A PROJECT REPORT

ON

RECOGNITION OF HUMAN ACTIVITY USING ENSEMBLE LEARNING OF MULTIPLE CONVOLUTIONAL NEURAL NETWORK

Submitted in the partial fulfilment of the requirements for the award of

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY

SUBMITTED BY

Gyanaji Harshitha	Bharadwaj Yagnik Rao	Gandla Uday Kiran
20BK1A1244	20BK1A1214	21BK15A1204

Under the guidance of
Dr R. Kalyani, M.Tech, Ph.D

DEPARTMENT OF INFORMATION TECHNOLOGY

St. Peter's Engineering College (UGC Autonomous)

Approved by AICTE, New Delhi, and NAAC Accrediated with 'A' Grade,

Affiliated to JNTU, Hyderabad, Telangana

2020-2024

St. PETER'S ENGINEERING COLLEGE
UGC - AUTONOMOUS
Affiliated to JNTUH, Approved by AICTE, Accredited by NAAC with "A" Grade, NBA Programe Accredited (EEE, CSE, ECE)

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

This is to certify that a Project entitled **"RECOGNITION OF HUMAN ACTIVITY USING ENSEMBLE LEARNING OF MULTIPLE CONVOLUTIONAL NEURAL NETWORK"** is carried out by **Gyanaji Harshitha (20BK1A1244), Bharadwaj Yagnik Rao (20BK1A1214) & Gandla Uday Kiran (21BK5A1204),** in partial fulfilment for the award of the degree of **Bachelor of Technology in Information Technology** is a record of bonafide work done by them under my supervision during the academic year 2023– 2024.

GUIDE
Dr R. Kalyani, M.Tech, Ph.D
Associate Professor
Department of IT
St. Peter's Engineering College,
Hyderabad

HEAD OF THE DEPARTMENT
Dr M. Dileep Kumar, M.Tech, Ph.D
Professor& HOD
Department of IT
St. Peters Engineering College,
Hyderabad

PROJECT COORDINATOR
Dr R. Kalyani , M.Tech, Ph.D
Associate Professor
Department of IT
St. Peters Engineering College,
Hyderabad

EXTERNAL EXAMINER

DEPARTMENT OF INFORMATION TECHNOLOGY

ACKNOWLEDGEMENT

We sincerely express our deep sense of gratitude to **Dr. R Kalyani,** for his valuable guidance, encouragement and cooperation during all phases of the project.

We are greatly indebted to our Project Coordinator **Dr. R Kalyani,** for providing valuable advice, constructive suggestions and encouragement without whom it would not been possible to complete this project.

It is a great opportunity to render our sincere thanks to **Dr. M. Dileep Kumar,** Head of the Department, Information Technology for his timely guidance and highly interactive attitude which helped us a lot in successful execution of the Project.

We are extremely thankful to our Principal **Dr. K. SREE LATHA,** who stood as an inspiration behind this project and heartfelt for her endorsement and valuable suggestions.

We respect and thank our Academic Director **Mrs. T. SAROJA REDDY** and secretary **Sri.T.V.REDDY,** for providing us an opportunity to do the project work at **St. PETER'S ENGINEERING COLLEGE** and we are extremely thankful to them for providing such a nice support and guidance which made us to complete the project.

We also acknowledge with a deep sense of reverence, our gratitude towards our parents, who have always supported us morally as well as economically. We also express gratitude to all our friends who have directly or indirectly helped us to complete this project work. We hope that we can build upon the experience and knowledge that we have gained and make a valuable contribution towards the growth of the society in coming future.

Gyanaji Harshitha (20BK1A1244)

Bharadwaj Yagnik Rao (20BK1A1214)

Gandla Uday Kiran (21BK5A1204)

DEPARTMENT OF INFORMATION TECHNOLOGY

INSTITUTE VISION

To be a renowned Educational Institution that moulds Students into Skilled Professionals fostering Technological Development, Research and Entrepreneurship meeting the societal needs.

INSTITUTE MISSION

IM1: Making students knowledgeable in the field of core and applied areas of Engineering to innovate Technological solutions to the problems in the Society.

IM2: Training the Students to impart the skills in cutting edge technologies, with the help of relevant stake holders.

IM3: Fostering conducive ambience that inculcates research attitude, identifying promising fields for entrepreneurship with ethical, moral and social responsibilities.

DEPARTMENT OF INFORMATION TECHNOLOGY

DEPARTMENT VISION

To be a vibrant nodal center for Computer Science Engineering Education, Research that make the students to contribute to technologies for IT, IT-Enabled Services; to involve in innovative research on thrust areas of industry and academia; to establish start-ups supporting major players in the industry.

DEPARTMENT MISSION

DM1: Emphasize project-based learning by employing the state-of art technologies, algorithms in software development for the problems in inter-disciplinary avenues.

DM2: Involve stakeholders to make the students industry ready with training in skill-oriented computer application software.

DM3: Facilitate to learn the theoretical nuances of Computer Science, Computer Engineering courses and motivate to carry out research in both core and applied areas of CSE.

DEPARTMENT OF INFORMATION TECHNOLOGY

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

PEO1: Graduates will be able to excel as IT Professional with Proficiency in understanding, applying, analyzing and designing for Information Technology relevant problems.

PEO2: Graduates will be able to pursue higher studies with good knowledge in core areas of Information Technology and promote collaborative research.

PEO3: Graduates will be able to exhibit professionalism, teamwork, leadership skills and exposure to current needs.

PEO4: Graduates will be able to excel as entrepreneurs with the potential knowledge to design software-based solutions for societal needs**.**

DEPARTMENT OF INFORMATION TECHNOLOGY

PROGRAM OUTCOMES (POs)

Engineering Graduates will be able to:

1: ENGINEERING KNOWLEDGE: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

2: PROBLEM ANALYSIS: Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using the first principles of mathematics, natural sciences, and engineering sciences.

3: DESIGN/DEVELOPMENT OF SOLUTIONS: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for public health and safety, and the cultural, societal, and environmental considerations.

4: CONDUCT INVESTIGATIONS OF COMPLEX PROBLEMS: Use research-based knowledge and research methods including design of experiments, analysis, interpretation of data, and synthesis of the information to provide valid conclusions.

5: MODERN TOOL USAGE: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

6: THE ENGINEER AND SOCIETY: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues, and the consequent responsibilities relevant to the professional engineering practice

7: ENVIRONMENT AND SUSTAINABILITY: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

8: ETHICS: Apply ethical principles and commit to professional ethics and, responsibilities and norms of the engineering practice.

9: INDIVIDUAL AND TEAM WORK: Function effectively as an individual, and as a member or leader in diverse teams, and multidisciplinary settings.

10: COMMUNICATION: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as being able to comprehend and draft effective reports and design documentation, make an effective presentation, give, and receive clear instructions.

11: PROJECT MANAGEMENT AND FINANCE: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's work, as a member and leader in a team, to manage projects and in a multidisciplinary environment.

12: LIFE-LONG LEARNING: Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broadcast context of technological changes.

DEPARTMENT OF INFORMATION TECHNOLOGY

PROGRAM SPECIFIC OBJECTIVES (PSO'S)

PSO1: Competent in Emerging Trends: Apply software design and development practices to create software applications in emerging areas such as Cloud computing, Data Analytics, AI and Cyber Security.

PSO2: Successful Career and Entrepreneurship: The ability to employ modern computer languages in all the environments to create an innovative carrier to be an entrepreneur and a zest for higher studies.

DEPARTMENT OF INFORMATION TECHNOLOGY

DECLARATION

We declare that a Project entitled "**Recognition of Human Activity Using Ensemble Learning of Multiple Convolutional Neural Network**" is an Original Work submitted by the following group members who have actively contributed and submitted in partial fulfilment for the award of degree in **"Bachelor of Technology in Information Technology",** at **St. Peter's Engineering College**, Hyderabad, and this project work has not been submitted by me to any other college or university for the award of any kind of degree.

Date Submitted:

Name	Roll Number	Signature
Gyanaji Harshitha	**20BK1A1244**	
Bharadwaj Yagnik Rao	**20BK1A1214**	
Gandla Uday Kiran	**21BK5A1204**	

ABSTRACT

Human Activity Recognition is a field concerned with the recognition of physical human activities based on the interpretation of sensor data, including one-dimensional time series data. Traditionally, hand-crafted features are relied upon to develop the machine learning models for activity recognition. However, that is a challenging task and requires a high degree of domain expertise and feature engineering. With the development in deep neural networks, it is much easier as models can automatically learn features from raw sensor data, yielding improved classification results. In this paper, we present a novel approach for human activity recognition using ensemble learning of multiple Convolutionalal neural network (CNN) models. Three different CNN models are trained on the publicly available data set and multiple ensembles of the models are created. The ensemble of the first two models gives an accuracy of 94% which is better than the methods available in the literature.

LIST OF FIGURES

LIST OF ACRONYMS AND DEFINITIONS

S.NO	ACRONYM	DEFINITION
01.	CNN	Convolutionalal Neural Network
02.	UML	Unified Modelling Language
03.	GP	Genetic Programming
04.	MCRP	Markov chain extended with rainfall prediction
05.	ANN	Artificial Neural Networks
06.	DFD	Data Flow Diagram
07.	RNN	Recurrent Neural Network
08.	WISDM	Wireless Sensor Data Mining

CONTENTS

CHAPTER 1
INTRODUCTION

Human Activty Recognition (HAR) has developed its application in multiple fields and for a variety of tasks, including smart healthcare systems, automated surveillance and security. For the purpose of HAR, on-body sensors, such as tri-axial accelerometers, gyroscopes, proximity sensors, magnetometers, and temperature sensors, etc., are preferred for three main reasons; firstly, they omit the privacy issues posed by video sensors. Secondly, an entire body sensor network (BSN) allows for more accurate deployment of a signal acquisition system, and finally, they reduce the restrictions in place due to the environment and the stationary placements of cameras, as in video based datasets. HAR has identified its crucial place in ubiquitous computing because of the eased process of collecting data with embedded sensors, especially with the widespread use of smart phones in the last decade. Therefore, smart phones can essentially serve as a network of on-body sensors for data collection. Previously, the process of feature extraction from these collected data points remained heuristic, hand-crafted, and task dependent. The feature extraction process has been dependent on the usage of features that are purposefully designed and selected depending on the application. Statistical measures, such as the mean, and the variance, and transform coding measures, such as Fourier transforms, were extracted from the raw signals and subsequently used for classification methods.

These methods posses the limitation of being bound to the specific classification tasks they are designed for. Manual selection of features also poses the problem of losing out on information from the raw signal. With the advancements in deep learning, this process has been accelerated with unprecedented improvements, especially with the availability of more data and higher computation power. The feature selection process no longer needs to be task-dependent as deep learning models can extract features automatically and they can be applied to a range of classification tasks. With the provision of deep learning, it makes a logical flow to maximize the use of this state-of-the-art research for concentrated implementation of HAR. Therefore, the aim of this work is to develop a systematic process for a more streamlined feature representation for improved activity recognition through ensemble learning of CNN's.

CHAPTER 2

LITERATURE SURVEY

1) "Feature learning for ¨ activity recognition in ubiquitous computing Authors: T. Plotz, N. Y. Hammerla, and P. L. Olivier

Feature extraction for activity recognition in context-aware ubiquitous computing applications is usually a heuristic process, informed by underlying domain knowledge. Relying on such explicit knowledge is problematic when aiming to generalize across different application domains. We investigate the potential of recent machine learning methods for discovering universal features for context-aware applications of activity recognition. We also describe an alternative data representation based on the empirical cumulative distribution function of the raw data, which effectively abstracts from absolute values. Experiments on accelerometer data from four publicly available activity recognition datasets demonstrate the significant potential of our approach to address both contemporary activity recognition tasks and next generation problems such as skill assessment and the detection of novel activities.

2. "LSTM networks for mobile human activity recognition," Authors: Y. Chen, K. Zhong, J. Zhang, Q. Sun, and X. Zhao,

A lot of real-life mobile sensing applications are becoming available. These applications use mobile sensors embedded in smart phones to recognize human activities in order to get a better understanding of human behavior. In this paper, we propose a LSTM-based feature extraction approach to recognize human activities using tri-axial accelerometers data. The experimental results on the (WISDM) Lab public datasets indicate that our LSTM-based approach is practical and achieves 92.1% accuracy.

3 Deep Convolutional neural networks on multichannel time series for human activity recognition

AUTHORS: J. Yang, M. N. Nguyen, P. P. San, X. L. Li, and S. Krishnaswamy

This paper focuses on human activity recognition (HAR) problem, in which inputs are multichannel time series signals acquired from a set of body-worn inertial sensors and outputs are predefined human activities. In this problem, extracting effective features for identifying activities is a critical but challenging task. Most existing work relies on heuristic hand-crafted feature design and shallow feature learning architectures, which cannot find those distinguishing features to accurately classify different activities. In this

paper, we propose a systematic feature learning method for HAR problem. This method adopts a deep Convolutionalal neural networks (CNN) to automate feature learning from the raw inputs in a systematic way. Through the deep architecture, the learned features are deemed as the higher level abstract representation of low level raw time series signals. By leveraging the labelled information via supervised learning, the learned features are endowed with more discrimi-native power. Unified in one model, feature learning and classification are mutually enhanced. All these unique advantages of the CNN make it out-perform other HAR algorithms, as verified in the experiments on the Opportunity Activity Recognition Challenge and other benchmark datasets.

4. "CNN based approach for activity recognition using a wrist-worn accelerometer," AUTHORS: M. Panwar, S. R. Dyuthi, K. C. Prakash, D. Biswas, A. Acharyya, K. Maharatna, A. Gautam, and G. R. Naik

In recent years, significant advancements have taken place in human activity recognition using various machine learning approaches. However, feature engineering have dominated conventional methods involving the difficult process of optimal feature selection. This problem has been mitigated by using a novel methodology based on deep learning framework which automatically extracts the useful features and reduces the computational cost. As a proof of concept, we have attempted to design a generalized model for recognition of three fundamental movements of the human forearm performed in daily life where data is collected from four different subjects using a single wrist worn accelerometer sensor. The validation of the proposed model is done with different pre- processing and noisy data condition which is evaluated using three possible methods. The results show that our proposed methodology achieves an average recognition rate of 99.8% as opposed to conventional methods based on K-means clustering, linear discriminant analysis and support vector machine.

5. Human activity recognition using LSTM-RNN deep neural network architecture AUTHORS: S. W. Pienaar and R. Malekian

Using raw sensor data to model and train networks for Human Activity Recognition can be used in many different applications, from fitness tracking to safety monitoring applications. These models can be easily extended to be trained with different data sources for increased accuracies or an extension of classifications for different prediction classes. This paper goes into the discussion on the available dataset provided by WISDM and the unique features of each class for the different axes. Furthermore, the design of a Long Short Term Memory (LSTM) architecture model is outlined for the application of HAR .

CHAPTER 3

SYSTEM ANALYSIS AND DESIGN

3.1 EXISTING SYSTEM:

In the existing work with wearable based or non-wearable based. Wearable based HAR system make use of wearable sensors that are attached on the human body. Wearable based HAR system are intrusive in nature. Non-wearable based HAR system do not require any sensors to attach on the human or to carry any device for activity recognition. Non-wearable based approach can be further categorized into sensor based HAR systems . Sensor based technology use RF signals from sensors, such as RFID, PIR sensors and WiFi signals to detect human activities. Sensor based HAR system are non-intrusive in nature but may not provide high accuracy.

DISADVANTAGES OF EXISTING SYSTEM:

- Require the optical sensors to be attached on the human and also demand the need of multiple camera settings.
- Wearable dives cost are high.
- **Algorithm**: Marker based motion Capture (MoCap) Framework.

3.2 PROPOSED SYSTEM:

we present an approach that is of forming three CNNs with one-dimensional (1D) Convolutionalal layers for activity prediction on 1D HAR dataset. The 1D Convolutional layer creates a Convolutional kernel that is convolved with the layer input over a single spatial or temporal dimension. It then further goes to ensemble the three individual models together by taking their averages. This, ensemble learning as a multiple classifier system , improves the model performance and produces results better than those of individual models. We then further on create other possible ensembles of two models, i.e. of first and second, second and third, and, first and third models. Ensemble learning as a whole has shown an improvement in the performance of the model.

ADVANTAGES OF PROPOSED SYSTEM:

- we developed three different CNN based models, using which, a series of ensemble learning models were also created. Ensemble learning is a paradigm by which multiple models or learners are trained to solve the same problem.
- The advantage of this paradigm lies in its ability to generalize. It has the ability to boost the learning effect of learners that are weaker, leading to an improved performance output. The following sub-sections describe the overview and characteristics of the models.

Algorithm: ensemble learning, deep learning, Convolutional neural networks

CHAPTER 4

SYSTEM REQUIREMENTS & SPECIFICATIONS

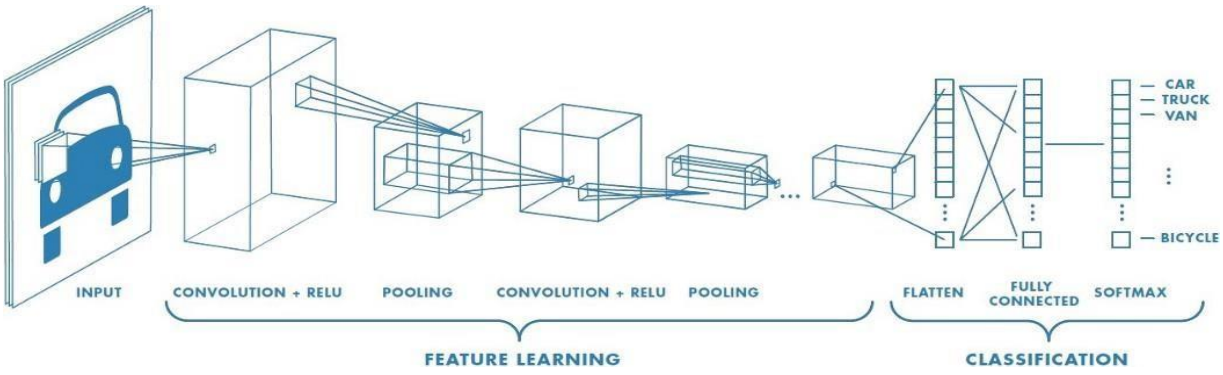
4.1 CNN ALGORITHM:

CONVOLUTIONAL NEURAL NETWORK:

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm that can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image, and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification methods.

The Convolutional Neural Network (CNN) algorithm is a type of deep learning neural network that is widely used in computer vision tasks, such as image classification, object detection, and segmentation. CNNs are designed to automatically learn and extract features from input images, allowing them to accurately classify images based on their content.

The basic architecture of a CNN consists of multiple layers of Convolutional, pooling, and fully connected layers. In the first layer, the input image is convolved with a set of learnable filters, which helps to extract useful features from the image. The resulting feature maps are then passed through a pooling layer, which down-samples the feature maps and reduces the dimensionality of the data.



After multiple Convolutional and pooling layers, the feature maps are flattened and passed through fully connected layers, which perform classification tasks based on the extracted features. The output of the final layer represents the predicted class of the input image.

During training, the weights of the filters and fully connected layers are updated using back propagation and stochastic gradient descent, allowing the network to learn to recognize patterns and features in the input images.

CNNs have demonstrated state-of-the-art performance in a wide range of computer vision tasks, including image classification, object detection, and segmentation, and they have been applied in various fields, such as autonomous driving, medical diagnosis, and natural language processing.

The first layer in a CNN is typically a Convolutional layer, which applies a set of filters (also known as kernels) to the input image. Each filter is designed to detect a specific feature, such as edges, corners, or textures. The output of the Convolutional layer is then passed through a non-linear activation function, such as the Rectified Linear Unit (ReLU) function, which introduces non-linearity into the model.

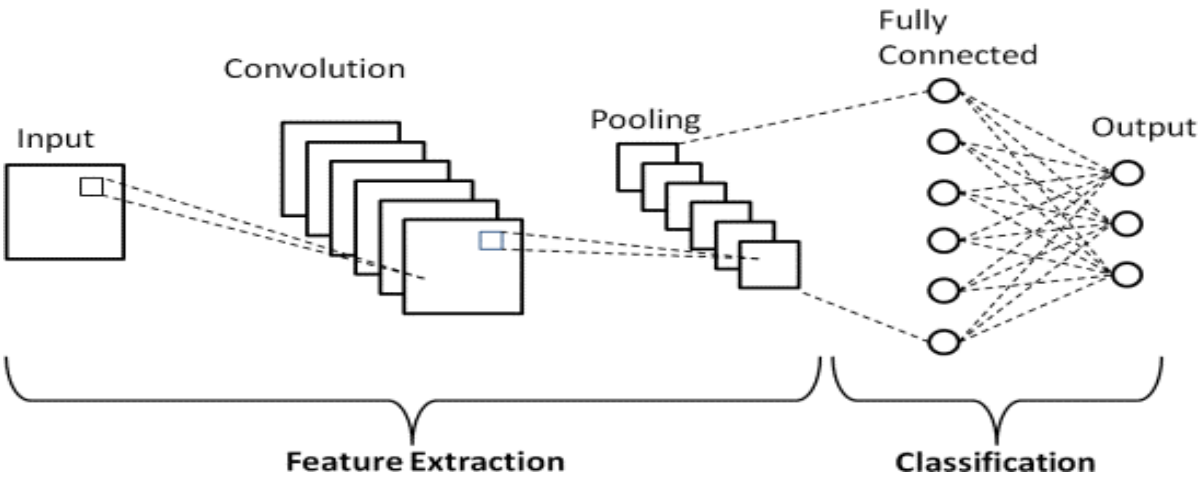
After the Convolutional layer, the output is typically passed through a pooling layer, which reduces the spatial size of the output by subsampling it. This helps to reduce the computational complexity of the model and prevent overfitting.

The process of applying Convolutional and pooling layers is repeated several times, with each layer learning more complex features from the input image. This is known as feature extraction, and the layers that perform this task are often referred to as the Convolutional feature extractor.

Once the Convolutional feature extractor has extracted relevant features from the input image, the output is typically passed through one or more fully connected layers. These layers use the extracted features to classify the input image or perform another task, such as object detection or segmentation

During the training process, the weights of the CNN are adjusted using an optimization algorithm, such as stochastic gradient descent. The goal is to minimize the difference between the predicted output of the model and the true labels of the images in the training set.

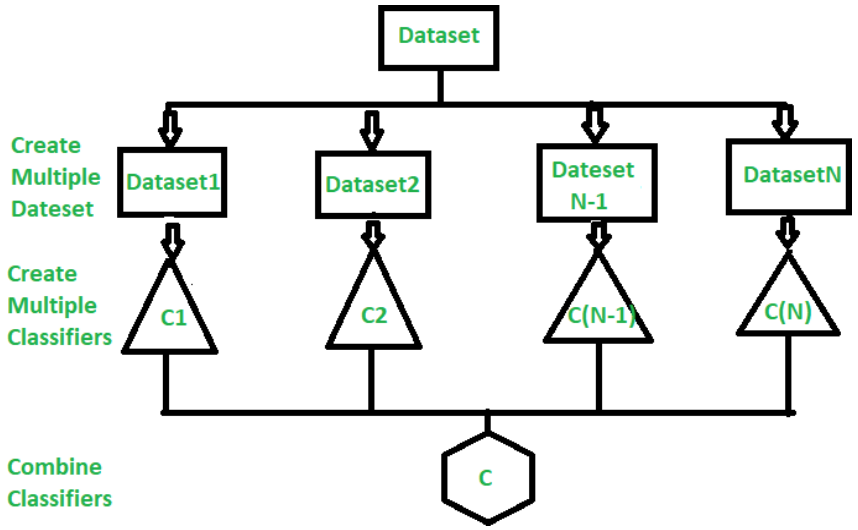
Overall, CNNs are a powerful tool for image processing tasks, and have been used in a wide variety of applications, from self-driving cars to medical image analysis. The deep architecture of CNNs allows them to learn complex features and perform highly accurate predictions, making them a popular choice for many computer vision tasks.



CLASSIFICATION BASED ON CONVOLUTIONAL NEURAL NETWORK

4.2 ENSEMBLE LEARNING :

Ensemble learning helps improve machine learning results by combining several models. This approach allows the production of better predictive performance compared to a single model. Basic idea is to learn a set of classifiers (experts) and to allow them to vote.



Ensemble learning is a machine learning technique that involves combining multiple models to improve the overall performance of a prediction task. The basic idea is that by combining the predictions of multiple models, the resulting prediction will be more accurate than any single model.

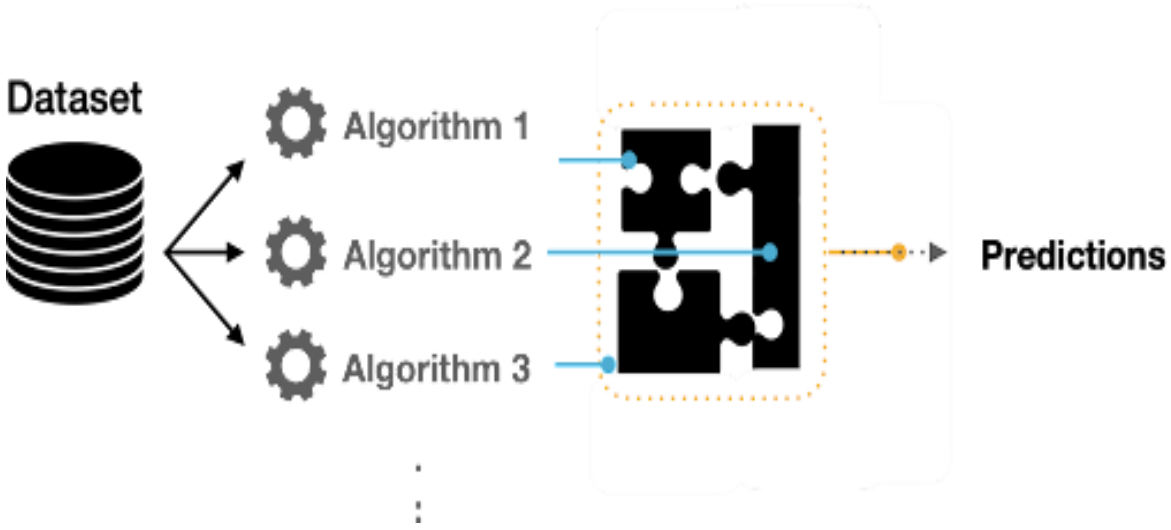
Ensemble learning can be used with a wide variety of machine learning algorithms, including decision trees, neural networks, and support vector machines. By combining the strengths of multiple models, ensemble learning can improve the accuracy, robustness, and generalization of a model, making it a powerful tool for many prediction tasks.

Ensemble learning of multiple CNNs (Convolutional Neural Networks) is a powerful technique for improving the accuracy and robustness of image recognition tasks. There are several ways to combine the outputs of multiple CNNs, including:

1. Model averaging: In model averaging, the outputs of multiple CNNs are combined by taking the average of their predictions. This is a simple and effective way to reduce the variance of the predictions and improve the accuracy of the model.
2. Majority voting: In majority voting, the outputs of multiple CNNs are combined by taking the most common prediction among the models. This is a robust way to handle noisy data and reduce the risk of overfitting.
3. Stacking: In stacking, the outputs of multiple CNNs are combined by training a meta-learner that takes the individual predictions as input and learns how to combine them to make the final prediction. Stacking can improve the accuracy of the model by learning more complex relationships between the individual predictions.

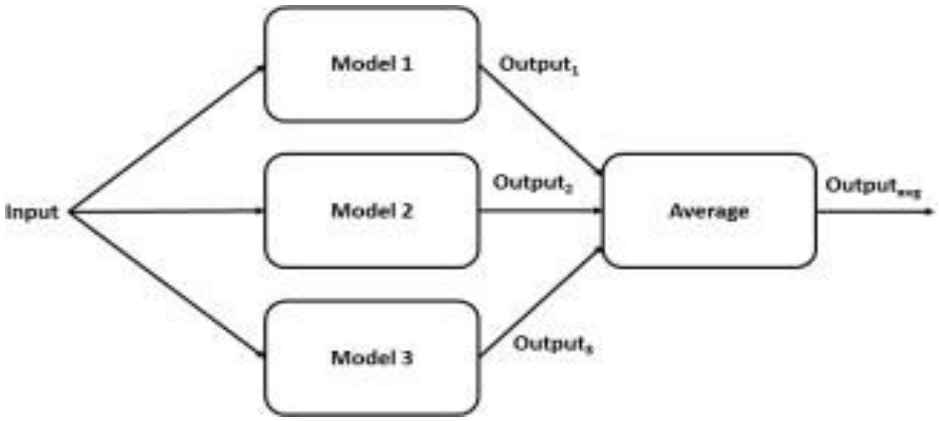
To implement ensemble learning of multiple CNNs, one approach is to train each CNN independently on the training data, using different architectures or hyperparameters. Once the individual CNNs are trained, their outputs can be combined using one of the above methods to make the final prediction.

Ensemble learning of multiple CNNs can significantly improve the accuracy and robustness of image recognition tasks, especially in cases where the individual models may be prone to errors or overfitting. However, it can also increase the computational cost and training time, so it is important to carefully consider the trade-offs between performance and resource usage when using this technique.



4.3 SYSTEM DESIGN

4.3.1 SYSTEM ARCHITECTURE:

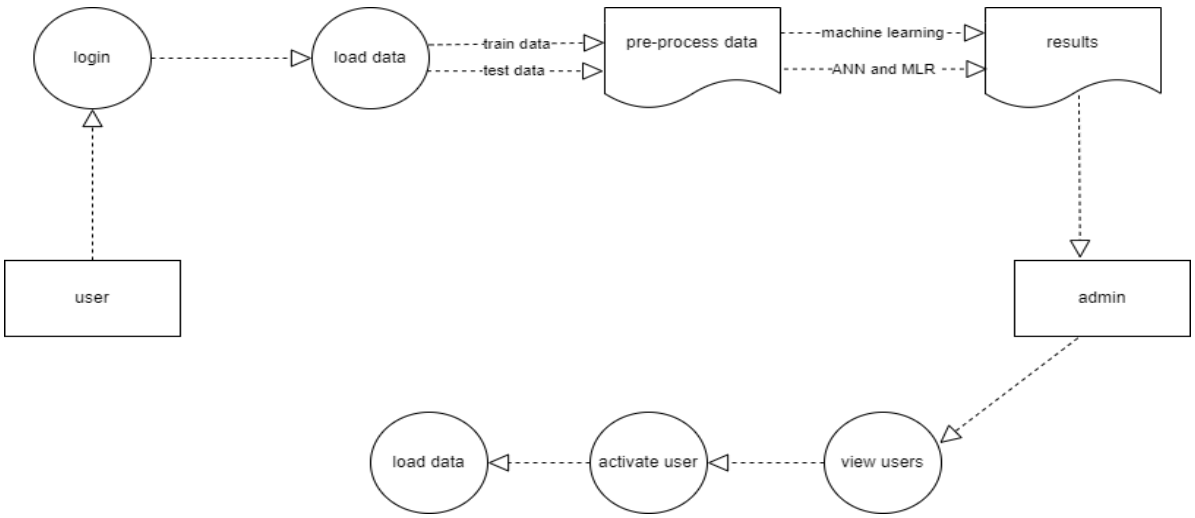


4.3.2 DATA FLOW DIAGRAM:

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.

2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.

3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.



4.3.3 UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group. The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML. The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

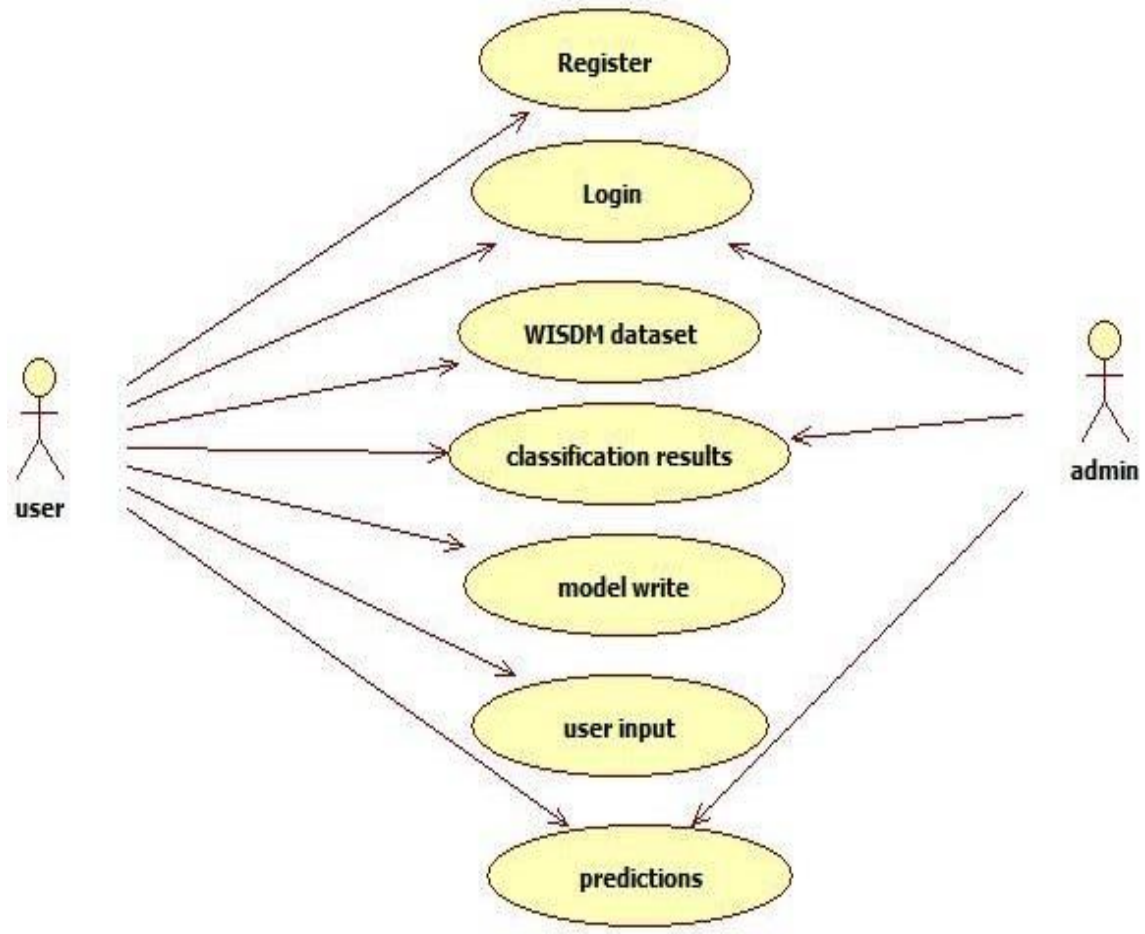
GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

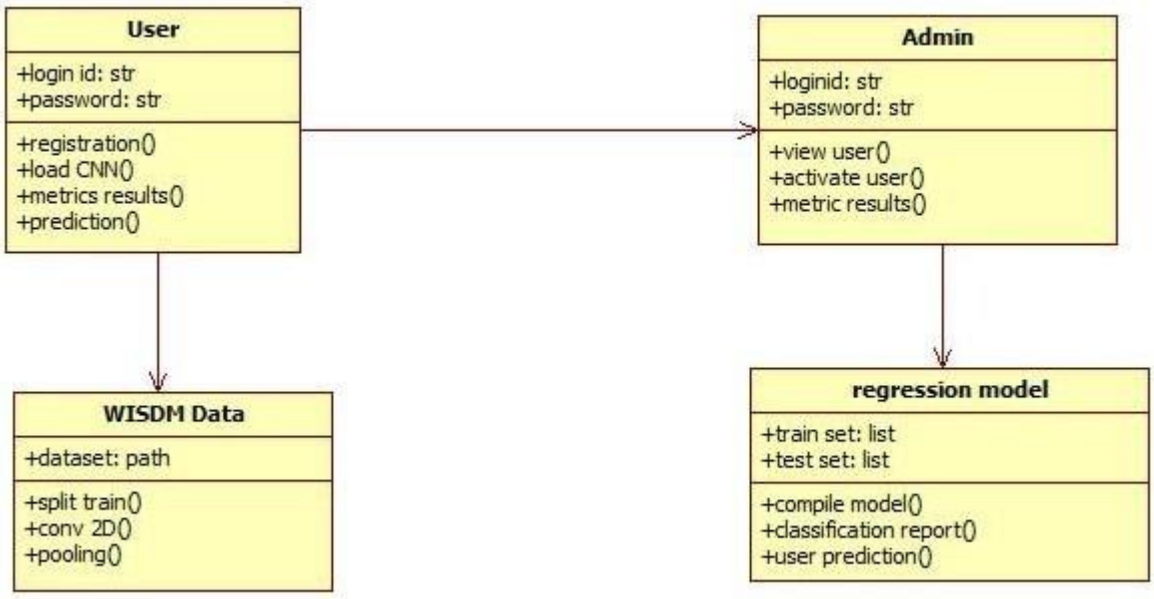
4.3.4 USE CASE DIAGRAM:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



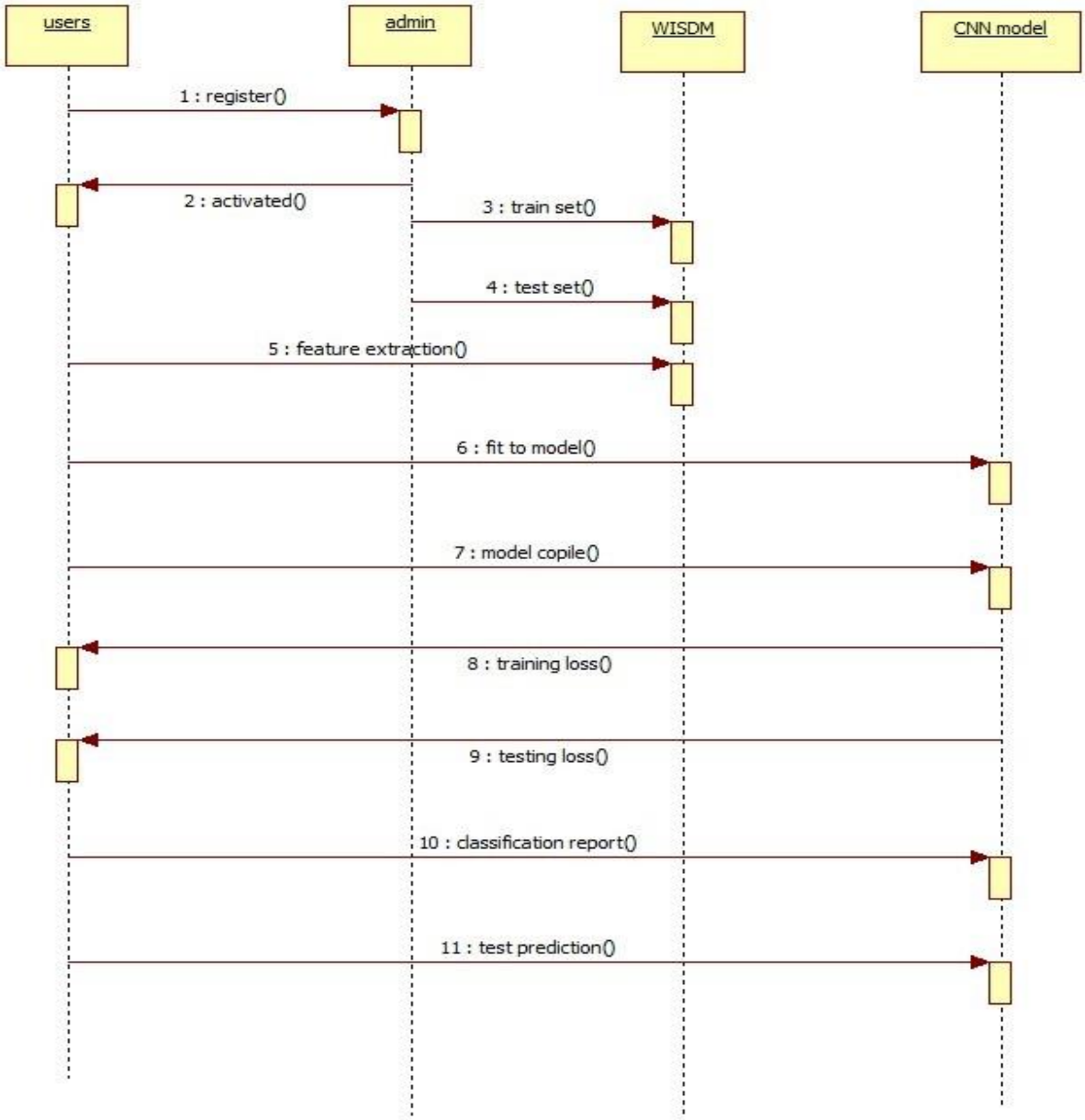
4.3.5 CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



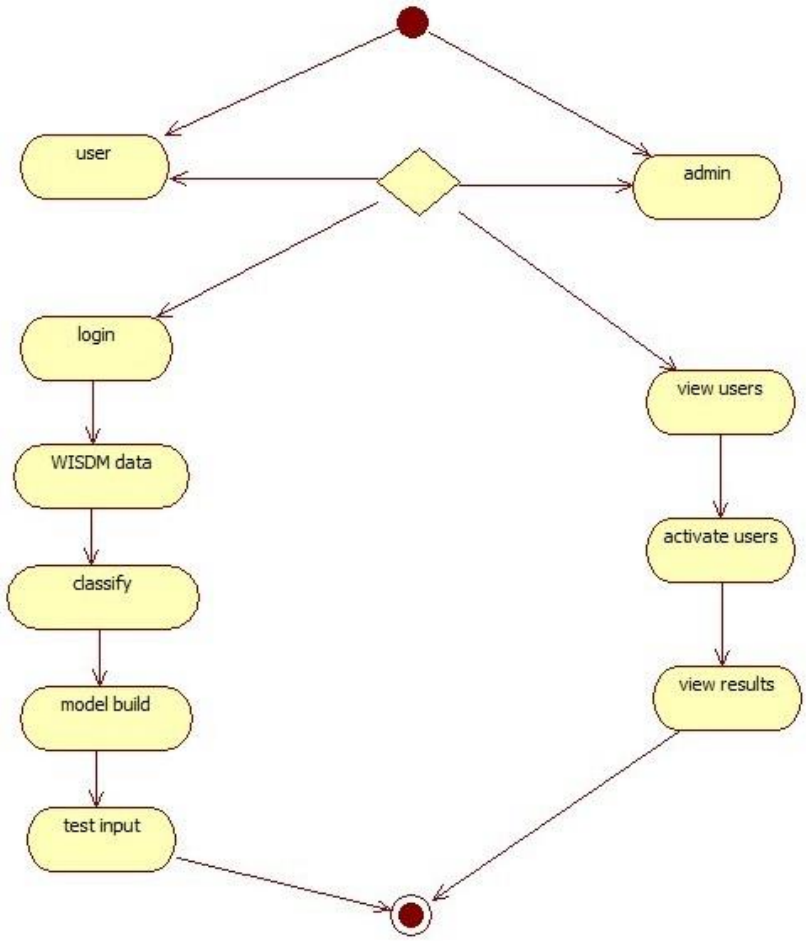
4.3.6 SEQUENCE DIAGRAM:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.



4.3.7 ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.



4.4 MODULES:

- User
- Admin
- Data Preprocessing
- Machine Learning Results

4.4.1 MODULES DESCRIPTION:

User:

The User can register the first. While registering he required a valid user email and mobile for further communications. Once the user register then admin can activate the user. Once admin activated the user then user can login into our system. User can upload the dataset based on our dataset column matched. For algorithm execution data must be in float format. Here we took Wireless Sensor Data Mining (WISDM) dataset. User can

also add the new data for existing dataset based on our Django application. User can click the Classification in the web page so that the data calculated Accuracy, Loss based on the algorithms.

Admin:

Admin can login with his login details. Admin can activate the registered users. Once he activate then only the user can login into our system. Admin can view the overall data in the browser. Admin can click the Results in the web page so calculated Accuracy, Loss based on the algorithms is displayed. All algorithms execution complete then admin can see the overall accuracy in web page.

Data Preprocessing:

The dataset used for the application of ensemble learning on Human Activity Recognition is the one provided by Wireless Sensor Data Mining (WISDM) Lab [13]. This dataset has been collected by 36 users each with a smart phone. The sensor used for collecting data is the embedded tri-axial accelerometer (x-axis, y-axis, z-axis). The samples are taken with a frequency of 20 Hz (20 samples/second). The dataset includes 6 daily activities as classes. The class distribution is as follows: • Walking: 424,400 examples, 38.6%,

• Jogging: 342,177 examples, 31.2%,

• Upstairs: 122,869 examples, 11.2%,

• Downstairs: 100,427 examples, 9.1%,

• Sitting: 59,939 examples, 5.5%,

• Standing: 48,395 examples, 4.4%

First we pre-process the data such that it is normalised, that is to have zero-mean axis. The 3 axis graph for one activity (walking) After pre-processing, we segment and shape the data such that it is compatible with 1D Convolutional layers of the deployed CNNs. For segmenting the data, we first generate indexes of a fixed size window and move by half the window size. Then we generate fixed size segments and append the windowed data such that the dataset created has dimensions such as [total segments, input width and input channel] which is [24403, 90, 3] for our case.

Deep Learning Results:

Based on the split criterion, the cleansed data is split into 60% training and 40% test, then the dataset is subjected to CNN Deep learning classifiers such as, we developed three different CNN based models, the first model is inspired from the ConvPool-CNN-C, The second model used is adapted from the ALL-CNN-C,

The third model is inspired from the 'network in a network' model. The accuracy and loss of the classifiers was calculated and displayed in my results. The classifier which bags up the highest accuracy and Loss could be determined as the best classifier.

4.5 SYSTEM REQUIRMENTS

4.5.1 HARDWARE REQUIRMENTS:

For developing the application the following are the Hardware Requirements:

- Processor: Intel i9
- RAM: 32 GB
- Space on Hard Disk: minimum 1 TB

4.5.2 SOFTWARE REQUIREMENTS:

For developing the application the following are the Software Requirements:

1. Python
2. Django

Operating Systems supported

1. Windows 10 64 bit OS

Technologies and Languages used to Develop

1. Python

Debugger and Emulator

- Any Browser (Particularly Chrome)

4.5.3 SOFTWARE ENVIRONMENT:

PYTHON:

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. An interpreted language, Python has a design philosophy that emphasizes code readability (notably using whitespace indentation to delimit code blocks rather than curly brackets or keywords), and a syntax that allows programmers to express concepts in fewer lines of code than might be used in languages such as C++or Java. It provides constructs that enable clear programming on both small and large scales. Python interpreters are available for many operating systems. CPython, the reference implementation of Python, is open source software and has a community-based development model, as do nearly all of its variant implementations. CPython is managed by the non-profit Python Software Foundation. Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

Python Identifiers

A Python identifier is a name used to identify a variable, function, class, module or other object. An identifier starts with a letter A to Z or a to z or an underscore (_) followed by zero or more letters, underscores and digits (0 to 9).

Python does not allow punctuation characters such as @, $, and % within identifiers. Python is a case sensitive programming language. Thus, Manpower and manpower are two different identifiers in Python.

Here are naming conventions for Python identifiers −

- Class names start with an uppercase letter. All other identifiers start with a lowercase letter.
- Starting an identifier with a single leading underscore indicates that the identifier is private.
- Starting an identifier with two leading underscores indicates a strongly private identifier.
- If the identifier also ends with two trailing underscores, the identifier is a language-defined special name.

Reserved Words

The following list shows the Python keywords. These are reserved words and you cannot use them as constant or variable or any other identifier names. All the Python keywords contain lowercase letters only.

and	exec	not
assert	finally	or
break	for	pass
class	from	print
continue	global	raise
def	if	return
del	import	try
elif	in	while
else	is	with
except	lambda	yield

Lines and Indentation

Python provides no braces to indicate blocks of code for class and function definitions or flow control.

Blocks of code are denoted by line indentation, which is rigidly enforced.

The number of spaces in the indentation is variable, but all statements within the block must be indented the same amount.

Command Line Arguments

Many programs can be run to provide you with some basic information about how they should be run.

Python enables you to do this with -h −

$ python -h

usage: python [option] ... [-c cmd | -m mod | file | -] [arg] ... Options and arguments (and corresponding

environment variables):

-c cmd : program passed in as string (terminates option list)

-d : debug output from parser (also PYTHONDEBUG=x)

-E : ignore environment variables (such as PYTHONPATH)

-h : print this help message and exit

You can also program your script in such a way that it should accept various options. Command Line Arguments is an advanced topic and should be studied a bit later once you have gone through rest of the Python concepts.

Python Lists

The list is a most versatile datatype available in Python which can be written as a list of comma-separated values (items) between square brackets. Important thing about a list is that items in a list need not be of the same type.

Creating a list is as simple as putting different comma-separated values between square brackets. For example

− list1 = ['physics', 'chemistry', 1997, 2000];

list2 = [1, 2, 3, 4, 5];

list3 = ["a", "b", "c", "d"]

Similar to string indices, list indices start at 0, and lists can be sliced, concatenated and so on.

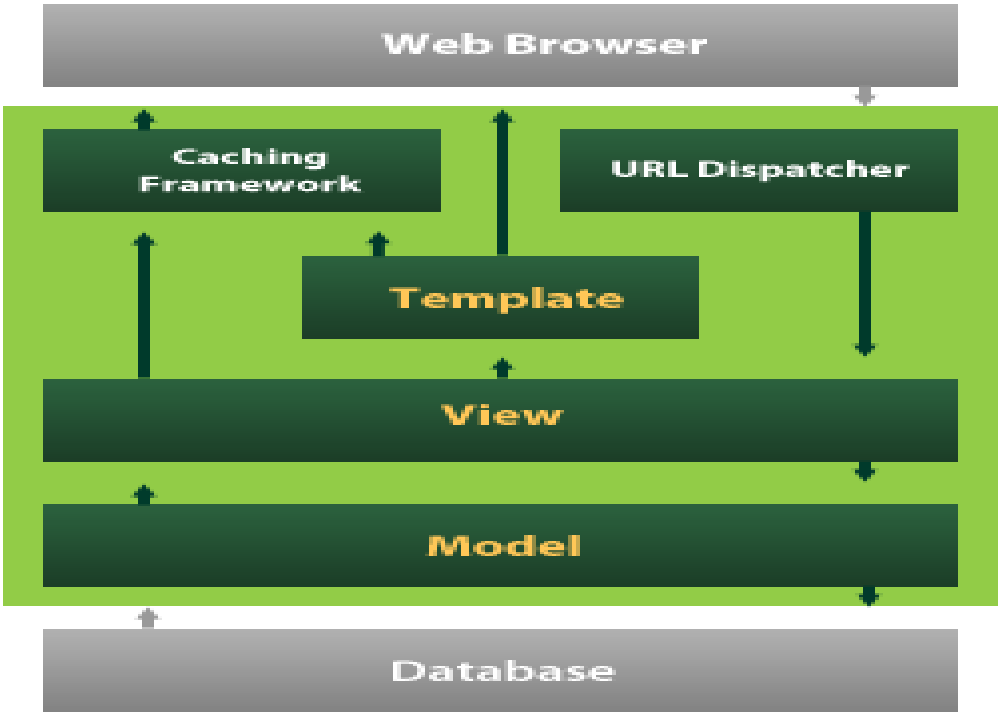
A tuple is a sequence of immutable Python objects. Tuples are sequences, just like lists. The differences between tuples and lists are, the tuples cannot be changed unlike lists and tuples use parentheses, whereas lists use square brackets.

Creating a tuple is as simple as putting different comma-separated values. Optionally you can put these comma- separated values between parentheses also.

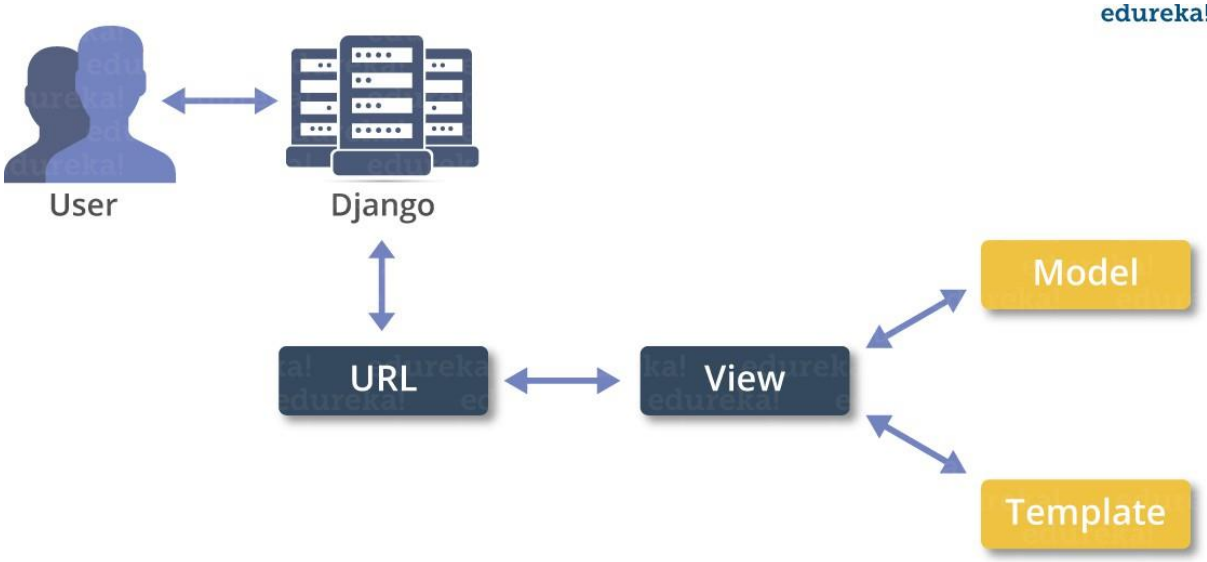
DJANGO

Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. It’s free and open source.

Django's primary goal is to ease the creation of complex, database-driven websites. Django emphasizes reusability and "pluggability" of components, rapid development, and the principle of don't repeat yourself. Python is used throughout, even for settings files and data models.



Django also provides an optional administrative create, read, update and delete interface that is generated dynamically through introspection and configured via admin models



Create a Project

Whether you are on Windows or Linux, just get a terminal or a cmd prompt and navigate to the place you want your project to be created, then use this code −

$ django-admin startproject myproject

This will create a "myproject" folder with the following structure − myproject/

manage.py myproject/

__init__.py

settings.py urls.py wsgi.py

The Project Structure

The “myproject” folder is just your project container, it actually contains two elements −

manage.py − This file is kind of your project local django-admin for interacting with your project via command line (start the development server, sync db...). To get a full list of command accessible via manage.py you can use the code −

$ python manage.py help

The “myproject” subfolder − This folder is the actual python package of your project. It contains four files −

__init__.py − Just for python, treat this folder as package. settings.py − As the name indicates, your project settings.

urls.py − All links of your project and the function to call. A kind of ToC of your project. wsgi.py − If you need to deploy your project over WSGI.

Setting Up Your Project

Your project is set up in the subfolder myproject/settings.py. Following are some important options you might need to set −

DEBUG = True

This option lets you set if your project is in debug mode or not. Debug mode lets you get more information about your project's error. Never set it to ‘True’ for a live project. However, this has to be set to ‘True’ if you want the Django light server to serve static files. Do it only in the development mode.

```
DATABASES = {
  'default': {
    'ENGINE': 'django.db.backends.sqlite3', 'NAME': 'database.sql',
    'USER': '',
```

```
  'PASSWORD': '',
      'HOST': '',
      'PORT': '',
    }
}
```

Database is set in the ‘Database’ dictionary. The example above is for SQLite engine. As stated earlier, Django also supports −

MySQL (django.db.backends.mysql)

PostGreSQL (django.db.backends.postgresql_psycopg2) Oracle (django.db.backends.oracle) and NoSQL DB

MongoDB (django_mongodb_engine)

Before setting any new engine, make sure you have the correct db driver installed.

You can also set others options like: TIME_ZONE, LANGUAGE_CODE, TEMPLATE…

Now that your project is created and configured make sure it's working −

$ python manage.py runserver

You will get something like the following on running the above code − Validating models...

0 errors found

December 03, 2022 - 11:41:50

Django version 1.6.11, using settings 'myproject.settings' Starting development server at http://127.0.0.1:8000/

Quit the server with CONTROL-C.

A project is a sum of many applications. Every application has an objective and can be reused into another project, like the contact form on a website can be an application, and can be reused for others. See it as a module of your project.

Create an Application

We assume you are in your project folder. In our main “myproject” folder, the same folder then manage.py −

$ python manage.py startapp myapp

You just created myapp application and like project, Django create a “myapp” folder with the application structure −

myapp/

__init__.py

admin.py models.py tests.py views.py

__init__.py − Just to make sure python handles this folder as a package. admin.py − This file helps you make the app modifiable in the admin interface. models.py − This is where all the application models are stored.

tests.py − This is where your unit tests are.

views.py − This is where your application views are. Get the Project to Know About Your Application

At this stage we have our "myapp" application, now we need to register it with our Django project "myproject". To do so, update INSTALLED_APPS tuple in the settings.py file of your project (add your app name) −

```
INSTALLED_APPS = (
   'django.contrib.admin',
```

```
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
     'django.contrib.messages',
    'django.contrib.staticfiles',
     'myapp',
)
```

4.6 TESTING:

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

4.6.1 UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

4.6.2 INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by

successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

4.6.3 FUNCTIONAL TESTING:

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input: identified classes of valid input must be accepted.

Invalid Input: identified classes of invalid input must be rejected.

Functions: identified functions must be exercised.

Output: identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

4.6.4 SYSTEM TESTING:

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

4.6.5 WHITE BOX TESTING:

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

4.6.6 BLACK BOX TESTING:

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot "see" into it. The test provides inputs and responds to outputs without considering how the software works.

4.6.7 UNIT TESTING:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

4.6.8 INTEGRATION TESTING

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

4.6.9 ACCEPTANCE TESTING:

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

CHAPTER 5

SOURCE CODE

```
from flask import Flask, request,render_template
from flask_restful import Resource, Api
from flask_httpauth import HTTPBasicAuth
from werkzeug.security import generate_password_hash, check_password_hash
from datapreparation.data_prep import data_prep
from train import train_class
from predict import predict_class import json
from werkzeug.utils import secure_filename import os
import shutil

app=Flask(_name_) app.config['UPLOAD_FOLDER']='raw'

api=Api(app) data_prep_instance=data_prep() train_instance=train_class()
predict_instance=predict_class()


with open('config.json', 'r') as f:
data = json.load(f)

auth=HTTPBasicAuth() users={
   "admin":generate_password_hash("okayboss")
}

@auth.verify_password
def verify_password(username,password):
   if username in users and check_password_hash(users.get(username),password): return username
```

```
@app.route('/')
# @auth.login_required def index():
   return render_template('index.html', msg="I'm working")

@app.route('/data_present',  methods = ['GET', 'POST']) def trained_data():
   listOfTrained=data_prep_instance.data_trained(data['csv_path']) return render_template('index.html',
   list_status=listOfTrained)


@app.route('/train',  methods = ['GET', 'POST']) def train():
   return render_template('train.html')


@app.route('/upload',  methods = ['GET', 'POST']) def upload_file():
   if request.method == 'POST':
      files = request.files.getlist('files[]') folder_name = request.form['text']

   create_folder_path=app.config['UPLOAD_FOLDER'] if not os.path.exists(create_folder_path):
      # create the folder os.mkdir(create_folder_path) print("Folder created successfully.")
   else:
      print("Folder already exists.")


   for file in files:
      filename = secure_filename(file.filename) try:
         directory = app.config['UPLOAD_FOLDER'] +'/'+ folder_name os.mkdir(directory)
      except FileExistsError: pass
      file.save(os.path.join(directory,  filename))
```

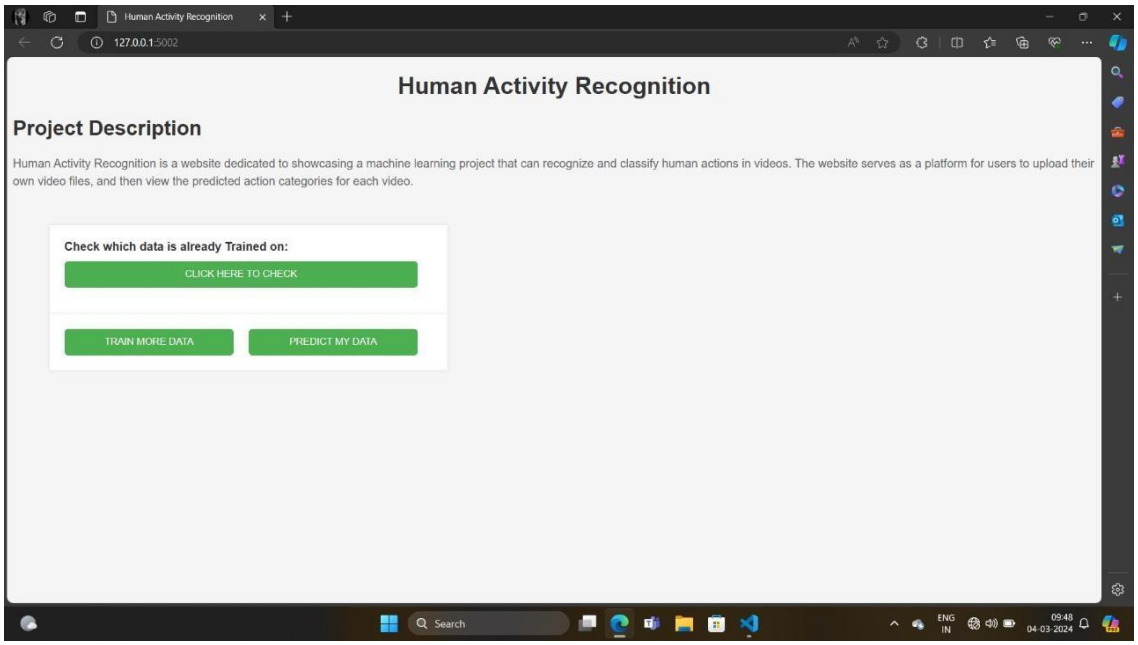
```
data_prep_instance.process_data(data['images_dir'],data['csv_path'],data['pose_model'],data['body_dict'],
'train')


    # check if the folder exists before attempting to delete it if os.path.exists(create_folder_path):
    # use shutil.rmtree() function to delete the folder and all its contents
    shutil.rmtree(create_folder_path, ignore_errors=True)
    print("Folder deleted successfully.") else:
    print("Folder does not exist.")

  return render_template('train.html', file_status="'Files uploaded successfully!")


@app.route('/data_prep_fun',  methods = ['GET', 'POST']) def data_prep_fun():
  acc=train_instance.train_model(data['csv_path'])
  return render_template('train.html', data_prep_fun_status="'Model accuracy is: " + str(acc))


@app.route('/predict',  methods = ['GET', 'POST']) def predict():
  return render_template('predict.html')


@app.route('/upload_predict',  methods = ['GET', 'POST']) def upload_predict_file():
  if request.method == 'POST': file = request.files['file'] folder_name = 'unknow'
  os.mkdir('upload')
  filename = secure_filename(file.filename) try:
    directory = 'upload' +'/'+ folder_name os.mkdir(directory)
  except FileExistsError: pass
  file.save(os.path.join(directory,  filename))
```

```
    data_prep_instance.process_data(data['predict_video'],data['predict_csv'],data['pose_model'],data
    ['body_dict'],'predict')

    if os.path.exists('upload'):
        # use shutil.rmtree() function to delete the folder and all its contents shutil.rmtree('upload',
        ignore_errors=True)
        print("Folder deleted successfully.") else:
        print("Folder does not exist.")

  return render_template('predict.html', file_status="Predict File uploaded successfully!")


@app.route('/prdict_data',  methods = ['GET', 'POST']) def prdict_data():
    result=predict_instance.predict_model(data['predict_csv']) return render_template('predict.html',
    predict_status=result)


@app.route('/backtohome', methods = ['GET', 'POST']) def backtohome():
    return render_template('index.html')


if _name== 'main_': app.run(debug=True,host='0.0.0.0',port=5002)
```

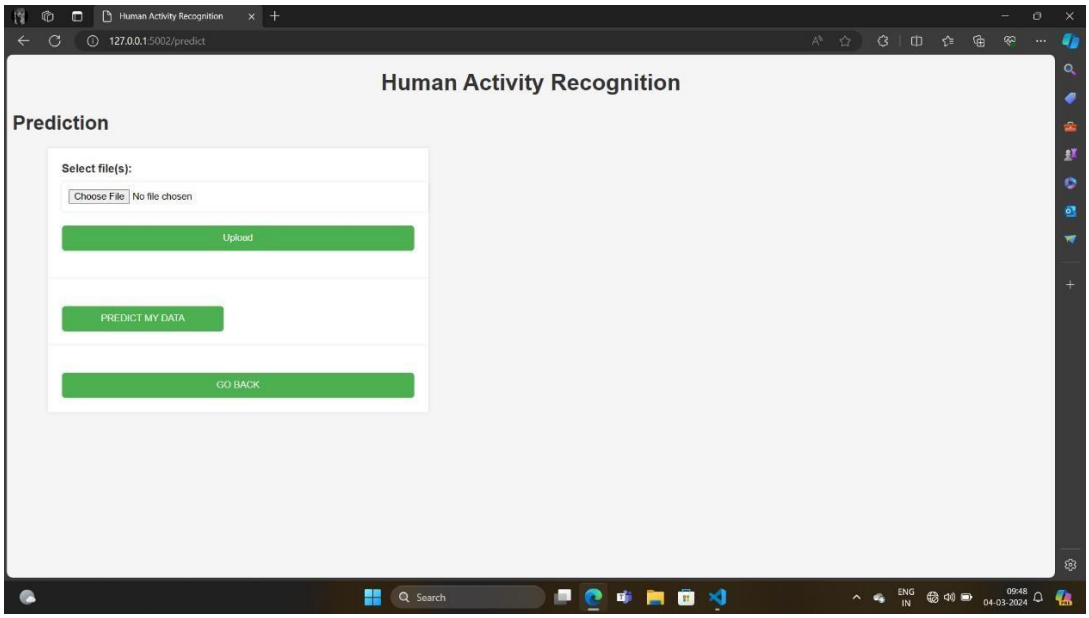
CHAPTER 6

EXPERIMENTAL RESULTS

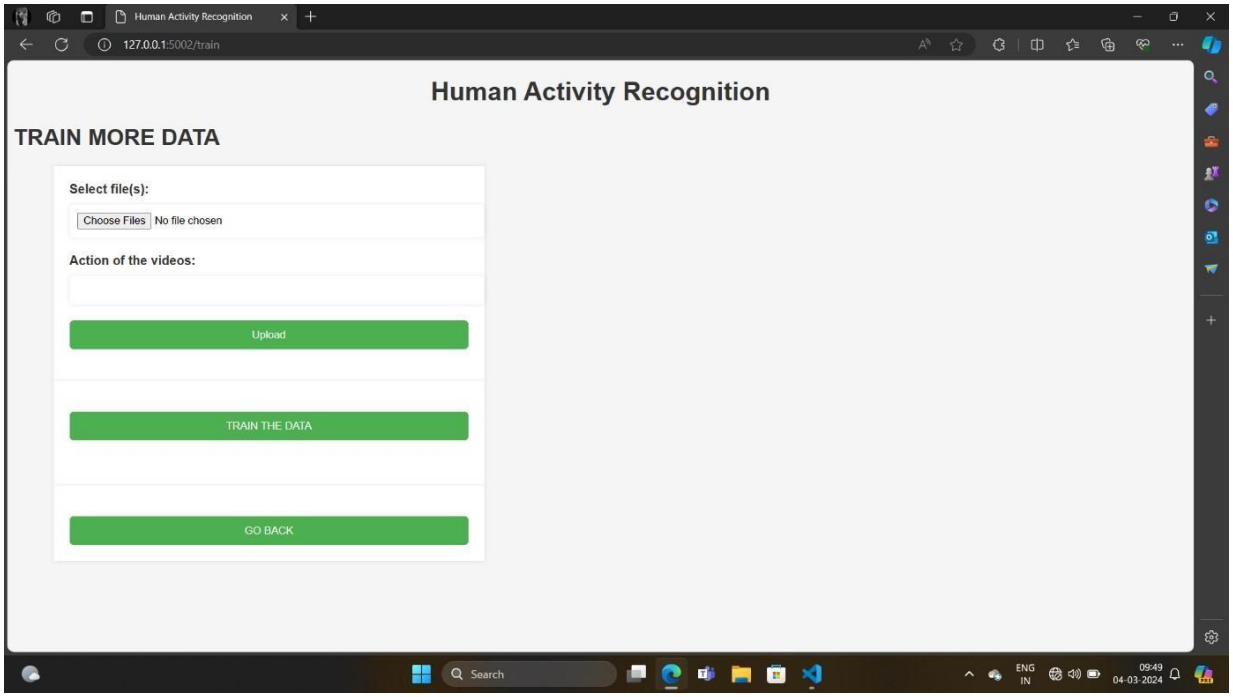
Home Page :



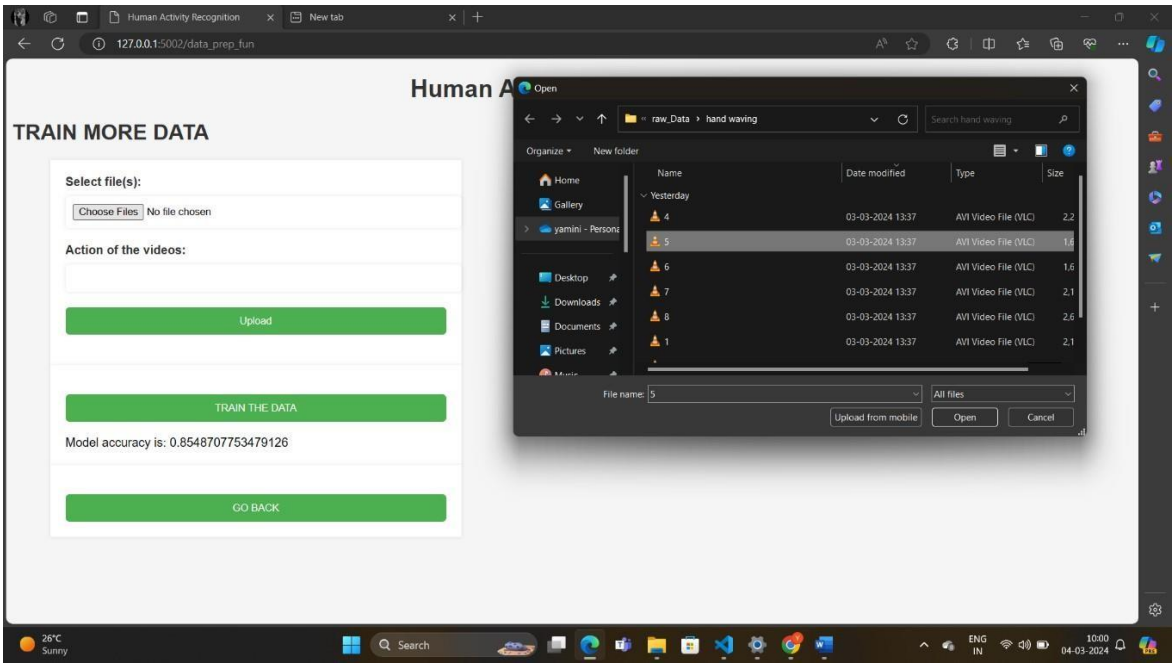
Prediction Data Uploading:



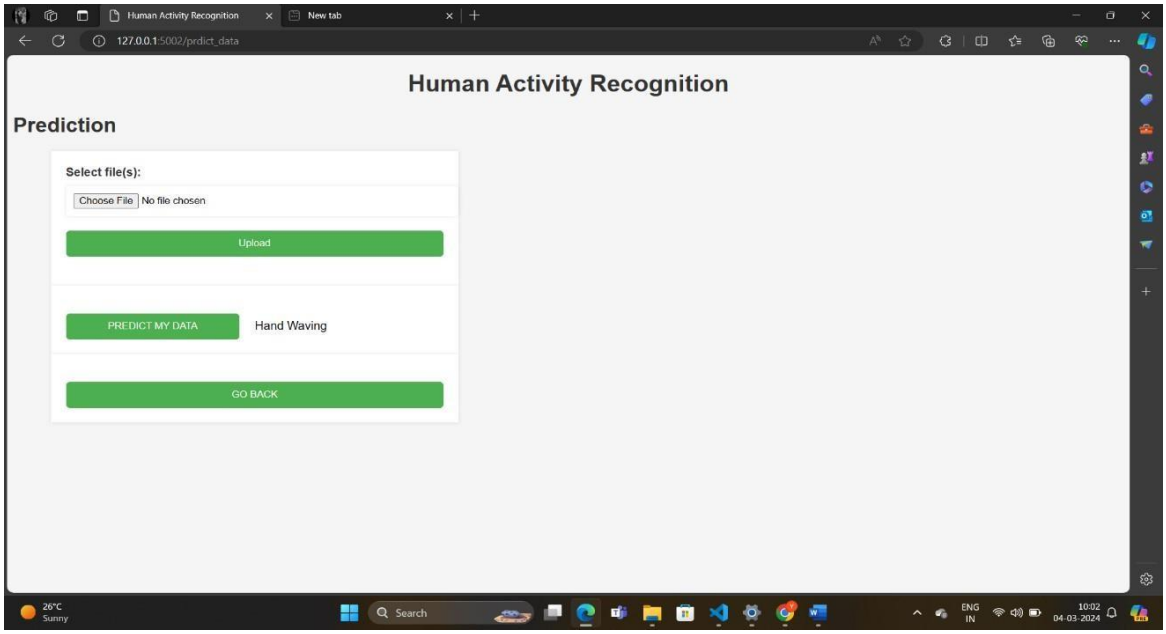
Data Training:



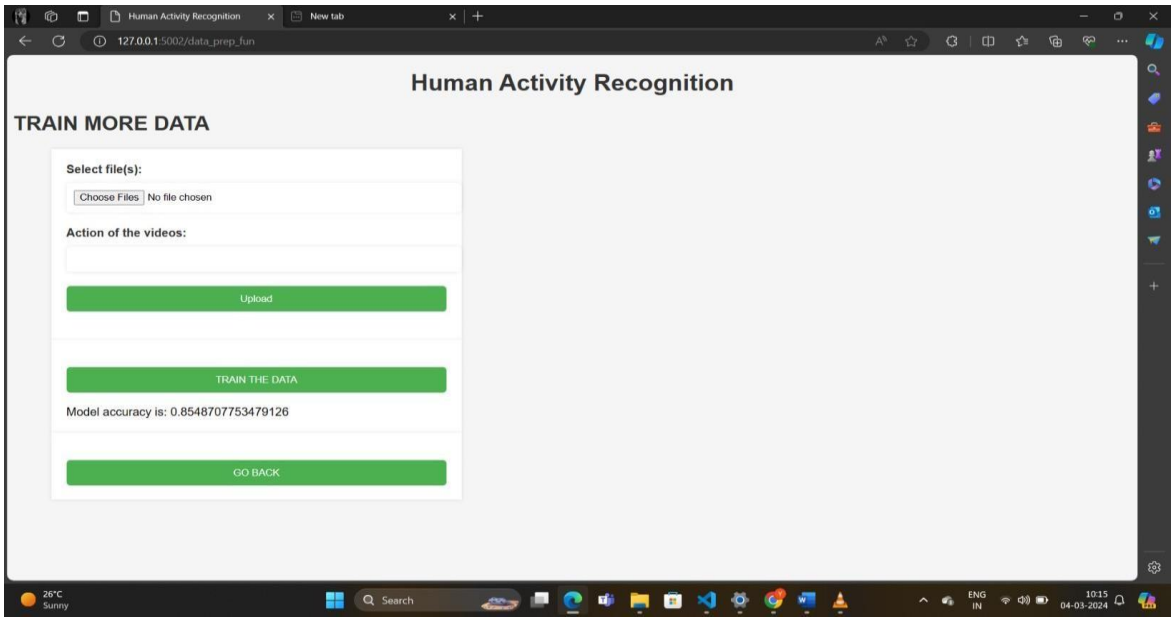
Data Uploading:



Data Prediction:

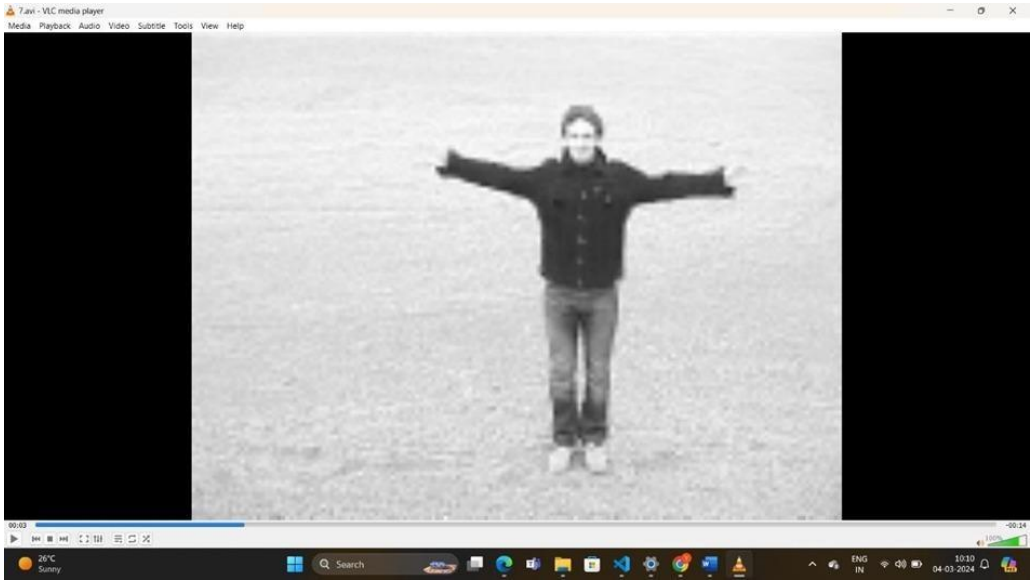


Model Accuracy:



Data Input :

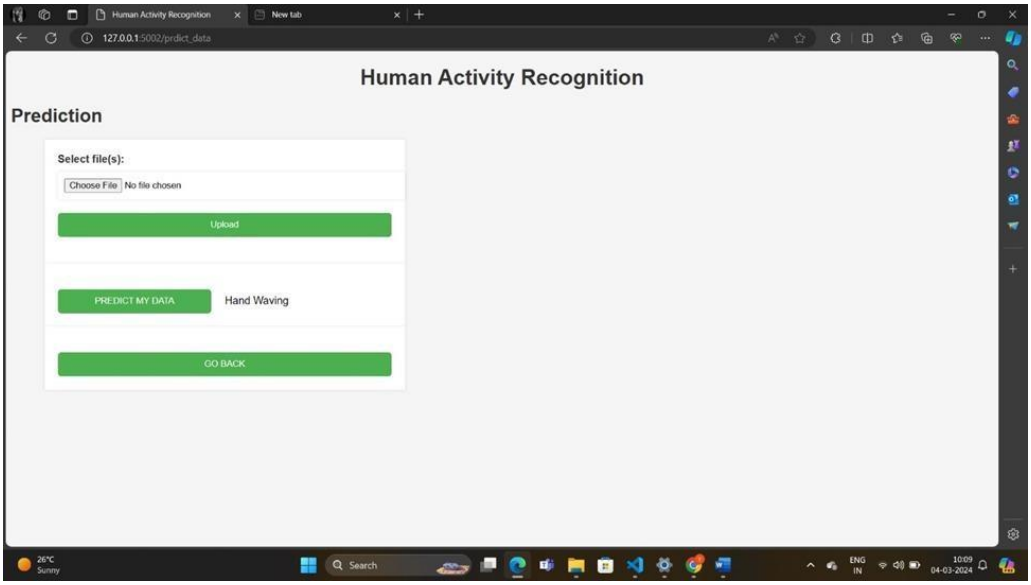
Hand Waving:



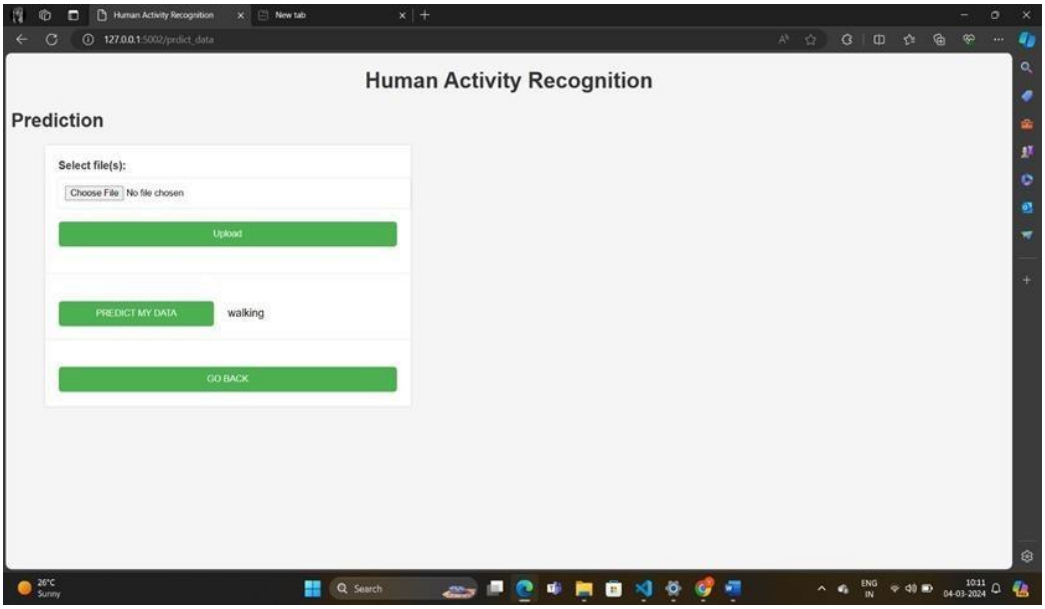
Walking:

Output:

Prediction:



Walking:



CHAPTER 7

CONCLUSION AND FURTHER ENHANCEMENT

7.1 CONCLUSION:

We presented three CNN based models as well as their ensembles for WISDM dataset of HAR. It was found that the performance of the ensemble model is better than that of individual models. One of the ensemble model performed better than the methods in the literature. In the dataset we used, we see a class imbalance such that we have 38% samples for walking class but hardly 5% for sitting and standing. In future, the results might be improved even more, if we can remove the class imbalance from dataset. Moreover, currently an ensemble of average of the three models is created but for further exploration, a future direction can be performing weighted ensemble learning such that the best performing model has the most effect in the ensemble

7.2 : FUTURE ENHANCEMENT:

Furthermore, we can explore the area of ensemble learning for a hybrid model, that is, ensemble learning of CNN and RNN

REFERENCES

[1] T. Plotz, N. Y. Hammerla, and P. L. Olivier, "Feature learning for ¨ activity recognition in ubiquitous computing," in 22nd International Joint Conference on Artificial Intelligence, 2011.

[2] Y. Chen, K. Zhong, J. Zhang, Q. Sun, and X. Zhao, "LSTM networks for mobile human activity recognition," in International Conference on Artificial Intelligence: Technologies and Applications (ICAITA 2016), 2016.

[3] L. K. Hansen and P. Salamon, "Neural network ensembles," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 12, no. 10, pp. 993–1001, 1990.

[4] J. Yang, M. N. Nguyen, P. P. San, X. L. Li, and S. Krishnaswamy, "Deep Convolutionalal neural networks on multichannel time series for human activity recognition," in 24th International Joint Conference on Artificial Intelligence, 2015.

[5] M. Panwar, S. R. Dyuthi, K. C. Prakash, D. Biswas, A. Acharyya, K. Maharatna, A. Gautam, and G. R. Naik, "CNN based approach for activity recognition using a wrist-worn accelerometer," in 2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC). IEEE, 2017, pp. 2438–2441.

[6] S. W. Pienaar and R. Malekian, "Human activity recognition using LSTM-RNN deep neural network architecture," in 2019 IEEE 2nd Wireless Africa Conference (WAC). IEEE, 2019, pp. 1–5.

[7] J. Wang, Y. Chen, S. Hao, X. Peng, and L. Hu, "Deep learning for sensorbased activity recognition: A survey," Pattern Recognition Letters, vol. 119, pp. 3–11, 2019

[8] M. M. Hassan, M. Z. Uddin, A. Mohamed, and A. Almogren, "A robust human activity recognition system using smartphone sensors and deep learning," Future Generation Computer Systems, vol. 81, pp. 307–313, 2018.

[9] J. Sun, Y. Fu, S. Li, J. He, C. Xu, and L. Tan, "Sequential human activity recognition based on deep Convolutionalal network and extreme learning machine using wearable sensors," Journal of Sensors, vol. 2018, pp. 1–10, 09 2018.

[10] K. Xia, J. Huang, and H. Wang, "LSTM-CNN Architecture for Human Activity Recognition," IEEE Access, vol. 8, pp. 56 855–56 866, 2020.

[11] D. Ravi, C. Wong, B. Lo, and G. Yang, "Deep learning for human activity recognition: A resource efficient implementation on low-power devices," in 2016 IEEE 13th International Conference on Wearable and Implantable Body Sensor Networks (BSN), 2016, pp. 71–76.

[12] M. Wozniak, M. Gra ´ na, and E. Corchado, "A survey of multiple classifier ˜ systems as hybrid systems," Information Fusion, vol. 16, pp. 3–17, 2014.

[13] J. R. Kwapisz, G. M. Weiss, and S. A. Moore, "Activity recognition using cell phone accelerometers," ACM SigKDD Explorations Newsletter, vol. 12, no. 2, pp. 74–82, 2011.

[14] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all Convolutionalal net," arXiv preprint arXiv:1412.6806, 2014.

[15] M. Lin, Q. Chen, and S. Yan, "Network in Network," arXiv preprint arXiv:1312.4400, 2013.

[16] C. A. Ronao and S.-B. Cho, “Human activity recognition with smartphone sensors using deep learning neural networks,” Expert Systems with Applications, vol. 59, pp. 235 – 244, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0957417416302056