MAJOR- 2 PROJECT

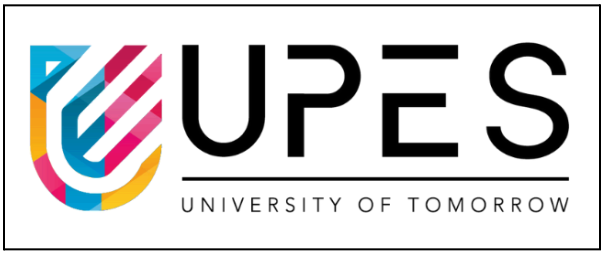
Report

For

“ VOLUME CONTROL USING GESTURES ”

Submitted By:

SPECIALIZATION	SAP ID	NAME
AIML	500087226	Kalpana Srivastava
AIML	500086694	Yachika Maheshwari
AIML	500083108	Yukti Panwar



Cluster of Artificial Intelligence
School of Computer Science
UNIVERSITY OF PETROLEUM AND ENERGY STUDIES
Dehradun - 248001, Uttarakhand

Dr. Ambika Aggarwal
Project Guide

Dr. Anil Kumar
Cluster Head

ABSTRACT

With technology constantly evolving, it's no surprise that how we interact with computers is changing too. Gesture recognition is a way for computers to understand hand movements as commands. It's becoming more popular because gestures are something everyone naturally understands. By using Human Computer Interaction(HCI) and deep learning technology, the project allows users to control their computer's system volume just by moving their hands in front of the camera, without needing any extra equipment. Instead of pressing buttons or using a remote, users can simply make hand signals to adjust the volume. Integration with PyAutoGUI and OpenCV makes it convenient for users to operate their computers from a distance, improving efficiency and making interaction smoother. In this project, four gestures are defined to control the system volume using hand gestures. The CNN-based model is trained on a dataset consisting of diverse hand gestures and environmental conditions associated with specific volume levels, enabling it to learn the underlying patterns and make accurate predictions in real-time. By focusing on feature extraction from the hand, our model successfully recognizes a diverse range of gestures with a remarkable accuracy of 95.00%. Through continued refinement and deployment, our model holds promise for empowering individuals with enhanced communication capabilities in real-world settings.

Keywords: Human-Computer Interaction(HCI), deep learning, PyAutoGUI, OpenCV, CNN

INTRODUCTION

Imagine a world where controlling the volume of your devices is as simple as a wave of your hand or a flick of your fingers. Picture yourself seamlessly adjusting the audio levels of your music player, television, or smart home system without reaching for a remote or tapping on a screen. This seemingly futuristic scenario is not just a fantasy—it's the reality shaped by the collaboration of numerous leading technology companies. Through the fusion of cutting-edge technology and human-computer interaction (HCI), companies like Intel Corporation, Apple Inc., Microsoft Corporation, Google Inc., Sony Corporation, and others are contributing to the advancement of gesture recognition technology [1].

The global Gesture Recognition market, valued at USD 16894.0 million in 2022 and expected to expand at a CAGR of 22.73 % during the forecast period, reaching USD 57730.68 million by 2028, reflects the growing demand for intuitive user interfaces. Gesture recognition, along with facial recognition, voice recognition, eye tracking, and lip movement recognition, are components of what developers refer to as a perceptual user interface (PUI). The goal of PUI is to enhance the efficiency and ease of use for the underlying logical design of a stored program, a design discipline known as usability. [1]

In this research paper, we got into the depth of gesture recognition, using Convolution Neural Network (CNN) to develop a responsive volume control system. By using Machine learning, we aim to increase, decrease and mute/Unmute the volume.

The use of CNNs enabled us to create a sophisticated model capable of learning and adapting to a wide range of gestures, regardless of variations in lighting conditions, background clutter or individual differences in gesture execution. By extensively training the model on diverse datasets, the model learns to generalize the patterns and effectively differentiate between different gestures with high accuracy.
Moreover, CNN facilitated real-time processing, allowing for seamless interaction between users and devices. [2]

The utilization of PyAutoGUI, a Python library for programmatically controlling the mouse and keyboard, complements our CNN-based approach by providing seamless integration with the operating system and enabling the execution of system-level commands triggered by recognized gestures. PyAutoGUI facilitates the translation of gesture inputs into corresponding actions, such as adjusting the volume settings or triggering specific functions within applications, thereby enabling comprehensive control over various aspects of device functionality. [2]

Imagine standing across the room, wanting to lower the volume of your music without disrupting your activities. With the CNN-based volume control system, a simple hand gesture becomes your

command. Whether it's a subtle gesture or a bold motion, the system interprets your movements with precision, translating them into actionable commands in real-time.

But why gestures?
Gestures are deeply ingrained in human communication—they are natural, instinctive, and effortless. By using this innate form of expression, we bridge the gap between humans and machines, creating a symbiotic relationship where interaction feels intuitive and seamless. [3]

In this research paper, we dug deep into gesture recognition, using Convolution Neural Networks (CNNs) to create a volume control system that responds quickly. Our goal is to make interaction with devices easier and more enjoyable by using machine learning. With this system, one can perform a bunch of actions like increasing and decreasing the volume, and mute and unmute by just moving the hands. We trained our model well, so it can understand even small hand movements and do exactly what you want with the volume. CNN makes it simple to control media without needing to touch anything. Whether you're pausing to answer a call, turning up the volume, avoiding messing your remote with your cooking hands or just skipping through the songs effortlessly.
By exploring gesture recognition, we're showing how cool and easy machine learning can be for making technology more user-friendly. We're excited about the possibilities it opens up for making interaction with devices smoother and more fun for everyone.

LITERATURE REVIEW

Abhishek B. et. al., [4] developed a system that integrates both spatial and temporal modelling for accurate gesture recognition from live webcam feeds or prerecorded videos. Initially, an adaptive algorithm fixes the user's skin colour based on lighting and camera parameters, followed by hand localization using histogram clustering. Subsequently, a machine learning algorithm detects gestures in consecutive frames, enabling seamless interaction with computer applications. The system comprises three subsystems: Hand and motion detection, utilizing OpenCV and TensorFlow Object Detector for capturing hand movements and performing edge and skin detection, which feeds into a 3D CNN trained on datasets for hand and gesture recognition. The CNN comprising input, hidden and output layers, undergoes training and verification for efficient recognition of gestures, facilitating various human-computer interactions like page-turning and zooming. Interaction with the computer is facilitated through PyAutoGUI. The model employs OpenCV and TensorFlow for hand detection and interpretation of gestures for computer actions such as page switching and scrolling.

Mrs. Akashya. et. al., [5] proposed a system that employs a Machine Learning algorithm, specifically Artificial Neural Networks (ANN), for recognizing hand gestures and facilitating communication for individuals with disabilities. Initially, the Video Processing module processes live video feeds, extracting clear images and adjusting parameters like brightness. Following this, the Feature Extraction module collects image data, smoothing out imperfections to ensure accurate gesture recognition. The Database Module stores hand movement data for training the ANN algorithm, allowing it to understand and predict gestures effectively. Once trained, the Prediction module utilizes the ANN algorithm to analyze hand movements and translate them into text and speech. Through this process, the system aims to bridge communication barriers, particularly for those who rely on sign language. In evaluation, the ANN model achieves an impressive average accuracy of 98.0%, indicating its robustness in recognizing and interpreting sign language gestures for seamless communication and interaction.

Rubèn E Nogales and Marco E Benalcàzar [6], proposed a research that addresses the pressing issue of hand gesture recognition, pivotal for communication and interaction with both people and machines. Focusing on high-dimensional pattern recognition, the study evaluates models utilizing both manual and automatic feature extraction techniques. Manual extraction employs statistical functions of central tendency, while automatic extraction employs Convolutional Neural Networks (CNN) and Bidirectional Long Short-Term Memory (BiLSTM). The best-performing model, BiLSTM-ANN, achieves an outstanding accuracy of 99.9912%. Evaluations also consider processing time, crucial for real-time applications. Manual extraction yields 92.936% accuracy with an Artificial Neural Network (ANN) classifier, while automatic extraction through CNN-ANN results in a remarkable 99.795% accuracy. The study further explores the BiLSTM model, achieving 95.6161% accuracy with Softmax classification and

99.9912% accuracy with ANN classification. Notably, both CNN-ANN and BiLSTM-ANN exhibit negligible differences in performance despite their architectural complexities. The study underscores the potential of deep learning models like CNN-ANN for portable hand recognition systems and highlights their real-time processing capabilities, crucial for practical deployment.

Anjali Patil and Shilendra Patil, [7] research presents a Hand Gesture Recognition (HGR) system aimed at enabling real-time interaction between users and the VLC media player, a software requiring physical gestures for commands like play, pause, and stop. Leveraging Convolutional Neural Networks (CNNs) for computer vision challenges, the study builds upon prior research and utilizes YOLO architecture for Region of Interest (ROI) detection. The ROI is fed into multiple models: a decision-based algorithm for smaller ROIs and Inception V3 architecture for larger ones. Transfer learning is employed due to limited datasets, with MobileNet 0.5 and 1.0 pre-trained models used for training. Additionally, edge detection is applied to create edge maps for further training. The testing phase involves YOLO for ROI detection, followed by probability calculation using the trained models for gesture recognition. System specifications include a Linux OS, GPU, and Tensorflow for model implementation. The testing dataset comprises 200 photos per gesture class, with Inception V3 achieving 92% accuracy and MobileNet 93%. The proposed two-stream architecture attains 100% accuracy on testing data, demonstrating robust real-time control of the VLC media player through hand gestures. The system's successful operation underscores its potential for user-friendly human-computer interaction, particularly in multimedia applications.

Noorain Zaidi et al., [8] explored the effectiveness of machine learning models in recognising hand gesture characters, including letters and numbers of different sizes, using sensor data from wrist-worn devices. The focus was on accelerometer and gyroscope sensors and analyzing various machine learning algorithms. The data collection involved novel methods with off-the-shelf wrist-worn sensor devices. A set of features was devised to ensure robust and accurate character detection. The Random Forest model achieved the highest accuracy of 90.40% when using data from both accelerometer and gyroscope sensors. However, when using only the accelerometer sensor the accuracy dropped to 82.51% for the Random Forest model. On the other hand, the Forward Neural Network (FNN) model achieved an average accuracy of 80.16% using only the gyroscope sensor. Despite the superior performance of the model utilizing data from both sensors, the evaluation demonstrates the feasibility of developing a machine-learning model using a single sensor to detect hand gesture characters with reasonable accuracy (>=80%).

Bin Ma et al., [9] research addressed communication barriers for deaf-mutes by developing a lightweight gesture detection algorithm based on pose estimation, enabling efficient recognition and translation of sign language. Initially, a small dataset of common hand gestures is compiled for training and testing the network. Optimization of the OpenPose network structure enhances its lightweight nature, followed by a pruning algorithm to compress the model. Post-Pruning, the

model size is reduced to 7MB, achieving an inference speed of 106fps with minimal accuracy loss. Fine-tuning after pruning further improves accuracy, surpassing the original. Deployed on AR9301 chip, hardware acceleration and quantization enhance performance, reaching a detection rate of 75fps with an average accuracy of 95.3%. Post-pruning, accuracy remains high at 94.4%, rising to 97% after fine-tuning, validating the proposed approach's effectiveness. This enables quantitative deployment on mobile terminals, resolving communication barriers effectively.

Deborah C. Attwood, [10] proposed a system iSign, that utilizes a Kinect Sensor for hand movement detection and capture, enabling in-air signature recognition without direct touch on any device. The captured video data is then transformed into feature templates. Various machine learning algorithms, including Bayesian network, Naïve Bayes, Decision Tree, and Random Forest, are evaluated for classifying hand gesture signatures. Empirical results demonstrate that Random Forest outperforms other algorithms with an accuracy of 93%.

Saurabh Adhikari et al., [11] for addressing challenges in gesture recognition using a Machine learning-based gesture recognition Framework (ML-GRF) is proposed. This framework utilizes machine learning algorithms to detect gesture sequences in data streams, reducing computing costs through similarity matching-based classification and efficient feature extraction. In simulations, ML-GRF achieved high accuracy (98.97%), precision (97.65%), recognition rate (98.04%), sensitivity (96.99%), and efficiency (95.12%). The methodology involves understanding hand anatomy, distinguishing between hand postures and gestures, and utilizing IoT-integrated sensors for real-time data transmission. Protocols like MQTT, HTTP, or CoAP facilitate data transmission to a central application for analysis. The data collection process involves multiple layers ensuring accuracy in time-series data. Training systems with IoT sensor data enable real-time gesture detection and algorithm evaluation using recorded sessions.

Hissah Almutairi et al., [12] presented a method for computer recognition of hand gestures using a color glove and a random forest classifier, achieving real-time prediction of hand movements with an average speed of 3 frames per second, irrespective of environmental variations. By integrating human-computer interaction (HCI) into the training process, they enhance prediction accuracy, particularly evident when using support vector machine (SVM) as the classifier, which reached a peak accuracy of 94.02%. The study delves into the complexity of hand gestures, distinguishing between static and dynamic movements and highlighting their relevance in various contexts, such as manufacturing lines.

A summary of Studies cited in the literature review:

Author	Method	Accuracy
Abhishek B. et. al., 2020	3D CNN	-
Mrs. Akashya. et. al., 2023	ANN	98%
Rubèn E Nogales and Marco E Benalcàzar, 2023	BiLSTM-ANN & CNN-ANN	99.9912% & 99.795%
Anjali Patil and Shilendra Patil,2022	CNN and YOLO	92%
Noorain Zaidi et al., 2022	Random Forest Classifier	90.40%
Bin Ma et al.,2022	OpenPose Network	94.4%
Deborah C. Attwood, 2023	Random Forest Classifier Sensor: Kinetic	93%
Saurabh Adhikari et al., 2023	ML Framework: ML-GRF	98.97%
Hissah Almutairi et al., 2022	Random Forest Classifier and SVM. Sensor: Color Glove	94.02%

MODEL OVERVIEW

The project is about controlling video playback functions like mute, Increasing /decreasing volume using hand gestures: Fist, Thumbs-up, and Thumbs-down, as shown in Fig 1.

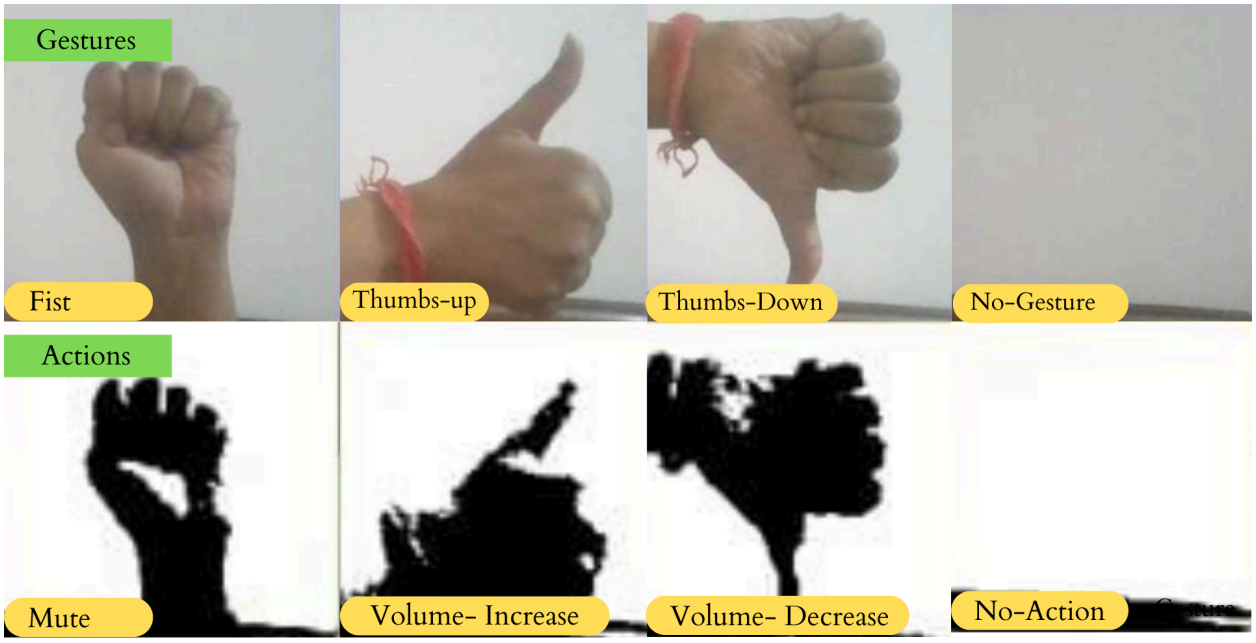


Fig. 1 (Gestures & Actions)

We started by capturing images of these gestures to create our own dataset. These images were then pre-processed by converting them to grayscale, performing data augmentation and resizing them to a standard size.

Next we trained a Convolutional Neural Network (CNN) model using this dataset. We employed ReLU and Softmax activation functions and experimented with different numbers of layers to improve the model's ability to recognize and differentiate between gestures.

After training, we used the model to predict gestures by performing corresponding actions such as increasing/ decreasing volume, mute, etc
Furthermore, we analyzed the result by checking the accuracy and other factors using the training dataset to evaluate the performance of the model.

IMPLEMENTATION

The Model architecture is as follows (Fig. 2):

Importing the Libraries and Modules:
Importing required libraries and modules including OpenCV, numpy, os, PIL, matplotlib, Keras, pyautogui, etc.

Creating Directories:
To organize the data for training and testing were created. These directories hold the image corresponding to different hand gestures.

User Input for Mode Selection:
Users get the option to select the mode of operation: “train” or “test”. This selection determines whether the system captures the training dataset or test dataset for model training and testing as shown in Fig. 3.

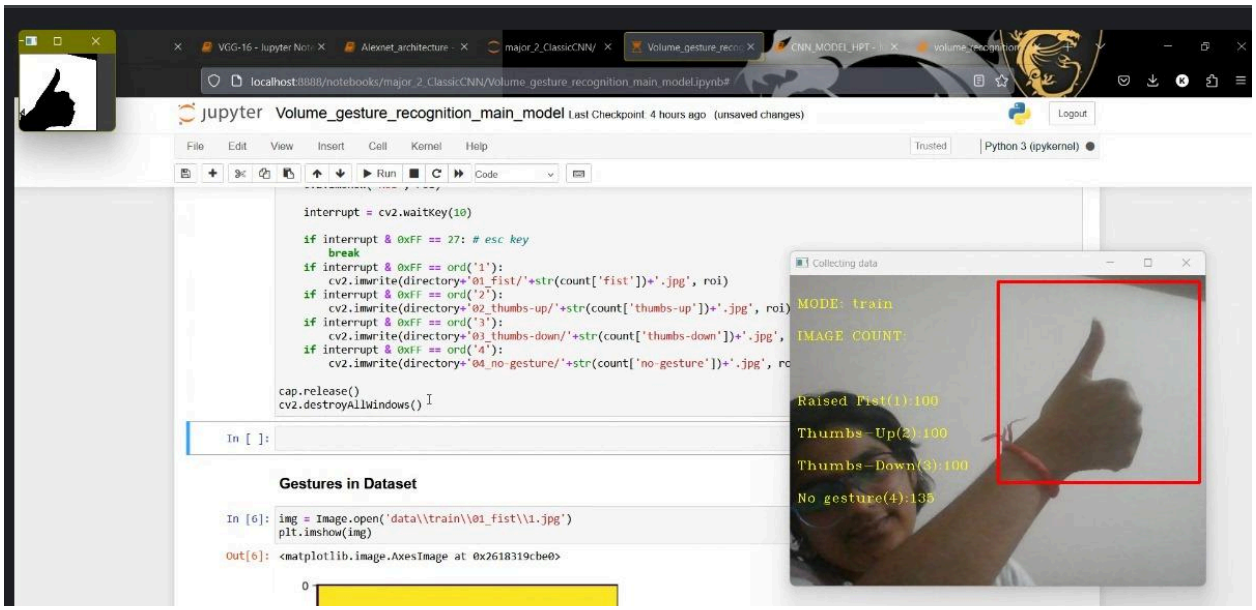


Fig. 3 (capturing images)

Building the CNN Model:
CNN Sequential model architecture is used, comprising Convolution layers, max-pooling layers, Flatten layers and Fully connected layers as shown in Fig. 4. Further, the model is compiled with the appropriate optimizer, loss function and evaluation metrics.

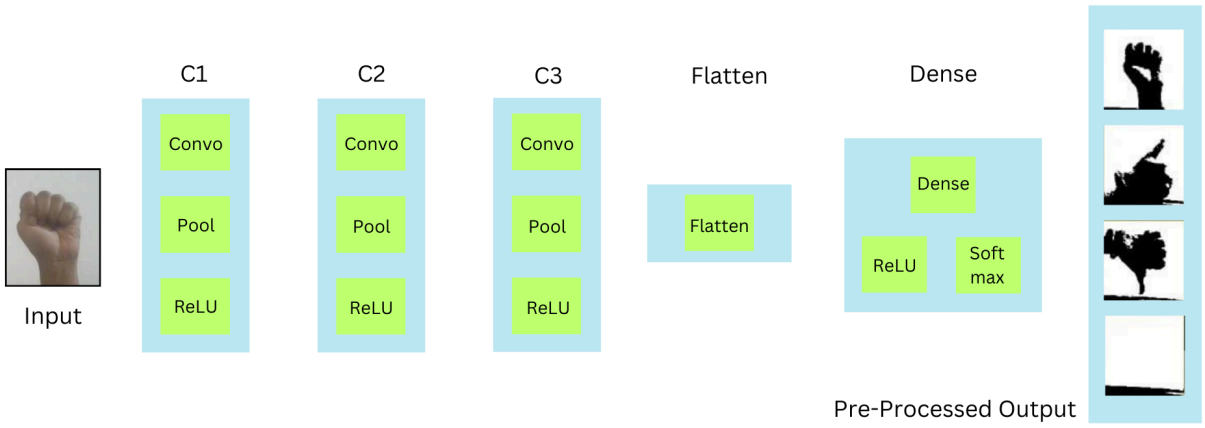


Fig. 4 (CNN Architecture)

Data Augmentation and Pre-processing:
Training data is augmented to artificially expand the diversity of training data using ImageDataGenerator. Augmentation techniques such as rescaling, shearing and zooming are applied to enhance the robustness of the model.

Training the Model:
The model is trained on the augmented training data. Training parameters such as epoch and validation steps are specified to monitor the training progress and performance.

Saving the model:
Upon completion of training, the trained model architecture is saved as a JSON file, while the model weights are saved in HDF5 format. This enables the model to be reused for future gesture recognition tasks without the need for training over and over again.

Real-time gesture recognition:
In the real-time gesture recognition phase, video frames from the webcam are continuously captured and processed. The region of interest (ROI) containing the hand gesture is extracted from each frame and preprocessed before being fed into the trained CNN model. The model predicts the gesture in real-time, and corresponding actions, such as adjusting volume or muting, are performed based on the recognized gesture as shown in Fig. 5.

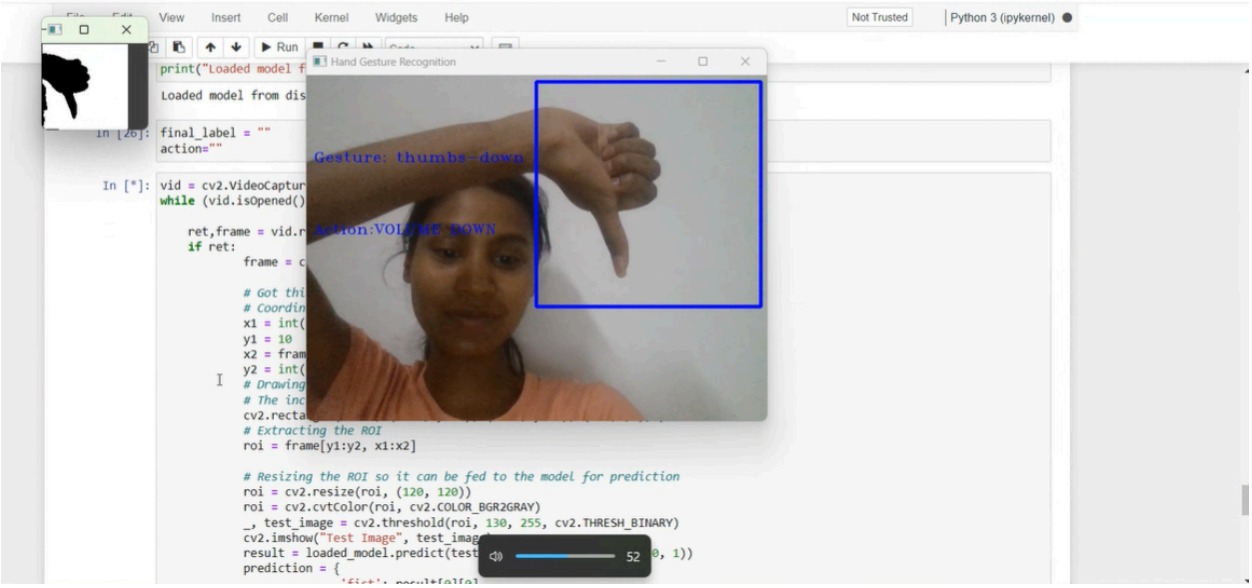


Fig. 5 (Real-time Gesture recognition)

Termination:

The system continues real-time gesture recognition until the user exits the loop by pressing the 'q' key. Resources such as the webcam feed are released, ensuring a clean termination of the application.

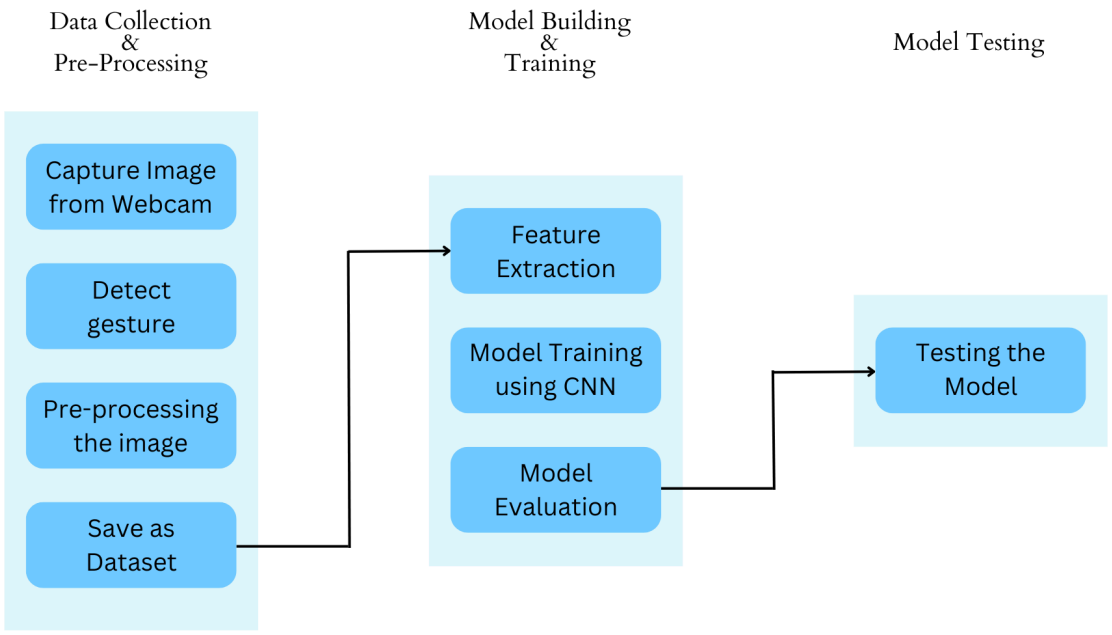


Fig. 2 (Model Architecture)

The table shows the different CNN models with respective accuracy:

CNN MODEL	ACCURACY
Layers: 5 No. of Filters: (32, 64, 128) Kerne Size: (3, 3) Optimizer: Adam Loss Function: Categorical Crossentropy	≈ 95.00%
AlexNet Layers: 8 Filter Size: (3, 3) No. of Filters: (96, 256, 384, 384, 256) Optimizer: Stochastic Gradient Descent Loss Function: Categorical Crossentropy	98.98%
VGG 16 Layers: 16 No. of Filters: (64, 64, 128, 128, 256, 256, 256, 512, 512, 512, 512, 512, 512) Kernel Size: (3, 3) Optimizer: Rmsprop Loss Function: Categorical Crossentropy	26.44%

CONVOLUTIONAL NEURAL NETWORK (CNN)

Convolutional Neural Networks (CNNs) as shown in Fig. 4, is a type of deep learning model specifically designed for processing structured grid-like data, such as images. They have gained widespread popularity in computer vision tasks due to their ability to automatically learn hierarchical representations of features directly from raw data.

Convolutional Layer:
The convolutional Layer is the fundamental building block of CNNs. It applies filters (also known as kernels) to the input image. Each filter extracts different features from the input, such as edges, textures, or patterns, by performing a convolution operation. The output of this operation is a feature map. Also, the Convolution Layer uses the Relu Activation Function. [13]

Pooling Layer:
Pooling layers are used to reduce the dimension of the feature maps produced by the convolutional layer. There are three types of Pooling: Min Pooling, Max Pooling and Average Pooling. The most common pooling operation is Max pooling, which extracts the maximum value from a set of values from the region. This downsampling operation helps reduce the computational complexity of the network and makes the learned features more robust to variations in translation and distortion. [13]

Flatten Layer:
Flatten layer converts the 2D feature map obtained from the convolution and pooling layer into a 1D vector.

Fully Connected Layer:
In this layer, each neuron is connected to every neuron in the previous layer.

Output Layer:
The output layer of a CNN is responsible for producing the final predictions or classifications. The number of neurons in this layer depends on the task at hand. For example, in image classification tasks, the output layer typically has one neuron per class, and the predicted class is determined by taking the sigmoid/ softmax of the output values.

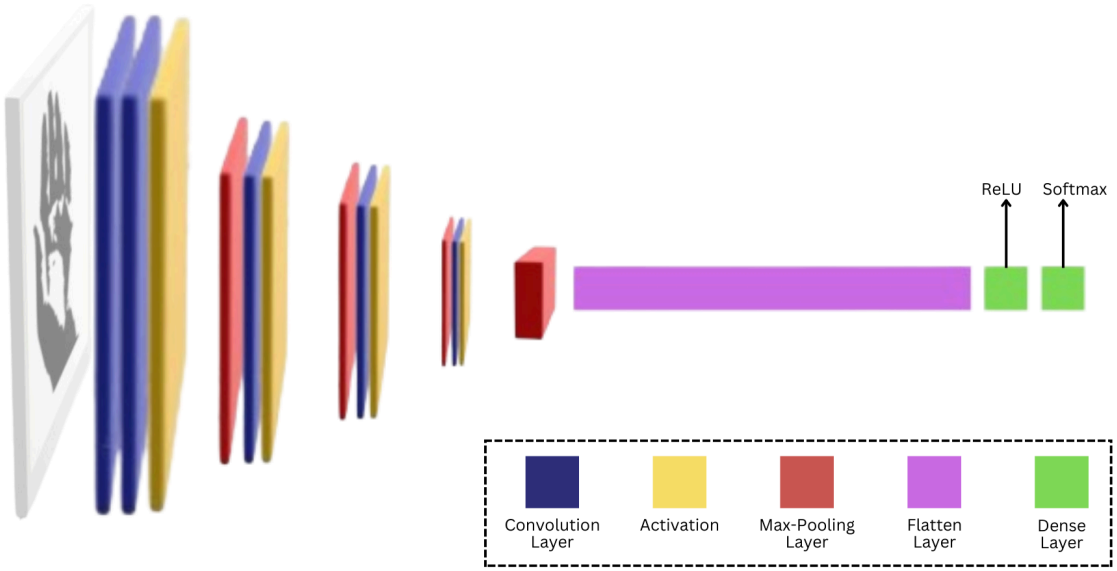


Fig. 4 (Convolution Neural Network (CNN))

CONCLUSION

The development of a volume detection system using hand gestures presents a significant leap forward in human-computer interaction. Through the utilization of deep learning techniques, we've crafted a system that offers users a seamless and instinctive means of controlling electronic devices. Our project emerged from recognizing the limitations of traditional volume control methods, driving us to explore innovative alternatives. By harnessing deep learning's prowess, we trained a model to accurately interpret hand gestures and correlate them with specific volume adjustments. Through rigorous experimentation, our CNN model demonstrates remarkable accuracy in detecting and recognizing hand gestures, even in challenging environments. As we continue exploring gesture recognition, we envision further advancements in human-computer interaction and accessibility technology. Our project represents a significant step towards a future where seamless communication between humans and machines is the norm. We are excited about the transformative potential of gesture recognition in making interactions with technology more intuitive and enjoyable.

REFERENCES

[1] Gesture Recognition Market Trends: Size 2024, Latest Developments, and Forecast till 2031

[2] Nagalapuram, G. D., Roopashree, S., Varshashree, D., Dheeraj, D., & Nazareth, D. J. (2021). Controlling Media Player with Hand Gestures using Convolutional Neural Network.

[3] Arun, N., V, N., Dutta, A., B, S., Dept. of Electronics and Telecommunication Engineering, & B.M.S. College of Engineering. (2022). Hand Gesture Recognition and Volume Control

[4] Abhishek. B. J., Krishi, K., Meghana, M., Daaniyaal, M., & Anupama, H. S. (2020). Hand gesture recognition using machine learning algorithms.

[5] Akashya, C., Ragupathy, V., Kuberaselvan, D., Muthamizhselvan, M., & Praveen, N. (2023). Hand gesture communication using deep learning and ANN algorithm.

[6] Nogales, R., & Benalcázar, M. E. (2023). Hand Gesture Recognition Using Automatic Feature Extraction and Deep Learning Algorithms with Memory

[7] Patil, A., & Patil, S. (2022). Hand Gesture Recognition System for controlling VLC Media Player based on two stream transfer learning.

[8] Zaidi, N., Kumari, P., Rajasegarar, S., & Karmakar, C. (2022). Machine Learning Aided Minimal Sensor based Hand Gesture Character Recognition.

[9] Ma, B., Lv, X., & Hu, Y. (2022). A hand gesture detection algorithm and quantization implementation.

[10] Deborah C. Attwood (2023). Hand Gesture Signature Recognition with Machine Learning Algorithms

[11] Adhikari, S., Gangopadhayay, T. K., Pal, S., Akila, D., Humayun, M., Alfayad, M., & Jhanjhi, N. Z. (2023). A novel machine Learning–Based hand gesture recognition using HCI on IoT assisted cloud platform

[12] Al-Juboori, S. A. M., Almutairi, H., Almajed, R., Ibrahim, A., & Gheni, H. M. (2022). Detection of hand gestures with human computer recognition by using support vector machine.

[13] What are Convolutional Neural Networks? | IBM.