A

Mini Project

On

Human Action Recognition From Depth Maps and Postures Using Deep Learning

(Submitted in partial fulfillment of the requirements for the award of Degree)

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

By

BIRADAR NITESH (207R1A05M4)

MARELLI RAMYA (207R1A05M1)

POLABOINA SREEJA (207R1A05M6)

Under the Guidence of

MRS. NAJEEMA AFRIN

(Assistant Professor)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR TECHNICAL CAMPUS

UGC AUTONOMOUS

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi) Recognized Under Section 2(f) & 12(B) of the UGCAct.1956, Kandlakoya (V), Medchal Road, Hyderabad-501401.

2020-2024

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project entitled **"HUMAN ACTION RECOGNITION FROM DEPTH MAPS AND POSTURES USING DEEP LEARNIN"** being submitted by **B. NITESH(207R1A05M4),M. RAMYA (207R1A05M1) & P. SREEJA(207R1A05M6)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by them under our guidance and supervision during the year 2023-24.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

MRS. NAJEEMA AFRIN (Assistant Professor) INTERNAL GUIDE	**Dr. A. RAJI REDDY** DIRECTOR
DR.K. SRUJAN RAJU HOD	**EXTERNAL EXAMINER**

Submitted for viva voice Examination held on ________________________________

ACKNOWLEDGEMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We take this opportunity to express my profound gratitude and deep regard to my guide **NAJEEMA AFRIN,** Assistant Professor for his exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by him shall carry us a long way in the journey of life on which we are about to embark.

We also take this opportunity to express a deep sense of gratitude to the Project Review Committee (PRC) **G.Vinesh Shanker, Dr. J. Narasimharao, Ms. Shilpa, & Dr. K. Maheswari** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. K. Srujan Raju,** Head, Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy,** Director for being cooperative throughout the course of this project. We also express our sincere gratitude to Sri. **Ch. Gopal Reddy,** Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

B. NITESH (207R1A05M4)

M.RAMYA (207R1A05M1)

P. SREEJA (207R1A05M6)

ABSTRACT

Human Activity Recognition is one of the active research areas in computer vision for various contexts like security surveillance, healthcare and human computer interaction. Over the past years, several methods published for human action recognition using RGB (red, green, and blue), depth, and skeleton datasets. Most of the methods introduced for action classification using skeleton datasets are constrained in some perspectives including features representation, complexity, and performance. However, there is still a challenging problem of providing an effective and efficient method for human action discrimination using a skeleton dataset. The first input is depth images and second input is a proposed moving joints descriptor (MJD) which represents the motion of body joints over time, in order to maximize feature extraction for accurate action classification, CNN channels are trained with different inputs and for Score fusion we are planning to use neural networks. Our proposed method was implementation on public datasets like MSR Action 3D.

Furthermore, we discuss the potential application of our approach in real-world settings, such as surveillance system, human-robot interaction, and gesture-based interfaces. The ability to recognize human action accurately and in real-time using depth maps and postures opens up new opportunities for improving the efficiency and effectiveness of various human-centric technologies.

LIST OF TABLES/FIGURES

FIGURE NO	FIGURE NAME	PAGE NO
Figure 3.1	Project Architecture of Human Action Recognition from Depth Maps and Postures using learning	6
Figure 3.2	Use Case Diagram of Human Action Recognition from Depth Maps and Postures using Deep learning	7
Figure 3.3	Class Diagram of Human Action recognition from Depth Maps and Postures using Deep learning	8
Figure 3.4	Sequence diagram of Human Action Recognition from Depth Maps and Postures using Deep learning	9
Figure 3.5	Activity diagram of Human Action Recognition from Depth Maps and Postures using Deep learning	10

LIST OF SCREENSHOTS

SCREENSHOT NO	SCREENSHOT NAME	PAGE NO
Screenshot 5.1	Upload MSRAction 3D Image	19
Screenshot 5.2	MSR ACTION dataset	19
Screenshot 5.3	Features Extraction	20
Screenshot 5.4	Train CNN Algorithm	20
Screenshot 5.5	CNN Performance graph	21

TABLE OF CONTENTS

ABSTRACT	i
LIST OF FIGURES	ii
LIST OF TABLES	iii
1.INTRODUCTION	**1**
1.1 PROJECT SCOPE	1
1.2 PROJECT PURPOSE	1
1.3 PROJECT FEATURES	1
2.SYSTEM ANALYSIS	**2**
2.2 EXISTING SYSTEM	2
2.2.1 DISADVANTAGES OF EXISTING SYSTEM	2
2.3 PROPOSED SYSTEM	3
2.3.1 ADVANTAGES OF PROPOSED SYSTEM	3
2.4 FEASIBILITY STUDY	3
2.4.1 ECONOMIC FEASIBILITY	4
2.4.2 TECHNICAL FEASIBILITY	4
2.4.3 SOCIAL FEASIBILITY	4
2.5 HARDWARE & SOFTWARE REQUIREMENTS	5
2.5.1 HARDWARE REQUIREMENTS	5
2.5.2 SOFTWARE REQUIREMENTS	5
3.ARCHITECTURE	**6**
3.1 PROJECT ARCHITECTURE	6
3.2 DESCRIPTION	7
3.3 USE CASE DIAGRAM	7
3.4 CLASS DIAGRAM	8
3.5 SEQUENCE DIAGRAM	9
3.6 ACTIVITY DIAGRAM	10
4.IMPLEMENTATION	**11**
4.1 SAMPLE CODE	11

5.SCREENSHOTS 19

6.TESTING **22**

6.1 INTRODUCTION TO TESTING 22

6.2 TYPES OF TESTING 22

6.2.1 UNIT TEST 22

6.2.2 INTEGRATION TESTING 23

6.2.3 FUNCTIONAL TESTING 23

6.2.4 SYSTEM TESTING 23

6.2.5 WHITE BOX TESTING 24

6.2.6 BLACK BOX TESTING 24

6.4 TEST CASES 24

6.4.1 CLASSIFICATION 24

7.CONCLUSION & FUTURE SCOPE **25**

7.1 PROJECT CONCLUSION 25

7.2 FUTURE SCOPE 25

8.BIBLIOGRAPHY **26**

8.1 REFERENCES 26

8.2 GITHUBLINK 26

1.INTRODUCTION

1.1 PROJECT SCOPE

The project aims to develop a robust and accurate Human Action Recognition System using Deep Learning techniques, specifically focusing on leveraging depth maps and Postures input modalities. The scope encompasses the design, training, and evaluation of a deep neural network architecture that combines Convolutional Neural Networks(CNNs) and Recurrent Neural Network(RNNs) to capture temporal dependencies and spatial information from depth maps and postures. The system will be evaluated on benchmark datasets to assess its performance against existing methods. Additionally, potential real-world applications, including surveillance, human-computer interaction, and healthcare, will be explored to demonstrate the practical utility of the developed system.

1.2 PROJECT PURPOSE

The purpose of this project is to advance the field of human action recognition by harnessing the capabilities of Deep Learning in combination with depth maps and postures as input data sources. Through the development of an innovative deep neural network architecture, our aim is to enhance the accuracy and robustness of recognizing human actions in diverse environments and lighting conditions. By leveraging depth maps and postures, offering more reliable and 3D-aware action recognition. The computer project also seeks to explore practical applications, paving the way for improved human- interaction and automation in various domains.

1.3 PROJECT FEATURES

The Project’s key features revolve around the innovative integration of depth maps and postures using deep learning techniques for human action recognition. It employs depth maps to provide a comprehensive, 3-dimensional perspective of human. Its deep learning architecture combines Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) for feature extraction and temporal modeling. Rigorous benchmark evaluations on datasets like NTU RGB+D and MSR Action3D validate its performance against existing methods.

2.SYSTEM ANALYSIS

2.1 SYSTEM ANALYSIS

System Analysis is the important phase in the system development process. The System is studied to the minute details and analyzed. The system analyst plays an important role of an interrogator and dwells deep into the working of the present system. In analysis, a detailed study of these operations performed by the system and their relationships within and outside the system is done. A key question considered here is, “what must be done to solve the problem?” The system is viewed as a whole and the inputs to the system are identified. Once analysis is completed the analyst has a firm understanding of what is to be done.

2.2 EXISTING SYSTEM

There are a variety of human action recognition systems, such as video-based human action recognition, wearable sensor-based human action recognition, wireless sensor network-based human action recognition, etc...

Among these studies, due to high recognition accuracy and easy deployment, video-based human action recognition techniques have got more research attention and been widely applied into lots of industrial application.

2.2.1 DISADVANTAGES OF EXISTING SYSTEM

The following are the Disadvantages of the Existing system:

- Complexity and Computation
- Robustness
- Interpretability
- Lack of Generalization

2.3 PROPOSED SYSTEM

The action recognition process introduced in this paper involves three CNN channels trained with DMI and MJD descriptors for feature extraction and classification. The first channel is trained with DMI., the second channel is a connection between two subchannels. One subchannel is trained with DMI and the other subchannel is trained with MJD. The third channel is trained with MJD only.

2.3.1 ADVANTAGES OF PROPOSED SYSTEM

The following are the Advantages of Proposed system:

- Accurate and Reliable
- Flexibility
- Real-time Performance
- Robustness
- Adaptable
- Potential for Automation

2.4 FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

2.4.1 ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

2.4.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

2.4.3 SOCIAL FEASIBILITY:

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

2.5 HARDWARE AND SOFTWARE REQUIREMENTS

2.5.1 HARDWARE REQUIREMENTS:

Hardware interfaces specify the logical characteristics of each interface between the software product and the hardware components of the system. The following are some hardware requirements.

- RAM: 4GB DD RAM
- HAED DISK: 250 GB

2.5.2 SOFTWARE REQUIREMENTS:

Software Requirements specifies the logical characteristics of each interface and software components of the system. The following are some software requirements.

- Operating System : Windows 7/8/10
- Platform : Visual Studio Code
- Programming Language : Python
- Front End : Visual Studio Code

3.ARCHITECTURE

3.1 PROJECT ARCHITECTURE

This project architecture shows the procedure followed for classification, starting from input to final prediction.

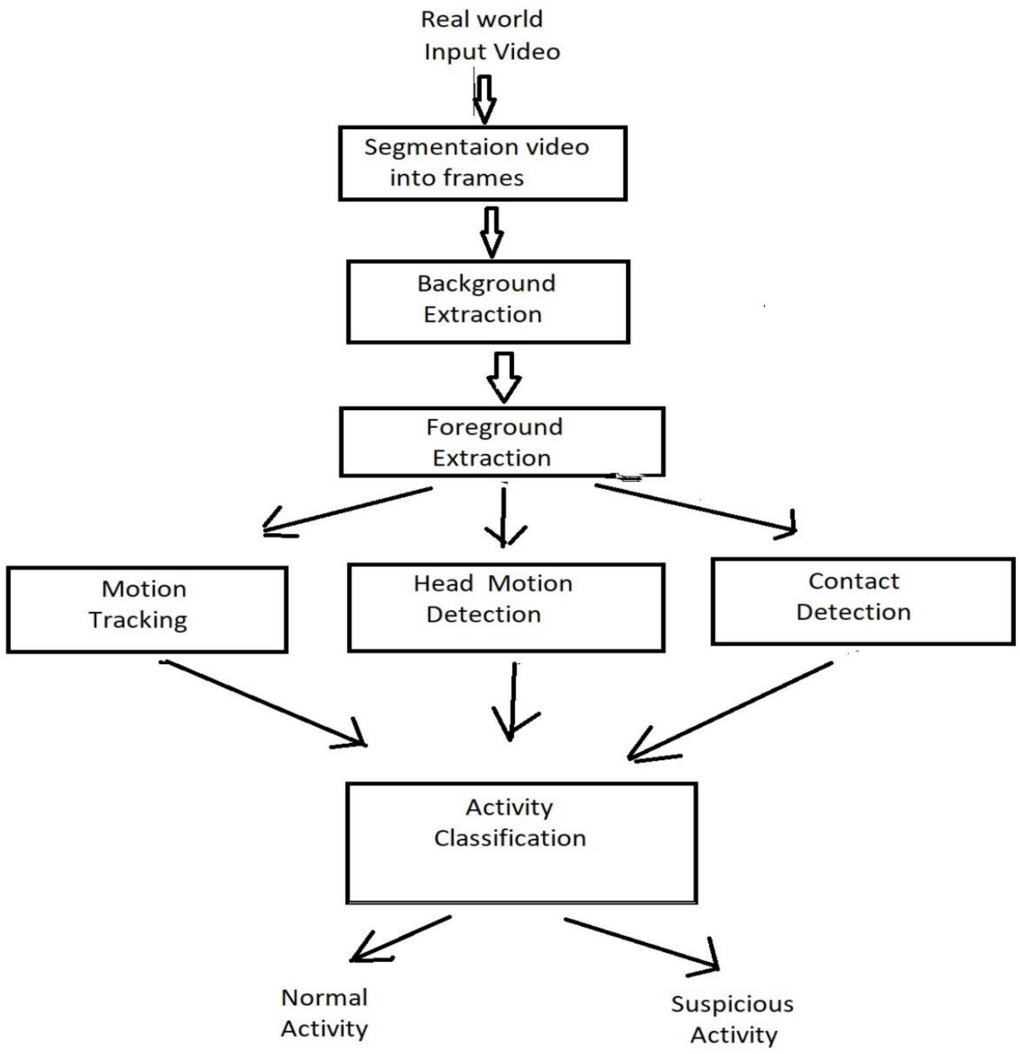


Figure 3.1: Project Architecture of Human Action Recognition from Depth Maps and Postures Using Deep Learning

3.2 DESCRIPTION

The architecture for human action recognition from depth maps and postures using deep learning is a sophisticated system that combines depth maps and posture information to accurately classify human actions in a three-dimensional space. It starts with the input modalities, where depth maps capture 3D scene structure, and postures represent skeletal joint positions. Convolutional Neural Networks (CNNs) extract spatial features from depth maps, while posture information is processed using techniques like graph convolutions or recurrent neural networks (RNNs) for temporal modeling.

3.3 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.

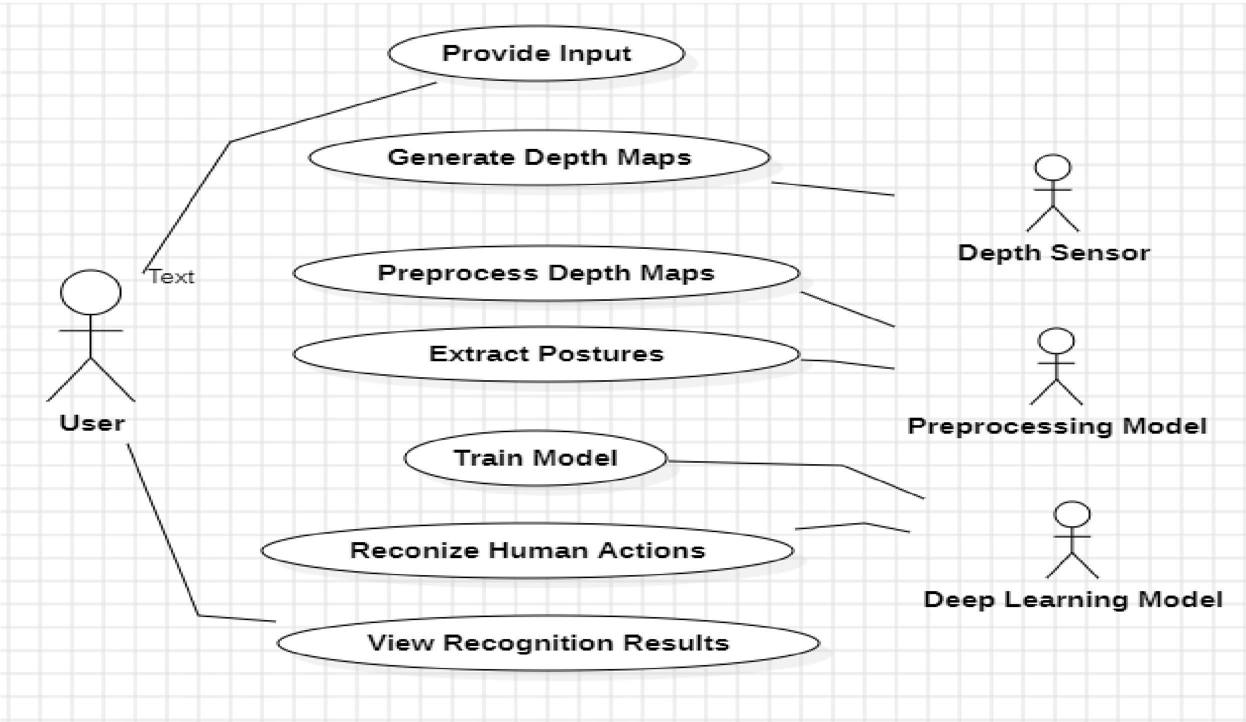


Figure 3.2: Use Case Diagram of Human Action Recognition from Depth Maps and Postures Using Deep Learning

3.4 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.

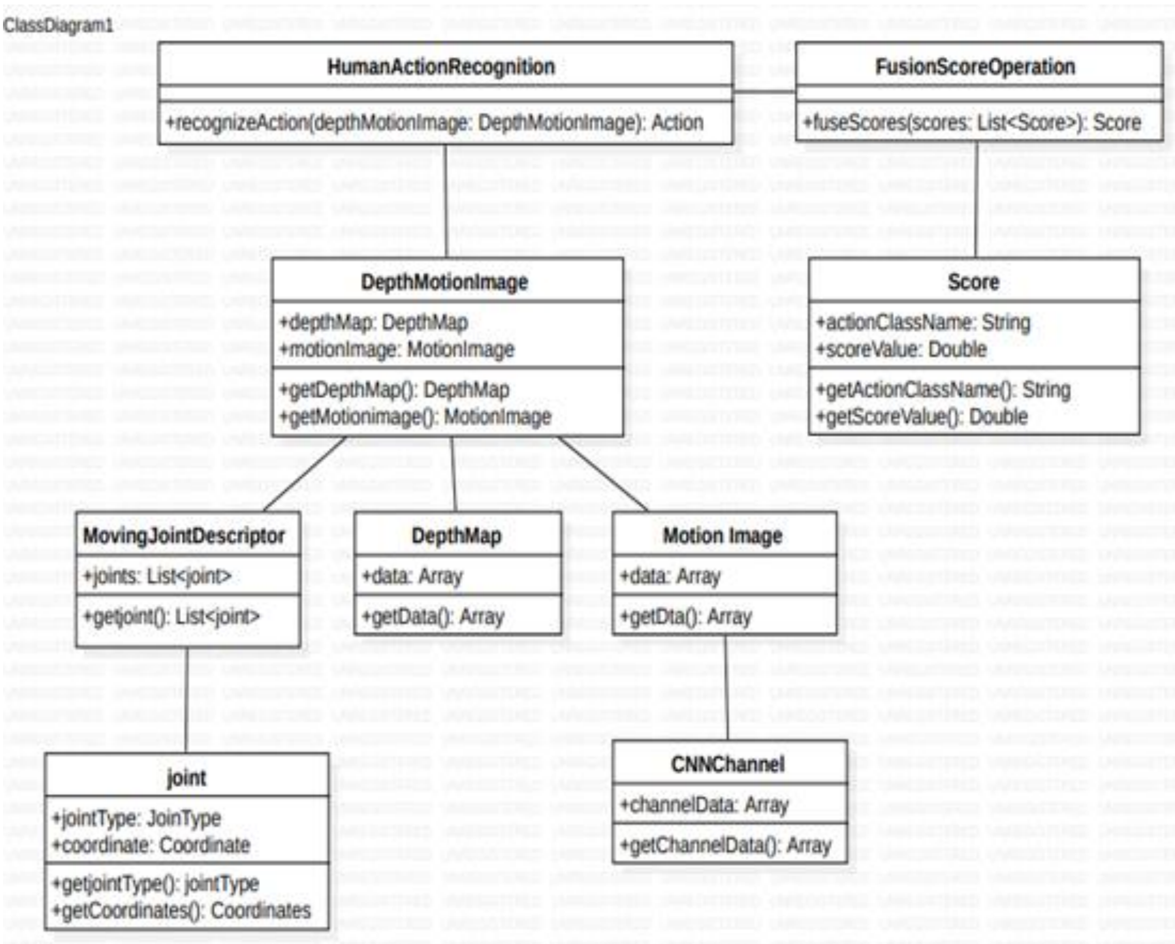


Figure 3.3: Class Diagram of Human Action Recognition from Depth Maps and Postures Using Deep Learning

3.5 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.

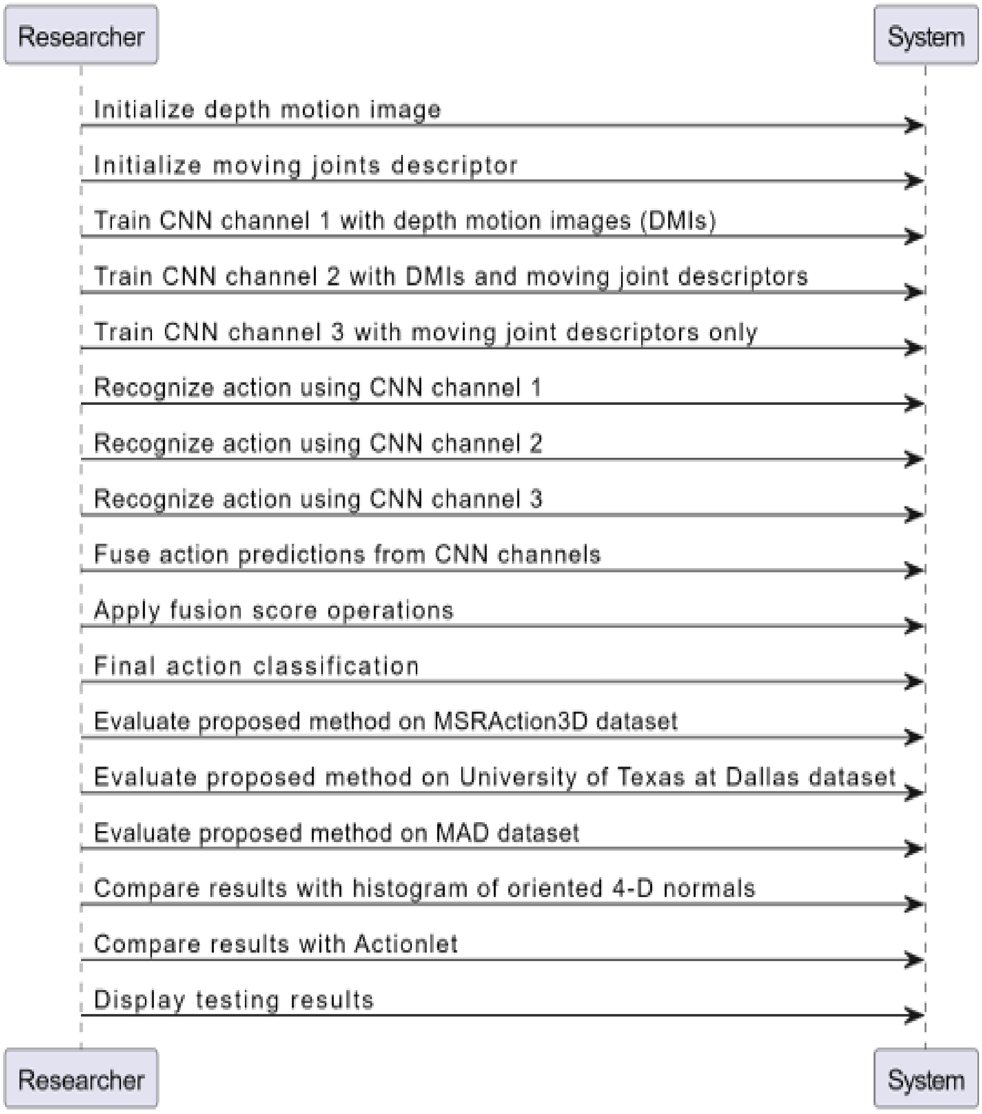


Figure 3.4: Sequence Diagram of Human Action Recognition from Depthmaps and Postures using DeepLearning

3.6 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. They can also include elements showing the flow of data between activities through one or more data stores.

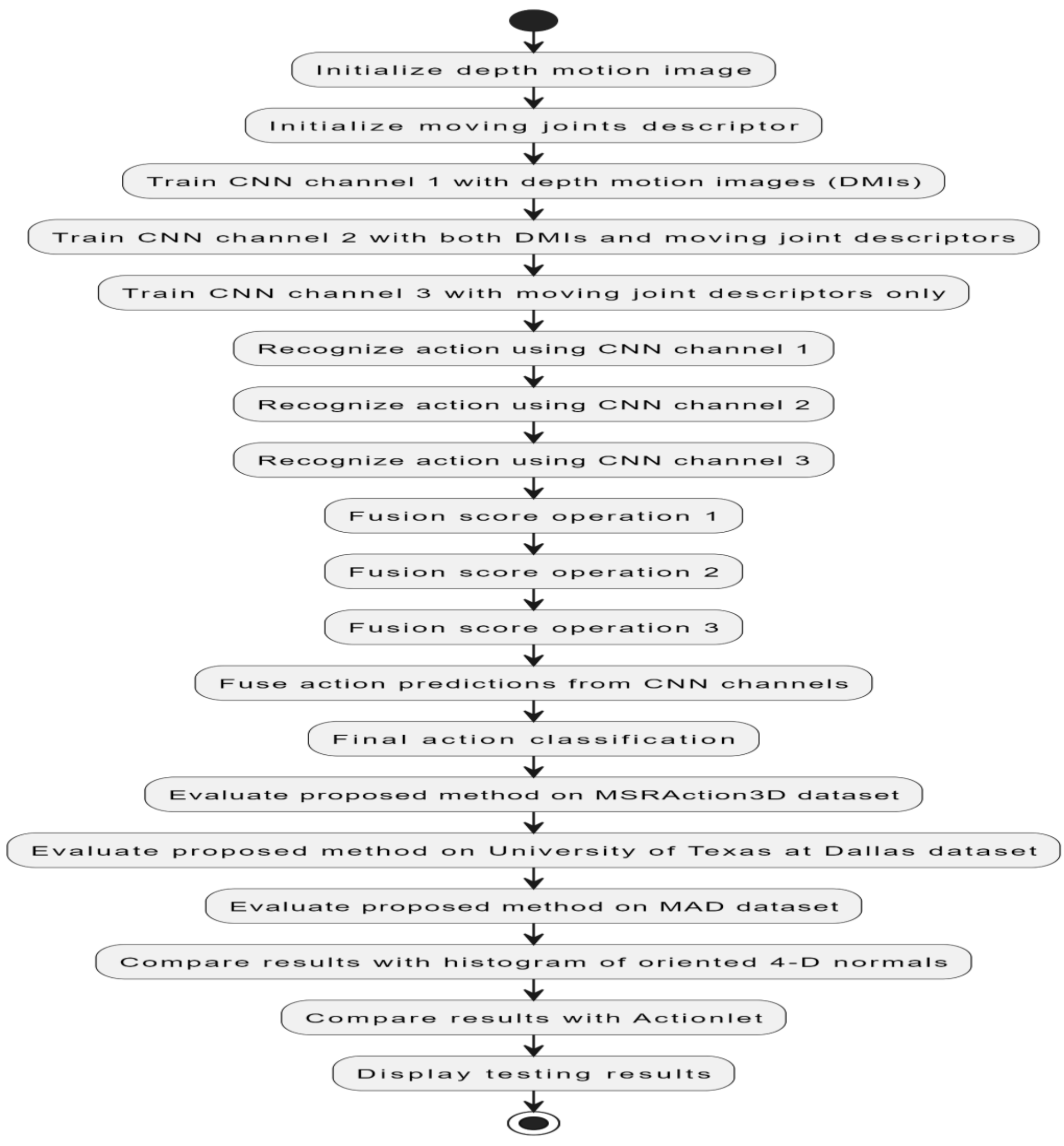


Figure 3.5: Activity Diagram of Human Action Recognition from Depth Maps and Postures Using Deep Learning

4.IMPLEMENTION

SAMPLE CODE

```
from tkinter import messagebox
from tkinter import *
from tkinter import simpledialog
import tkinter
from tkinter import filedialog
import matplotlib.pyplot as plt
import numpy as np
from tkinter.filedialog import askopenfilename
from keras.utils.np_utils import to_categorical
from keras.layers import  MaxPooling2D
from keras.layers import Dense, Dropout, Activation, Flatten
from keras.layers import Convolution2D
from keras.models import Sequential
from keras.models import model_from_json
import pickle
import mpl_toolkits.mplot3d.axes3d as p3
from matplotlib.animation import FuncAnimation
from sklearn.model_selection import train_test_split
from sklearn.metrics import confusion_matrix
import seaborn as sns
from sklearn.metrics import precision_score
from sklearn.metrics import recall_score
from sklearn.metrics import f1_score
from sklearn.metrics import accuracy_score
import os

main = tkinter.Tk()
```

```
main.title("Human Action Recognition from depth maps and Postures using Deep Learning")

main.geometry("1300x1200")


global filename

global classifier

global X, labels, datas, graph

class_labels = ['high arm wave', 'horizontal arm wave', 'hammer', 'hand catch', 'forward punch', 'high throw', 'draw x', 'draw tick', 'draw circle', 'hand clap',

        'two hand wave', 'side-boxing', 'bend', 'forward kick', 'side kick', 'jogging', 'tennis swing', 'tennis serve', 'golf swing', 'pick up & throw']

def upload():

    global filename

    text.delete('1.0', END)

    filename = filedialog.askdirectory(initialdir = ".")

    pathlabel.config(text=filename)

    text.insert(END,filename+" loaded\n")


def animate(i):

    graph._offsets3d = (datas[i,:,0], datas[i,:,2], datas[i,:,1])

    return graph


def loadData(action, subject, instance):

    ins = np.loadtxt(filename+'/a%02i_s%02i_e%02i_skeleton.txt'%(action, subject, instance))

    print("ins.shape[0]",ins.shape[0])

    ins = ins.reshape((ins.shape[0]//20, 20, 4))

    return ins


def featuresExtraction():
```

```
    global X, labels, datas, graph
    text.delete('1.0', END)
    X = []
    data = np.load("model/data.txt.npy",allow_pickle=True)
    labels = np.load("model/labels.txt.npy")
    data = np.asarray(data)
    labels = np.asarray(labels)
    for i in range(len(data)):
        values = data[i][1]
        X.append(np.asarray(values))
    X = np.asarray(X)
    labels = np.asarray(labels)

    indices = np.arange(X.shape[0])
    np.random.shuffle(indices)
    X = X[indices]
    labels = labels[indices]
    labels = to_categorical(labels)
    X = X.reshape(X.shape[0],5,4,3)
    text.insert(END,"Total subject & action images found in dataset:
"+str(data.shape[0])+"\n")
    text.insert(END,"Total actions found in dataset:\n\n")
    text.insert(END,str(class_labels))

    fig = plt.figure()
    ax = p3.Axes3D(fig)
    ax.set_xlim3d([0.0, 300.0])
    ax.set_xlabel('X')
    ax.set_ylim3d([0.0, 1400.0])
    ax.set_ylabel('Z')
```

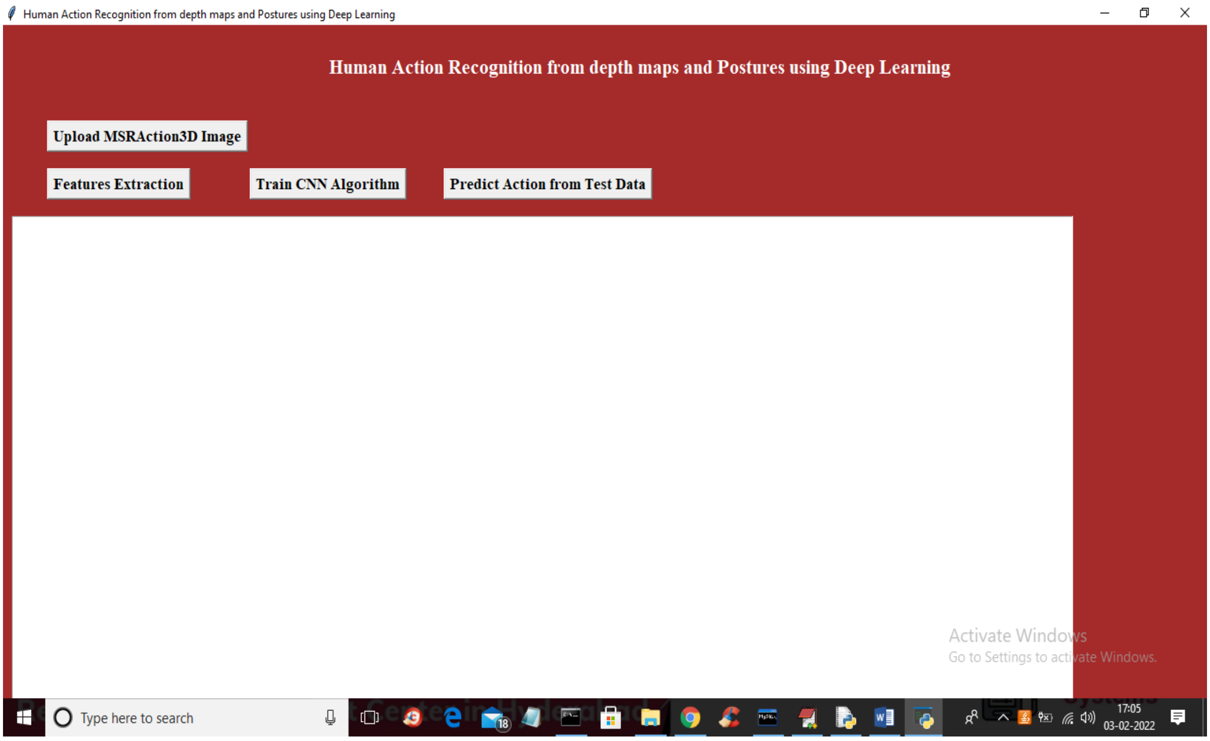
```
    ax.set_zlim3d([300.0, 0.0])
    ax.set_zlabel('Y')
    graph = ax.scatter([], [], [])
    datas = loadData(1,1,1)
    datas = np.asarray(datas)
    ani = FuncAnimation(fig, animate, frames=datas.shape[0], interval=100)
    plt.show()
def trainCNN():
    global X, labels,classifier
    text.delete('1.0', END)
    if os.path.exists('model/model.json'):
        with open('model/model.json', "r") as json_file:
            loaded_model_json = json_file.read()
            classifier = model_from_json(loaded_model_json)
        json_file.close()
        classifier.load_weights("model/model_weights.h5")
        classifier._make_predict_function()
    else:
        classifier = Sequential()
        classifier.add(Convolution2D(32, 1, 1, input_shape = (X.shape[1],
X.shape[2], X.shape[3]), activation = 'relu'))
        classifier.add(MaxPooling2D(pool_size = (1, 1)))
        classifier.add(Convolution2D(32, 1, 1, activation = 'relu'))
        classifier.add(MaxPooling2D(pool_size = (1, 1)))
        classifier.add(Flatten())
        classifier.add(Dense(output_dim = 256, activation = 'relu'))
        classifier.add(Dense(output_dim = labels.shape[1], activation = 'softmax'))
        classifier.compile(optimizer = 'adam', loss = 'categorical_crossentropy',
metrics = ['accuracy'
```

```
        hist = classifier.fit(X, labels, batch_size=16, epochs=1000, shuffle=True,
verbose=2)
        classifier.save_weights('model/model_weights.h5')
        model_json = classifier.to_json()
        with open("model/model.json", "w") as json_file:
            json_file.write(model_json)
        json_file.close()
        f = open('model/history.pckl', 'wb')
        pickle.dump(hist.history, f)
        f.close()
    X_train, X_test, y_train, y_test = train_test_split(X, labels, test_size=0.2)
    predict = classifier.predict(X_test)
    predict = np.argmax(predict, axis=1)
    testY = np.argmax(y_test, axis=1)
    p = precision_score(testY, predict,average='macro') * 100
    r = recall_score(testY, predict,average='macro') * 100
    f = f1_score(testY, predict,average='macro') * 100
    a = accuracy_score(testY,predict)*100
    text.insert(END,'CNN Activity Recognition Accuracy  : '+str(a)+"\n")
    text.insert(END,'CNN Activity Recognition Precision : '+str(p)+"\n")
    text.insert(END,'CNN Activity Recognition Recall    : '+str(r)+"\n")
    text.insert(END,'CNN Activity Recognition FMeasure  : '+str(f)+"\n\n")
    conf_matrix = confusion_matrix(testY, predict)
    plt.figure(figsize =(6, 6))
    ax = sns.heatmap(conf_matrix, xticklabels = class_labels, yticklabels =
class_labels, annot = True, cmap="viridis" ,fmt ="g");
    ax.set_ylim([0,10])
    plt.title("CNN Depth Maps Confusion matrix")
    plt.ylabel('True class')
    plt.xlabel('Predicted class')
    plt.show()
```

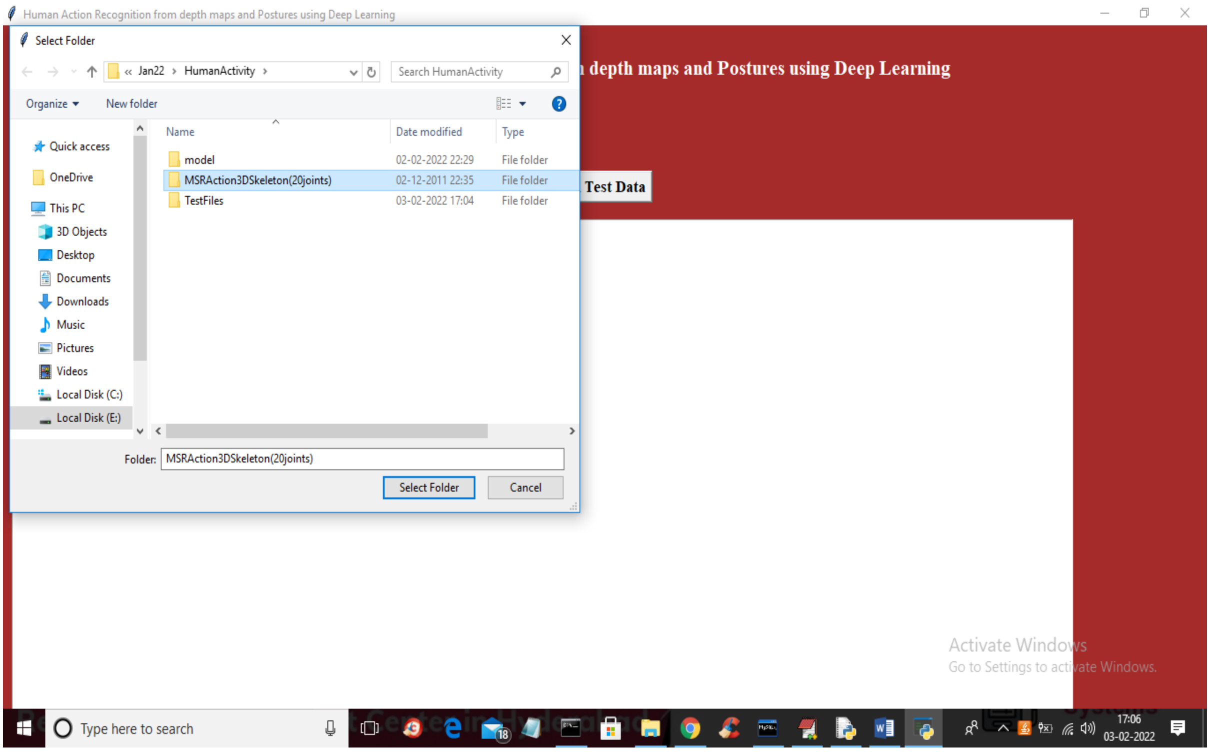
```
def read(file):
    data, labels, lens, subjects = [], [], [], []
    action = np.loadtxt(file)[:, :3].flatten()
    frame_size = len(action) // 60  # 20 iskeleton num x,y,z 3D points
    action = np.asarray(action).reshape(frame_size, 60)
    new_act = []
    for frame in action:
        new_act.append(frame)
    data.append(new_act)
    data = np.asarray(data)
    return data

def loadTestData(filename):
    ins = np.loadtxt(filename)
    print("ins.shape[0]",ins.shape[0])
    ins = ins.reshape((ins.shape[0]//20, 20, 4))
    return ins

def predictAction():
    global datas, graph
    text.delete('1.0', END)
    filename = filedialog.askopenfilename(initialdir="TestFiles")
    test = []
    data = read(filename)
    for i in range(len(data)):
        values = data[i][1]
        test.append(values)
    test = np.asarray(test
```

```
    test = test.reshape(test.shape[0],5,4,3)
    preds = classifier.predict(test)
    predict = np.argmax(preds)
    text.insert(END,"Skeleton values : "+str(test)+" ACTIVITY RECOGNIZE AS: "+class_labels[predict])
    text.update_idletasks()
    fig = plt.figure()
    ax = p3.Axes3D(fig)
    ax.set_xlim3d([0.0, 300.0])
    ax.set_xlabel('X')
    ax.set_ylim3d([0.0, 1400.0])
    ax.set_ylabel('Z')
    ax.set_zlim3d([300.0, 0.0])
    ax.set_zlabel('Y')
    graph = ax.scatter([], [], [])
    datas = loadTestData(filename)
    datas = np.asarray(datas)
    ani = FuncAnimation(fig, animate, frames=datas.shape[0], interval=100)
    plt.show()


font = ('times', 16, 'bold')
title = Label(main, text='Human Action Recognition from depth maps and Postures using Deep Learning')
title.config(bg='brown', fg='white')
title.config(font=font)
title.config(height=3, width=120)
title.place(x=0,y=5)


font1 = ('times', 13, 'bold
```

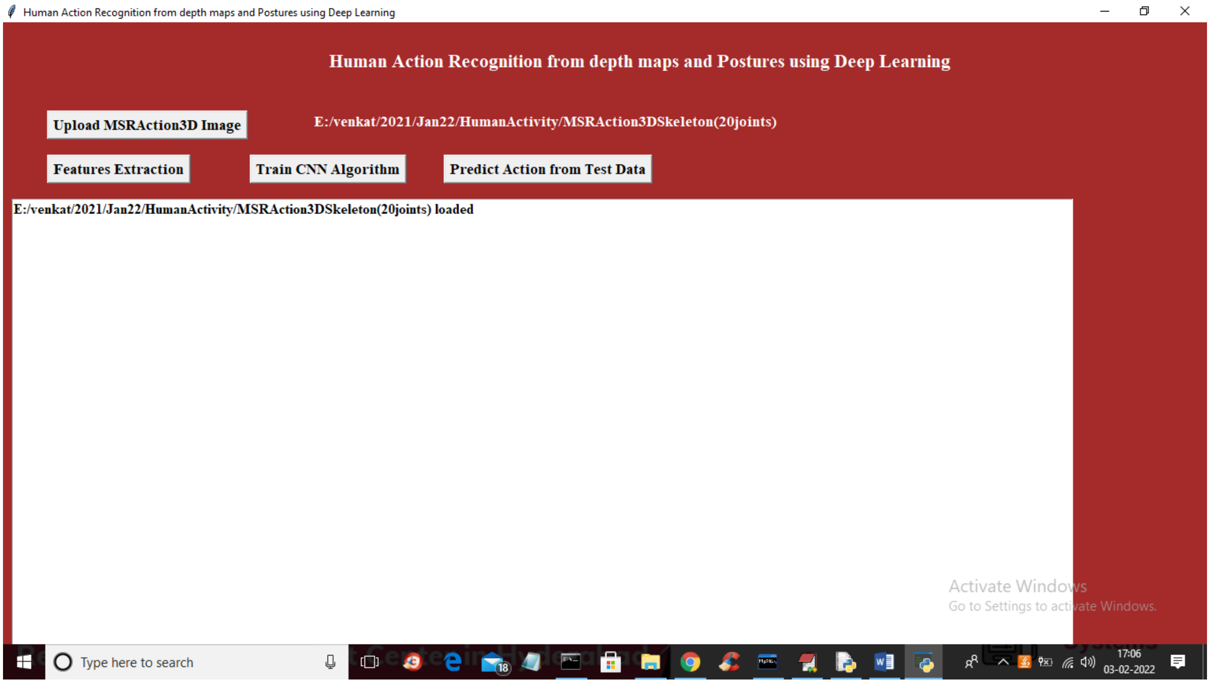
```
upload = Button(main, text="Upload MSRAction3D Image", command=upload)
upload.place(x=50,y=100)
upload.config(font=font1)

pathlabel = Label(main)
pathlabel.config(bg='brown', fg='white')
pathlabel.config(font=font1)
pathlabel.place(x=350,y=100)

feButton = Button(main, text="Features Extraction", command=featuresExtraction)
feButton.place(x=50,y=150)
feButton.config(font=font1)

cnnButton = Button(main, text="Train CNN Algorithm", command=trainCNN)
cnnButton.place(x=280,y=150)
cnnButton.config(font=font1)

predictButton = Button(main, text="Predict Action from Test Data", command=predictAction)
predictButton.place(x=500,y=150)
predictButton.config(font=font1)

font1 = ('times', 12, 'bold')
text=Text(main,height=30,width=150)
scroll=Scrollbar(text)
text.configure(yscrollcommand=scroll.set)
text.place(x=10,y=200)
text.config(font=font1)
main.config(bg='brown')
main.mainloop()
```



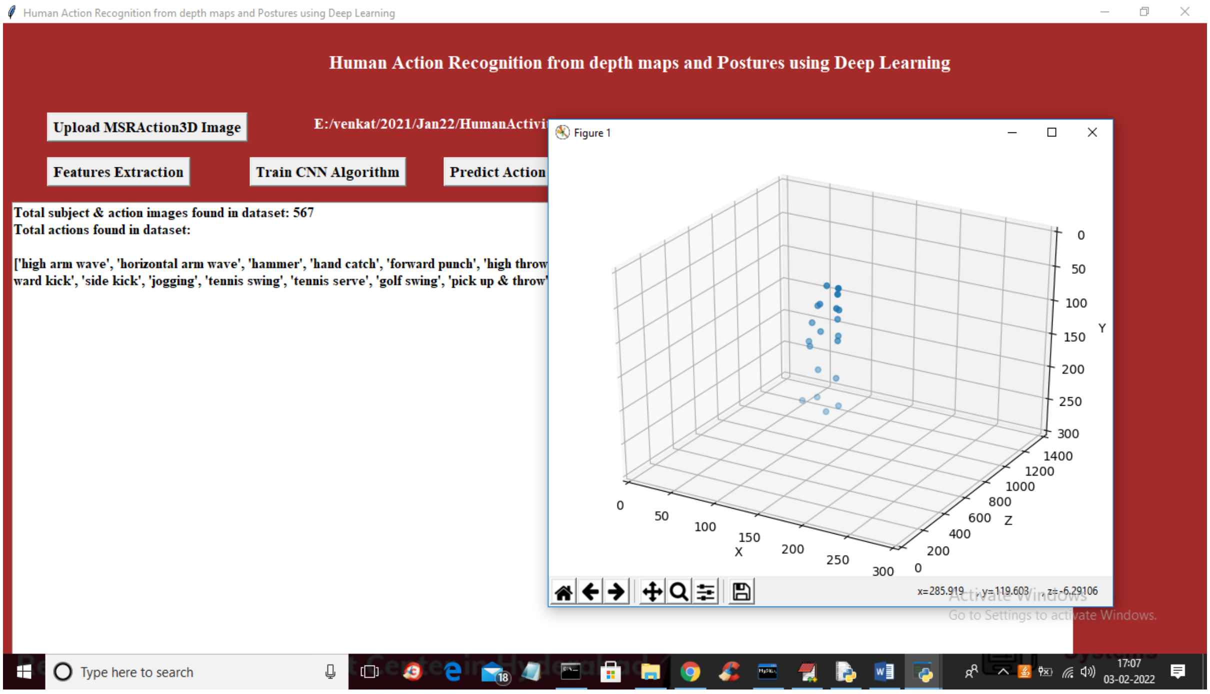
Screenshot 5.1: Upload MSRAction3D Image



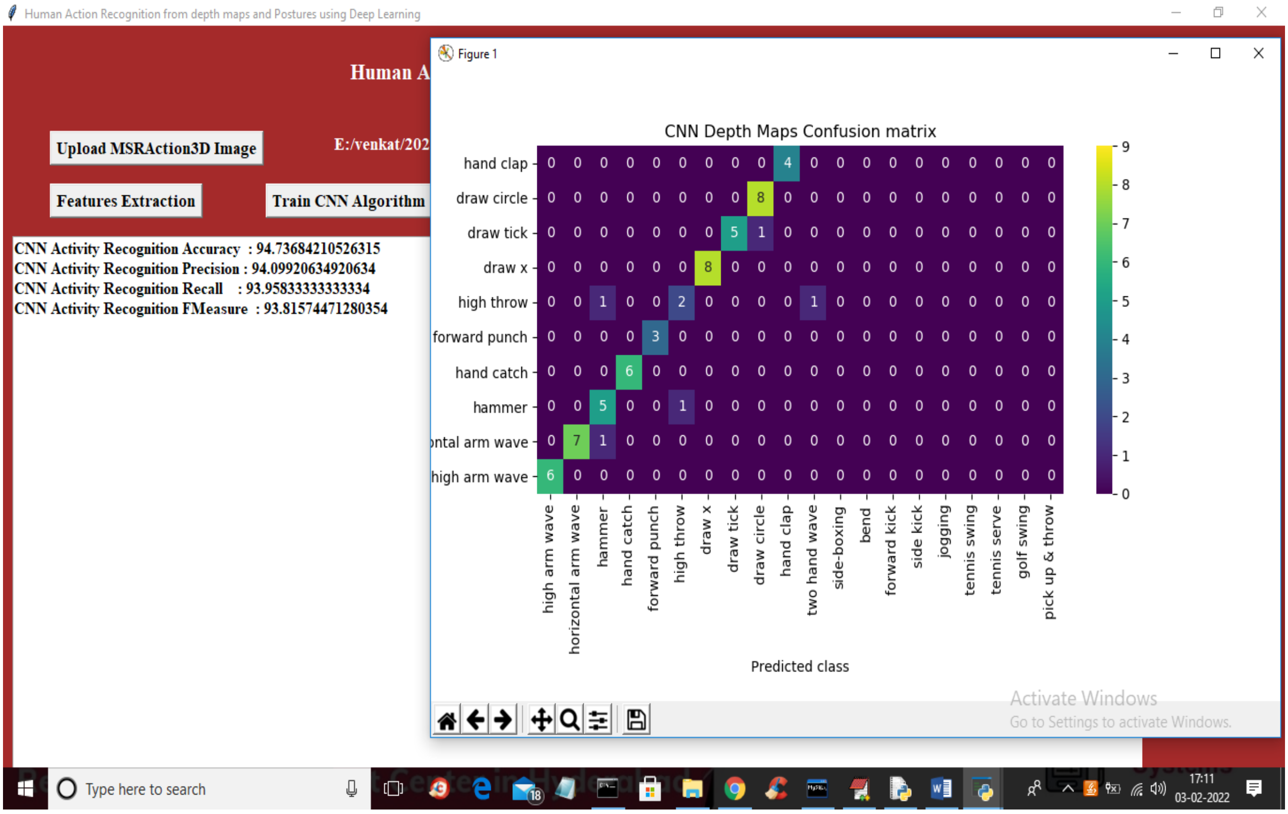
Screenshot 5.2: MSRACTION dataset



Screenshot 5.3: Features Extraction



Screenshot 5.4: Train CNN Algorithm



Screenshot 5.5: CNN Performance graph

6. TESTING

6.1 INTRODUCTION TO 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.

6.2 TYPES OF TESTING

6.2.1 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.

6.2.2.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.

6.2.3 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.

6.2.4 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.

6.2.5 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.

6.2.6 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.

6.4 TEST CASES

6.4.1 CLASSIFICATION

Test case ID	Test case name	Purpose	Input	Output
1	Action Recognition	To detect human action.	The user gives the input in the form of dataset.	An output is skeleton image is produced.
2	Action Recognition	To detect the human action.	The user gives input i.e., 14.txt file	An output is accuracy of recognition Using CNN.

7. CONCLUSION AND FUTURE SCOPE

7.1 PROJECT CONCLUSION

In conclusion, this project has successfully addressed the challenging task of human action recognition from depth maps and postures using deep learning techniques. Through the development of a sophisticated architecture, we have demonstrated the ability to leverage the complementary information provided by depth maps and postures, resulting in robust and accurate action recognition. The use of Convolutional Neural Networks (CNNs) for spatial feature extraction, Recurrent Neural Networks (RNNs) for temporal modeling, and fusion strategies has allowed us to effectively capture the nuances of human actions in three-dimensional space.

7.2 PROJECT FUTURE SCOPE

The future scope of the project on human action recognition from depth maps and postures using deep learning is exceptionally promising. It extends to fine-grained action recognition, real-time performance optimization, and multi-modal fusion for more comprehensive understanding of human actions. By delving into semi-supervised and self-supervised learning, the system can become more adaptable and data-efficient. Moreover, exploring adaptive human-computer interaction and cross-modal action recognition opens doors to transformative applications in fields like virtual reality and multimedia analysis. However, it's imperative to consider the ethical dimensions, especially concerning privacy, as the technology evolves. As view benchmark datasets and collaborative opportunities arise, this project's future lies in enhancing accuracy, versatility, and its positive impact on various human-centric domains

8. BIBLIOGRAPHY

8.1 REFERENCES

- J. Sun, X. Wu, S. Yan, L.-F. Cheong, T.-S. Chua, and J. Li, “Hierarchical spatio-temporal context modeling for action recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Miami, FL, USA, Jun. 2009, pp. 2004–2011.

- I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld, “Learning realistic human actions from movies,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Anchorage, AK, USA, Jun. 2008, pp. 1–8.

- W. Li, Z. Zhang, and Z. Liu, “Action recognition based on a bag of 3D points,” in Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Workshops, San Francisco, CA, USA, Jun. 2010, pp. 9–14.

8.2 GITHUB LINK

https://github.com/Biradanitesh/Human-Action-Recognition-from-depth-maps-and-Postures-using-Deep-Learning