Received Date: 9th April, 2024
Revision Date:6th May, 2024
Accepted Date: 9th July, 2024

Nepali Virtual AI Assistant

Sajag Majhi [1*], Shubham Kaphle [2], Shyam Dahal [3]

[1] Dept. of Computer Engineering, Kathmandu Engineering College. E-mail: sajagmajhi666@gmail.com
[2] Dept. of Computer Engineering, Kathmandu Engineering College. E-mail: shubhamkaphle3@gmail.com
[3] Assoc. Professor, Dept. of Computer Engineering, Kathmandu Engineering College. E-mail: shyam.dahal@kecktm.edu.np

***Abstract*— With the development of Artificial Intelligence (AI), much of our regular redundant work that does not require critical thinking is replaced by machines. The scope of the Internet of Things (IoT) is also increasing due to AI. IoT acts as a suitable platform for the application of AI in real life. In this paper, we present a Virtual AI Assistant that combines AI with IoT. Our system uses an ESP8266 (Node MCU) chip as the core microcontroller for the home automation application. It is an IoT-compatible microcontroller with a cloud connectivity feature that is accomplished with the help of WiFi (Wireless fidelity). This allows home appliances such as lights, fans, rice cookers, etc. to be controlled by desktop or mobile applications from any part of the world. Our system uses speech recognition as its primary source for controlling the application. The speech recognition feature can detect and distinguish people's voices which is a handy feature in terms of security and privacy. Other than IoT, software features such as note-making, application initiators, web navigation, and automatic typing through speech recognition are also implemented in this system. Our work also makes use of the Nepali language for IoT applications for switching home appliances on and off. The AI model was trained from scratch manually for the Nepali language as proper audio datasets for the Nepali language are not available. Our work in the development of Nepali Virtual AI Assistant is to integrate hardware and software with AI to create user-friendly automation and ease in people's daily lives.**

Keywords – AI, IoT, Speech recognition, Node MCU, WiFi

Introduction

In this paper, the cloud connectivity feature of *'Node MCU'* has been used for communication between a user and household appliances such as lights, fans, etc. It allows the user to be connected to the microcontroller system from any part of the world, provided that both *'Node MCU'* and the user's device i.e., desktop or mobile are connected to the internet. This allows the user to send data to the microprocessor wirelessly. There are two ways of sending data to the microcontroller from the user in our system, manual button control and speech recognition input. Manual button control within the mobile or desktop app can be used especially when the user is present in a noisy environment because of the susceptible interference in speech recognition input from noise. Speech recognition input can be given by the user for ease of control provided that the user is in a less noisy environment.

Coming to software applications, our system can be implemented on both desktop and mobile as mentioned before. Features such as note-making are accomplished through speech recognition input from the user. The note-making section in the application stores the audio input taken from the user in an organized manner. Note-making is one of the most demanded features in mobile phones as in the modern era people don't carry paper and pens regularly with them. Our work digitalizes and provides ease in note-making as people always have mobile phones with them most of the time nowadays.

Nepali Virtual AI Assistant also implements a web navigation feature. It is described as being able to surf through the internet such as Wikipedia or googling something through speech recognition input from the user. It also involves the opening of YouTube, Facebook, and other internet platforms through speech recognition input. This involves complete elimination of typing to surf the internet which can save time and increase ease of use.

Another in-app feature of our work is the application initiator. It is described as opening desktop applications such as *File Explorer, MS Word, Microsoft Edge,* etc. Similar to web navigation and note-making this feature also implements speech recognition input from the user for performing the above-mentioned actions. This is especially more convenient for desktops rather than for phones because opening applications on mobile phones is already easier.

Related Works

a voice-controlled robot using ATMEGA328P (Arduino UNO) as a core microcontroller is described in paper [1]. In paper [2] the authors have presented a way to control a vehicle through voice input from a user using Arduino. The authors focus on the different neural network-related methods that can be used for speech recognition and compare their advantages and disadvantages and the conclusion is given on the most suitable method in paper [3]. An alternative viewpoint on the possibilities of ASR systems is presented in paper [4] where the researchers demonstrated the

* *Corresponding Author*

effectiveness of deep learning in improving the performance of virtual assistants by introducing an end-to-end trainable speech recognition system that surpassed existing models. The development of AI-based virtual assistants is thoroughly covered in paper [5] where the assessment highlights the problems with modern virtual assistants, particularly their need for internet access.

Problem Statement

In the fast-paced, digitally-driven era, the need for effective and tailored user experiences is continuously growing. A solution that enables users to interact with software programs naturally and effortlessly is in high demand. This brings to the fore the role of Virtual AI Assistants, which aim to revolutionize user interactions through voice recognition capabilities as opposed to conventional typing and manual navigation-based requests.

However, the development of a comprehensive voice-activated virtual assistant presents unique challenges. The heterogeneity of speech patterns, dialects, and languages makes voice recognition and accurate interpretation of voice commands for various tasks a complex undertaking. Current AI assistants often struggle with understanding complex or ambiguous instructions, recognizing multiple languages, and handling various accents. This sometimes leads to the misinterpretation of user instructions.

Furthermore, they often lack contextual understanding, resulting in irrelevant responses and struggles with words that have multiple meanings. The functionality of such assistants can also be hindered due to hardware limitations, and privacy concerns are prevalent due to the necessary access to personal data for optimal operation.

Addressing these issues, our work is to offer an improved user experience and operational efficiency and aim to seamlessly integrate voice recognition, natural language processing, application control, data storage, and Application programming interface (API) integration.

This paper employs advanced AI algorithms for better contextual understanding. Ensuring data privacy and security will be a key priority. Lastly, the system's design will consider hardware limitations, ensuring maximum compatibility and flexibility in the execution of commands. Our work thus presents a solution to the existing challenges and limitations in the field of AI voice assistants.

Methodology

i. IoT

It refers to an interconnected network of various devices where communication between the devices is performed for the exchange of data between the devices. Sensors are the backbone of IoT. For simplicity, we have not implemented the use of sensors in our work. Sensors can also be used in home automation applications. For e.g., the Water level of the water tank in the home can be detected by an ultrasonic sensor and programmed accordingly for the automatic water filling process. Another example includes maintaining the temperature of certain rooms of the house to the desired value by detecting the surrounding temperature by thermistors and thermocouples and programming accordingly for automatic temperature adjustment.

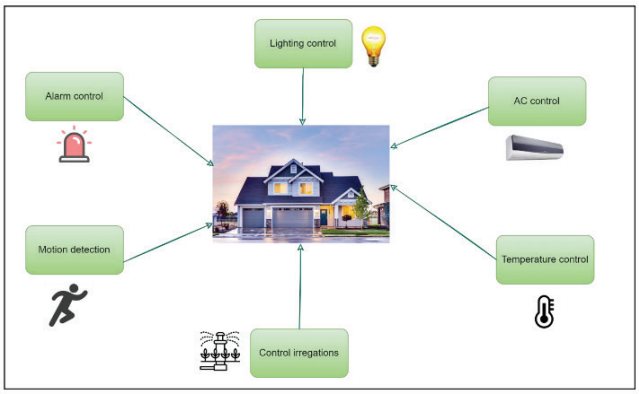

Fig. 1. IoT home automation

A simple IoT home automation has been shown in Fig.1 which includes the interconnection of several home appliances with a common microprocessor. Smart devices and appliances are effortlessly integrated by IoT home automation, revolutionizing living spaces. Homeowners may remotely control security systems, lighting, thermostats, and more with the help of networked sensors and internet access. With the help of this technology, homes may become more responsive, intelligent, and energy-efficient while also improving convenience and security.

ii. Artificial Intelligence (AI)

Artificial intelligence is the concept of machines being able to act rationally similar to human abilities by observing their environment to achieve an objective. AI can be understood as a big bubble that has many smaller subsets within it. Machine learning, a subset of AI, can be defined as a way of developing an AI where machines learn from data.

iii. Hypothesis issues

For our system, we chose spectrograms for pre-processing as they were much more compatible with Convolutional Neural Networks (CNNs). We also did a few tests by using MFCCs with Long Short-Term Memory (LSTM) cells but due to limited data, the spectrograms and CNNs worked better.

iv. Algorithm

Convolutional neural networks (CNN) are a type of deep neural network that is capable of learning directly with data. They are useful for identifying patterns, edges, and textures. The components of CNN such as convolution, pooling, and

flattening layers learn to extract features from input data.

Designs

a) Dataset Construction

The system is focused on recognizing short voice commands given in the Nepali language so audio dataset files were created accordingly. The duration of each audio file was created to be around three seconds long and a sampling rate of 44100 was used. We selected the sampling rate of 44100 because most of the electronic devices manufactured today use microphones whose sampling rate is around 44100. Different commands' audio dataset was recorded and each were stored in a unique folder for ease of access during model training. Background noises were also recorded in the same way and used for training the model.

b) Preprocessing

Raw audio data is rarely directly fed into any machine learning models. Some methods are used in order to preprocess the raw audio data into a suitable form for feeding into the machine learning architectures. The most popular methods for preprocessing are spectrograms and Mel-Frequency Cepstral Coefficients (MFCCs). For generating a spectrogram, we first need to compute the Short-Time Fourier Transform (STFT) of the signal. STFT is a sequence of Fourier Transforms that are computed over short and overlapping windows of the input signal. In this way, we can convert a time-domain signal into a time-frequency domain signal.

The Fourier transform of a function f with angular frequency ω is denoted by $F(f)$ and the formula is given as

$$F(f)(\omega) = \int_{t1}^{t2} f(t)e^{-j\omega t}dt \quad (1)$$

The Short Time Fourier Transform of a function $f(t)$, frequency ω, time shift τ and window function $g(t)$ is given as

$$STFT\{f(t)\}(\tau,\omega) = \int_{-\infty}^{\infty} f(t)g(t-\tau)e^{-j\omega t}dt \quad (2)$$

For the spectrogram, let b be the time-shift variable denoted by T_b in the Fourier Transform and k be the angular frequency, then the short-time Fourier transform of a function f with respect to a window function g is given as

$$STFT\{f(t)\}(b,k) = \int_{-\infty}^{\infty} f(t)\cdot g(t-T_b)e^{-jkt}dt = F(f\,.\,T_b g)(k) \quad \{from\,(1)\,and\,(2)\}$$

Let the above equation be noted as

$$V_g f(b,k) = F(f\,.\,T_b g)(k) \quad (3)$$

Hence, we obtain the spectrogram as:

$$S_o(b,k) = |V_g f(b,k)|^2 \quad (4)$$

The above-mentioned expressions are mathematically detailed but they are not suitable for directly processing digital signals. There are libraries such as Librosa that make the implementation of these complex mathematical concepts more practical for application development. Using Librosa we further used Mel-Spectrograms, which are obtained by passing spectrograms through Mel filter banks. Mel-Spectrograms have been found to improve the performance of machine learning models in audio-related tasks as they provide a more meaningful representation of the audio signal in terms of the human auditory perception. The waveforms of some of the recorded commands along with background noises are as follows:

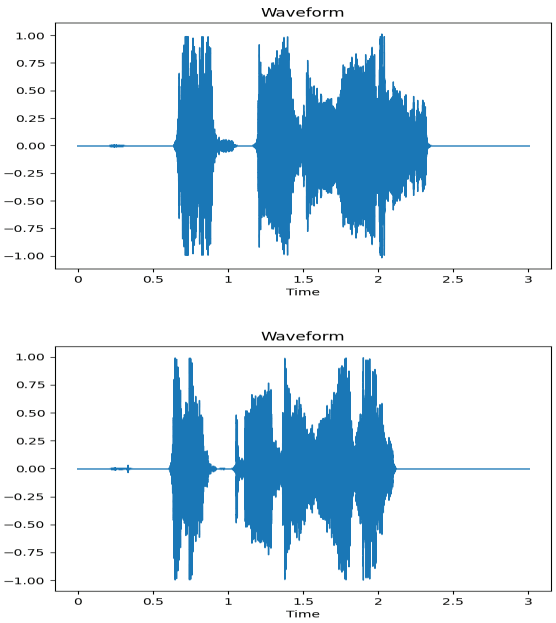

Fig. 2. Waveform samples of the commands recorded as "बत्ती बाल्नुस्"

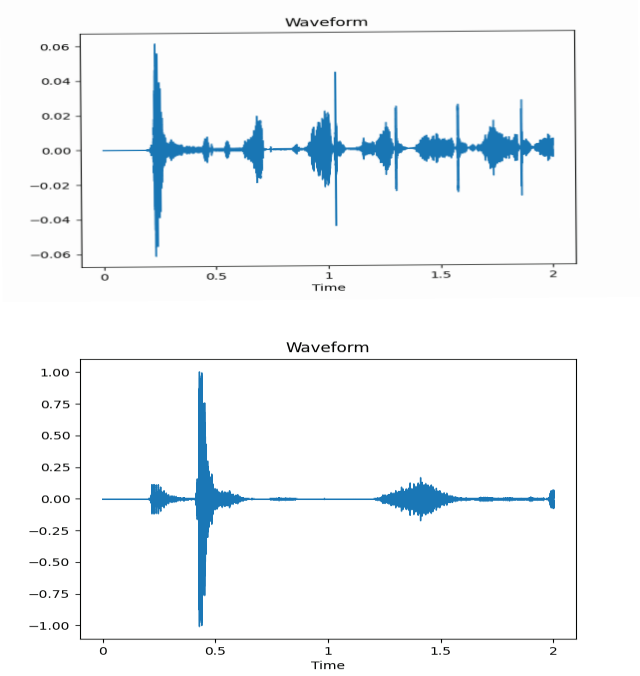

Fig. 3. Waveform samples of background noises

The obtained Mel-spectrograms of the above waveforms are shown below:

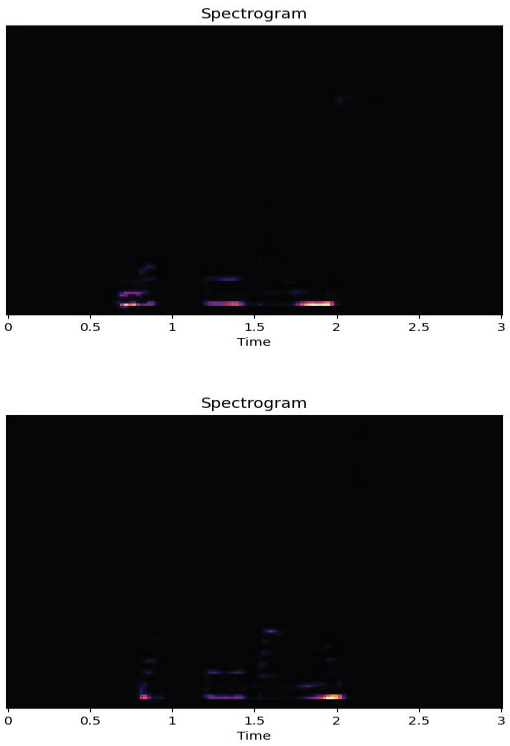

Fig. 4. Mel-spectrogram samples of the commands recorded as "बत्ती बाल्नुस्"

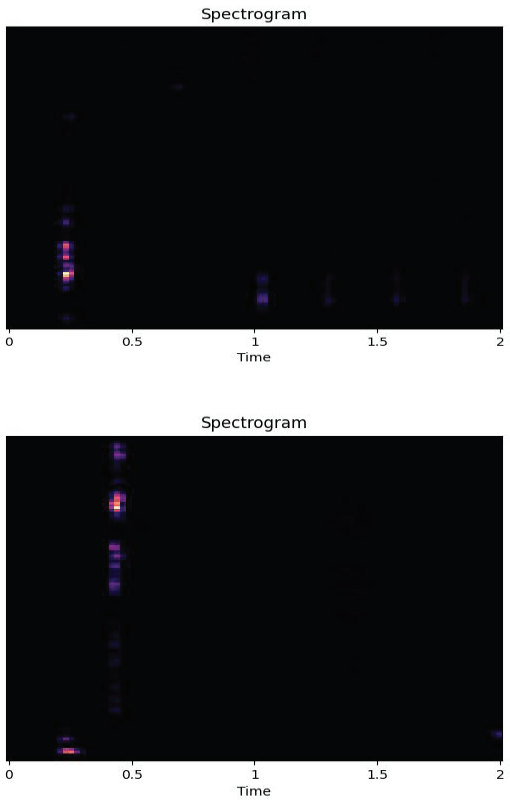

Fig. 5. Mel-spectrogram samples of background noises

c) Model Architecture

We built our model architecture with two convolutional neural network (CNN) layers, two max-pooling layers, a flatten layer, a dropout layer, and two fully connected dense layers.

The first CNN layer was used, with ReLU activation, for scanning the spectrogram using 4 different 3x3 filters. This layer is used for locating patterns like edges or textures of the spectrogram. After the first CNN layer, a max-pooling layer was used to reduce the size of the output from the previous layer by a factor of 2. This max-pooling layer takes the most important features obtained from the spectrogram and discards the rest.

The second CNN layer is similar to the first CNN layer but it looks for complex patterns in the data that have already been simplified by the first two layers. A second max-pooling layer was also used to further reduce the size of the feature map and only keep the essential elements.

After the CNN and max-pooling layers, a flatten layer was used for converting 2D feature maps into a 1D array to prepare it for input into the dense layers. A dropout layer was also used for randomly setting 20% of the input units to 0 at each training in order to prevent overfitting.

The first dense layer was used as a fully connected layer with 40 neurons and the activation function was set to ReLU. This layer helps the model to make decisions based on the extracted features by the previous layers. The final dense layer was used as an output layer with the activation set as a softmax function which is suitable for multi-class classifications.

TABLE 1

MACHINE LEARNING MODEL PARAMETERS

Neural Network Model Parameters	
Parameter Type	**Count**
Total Parameters	328,113
Trainable Parameters	328,113
Non-trainable Parameters	0

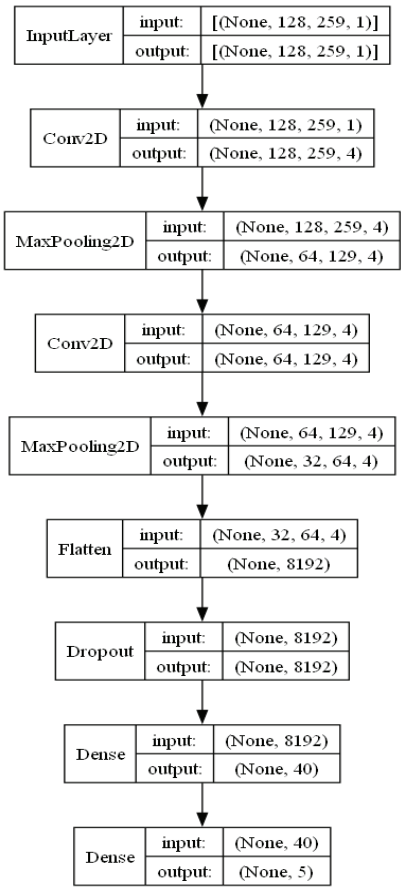

Fig. 6. Model architecture visualization

d) Training the model

For training our machine learning model, we used different Python libraries. We used Pandas and Numpy for loading the preprocessed audio data into memory and converting the obtained list into a NumPy array.

After obtaining the array, we split the dataset into training and testing subsets by utilizing the scikit-learn library. 20% of the data was used for testing and the remaining 80% was used for training the model. The width and height of the spectrogram images were also extracted by using NumPy. These dimensions are important for defining the input layer of our neural network.

The model architecture was defined with the help of Keras library and complied for training. The model was compiled using categorical cross-entropy as the loss function. This function measures how well the model predicts the class of each input. After training, we saved the model in the H5 format. After the model was trained and saved, it was evaluated using the test sets in order to see its performance against the data it did not encounter during the training phase. Predictions were made on the test set and the class with the highest probability was chosen as the model's prediction.

At the end of the training and evaluation, a classification report was created in order to view a detailed report of metrics like precision, recall, and F1-score for each class. A confusion matrix was also plotted in order to view the true labels and the predicted labels.

TABLE 2

CLASSIFICATION REPORT

Class Number	Class Name	Precision	Recall	F1-Score	Support
0	पृष्ठभूमि ध्वनि	1.00	1.00	1.00	28
1	बत्ती बाल्नुस्	0.93	1.00	0.97	14
2	बत्ती निभाउनुस्	0.91	1.00	0.95	10
3	पंखा चलाउनुस्	0.96	0.96	0.96	24
4	पंखा निभाउनुस्	0.95	0.88	0.91	24

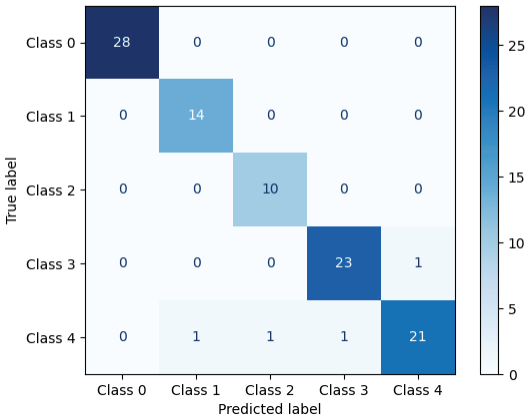

Fig. 7. Confusion matrix plot.

Software Implementation

A simple app was made to interface with the machine-learning model architecture and hardware architecture. PyQt was used to develop the software interface for the PC. For Android application development Android Studio and Java was used. The necessary software/libraries along with the versions we used are mentioned in Table 3.

TABLE 3

SOFTWARE/LIBRARY VERSION

Category	Software/Library	Version
Python	Python	3.11
	Numpy	1.23.5
	Pandas	2.0.3
	Scikit-learn	1.3.0
	Keras	2.12.0
	Matplotlib	3.6.3
	Librosa	0.9.2
	Scipy	1.11.1
	PyQt	5
Android Development	Android Gradle	8.0.0
	Java	8

Hardware implementation

A. Node MCU

It is a development board with built-in WIFI module which is specially used for connecting the ESP8266 microcontroller to the WIFI network. It includes open-source firmware which written in Lua programming language and consists of necessary computer elements Central Processing Unit (CPU), Random Access Memory (RAM), etc.

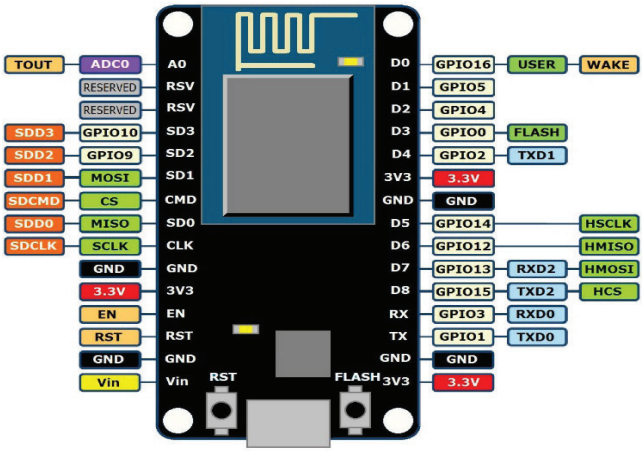

Fig. 8. ESP8266 node MCU pin diagram

B. Four-Channel Relay Module

A relay module is a switch in which the flow of high current due to high voltages (such as 220v mains) is controlled through low voltage signals from microcontrollers. It allows low-power design devices such as *"Node MCU"* to interact with high-power devices like bulbs and heaters i.e. home appliances. For this work, we have designed our own custom four-channel relay module using a custom Printed Circuit Board (PCB) design. The PCB design was carried out using *EasyEDA* software.

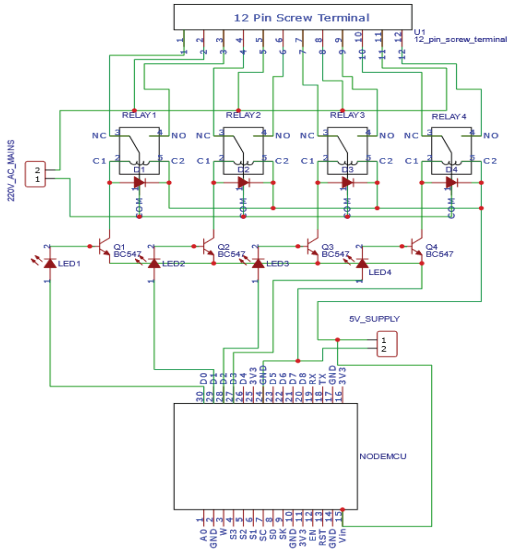

Fig. 9. Schematic of 4-channel relay module with Node MCU

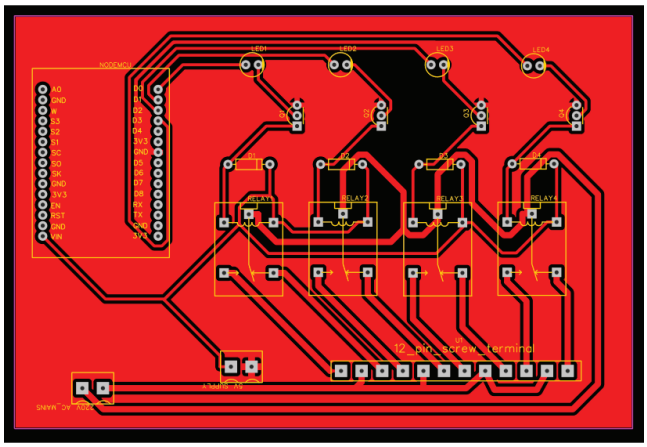

Fig. 10. Top layer and top silk layer of PCB

The PCB was prepared by implementation of toner transfer using heat and the PCB was etched using the Ferric Chloride ($FeCl_3$) etching method. The toner was easily removed using acetone. Individual holes and vias of PCB are drilled and the components were soldered.

C. Node MCU Network

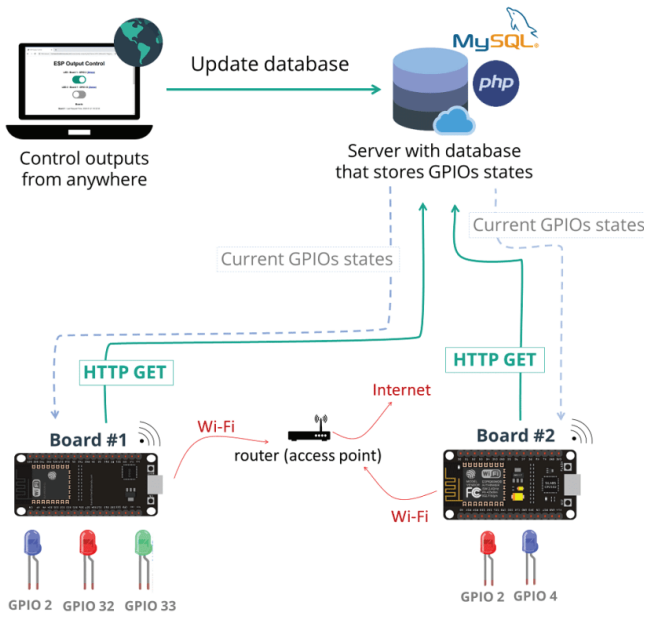

Fig. 11. Node MCU network through cloud database

In this paper, a real-time database of Google Firebase is implemented which allows us to control the General Purpose Input Output (GPIO) pins of "Node MCU" from any part of the world as depicted in Fig. 15.

The Firebase real-time database is linked with the mobile and desktop application which can recognize voice inputs and send data to Node MCU to control the GPIO pins. This is achieved through the predefined Java snippet for mobile applications and Python snippet for Windows applications which is given by Google Firebase. The database in a cloud-based server is updated when each command is given. The

ESP8266 gets the commands through the cloud database with the help of a request library of Python or Java in desktop or mobile applications.

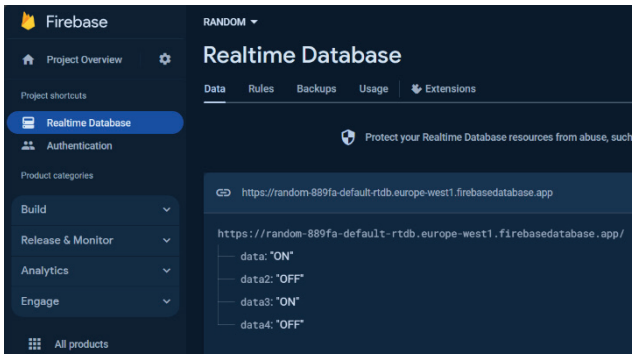

Fig. 12. Firebase real-time database

The database represented in Fig. 16. consists of four nodes data, data2, data3, and data4 which controls four GPIO pins of ESP8266. In our work, we have implemented two light bulbs controlled by data, data2, and two fans controlled by data3, data4. The firebase is connected to ESP8266 with the help of two parameters, the firebase host Uniform Resource Locator (URL) and the firebase authentication key also known as the "secret key".

The programming of the ESP8266 microcontroller is carried out in the Arduino Integrated Development Environment (IDE). The Arduino IDE libraries used for establishing a connection between node MCU and firebase are given below in the following table with their versions.

TABLE 4

ARDUINO IDE LIBRARIES

Header File	Library/ Package	Version
ESP8266WiFi.h	esp8266 by ESP8266 community	2.6.3
FirebaseESP8266.h	Firebase ESP8266 Client by Mobitzt	3.1.13

The ESP8266WiFi.h header file is used for connecting the Node MCU to the router via WIFI and the FirebaseESP8266.h header file is used for establishing a connection between the Node MCU and real-time database in Firebase provided that the Node MCU is connected to the router.

Each node in the real-time database can contain two values, "ON" and "OFF" which are of string datatype. This string datatype is fetched and compared in the conditional statements of the Node MCU source code through which the GPIO pins are switched "ON" or "OFF". The real-time database can also be shared among different users with the permission of the owner of the database which implies, that multiple users can control the GPIO pins of the same microcontroller from any part of the world.

Conclusions

The major purpose for doing this research is to implement Nepali speech recognition for home automation. To achieve this, different software development environments and hardware development boards, circuits, and modules were integrated. Controlling basic home appliances such as lights, fans, heaters, etc. with the Nepali language as voice input to the mobile and desktop applications was successfully achieved. The GPIO pins of ESP8266 (Node MCU) can be controlled using our application from anywhere in the world, provided that the Node MCU and the user is connected to the internet. Lastly, for additional features, note-making, application initiators, web navigation, and automatic typing in the English language were also implemented.

Acknowledgement

We express gratitude to our supervisor, Assoc. Professor Er. Shyam Dahal, whose unwavering support and guidance were instrumental throughout the research. We are grateful to the Research and Innovation Unit of Kathmandu Engineering College, for providing us with the opportunity to undertake this research, enabling us to apply the knowledge acquired over the years. This opportunity significantly enriched our understanding and fostered teamwork skills.

References

[1] A. Chaudhry, M. Batra, P. Gupta, S. Lamba and S. Gupta, "Arduino Based Voice Controlled Robot," *2019 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, Greater Noida, India, 2019.

[2] Kumar, Kantakar. (2022). Voice Controlled Robot Vehicle Using Arduino. International Journal for Research in Applied Science and Engineering Technology.

[3] Bhushan, C. and Kamble. "Speech Recognition Using Artificial Neural Network – A Review." (2016).

[4] Hannun, Awni Y. et al. "Deep Speech: Scaling up end-to-end speech recognition."

[5] Manojkumar, Patil & Patil, Aditi & Shinde, Sakshi & Patra, Shaktiprasad & Patil, Saloni. (2023). AI-Based Virtual Assistant Using Python: A Systematic Review. International Journal for Research in Applied Science and Engineering Technology.