International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

A Multimodal Architecture with Visual-Level Framework for Virtual Assistant

Shree Varshan V[1], Gayathri Rajakumaran[2*], Shola Usharani[3], Rajiv Vincent[4]

Submitted: 21/11/2022 **Revised:** 22/01/2023 **Accepted:** 20/02/2023

Abstract: Virtual personal assistants (VPAs) have grown in popularity due to their scalability and accessibility. The development of machine learning (ML) and natural language processing (NLP) have changed how we use technology and the growth of digital assistant technology. VPAs uses artificial intelligence to personalize, simplify, and automate user tasks and use computer vision to recognize visual cues. Consequently, increasing their versatility and functionality across other inputs. The proposed architecture for the VPA is based on two main components that will work in tandem: a chat-like model that can accept text or speech input to help the user and a computer vision framework that consists of multiple levels to enable control and interaction with the computer using visual input. The VPA utilizes NLP and ML algorithms to read and interpret user queries in both audio and text formats. Based on the classification and context of the query, automated actions are taken by the VPA to improve operational efficiency. The innovative computer vision framework lets the user to control the computer without a mouse or keyboard via gestures and level to represent action in the environment. Eventually, improving user experience by adding convenience and effectiveness. Automated tasks include opening particular apps, creating weather reports based on the location, altering screen brightness and system volume, virtual mouse operation, and manage launched applications such as minimize, maximize, or audio-input with hand gestures. Thus, VPAs save time, boost productivity, and enhance use of technology at home and in the workplace.

Keywords: *Interactive virtual personal assistant, LSTM hand-landmark Gesture recognition, LSTM intent classification chat bot, Multimodal input, System automation, Visual level framework.*

1. Introduction

Humans interacting with computers has become simpler and more effortless with the innovation of virtual personal assistants (VPAs) or virtual assistants [1][2]. The development of virtual assistant technology has made a variety of jobs and services easier to perform. These software agents or programs can help people and organizations to easily manage their daily duties and tasks, such as making appointments, sending emails, creating a workspace, maintaining social media accounts, and more. VPAs operate based on commands or queries and have significant advantages in terms of convenience by providing a quick and easy approach to gather information for work or other purposes. VPAs also save people's time by doing repetitive activities and exclude the necessity to keep track of multiple pieces of information, allowing people to be highly productive and effective like to concentrate on more crucial issues. People with disabilities or impairments may perform jobs that they may have struggled with previously using the support of these assistants [3]. For instance, people with visual impairments can use virtual assistant to handle their emails and social media accounts, book appointments, and carry out other duties independently.

The expansion of virtual assistants over the past decade has been genuinely astounding. Grand View Research reports that the value of virtual assistants in the global market is expected to grow at a compound annual growth rate (CAGR) of 24.3 percent by 2030, exceeding USD 14.10 billion [4]. The accuracy of virtual assistants in understanding the user's queries has significantly improved due to advancements in the ML and NLP of AI. The availability of the internet has made it simpler for users to connect to digital assistants to utilize from wherever and at any moment [5]. Developers can create and implement virtual assistants into their websites or applications through the accessibility of cloud and APIs services to accomplish tasks from a range of platforms and devices, including PCs, smartphones, and tablets. The popularity of smart devices like smart speakers and IoT (Internet of Things) devices has contributed to the popularity of digital assistants in homes, workplaces, and public areas. The development of online platforms has also been a major factor in adopting digital assistants to deliver

[1] *Student, Vellore Institute of Technology, Chennai, India.*
ORCID ID: 0009-0009-4627-5572
[2] *Assistant Professor Sr., Vellore Institute of Technology, Chennai, India.*
ORCID ID: 0000-0001-8995-1221
[3] *Associate Professor, Vellore Institute of Technology, Chennai, India.*
ORCID ID: 0000-0002-7480-7777
[4] *Assistant Professor Sr., Vellore Institute of Technology, Chennai, India.*
ORCID ID: 0000-0002-4012-6383

** Corresponding Author Email: gayathricse87@gmail.com*

more effective and customized services. Several companies use these assistants as chat bots to handle customer care, support, and sales [6][7]. As a result, consumer satisfaction has improved, revenues have climbed, and corporate costs have decreased. Greater benefits and functions for virtual assistants are expected as they continue to advance in many industries [8]. A few popular examples of virtual assistants available to users are Google Assistant, Apple's Siri, and Amazon's Alexa.

The steady improvements in the field of image processing over the past decade, such as the breakthrough invention in highly accurate deep learning algorithms and the availability of large data that were collected are used to recognize and understand complex visual data like emotions, facial expressions, objects, hand gestures, and body pose. The integration of computer vision has further enhanced virtual assistant’s capabilities and is expected to keep developing to becomee a part of everyday life. Google Lens, Microsoft Lens, and Bixby Vision are examples of the most popular and latest products available to the public, which utilizes images to process by providing quick information about the image, finding similar objects in the image, searching the image source, image to text function, and much more, making daily tasks and activities more convenient and efficient.

However, the current interaction with smart assistants is limited to using voice and text as inputs. The existing research has only been implemented until recognizing gestures, opening applications or performing task and returning to peripherals. The proposed VPA is a multimodal input program that aims to bridge the gap between existing smart assistants and user’s needs by allowing them to use computer vision as a hands-free input along with voice and text inputs.

2. Related works

G Iannizzotto *et al.* designed a prototype of a real-time interactive virtual assistant with Raspberry PI3 which consists few services. The graphical frontend with a red fox as an interactive character offers users to interact seamlessly by wake word detection using SnowBoy software and PocketSphinx module, then a face detection by Haar module for security protection followed by an NLP-based Smart assistant module Mycroft, to understand the question and response using Flite, a speech synthesis module to synchronize the character at run time graphically [9]. S Kumari *et al.* made a desktop application using PyAutoGUI that can perform tasks based on hand gestures and speech. The speech recognition is performed by Google speech recognition and hand gesture recognition is accomplished by the CNN model and exported to the Keras model [10]. Haria *et al.* the real-time feed follows a set of processes: noise removal, image smoothening, thresholding, contour extraction, convex hull, and convexity defects. Finally, the Haar Cascade classifier is used to detect gestures and open corresponding desktop applications [11]. Equivalent series of processes are followed in P Xu *et al.* on image sizes of 128*128, 64*64, and 32*32 but the CNN was implemented to play games, and control the robot in ROS using hand gestures [12]. Rautaray *et al.* developed a real-time multiple hand gesture recognition system for enhancing human-computer interaction. [13]. M Panwar *et al.* the real-time feed is enhanced and segmented into an image of size 320*240 and then the orientation is detected. Further centroid, peaks, Euclidean distance, and thumb detection are extracted from features. Finally classifies the hand gesture based on the features and outputs the corresponding alphabet letter to a file [14]. N R Impana and G R Manjula. made a desktop application that allows users by face recognition then the user can give tasks either as text by keyboard or voice that is converted into text by Google Speech API. Further, it searches for command words and executes the corresponding task to fetch the solution by API requests. The response is retrieved as system-readable files or JSON files from Google search [15]. V Geetha *et al.* did a python program in Microsoft Visual Code IDE that is fully functional by voice and assists day-to-day tasks using API requests and getting the response in JSON format. Speech recognition is achieved by pyttsx3 and then uses Google's speech recognition Engine to complete the queried task [16]. A Pandey *et al.* created a python program that uses Google speech recognition for speech-to-text conversion then the speech output is used in context extraction by methods of NLP, identifies the speech as a command or API call, and finally executes them [17]. A similar proposed system was found in A sakharkar *et al* [18]. S Sarda *et al.* developed a personal assistant that accepts command via speech making it user friendly. Addition of new commands and remote access feature is different from previous works mentioned [19]. K V Kulhali *et al.* created an offline voice recognition engine PARI that can be primarily used by blind persons. It can be used in android phones for basic functionalities like opening/closing apps, calls, SMS, and many others. Also, it can access the computer from the phone [20]. K Belattar *et al.* designed a drone prototype with Raspberry PI that can interact with a human by detecting gestures using YOLO and SSD mobileNetv2. Many IoT devices can be interacted with by gestures using this embedded system-based model [21].

3. Methodology

The proposed architecture of the virtual personal assistant (VPA) consists of three modules: Input module, Detection module and Execution module as in below Fig.

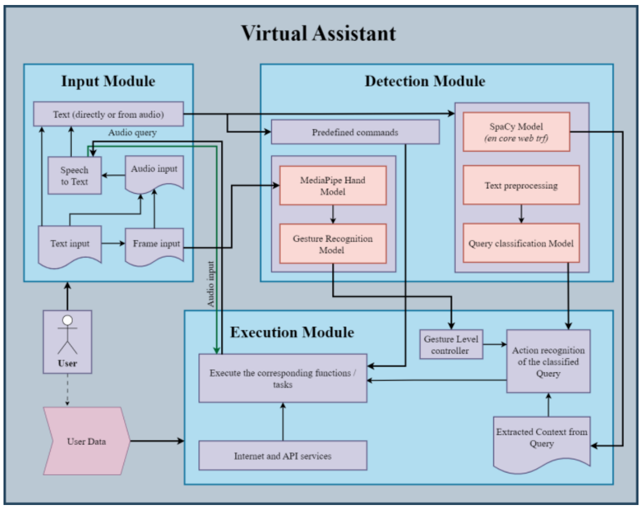

Virtual Assistant
Input Module
Text (directly or from audio)
Audio query
Speech to Text
Audio input
Text input
Frame input
Detection Module
Predefined commands
MediaPipe Hand Model
Gesture Recognition Model
SpaCy Model (en core web trf)
Text preprocessing
Query classification Model
Audio input
User
User Data
Execution Module
Execute the corresponding functions / tasks
Gesture Level controller
Action recognition of the classified Query
Internet and API services
Extracted Context from Query

Fig. 1. The proposed architecture of the VPA.

3.1. Input Module

The input module plays an important role in the VPA as it receives and processes the input data, which can be in text, audio, or image frame format. The processed input data can be used for further computation by the models. Parallel processing by threads enables the VPA to handle multiple input sources simultaneously and allowing the users to interact with the VPA in various ways. Text is the primary mode of input through which users can directly enter text. There are two types of text input: Predefined commands that directly execute the function when called and Text queries serve as requests from user. The audio function is called to collect audio as input from the computer's microphone using the PyAudio package. The collected audio chunk relies on a Google Speech Recognition python package to convert audio to text. Then the input query from audio or text is processed contextually to identify important words which are subsequently utilized to perform the action associated with the user's intent. The video input from the device camera allows users to interact with the system using hand gestures. Image frames are processed to identify gestures from a subset of frames continuously in real time, ensuring accurate classification even during transition hand movements. The identified gestures along with level representing the working environment is used to perform the action.

3.2. Detection Module

The detection module comprises three key components: Spacy *en_web_core_tf* pretrained model, query classification and gesture recognition. This section provides an overview of the detection module's architecture and functionality.

3.2.1. Spacy *en_web_core_tf* pretrained Model

The traditional neural network or LSTM models for intent-based chat bots can struggle with understanding context because they often rely solely on the word or sentence similarity to make predictions. This is because these models typically represent language as fixed-size vectors, which are then compared to other vectors to determine similarity. Suppose a user's query contains a new word that is not present in the training corpus data. In that case, its score is typically zero, and when the sentence similarity is calculated, which can result in incorrect classification. To improve the query's accuracy in such situations, the pretrained Spacy model was incorporated into the VPA to recognize all nouns and adjective using POS tagging and NER entities such as PERSON, GPE, and WORK_OF_ART, which are used to extract the context of the query like name of the person, location, video and product with adjective [22].

Spacy is an open-source Python module for NLP that provides a simple and easily operatable interface for handling text data. The advantage of using Spacy is that it includes some pretrained models designed to do various NLP tasks. The "en_core_web_trf" model is one such pretrained model based on transformer architecture that can handle English text data as it can process full sentences or paragraphs at once rather than individual words. The "en_core_web_trf" model has achieved state-of-the-art results on several NLP benchmarks, demonstrating its high level of performance. Thus the pretrained "en_core_web_trf" model does not need any additional training to process text data. Because of this, it is an ideal choice for any project that requires quick and accurate results on massive volumes of English text data.

3.2.2. Query Classification

The Query Classification model uses a deep learning architecture called Long Short-Term Memory (LSTM) model, which are particularly suited for processing data sequences such as words in sentences [23]. During training, the model is trained on a set of labeled queries, where each query is assigned to a specific intent.

The dataset used for training and testing was in JSON format customized for the VPA. Each intent was represented as an object with key-value pairs. The keys included "tag" to denote the intent or class name, "patterns" for a list of possible queries or utterances belonging to that intent, and "responses" for a list of potential chat bot responses associated with that intent. The following example illustrates the dataset's structure:

{“tag": "greeting", "patterns": ["Hi", "Hello", "Hey"], "responses": ["Hello!", "Hi there!"]}

During the preprocessing process, all letters in the training data are converted to lowercase, and any punctuation is removed as it reduces the size of the input space and makes it easier for the model to learn patterns in the data. Then the input sequences are converted into sequences of integers, and padding is added to ensure all training data have the same length and encode the category labels as integers.

The architecture of the LSTM model is as follows: The pretrained word embeddings, GloVe, is used as an embedding layer to the model that converts input text data into a vector representation of the fixed size. Then a Convolutional 1D layer is used to extract local features from the sequence, followed by a Dropout layer that drops out some of the neurons in the previous layer during training to prevent overfitting. Then the LSTM layer is used to process the sequence data in a recurrent manner and outputs a sequence of hidden states followed by another LSTM layer with a different configuration, resulting in a sequence of hidden states with different dimensionality. Now a Dense layer fully connects with the hidden states to a higher-dimensional space. Then a Dropout layer and finally, a Dense layer that maps the higher-dimensional representation to the final output space.

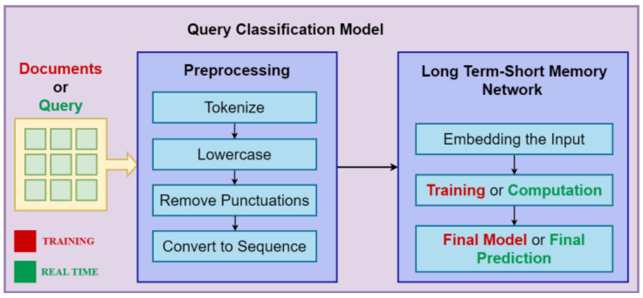


Fig. 2. The query classification system.

Now the preprocessed data is ready to pass into the LSTM model for training. With the total size of the training corpus and the number of output classes, we configure the model's architecture and compile the model. Once the training and fine-tuning processes are complete, the model can be evaluated on unseen data during production and make real-time predictions on new input sequences.

3.2.3. Gesture recognition

The MediaPipe hand model combined with LSTM is used for hand gesture recognition. The Mediapipe Hand model uses a Convolutional Neural Network (CNN) architecture to process the input image frames as they are well-suited for image processing and classification tasks due to their ability to automatically learn and extract meaningful features from raw pixel data [24]. The hand model consists of multiple layers of convolutional and pooling operations, which are used to extract increasingly abstract and higher-level features from the input data. The extracted features are then passed through a series of fully connected layers, which make the final classification. The Mediapipe Hand model is trained on large datasets of hand images and videos, which enables it to generalize well to new, unseen data and to work in real-time. The model uses the concept of land mark detection, where specific points or land marks on the hands are detected and tracked throughout the image sequence even when it is partially occluded or in motion.

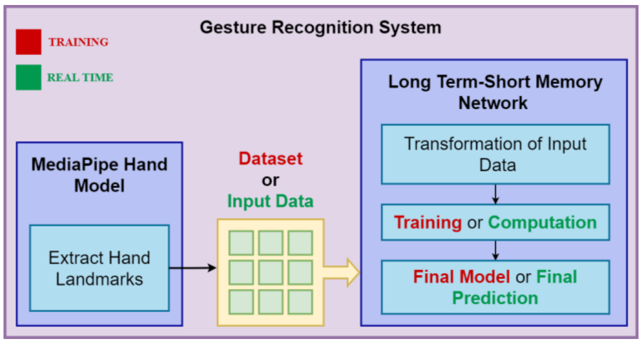

Gesture Recognition System
TRAINING
REAL TIME
MediaPipe Hand Model
Extract Hand Landmarks
Dataset or Input Data
Long Term-Short Memory Network
Transformation of Input Data
Training or Computation
Final Model or Final Prediction

Fig. 3. The gesture recognition system.

The dataset used for training the LSTM model for gesture recognition comprised of hand landmarks that were extracted using the MediaPipe hand model and stored in a CSV file format. Each row in the CSV file represents a single frame, with the hand landmarks for that frame listed in the columns. The final column of the CSV file contains the label for the corresponding gesture, such as "thumbs up" or "peace sign". The model architecture incorporates two LSTM layers for sequential data processing, dropout layers for regularization, and dense layers for classification. After the training process was complete, the model was converted into the TensorFlow Lite (tflite) format, which is a compressed format optimized for use on mobile and embedded devices. The MediaPipe hand model processes image frames to extract hand landmarks, which are then fed into the LSTM model for classification in real-time.

3.3. Execution module

The execution module is responsible for executing appropriate actions, providing accurate responses, and ensuring the fulfillment of user requests. In order to achieve this, the module utilizes a trained query classification model based on a dataset containing 21 intents. Among these intents, 7 are non-action intents, while the remaining 16 are specifically related to assistance tasks. New intents can be added to the dataset and the model can be trained accordingly for the corresponding added actions to be executed by VPA. The module utilizes stored user data, encompassing information such as known contact details, user location, and software data on the user's computer, to deliver personalized assistance. Furthermore, the module integrates with API to access services like weather report and AI Tools like ChatGPT or ChatSonic. The table below provides a summary of the assistance provided.

Table 1. Summary of VPA Assistance.

INTENT	ACTION	CONTEXT
About	Provide information about the VPA.	None
ChatAI	Request a query to be sent to ChatGPT or ChatSonic using	AI-Tool: ChatSonic or

	API.	ChatGPT
Close	Closes the detected application.	Application name
Commend	Responds with kind words.	None
Email	Open Gmail in a new browser tab and, if the recognized person is in the contact book, include their email ID.	Person name, entity: PERSON
Farewell	Responds with words of gratitude.	None
Greeting	Greets the user in return.	None
Help	Explains usage and capabilities of the VPA.	None
Message	Sends a WhatsApp message to the detected recipient.	Person name, entity: PERSON
Music	Detects the Music's name and open in YouTube Music website.	Nouns, entity: WORK_OF_ART
News	Opens specified news in Google News.	Nouns, adjectives
No	Response back with neutral response.	None
Notes	Write important notes in a text file.	None
Open	Launches the detected application.	Application name
Service	Search the query in browser.	None
Shopping	Detects the product name and open few online shopping websites.	Noun, adjectives
System	Executes system commands.	None
Time	Responses the time and date.	None
Video	Plays a YouTube video based on the query.	Nouns, entity: WORK_OF_ART
Weather	Detects the location and responses with weather report from API	Entity: GPE
Yes	Response back with neutral response.	None

The visual level framework comprises three levels: Home, Application, and System. Unlike previous works that typically assign a single functionality to one gestures, this framework introduces a novel approach of reusing gestures, reducing the total number of distinct gestures required for different actions. The new framework enhances scalability, adaptability, and minimizes user memorization for all actions.

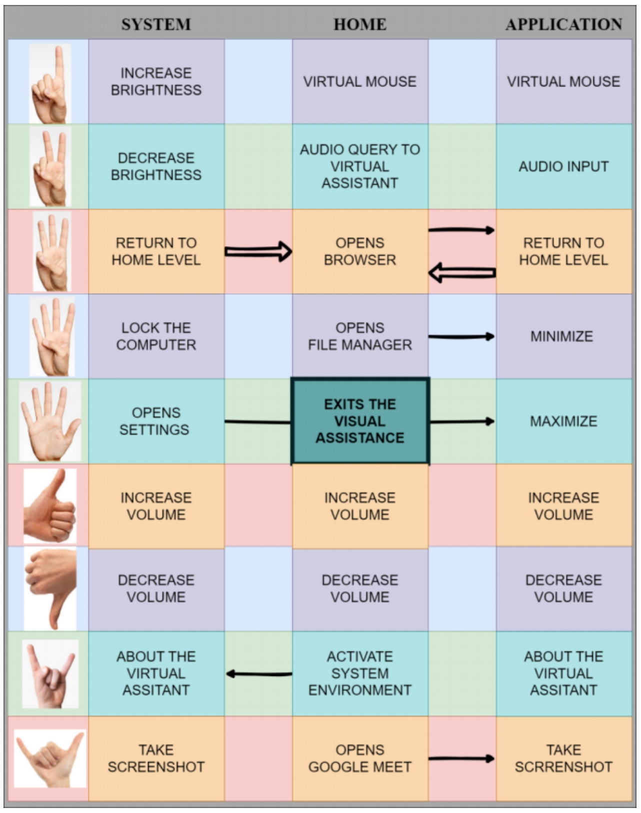


Fig. 4. Visual level framework.

The default level is Home, the environment that users arrive at when the visual support is activated. Users can utilize the virtual mouse to navigate across the system, use speech to avail aid from VPA. Users can also select and open one of the three preconfigured applications, switching to the Application level. The Application level provides advanced management features, including options to minimize, maximize the application window, take screenshots and seamlessly control the application interface using the Virtual Mouse. Furthermore, the framework allows users to input text into fields via audio, eliminating the need for manual typing. From the Home level, users can access the System level to control system-wide settings like adjusting brightness levels, managing audio volume, performing system locking actions, and accessing system settings such as network configurations, accessibility options, and display preferences using virtual mouse in application level. Users can easily switch between applications, facilitated by intuitive gestures and a streamlined interface, ensuring smooth transitions and minimizing the time required to navigate between different applications as shown in above Fig.

4. Results

The virtual personal assistant (VPA) was successfully developed using the proposed multimodal architecture with pioneered computer vision framework. The evaluation yielded promising outcomes in terms of user experience and convenience as user can interact with the computer in hands-free using computer vision and speech. The chat-like model underwent a data-centric approach for fine-tuning the dataset. The modifications continued until the model achieved satisfactory real-time performance in production environments. Notably, the Spacy model accurately extracted contextual information from user queries, which were crucial for task execution. The gesture recognition LSTM model exhibited an impressive accuracy of 95% on the test dataset, the confusion matrix and recall curve are given below:

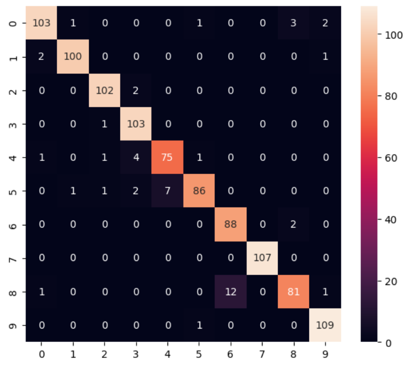


Fig. 5. The confusion matrix for gesture recognition.

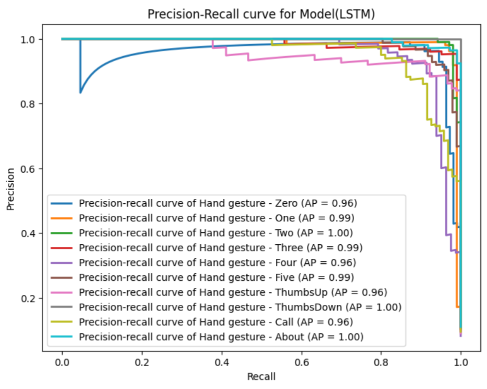


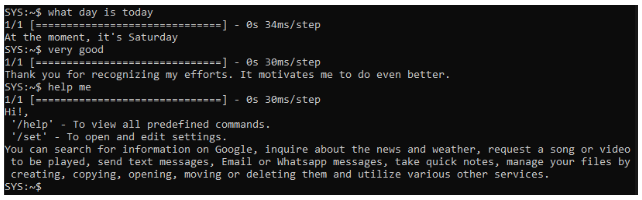
Fig. 6. The precision-recall curve for gesture recognition.

The high-performance of the gesture recognition LSTM model enabled smooth execution of the VPA without the need for real-time cross-checking by the user i.e., the image frames from the camera were sufficient for accurate classification, eliminating the requirement to display the processed image back to the user. Thus, reducing the usage of computer resources. Users have the option to view the processed image in live (live feed) or examine the last 750 processed frames (last ~1 minutes) stored from visual assistance for debugging.

Table 2. Resource consumption of proposed VPA.

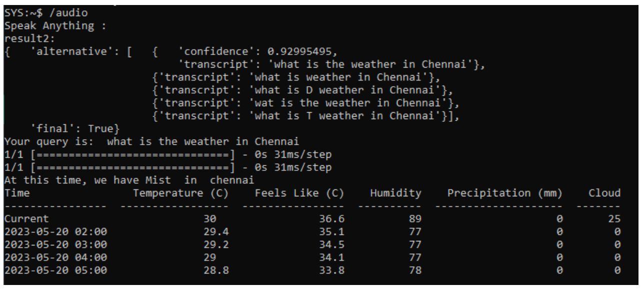
Scenario	**Resource Consumption**	
	CPU	**RAM**
Computer Idle (VPA not started)	10%	40%
Initial: VPA started	14% (+4)	47% (+7)
VPA for text or audio	16% (+6)	47% (+7)
VPA only visual assistance	43% (+33)	55% (+15)
VPA with visual live feed.	48% (+38)	47% (+7)
VPA only to show last 2 minutes (at end).	31% (+21)	47% (+7)
CPU used: Intel i5-6200U ;6th gen 2 cores 4 threads **RAM used:** 8GB 2133 MHz		

As depicted in the above table. The computer resource consumption is relatively low for the proposed VPA to at most of +38% of CPU and +15% of RAM -due to storing 750 frames. Displaying these stored frames reduces the RAM to initial state of VPA. 'VPA with visual live feed' where CPU is maximum of +38% but +7% RAM since we do not store frames. Here are a few sample outputs of the VPA.



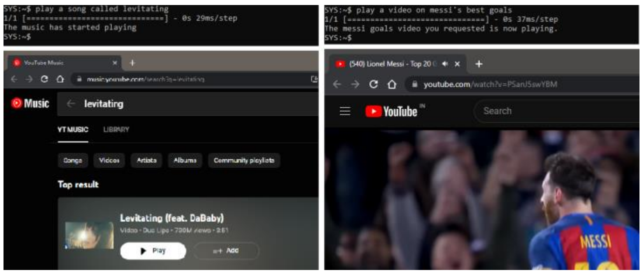
```
SYS:~$ what day is today
1/1 [==============================] - 0s 34ms/step
At the moment, it's Saturday
SYS:~$ very good
1/1 [==============================] - 0s 30ms/step
Thank you for recognizing my efforts. It motivates me to do even better.
SYS:~$ help me
1/1 [==============================] - 0s 30ms/step
Hi!,
 '/help' - To view all predefined commands.
 '/set' - To open and edit settings.
You can search for information on Google, inquire about the news and weather, request a song or video
 to be played, send text messages, Email or Whatsapp messages, take quick notes, manage your files by
 creating, copying, opening, moving or deleting them and utilize various other services.
SYS:~$
```

Fig. 7. Demonstrates conversational responses by text input.



```
SYS:~$ /audio
Speak Anything :
result2:
{   'alternative': [   {   'confidence': 0.92995495,
                           'transcript': 'what is the weather in Chennai'},
                       {'transcript': 'what is weather in Chennai'},
                       {'transcript': 'what is D weather in Chennai'},
                       {'transcript': 'wat is the weather in Chennai'},
                       {'transcript': 'what is T weather in Chennai'}],
    'final': True}
Your query is:  what is the weather in Chennai
1/1 [==============================] - 0s 31ms/step
1/1 [==============================] - 0s 31ms/step
At this time, we have Mist  in  chennai
Time                Temperature (C)    Feels Like (C)    Humidity    Precipitation (mm)    Cloud
----------------  -----------------  ----------------  ----------  --------------------  -------
Current                          30              36.6          89                     0       25
2023-05-20 02:00               29.4              35.1          77                     0        0
2023-05-20 03:00               29.2              34.5          77                     0        0
2023-05-20 04:00                 29              34.1          77                     0        0
2023-05-20 05:00               28.8              33.8          78                     0        0
```

Fig. 8. Demonstrates the speech input function and use of API services.

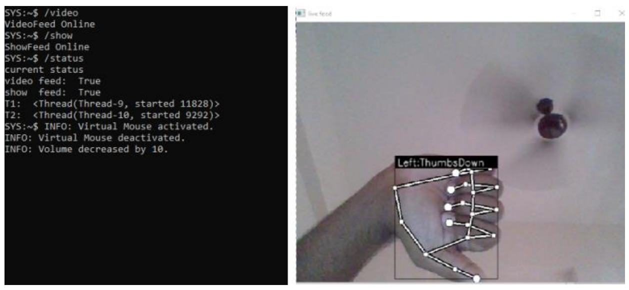


```
SYS:~$ play a song called levitating
1/1 [==============================] - 0s 29ms/step
The music has started playing
SYS:~$
```

```
SYS:~$ play a video on messi's best goals
1/1 [==============================] - 0s 37ms/step
The messi goals video you requested is now playing.
SYS:~$
```

Fig. 9. Few more examples.

The VPA uses a chat-like interface as shown in Fig 7. Fig 8 shows the execution of predefined command *'/audio'*, to get input from audio and convert it to text and uses weather API to get and display the weather report. The VPA plays song and video based on the context of query, identified using Spacy model in Fig 9.



```
SYS:~$ /video
VideoFeed Online
SYS:~$ /show
ShowFeed Online
SYS:~$ /status
current status
video feed:  True
show  feed:  True
T1:  <Thread(Thread-9, started 11828)>
T2:  <Thread(Thread-10, started 9292)>
SYS:~$ INFO: Virtual Mouse activated.
INFO: Virtual Mouse deactivated.
INFO: Volume decreased by 10.
```

Fig. 10. Controlling the system using computer vision.

Predefined commands: '/video' - to enable visual assistance and '/show' - to show the stored frames (both are called together for a live feed) are shown in Fig 10 along with the gesture 'thumbs down' - to decrease volume by 10. In Fig 11, the virtual mouse can move the cursor, perform left-click and close it virtually along with the output directory of the VPA, where the screenshots and the notes taken by the user are stored. Fig 12 displays a log of actions performed from operating the virtual mouse to terminating the visual assistance. During the interaction, the audio-input is used to search 'IPL score' in the launched browser. The file explorer was launched and a screenshot was taken. Finally terminating the visual assistance using gesture.

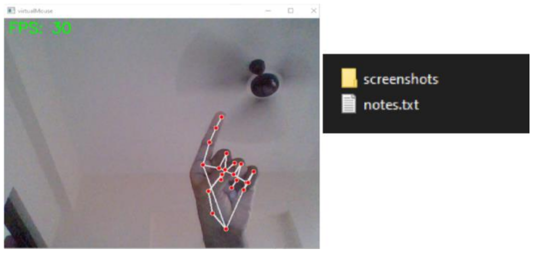
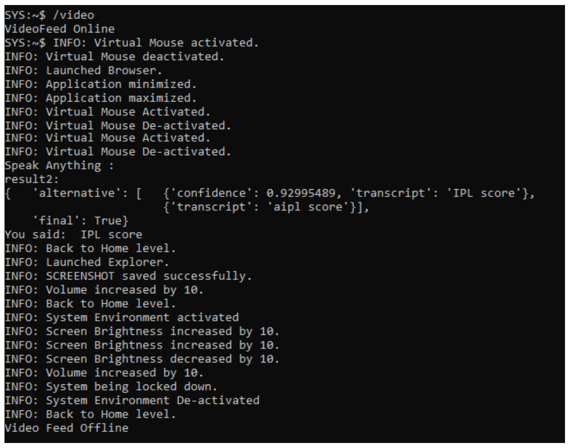


Fig.11. Virtual mouse using computer vision and Outputs of VPA.



```
SYS:~$ /video
VideoFeed Online
SYS:~$ INFO: Virtual Mouse activated.
INFO: Virtual Mouse deactivated.
INFO: Launched Browser.
INFO: Application minimized.
INFO: Application maximized.
INFO: Virtual Mouse Activated.
INFO: Virtual Mouse De-activated.
INFO: Virtual Mouse Activated.
INFO: Virtual Mouse De-activated.
Speak Anything :
result2:
{   'alternative': [   {'confidence': 0.92995489, 'transcript': 'IPL score'},
                       {'transcript': 'aipl score'}],
    'final': True}
You said:  IPL score
INFO: Back to Home level.
INFO: Launched Explorer.
INFO: SCREENSHOT saved successfully.
INFO: Volume increased by 10.
INFO: Back to Home level.
INFO: System Environment activated
INFO: Screen Brightness increased by 10.
INFO: Screen Brightness increased by 10.
INFO: Screen Brightness decreased by 10.
INFO: Volume increased by 10.
INFO: System being locked down.
INFO: System Environment De-activated
INFO: Back to Home level.
Video Feed Offline
```

Fig. 12. Log of vision-level framework.

It is important to note that all of the aforementioned actions were executed without the need to revert back to external peripherals, such as a physical mouse or keyboard as in Fig 4.

5. Conclusion

The research presented in this work introduces a comprehensive architecture for a virtual personal assistant (VPA) designed to facilitate multi-modal interactions which has the potential to revolutionize the way users interact and enhance their overall experience with these assistants. The use of NLP methods like Parts-Of-Speech tagging and named entity recognition on the input query has made the VPA to understand the user's request contextually to execute the task. The innovative visual-level framework provides a more engaging and personalized user experience to interact and control the application and the computer. The framework utilizes a unified set of gestures for different applications and systems promoting consistency and ease of use. This framework can find practical implementation in various domains, including mobile device interfaces, smart home control, virtual reality, and augmented reality. The ability to seamlessly integrate multi-modal interactions opens doors for innovative use cases and improved user engagement across different contexts. Moreover, the VPA holds promise for improving accessibility features, making interactions more intuitive for individuals with disabilities. The VPA architecture holds significant potential for a wide range of applications and can be deployed in various domains, including smart homes, healthcare, customer service, and education.

Conflicts of interest

The authors declare no conflicts of interest.

References

[1] Dubiel, M., Halvey, M., & Azzopardi, L. (2018). A survey investigating usage of virtual personal assistants. arXiv preprint arXiv:1807.04606.

[2] Almansor, E. H., & Hussain, F. K. (2020). Survey on intelligent chatbots: State-of-the-art and future research directions. In Complex, Intelligent, and Software Intensive Systems: Proceedings of the 13th International Conference on Complex, Intelligent, and Software Intensive Systems (CISIS-2019) (pp. 534-543). Springer International Publishing.

[3] Samim, A. (2023). A New Paradigm of Artificial Intelligence to Disabilities. International Journal of Science and Research (IJSR), 12(1), 478-482.

[4] Grand View Research. (2021). Intelligent Virtual Assistant Market Size, Share & Trend Analysis Report By Technology (Text-to-Speech, Text-based), By Product (Chatbot, Smart Speaker), By Application (IT & Telecom, Consumer Electronics), And Segment Forecasts, 2023 - 2030 (online). Retrieved from https://www.grandviewresearch.com/industry-analysis/intelligent-virtual-assistant-industry.

[5] Budzinski, O., Noskova, V., & Zhang, X. (2019). The brave new world of digital personal assistants: Benefits and challenges from an economic perspective. NETNOMICS: Economic Research and Electronic Networking, 20, 177-194.

[6] Franken, S., & Wattenberg, M. (2019, October). The impact of AI on employment and organisation in the industrial working environment of the future. In ECIAIR 2019 European Conference on the Impact of Artificial Intelligence and Robotics (Vol. 31). Academic Conferences and publishing limited.

[7] Barwal, R. K. ., Raheja, N. ., Bhiyana, M. ., & Rani, D. . (2023). Machine Learning-Based Hybrid Recommendation (SVOF-KNN) Model For Breast Cancer Coimbra Dataset Diagnosis. International Journal on Recent and Innovation Trends in Computing and Communication, 11(1s), 23–42. https://doi.org/10.17762/ijritcc.v11i1s.5991

[8] Majumder, S., & Mondal, A. (2021). Are chatbots really useful for human resource management?. International Journal of Speech Technology, 1-9.

[9] de Barcelos Silva, A., Gomes, M. M., da Costa, C. A., da Rosa Righi, R., Barbosa, J. L. V., Pessin, G., ... & Federizzi, G. (2020). Intelligent personal assistants: A systematic literature review. Expert Systems with Applications, 147, 113193.

[10] Iannizzotto, G., Bello, L. L., Nucita, A., & Grasso, G. M. (2018, July). A vision and speech enabled, customizable, virtual assistant for smart environments. In 2018 11th International Conference on Human System Interaction (HSI) (pp. 50-56). IEEE.

[11] Kumari, S., Mathesul, S., Shrivastav, P., & Rambhad, A. (2020). Hand gesture-based recognition for interactive human computer using tenser-flow. International Journal of Advanced Science and Technology, 29(7), 14186-14197.

[12] Haria, A., Subramanian, A., Asokkumar, N., Poddar, S., & Nayak, J. S. (2017). Hand gesture recognition for human computer interaction. Procedia computer science, 115, 367-374.

[13] Xu, P. (2017). A real-time hand gesture recognition and human-computer interaction system. arXiv preprint arXiv:1704.07296.

[14] Rautaray, S. S., & Agrawal, A. (2012). Real time multiple hand gesture recognition system for human computer interaction. International Journal of Intelligent Systems and Applications, 4(5), 56-64.

[15] Panwar, M., & Mehra, P. S. (2011, November). Hand gesture recognition for human computer interaction. In 2011 International Conference on Image Information Processing (pp. 1-7). IEEE.

[16] Impana, N. R., & Manjula, G. R. VOICE AND TEXT BASED VIRTUAL PERSONAL ASSISTANT FOR DESKTOP.

[17] Morzelona, R. (2021). Human Visual System Quality Assessment in The Images Using the IQA Model Integrated with Automated Machine Learning Model . Machine Learning Applications in Engineering Education and Management, 1(1), 13–18. Retrieved from http://yashikajournals.com/index.php/mlaeem/article/view/5

[18] Geetha, V., Gomathy, C. K., Vardhan, K. M. S., & Kumar, N. P. (2021). The voice enabled personal assistant for Pc using python. International Journal of Engineering and Advanced Technology, 10, 162-165.

[19] Pandey, A., Vashist, V., Tiwari, P., Sikka, S., & Makkar, P. (2020). Smart voice based virtual personal assistants with artificial intelligence. Artificial Computational Research Society, 1(3).

[20] Sakharkar, A., Tondawalkar, S., Thombare, P., & Sonawane, R. (2021). Python based AI assistant for computer. In Conference on advances in communication and control systems (CAC2S).

[21] Sarda, S., Shah, Y., Das, M., Saibewar, N., & Patil, S. (2017). VPA: Virtual Personal Assistant. International Journal of Computer Applications, 165(1).

[22] Kulhalli, K. V., Sirbi, K., & Patankar, M. A. J. (2017). Personal assistant with voice recognition intelligence. International Journal of Engineering Research and Technology, 10(1), 416-419.

[23] Belattar, K., Mehadjbia, A., Bala, A., & Kechida, A. (2022). An embedded system-based hand-gesture

recognition for human-drone interaction. International Journal of Embedded Systems, 15(4), 333-343.

[24] Nigam, A., Sahare, P., & Pandya, K. (2019, January). Intent detection and slots prompt in a closed-domain chatbot. In 2019 IEEE 13th international conference on semantic computing (ICSC) (pp. 340-343). IEEE.

[25] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural computation, 9(8), 1735-1780.

[26] Zhang, F., Bazarevsky, V., Vakunov, A., Tkachenka, A., Sung, G., Chang, C. L., & Grundmann, M. (2020). Mediapipe hands: On-device real-time hand tracking. arXiv preprint arXiv:2006.10214.