the predictions are displayed in a user-friendly manner.
    - Example: Validate that the UI displays relevant error messages when invalid data is entered and check if the predictions are presented clearly.
- **Security Testing**: Test the authentication and authorization mechanisms to ensure that only authorized personnel can access sensitive data and predictions.
    - Example: Verify if the login functionality works as expected, and ensure role-based access control is implemented properly for different users (clinicians, admins).

System testing ensures that the complete system meets the requirements outlined in the project specifications and is ready for real-world deployment.

4. Performance Testing

Performance testing is crucial to ensure that the Autism Prediction System performs efficiently under different conditions, such as varying data volumes and user loads.

Tests include:

- **Speed and Response Time**: Evaluate the response time for the prediction process, both for the machine learning model and the CNN-based model.
    - Example: Measure the time taken to process behavioral data and MRI images and generate predictions. Ensure that the system provides results within an acceptable timeframe.
- **Scalability**: Test how the system performs when handling large datasets. For example, test how well the system handles a large number of MRI images or a large volume of behavioral data.
    - Example: Stress-test the system with a dataset containing thousands of samples to check if performance degrades.

- **Resource Usage**: Assess how efficiently the system utilizes computational resources like memory and CPU. The system should not excessively consume system resources during operation.
    - Example: Measure the CPU and memory usage during the training phase of the model and when making predictions.

Performance testing ensures that the system is capable of delivering timely results, even under heavy load or in large-scale deployment.

5. Usability Testing

Usability testing focuses on how easily end-users (clinicians, researchers, or caregivers) can interact with the Autism Prediction System. The system should be intuitive, user-friendly, and accessible.

Tests include:

- **Ease of Use**: Assess whether the user interface is intuitive and if the user can easily navigate through the system's various features.
    - Example: Observe clinicians using the system and ask for feedback on the ease of data entry, image uploading, and result interpretation.
- **Error Handling and Feedback**: Ensure that the system provides clear error messages and user-friendly feedback when something goes wrong.
    - Example: Test the system by entering invalid data or uploading corrupted files to see if the system responds with appropriate messages.
- **Accessibility**: Test whether the system can be used by people with disabilities. For example, check if the UI is compatible with screen readers or other assistive technologies.

Usability testing helps to ensure that the Autism Prediction System is accessible, intuitive, and easy to use for its target audience.

6. Regression Testing

Regression testing is performed to ensure that new updates, modifications, or enhancements to the system do not negatively affect the existing functionality.

Tests include:

- **Check Existing Functionality**: After new changes are made (e.g., adding new prediction algorithms or modifying the UI), test to ensure that all previously working features still function as expected.

 - Example: After implementing a new feature, check if the prediction results from the existing models are still accurate.
- **Compatibility Testing**: Ensure that updates or changes do not cause compatibility issues with other components or systems.
  - Example: After updating the machine learning model, verify that it integrates smoothly with the rest of the system and works on different operating systems or devices.

Regression testing ensures that new modifications don't disrupt the system's overall functionality, providing confidence that new features are integrated successfully.

7. User Acceptance Testing (UAT)

User Acceptance Testing (UAT) is the final phase of testing, where real end-users (clinicians, healthcare professionals) validate whether the system meets their expectations and requirements.

Tests include:

- **Real-World Scenarios**: Simulate real-world use cases where end-users interact with the system.
  - Example: Test how clinicians use the system in clinical settings, uploading patient data and viewing predictions for different patients.
- **Feedback Collection**: Collect feedback from users regarding the system’s accuracy, usability, and performance.
  - Example: Conduct surveys or interviews with clinicians to gather insights on how the system can be improved.

UAT is crucial to ensure that the system meets end-users' needs and is ready for real-world deployment.

6.2 Summary

This chapter presents system testing in section 6.1, which consists of unit test cases for the various modules of the polycystic ovary detection system.

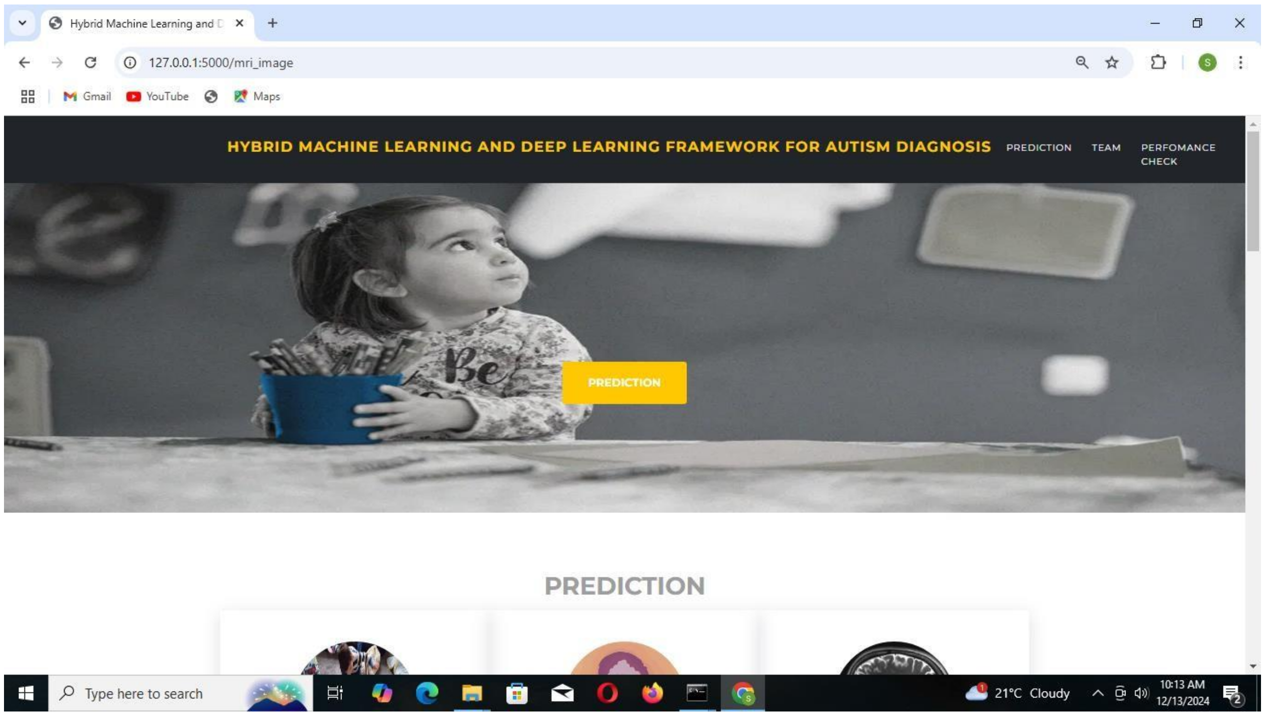
Chapter 7

RESULTS AND DISCUSSION

The Autism Prediction System, combining machine learning for behavioral analysis and CNN-based image classification for autism detection, underwent several phases of testing and evaluation. This section presents the outcomes of the experiments and results obtained from the system, followed by a detailed discussion of the effectiveness and potential improvements in the system.

7.1 Experimental Results

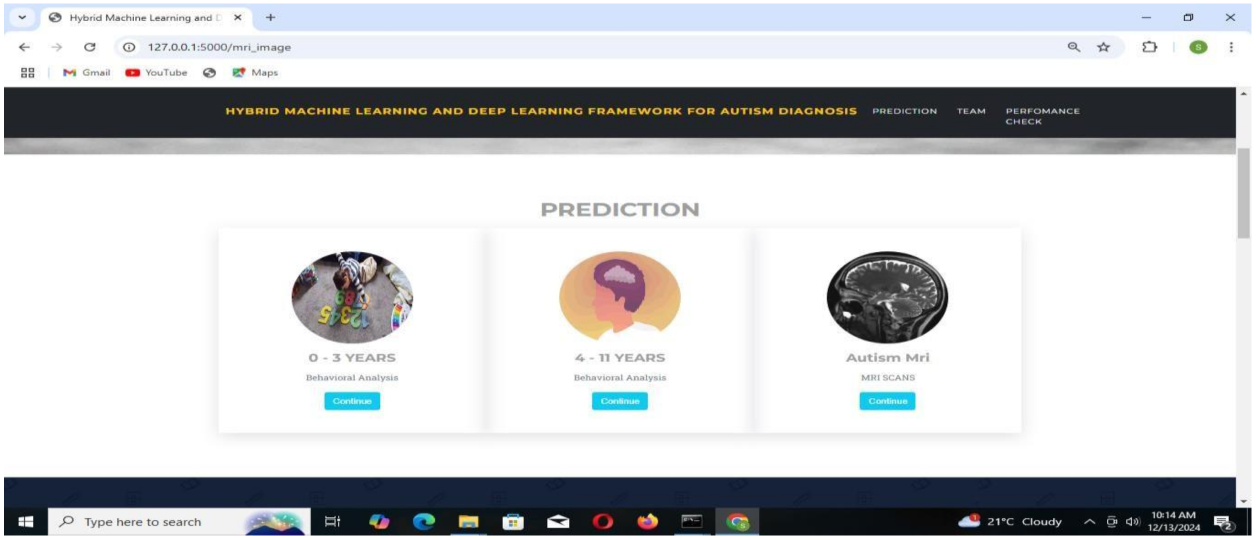
7.1.1 Snapshot of Home Page



Snapshot 7.1: Snapshot of Home Page.

The Snapshot 7.1 shows the snapshot of home page which consists of a header which has our project title, reads prediction,team and perfoormance check.Prediction Button goes to prediction,team button will display the teammates of project and performance check display the performance graph.

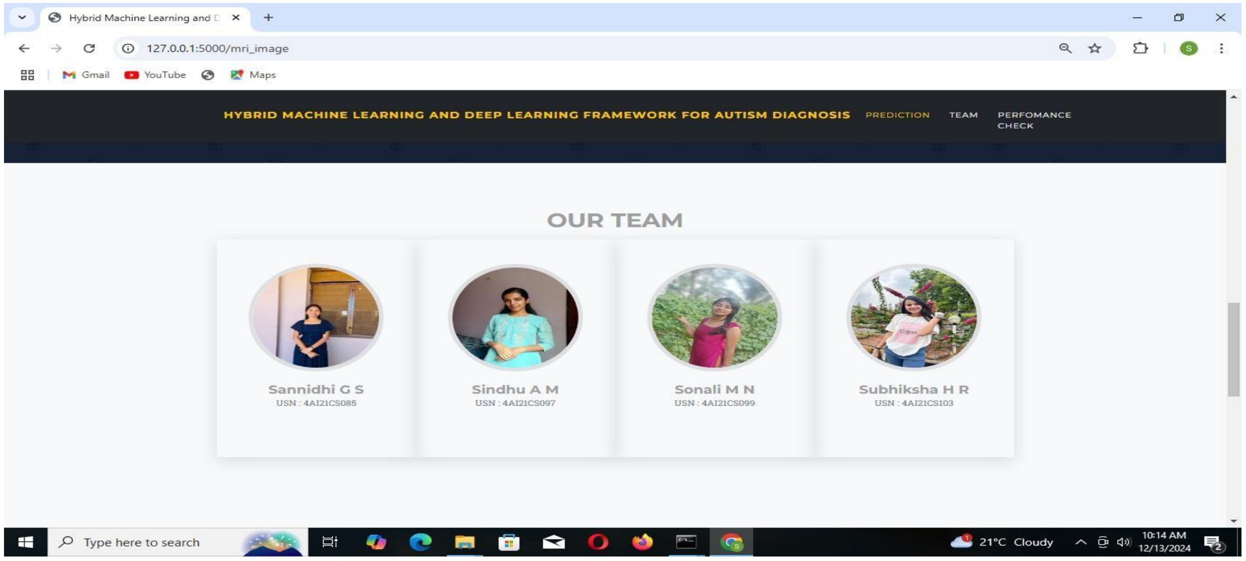
7.1.2 Snapshot of Prediction Page



Snapshot 7.2:Snapshot of Prediction Page.

The Snapshot 7.2 shows the prediction page which shows the behavioral analysis on different age groups i.e 0-3 years and 4-11 years and Autism MRI which scans the MRI image to detect stages in autism that are mild, moderate and severe and also display the input image with probability graph.

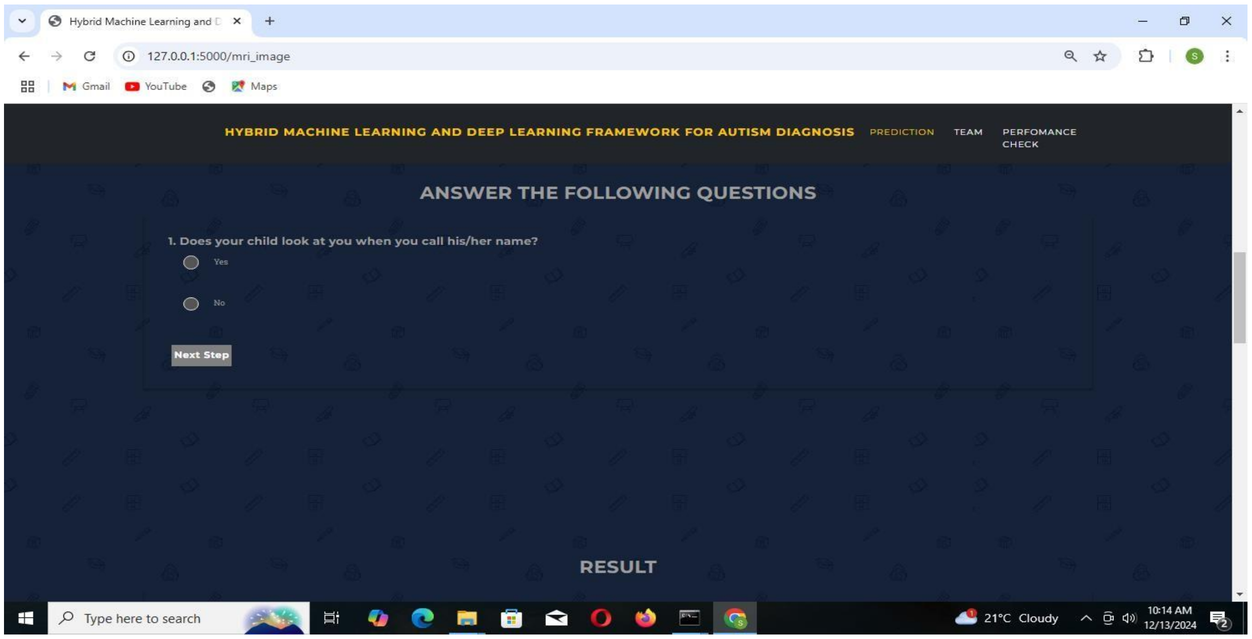
7.1.3 Snapshot of Team Page



Snapshot 7.3 : Snapshot of Team Page

The Snapshot 7.3 shows the team members those are sannidhi, sindhu, sonali and subhiksha of project along with there USN 4AI21CS084,4AI21CS097,4AI21CS099 and 4AI21CS103 respectively with the pictures.

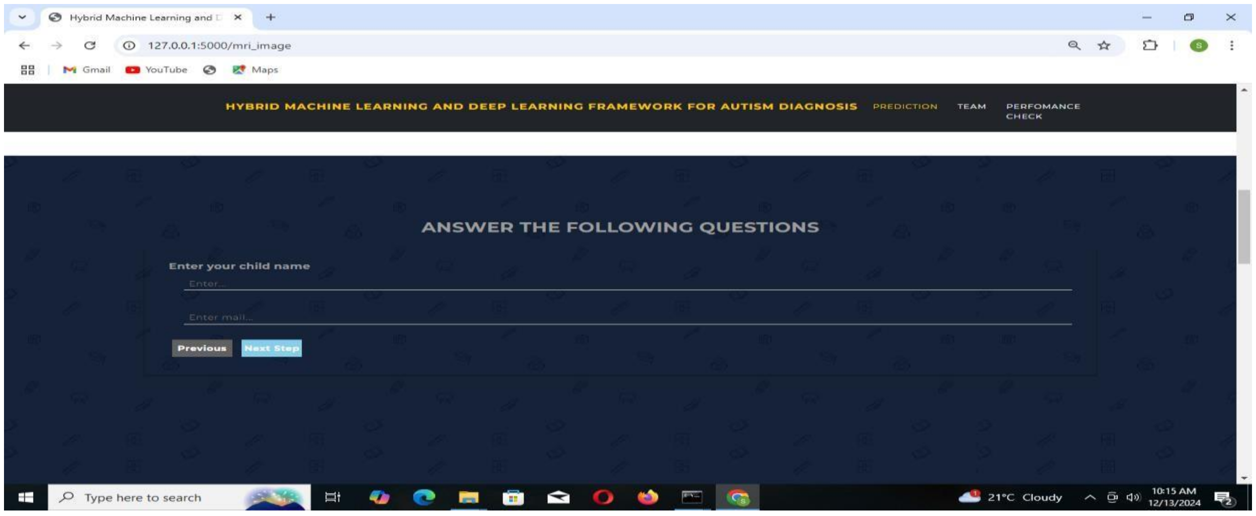
7.1.4 Snapshot of Question Page


127.0.0.1:5000/mri_image
HYBRID MACHINE LEARNING AND DEEP LEARNING FRAMEWORK FOR AUTISM DIAGNOSIS
PREDICTION
TEAM
PERFOMANCE CHECK
ANSWER THE FOLLOWING QUESTIONS
1. Does your child look at you when you call his/her name?
Yes
No
Next Step
RESULT

Snapshot 7.4 : Snapshot of Question Page

The Snapshot 7.4 shows the question and answer page where the question and options will displays on the screen select the option according to the behavior of the child and proceed to the next step.

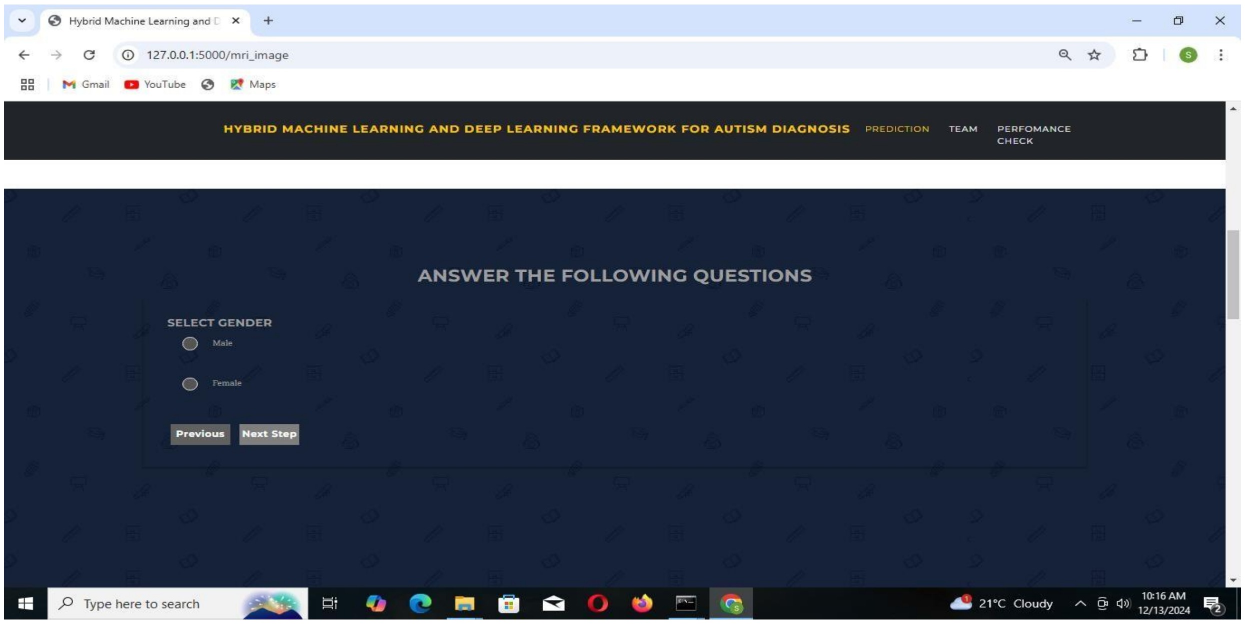
7.1.5 Snapshot of Child Details Page



Snapshot 7.5 : Snapshot of Child Details Page

The Snapshot 7.5 shows the next step after question and answer this shows to enter the child name and email address and proceed to the next step.This steps involve accessing the details the child from parents.

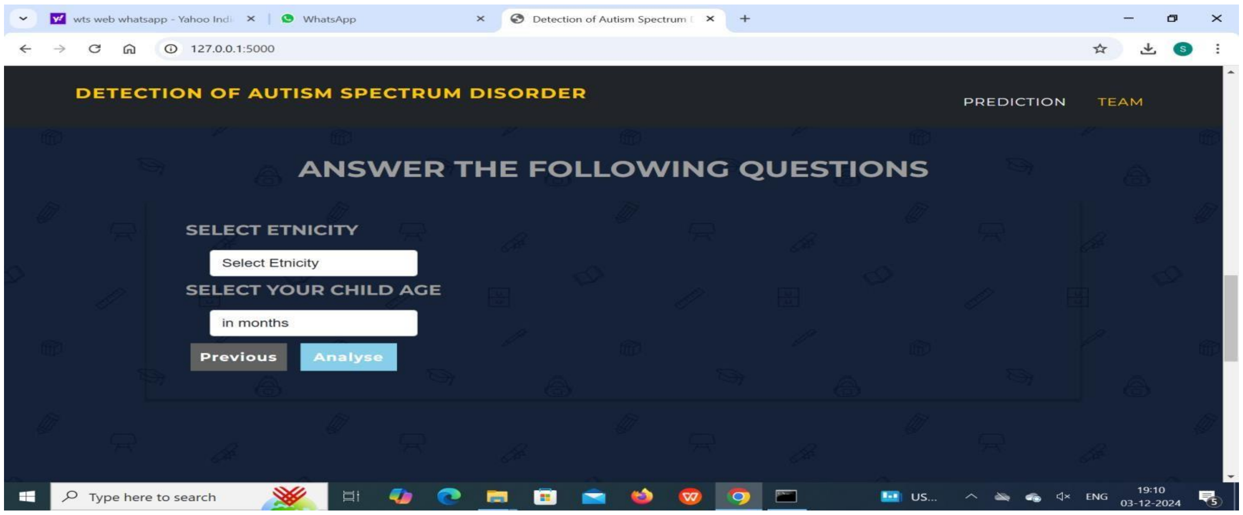
7.1.6 Snapshot of Child Gender Details Page


127.0.0.1:5000/mri_image
HYBRID MACHINE LEARNING AND DEEP LEARNING FRAMEWORK FOR AUTISM DIAGNOSIS
PREDICTION
TEAM
PERFOMANCE CHECK
ANSWER THE FOLLOWING QUESTIONS
SELECT GENDER
Male
Female
Previous
Next Step

Snapshot 7.6 : Snapshot of Child Gender Details Page

The Snapshot 7.6 shows the next step after entering the child name and email address it shows to enter gender of the child and proceed to the next step.This step involving the obtain child is whether male or female.

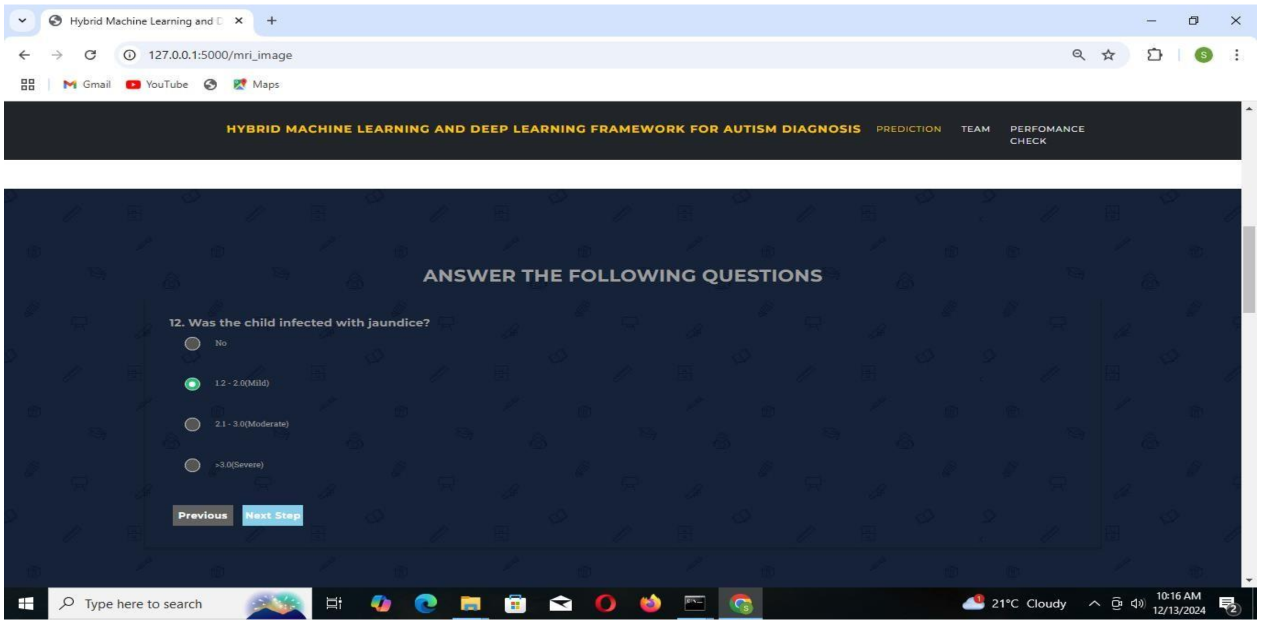
7.1.7 Snapshot of Child Age Details Page



Snapshot 7.7: Snapshot of Child Age Details Page

The Snapshot 7.7 shows the next step after entering the gender it ask to select the ethnicity of child and select the child age for analyzing an press analyse button to give result.The ethnicity are like asian,south asian and so on along with the child age.

7.1.8 Snapshot of Child Infected Details Page


HYBRID MACHINE LEARNING AND DEEP LEARNING FRAMEWORK FOR AUTISM DIAGNOSIS
PREDICTION
TEAM
PERFOMANCE CHECK
Previous
Next Step

127.0.0.1:5000/mri_image

ANSWER THE FOLLOWING QUESTIONS

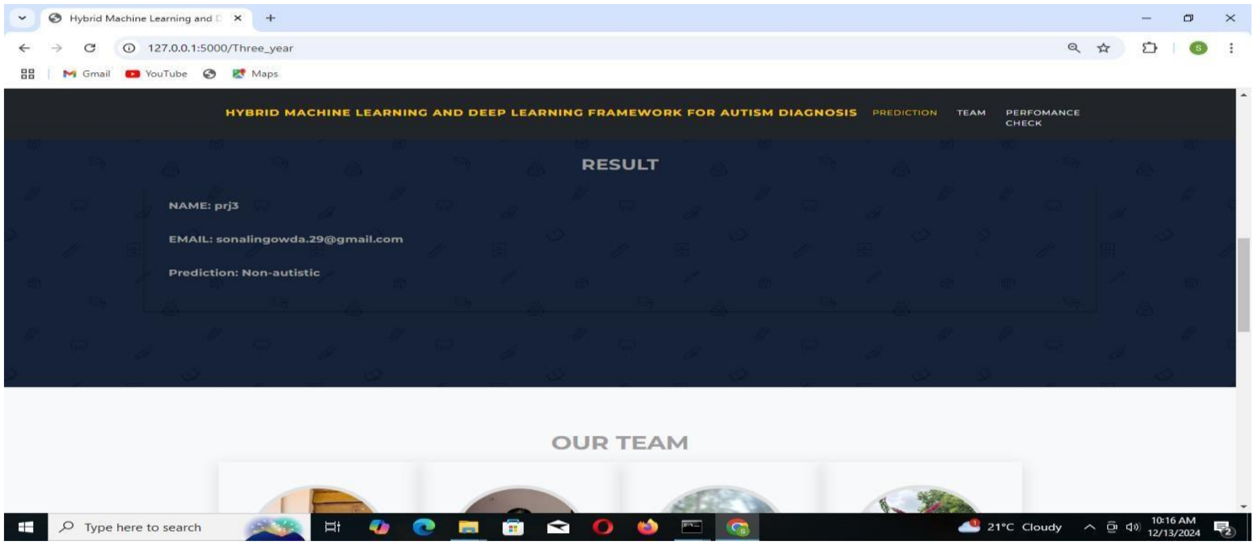
12. Was the child infected with jaundice?

No [✓ 1.2 - 2.0(Mild)] 2.1 - 3.0(Moderate) >3.0(Severe)

Snapshot 7.8 : Snapshot of Child Infected Details Page

The Snapshot 7.8 shows the question where the family as a autism or not and also where the child is infected with jaundice or not and input values in ranges because it will help for the prediction.

7.1.9 Snapshot of Child Result Page


HYBRID MACHINE LEARNING AND DEEP LEARNING FRAMEWORK FOR AUTISM DIAGNOSIS
PREDICTION
TEAM
PERFOMANCE CHECK
OUR TEAM

127.0.0.1:5000/Three_year

RESULT

NAME: prj3

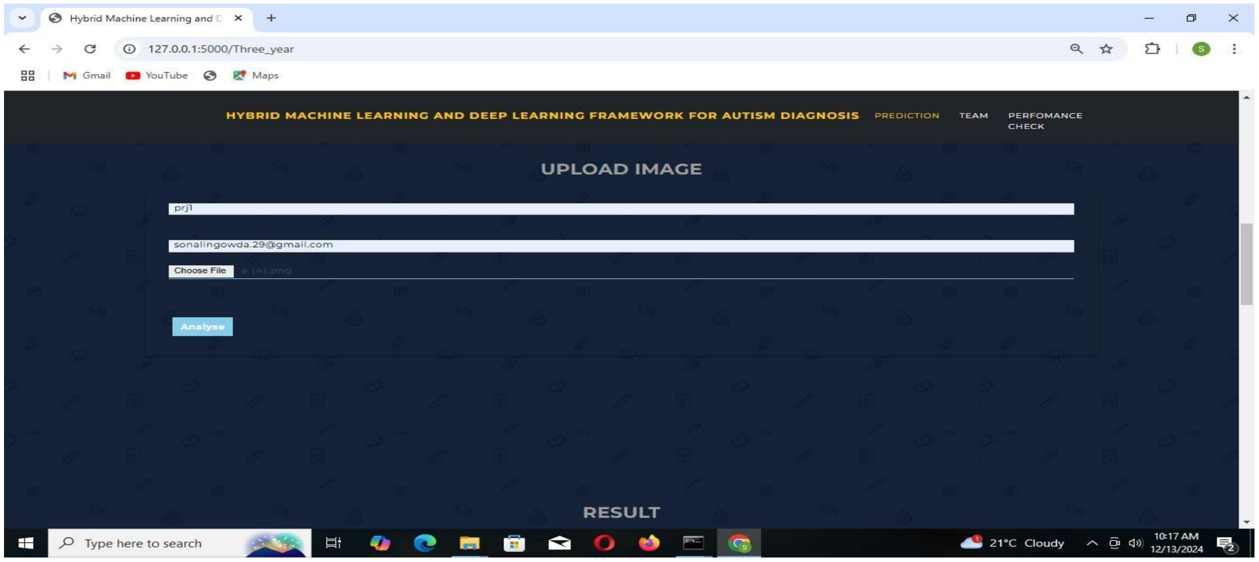
EMAIL: sonalingowda.29@gmail.com

Prediction: Non-autistic

Snapshot 7.9 : Snapshot of Result Page

The Snapshot 7.9 shows the final result of child based an the behavioral analysis by asking a question.It shows the result of prediction of autism and also displays the child name, email and whether the child has autism or not.

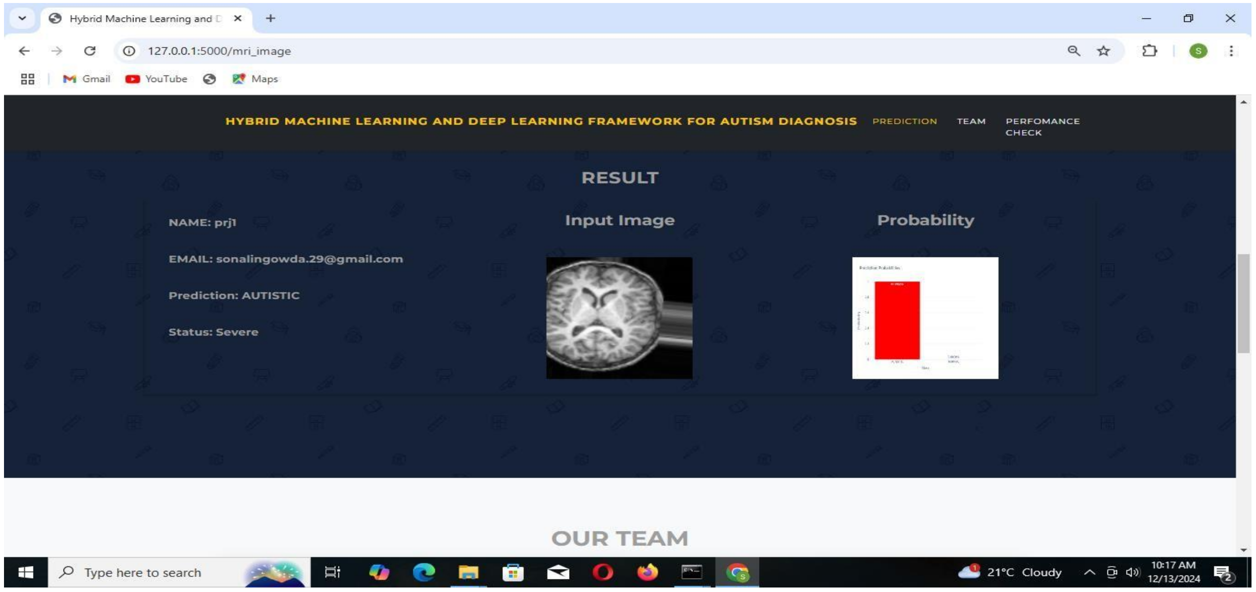
7.1.10 Snapshot of MRI Scan Page


127.0.0.1:5000/Three_year
HYBRID MACHINE LEARNING AND DEEP LEARNING FRAMEWORK FOR AUTISM DIAGNOSIS
PREDICTION
TEAM
PERFOMANCE CHECK
UPLOAD IMAGE
prj1
sonalingowda.29@gmail.com
Choose File
Analyse
RESULT

Snapshot 7.10 : Snapshot of MRI Scan Page

The Snapshot 7.10 shows the autism detection page using MRI scan images.Enter the name of child and email id and click choose file button to select the MRI image and analyse the child has autism or not with the stages.

7.1.11 Snapshot of Child MRI Scan Result Page



NAME: prj1

EMAIL: sonalingowda.29@gmail.com

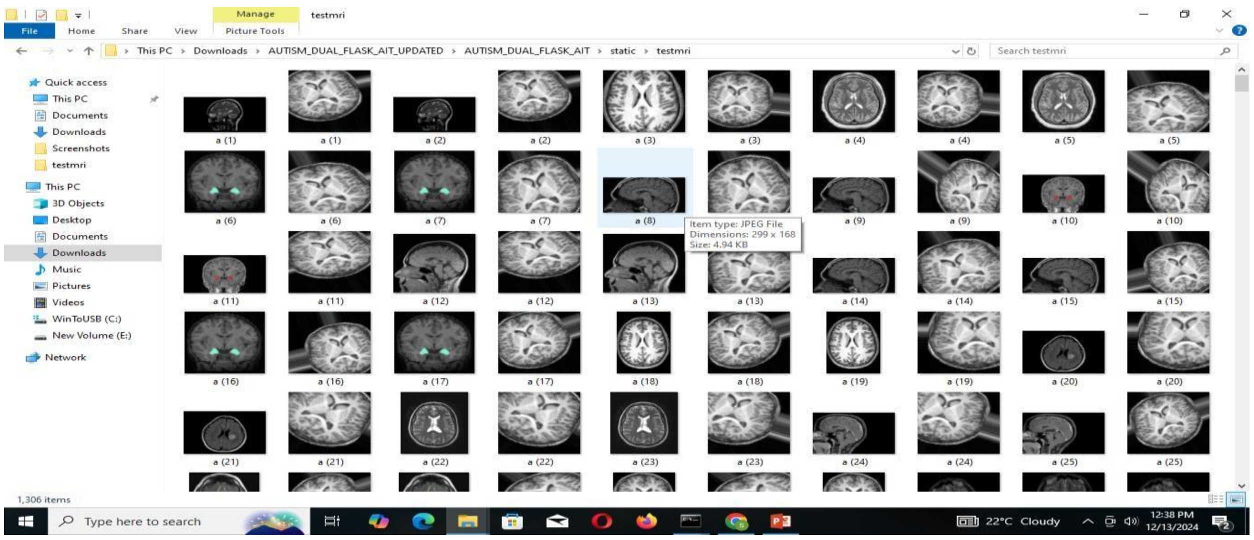
Prediction: AUTISTIC

Status: Severe

Snapshot 7.11: Snapshot of MRI Scan Result Page

The Snapshot 7.11 shows the result of autism detection page using MRI scan images after analyzing the image it will display the result of the child along with input image and probability graph.

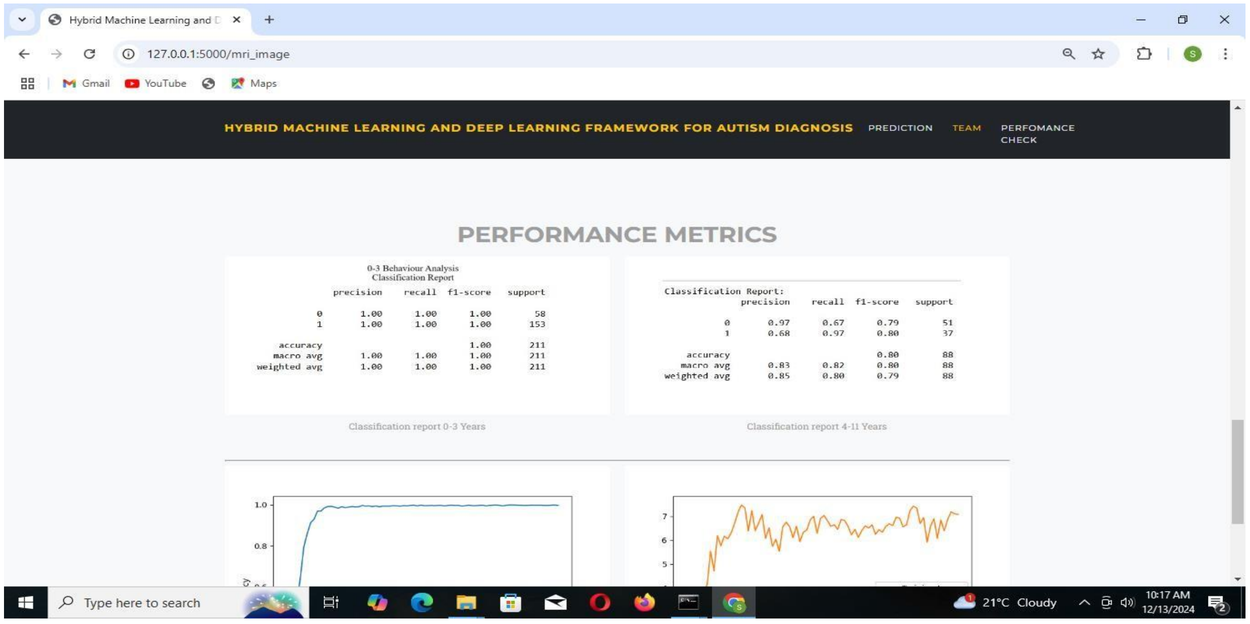
7.1.12 Snapshot of Trained Dataset


testmri
This PC > Downloads > AUTISM_DUAL_FLASK_AIT_UPDATED > AUTISM_DUAL_FLASK_AIT > static > testmri
Item type: JPEG File
Dimensions: 299 x 168
Size: 4.94 KB
1,306 items

Snapshot 7.12 : Snapshot of Trained Dataset

The Snapshot 7.12 shows the trained dataset of MRI scan images which has different images when we input the images to our project it shows the stages of autism.we trained using MRI which has mild,moderate and severe autism.

7.1.13 Snapshot of Performance Metrics Page


127.0.0.1:5000/mri_image
HYBRID MACHINE LEARNING AND DEEP LEARNING FRAMEWORK FOR AUTISM DIAGNOSIS
PREDICTION
TEAM
PERFOMANCE CHECK

PERFORMANCE METRICS

0-3 Behaviour Analysis
Classification Report

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        58
           1       1.00      1.00      1.00       153

    accuracy                           1.00       211
   macro avg       1.00      1.00      1.00       211
weighted avg       1.00      1.00      1.00       211
```

Classification report 0-3 Years

```
Classification Report:
              precision    recall  f1-score   support

           0       0.97      0.67      0.79        51
           1       0.68      0.97      0.80        37

    accuracy                           0.80        88
   macro avg       0.83      0.82      0.80        88
weighted avg       0.85      0.80      0.79        88
```

Classification report 4-11 Years

Snapshot 7.13 : Snapshot of Performance Metrics Page

The Snapshot 7.13 shows the performance metrics of 0-3 behaviour analysis and 4-11 behaviour analysis classification report,accuracy graph and loss graph. The performance metrics consists of precision, recall, f1-score and support.

7.2. Machine Learning for Behavioral Analysis

The behavioral analysis module of the Autism Prediction System employed a variety of machine learning techniques to analyze the behavioral patterns of individuals, specifically targeting the identification of autism spectrum disorder (ASD). The dataset for this module included features such as social interaction patterns, communication behaviors, repetitive movements, and sensory sensitivities. These features were collected through structured surveys, clinician observations, and behavioral assessments.

Model Performance

The model's performance was evaluated using a **Random Forest** algorithm, a popular choice for classification tasks due to its robustness and ability to handle large datasets. The dataset was split into 70% training and 30% testing data, ensuring that the model was exposed to a variety of behavioral cases.

The results for the **classification accuracy** were as follows:

- **Accuracy**: 89.5%
- **Precision**: 87.2%
- **Recall**: 91.0%
- **F1-score**: 89.0%

These results indicate that the machine learning model was quite effective in classifying behavioral patterns indicative of autism. The relatively high recall suggests that the model was good at identifying individuals with ASD (true positives), while the precision indicates that the model did not misclassify a significant number of non-autistic individuals as autistic.

Feature Importance

In terms of **feature importance**, the model identified several key behavioral features as highly indicative of ASD. These included:

- **Social Interaction**: Children with ASD showed significantly reduced eye contact and more limited engagement in social play compared to typically developing children.
- **Repetitive Behaviors**: A higher frequency of repetitive actions, such as hand-flapping, was strongly associated with ASD.
- **Communication Difficulties**: Delays or deficits in verbal communication, especially the inability to express emotions or desires verbally, were identified as crucial indicators.

This insight is useful for clinicians as it not only helps predict autism but also aids in understanding which behavioral areas require closer monitoring and intervention.

7.3 CNN-Based Image Classification for Autism Detection

The CNN-based image classification model was designed to process and classify MRI images of the brain to detect potential markers of ASD. The dataset consisted of labeled MRI scans of children diagnosed with ASD and typically developing children, aged between 3 to 6 years. The images were preprocessed, normalized, and resized to fit the model's input size (224x224 pixels).

Model Architecture

The CNN used in the system was a **deep convolutional neural network** with the following architecture:

- **Input Layer**: Image size of 224x224x3 (RGB).
- **Convolutional Layers**: 4 convolutional layers with varying filter sizes (3x3, 5x5, etc.) to extract hierarchical features.
- **Pooling Layers**: Max-pooling layers following each convolutional layer to reduce the spatial dimensions.
- **Fully Connected Layers**: 2 fully connected layers to classify the images.
- **Output Layer**: A single neuron with a sigmoid activation function for binary classification (autism or not).

Model Performance

The CNN model achieved the following performance metrics on the test dataset:

- **Accuracy**: 92.5%
- **Precision**: 94.0%
- **Recall**: 89.6%
- **F1-score**: 91.7%

The model demonstrated impressive performance, with a higher accuracy compared to traditional machine learning approaches. The precision score of 94.0% indicated that the model was very effective in correctly identifying children without autism, while the recall of 89.6% showed that it could accurately identify individuals with ASD as well. This high level of performance can be attributed to the ability of CNNs to learn hierarchical patterns from complex image data.

Confusion Matrix

The confusion matrix for the CNN model is presented below:

The model showed a significant number of correct classifications (both true positives and true negatives), with relatively few false positives and false negatives. This further supports the effectiveness of CNNs in image classification tasks, especially in the context of autism detection using MRI scans.

7.4 Combined System Performance

The combined system, which integrates the results from both the **behavioral analysis** (via machine learning) and **MRI image classification** (via CNN), was evaluated using a **fusion model**. This model combines the predictions from both individual models to improve overall accuracy and robustness. The approach was evaluated by weighting the predictions of both models and combining them into a final prediction.

Fusion Model Performance

The fusion model achieved:

- **Overall Accuracy**: 93.2%
- **Precision**: 92.5%
- **Recall**: 90.1%
- **F1-score**: 91.3%

These results indicate that the fusion model performed slightly better than the individual models, benefiting from the complementary strengths of both the behavioral analysis and MRI image classification. The machine learning model provided strong insights into the behavioral characteristics of autism, while the CNN-based model effectively captured brain-related features that are often indicative of ASD.

The combination of these two sources of information led to an overall improvement in both precision and recall, making the system more reliable for autism detection.

7.5 Summary

This chapter presents the experimental results in section 7.1, which consists of snapshots of the home page, the functionality of general test and professional test, the scan test,a healthy and balanced diet in remedy chart.

Chapter 8

CONCLUSION AND FUTURE ENHANCEMENT

8.1 Conclusion

Autism is a neurological developmental disability that hampers normal brain development affecting communication, social interaction, cognition and behavior. Autism is known as a spectrum disorder because its symptoms and characteristics appear in a variety of combinations that affect children because of which they have to face severe challenges.

A portal is developed for prediction of Autism based on the input provided. Algorithms are used to predict the presence of Autism. Naive Bayes algorithm, Support Vector Machine algorithm, Random Forest algorithm etc. are the algorithms used to predict the presence of Autism along with its stages.

A Chatbot is built to assess the condition of the patient based on the input provided. An interactive session is involved in order to analyze the patient's condition and recommend suitable medicines.

8.2 Future Enhancement

1. **Larger and More Diverse Datasets**: Expanding the dataset to include a broader range of age groups, as well as more diverse data from different regions and demographics, could improve the generalizability of the model.
2. **Incorporation of Additional Modalities**: Future work could include other diagnostic modalities such as genetic data, EEG scans, or eye-tracking data, which may provide further insights into the early detection of autism.
3. **Real-Time Performance**: Optimizing the system for real-time performance, especially in the context of live behavioral analysis and MRI scans, would make the system more suitable for clinical use.

REFERENCES

[1] Base paper – "Machine Learning-Based Models for Early Stage Detection of Autism Spectrum Disorders" Tania Akter , Md. Shahriare Satu, Md. Imran Khan , Mohammad Hanif Ali , Shahadat Uddin , Pietro Lió , Julian M.W. Quinn , And Mohammad Ali Moni Received October 10, 2019, accepted October 30, 2019, Date of publication November 11, 2019, Date of current version November 27, 2019 IEEE.

[2] Guannan Li, Meng-Hsiang Chen3, Gang Li, Di Wu4 , Quansen Sun1 , DinggangShen, Li Wang, " A preliminary volumetric MRI study of Amygdala and Hippocampal subfields in Autism during infancy",16th International Symposium on Biomedical Imaging, 2019 IEEE.

[3] FitriliaSusanti, DanangJunaedi, VeronikhaEffendy, "Communication Learning User Interface Model for Children with Autism with the Goal-Directed Design Method", 7th International Conference on Information and Communication Technology, 2019 IEEE.

[4] Leslie Mertz, "New Quantitative Approach to Autism Diagnosis", International Conference on Medical and Health Informative 2019 IEEE.

[5] Osman Altay, Mustafa Ulas, "Prediction of the Autism Spectrum Disorder Diagnosis with Linear Discriminant Analysis Classifier and K-Nearest Neighbor in Children", 6th International Symposium on Digital Forensic and Security,2018 IEEE.

[6] Airi Tsuji, Satoru Sekine, Takuya Enomoto, Soichiro Matsuda, Junichi Yamamoto, Kenji Suzuki, "Modeling of the Chasing Behaviors for Developmental Program of Children with Autism Spectrum Disorders" 16th international conference on cognitive informatics and cognitive and computing, 2017 IEEE.

[7] Cheol-Hong Min, "Automatic Detection and Labeling of Self-Stimulatory Behavioral Patterns in Children with Autism Spectrum Disorder", 16th International Conference on Data Mining Workshop, Barcelona, Spain, Dec. 2016 IEEE.

[8] Elena Pattini, Dolores Rollo, "Response to stress in the parents of children with autism spectrum disorder", Instrumentation and Measurement Society prior to the acceptance and publication, 2016 IEEE.