The North African Journal of Scientific Publishing (NAJSP)
مجلة شمال إفريقيا للنشر العلمي (NAJSP)
EISSN: 2959-4820
Volume 1, Issue 4, October-December 2023, Page No: 37-53
Website: https://najsp.com/index.php/home/index
SJIFactor 2023: 3.733 **0.63 :2023 (AIF) معامل التأثير العربي** **ISI 2023: 0.383**

Developing a Speech Recognition System using Deep Learning with LabVIEW in Smart Home Devices

Tajdeedah Alhousayn Ahfaaf [1*], Ashraf Auda Abomandel [2]
[1] Computer department, College of Education, Bani Waleed University, Bani Waleed, Libya
[2] Computer department, College of Applied Sciences Technology – Al-Awata, Libya

*Corresponding author: tjdedaehfaf@bwu.edu.ly

Received: September 18, 2023	Accepted: November 03, 2023	Published: November 07, 2023

Abstract:
The main goal of this paper is to enhance the response of smart devices to voice commands by employing LabVIEW and deep learning for voice recognition. The research works to create an audio sensing system that can effectively recognize voice inputs in real time, even in noisy or difficult conditions, giving consumers a natural interface that enables them to interact directly with their tools. To achieve this purpose, it is necessary to collect and analyze a set of audio data for the smart home device. A deep learning model was trained using this dataset and then tested to see how well it could recognize speech in a smart home environment. The research also addresses alternative methods for protecting user information and considers the privacy and security implications of using a speech detection system in a smart home device. The results showed the success of the experiment in creating a sensor system based on LabVIEW to recognize speech with smart home devices and maintain security and privacy at the same time.

Keywords: Voice commands, Deep learning, Speech recognition, LabVIEW.

Cite this article as: T. A. Ahfaaf, A. Auda Abomandel, "Developing a Speech Recognition System using Deep Learning with LabVIEW in Smart Home Devices," *The North African Journal of Scientific Publishing (NAJSP),* vol. 1, no. 4, pp. 37–53, October-December 2023

تطوير نظام التعرف على الكلام باستخدام التعلم العميق باستخدام LabVIEW في الأجهزة المنزلية الذكية

أ. تجديدة الحوسين الهادي احفاف [1*]، أ. أشرف عودة أبو منديل [2]
[1] قسم الحاسوب، كلية التربية، جامعة بني وليد، بني وليد، ليبيا
[2] قسم الحاسوب، كلية التقنية للعلوم التطبيقية، العواتة، ليبيا

الملخص
الهدف الرئيسي من هذا البحث هو تعزيز استجابة الأجهزة الذكية للأوامر الصوتية من خلال توظيف LabVIEW والتعلم العميق للتعرف على الصوت، حيث يعمل البحث على إنشاء نظام استشعار صوتي يمكنه التعرف بشكل فعال على المدخلات الصوتية في الوقت الحقيقي، وحتى في الظروف الصاخبة أو الصعبة، مما يمنح المستهلكين واجهة طبيعية تمكنهم من

التفاعل المباشر مع أدواتهم. ولتحقيق هذا الغرض يتطلب جمع وتحليل مجموعة من البيانات الصوتية لجهاز المنزل الذكي. تم تدريب نموذج التعلم العميق باستخدام مجموعة البيانات هذه، ثم اختباره لمعرفة مدى قدرته على التعرف على الكلام في بيئة المنزل الذكي، كما يتناول البحث الطرق البديلة لحماية معلومات المستخدم ويأخذ في الاعتبار الآثار المترتبة على الخصوصية والأمن لاستخدام نظام كشف الكلام في جهاز منزلي ذكي. وتوصلت النتائج الى نجاح تجربة إنشاء نظام استشعار يعتمد على LabVIEW في التعرف على الكلام مع الأجهزة المنزلية الذكية والحفاظ على الآمن والخصوصية في الوقت نفسه.

الكلمات المفتاحية: الأوامر الصوتية، التعلم العميق، التعرف على الكلام، LabVIEW.

Introduction:

The goal of the research project "Deep Learning for Speech Recognition in Smart Home to Control Devices by LabVIEW" is to create a speech recognition system that uses deep learning methods and integrates it with LabVIEW software to control devices in a smart house. The demand for more comfortable and natural ways to operate devices in a smart home is the driving force behind this study. Traditional controls, like remotes and smartphone applications, may be laborious and demand user input. Deep learning may enhance the accuracy and dependability of speech recognition systems, which provide a hands-free and natural method to operate technology.

The research involves the gathering of data, pre-processing, creation of a model, integration with LabVIEW, and performance assessment. To collect data, a dataset of voice samples must be assembled.

Data collection, pre-processing, model creation, interaction with LabVIEW, and performance assessment are all included in the research. To train and test the speech recognition system, a dataset of speech samples must be gathered. The dataset is prepared for use in the deep learning model during the pre-processing stage, and the deep learning model for voice recognition is created utilizing the dataset during the model development stage. The model's performance is assessed in terms of accuracy and efficiency during the training and assessment process [1]. The integration with LabVIEW involves developing a user interface for controlling devices. The performance evaluation stage compares the voice recognition system's performance to conventional control methods in a smart home. The study has the potential to increase effectiveness, convenience, and accessibility by giving consumers a more comfortable and natural way to operate gadgets in a smart home [2].

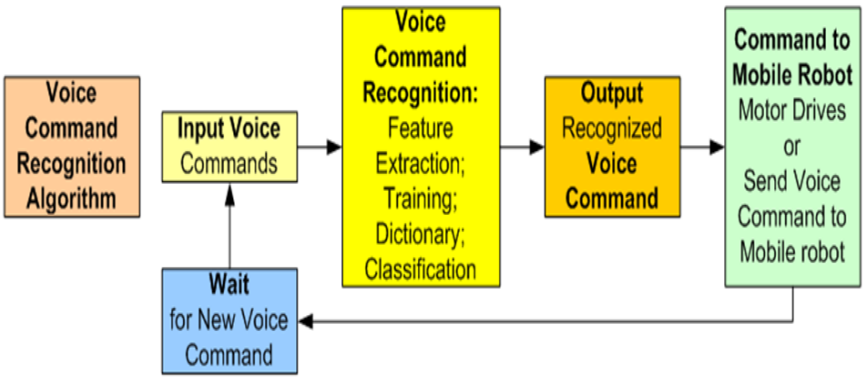


Figure 1: Block diagram of speech recognition program

Objectives:

- A natural and easy-to-use speech interface for smart home devices is provided by deep learning, which improves the user experience.
- Voice control increases accessibility for those with impairments and for those who find conventional user interfaces uncomfortable.
- By collecting complex speech qualities like tone, pitch, and rhythm, deep learning increases the accuracy and robustness of speech recognition.
- Reliable performance is made possible even in loud or difficult conditions via accurate voice recognition.

Innovative uses of deep learning include voice-controlled home automation and individualized media suggestions.

• Deep learning-powered intelligent assistants can carry out a variety of activities, boosting the usefulness of smart home gadgets.

Problem Definition:

Using deep learning for voice recognition in smart home devices presents a number of technical and practical difficulties. These consist of:

1. Acoustic variability: Accent, background noise, and speaking style are just a few examples of the many variables that can affect speech signals. Due to this, it may be tricky to identify voice inputs properly, especially in noisy or harsh circumstances.
2. Limited training data: To function well, deep learning models often need a lot of labelled training data. On the other hand, gathering and annotating voice data can be time- Costly and time-consuming, especially for less widely spoken languages or dialects.
3. Real-time processing: voice recognition systems that can process voice inputs in real-time, with low latency and high accuracy, are needed for smart home devices. With sophisticated deep learning models that need substantial processing resources, this can be difficult to do.
4. Privacy and security: In order to avoid unwanted access to sensitive information or unintentional device activation, speech recognition systems in smart home devices must be developed with privacy and security in mind.

Artificial Intelligence:

Unlike the natural intelligence displayed by humans and other animals, which is demonstrated by machines, artificial intelligence, also known as machine intelligence, may be comprehended by intelligence. It examines methods for creating intelligent machines and systems that can come up with innovative solutions to issues that are frequently viewed as human rights. AI therefore denotes a machine that mimics human behaviour in some way.

Machine Learning

A subset of artificial intelligence known as "machine learning" comprises of methods that let computers understand data and produce AI applications. In ML, a variety of algorithms—such as neural networks—helps solve problems.

Deep Learning

Deep learning, also known as deep neural learning or deep neural network, is a branch of machine learning that makes use of neural networks to analyse data in a manner like that of the human nervous system. It has networks that can autonomously learn from unstructured or unlabelled input.

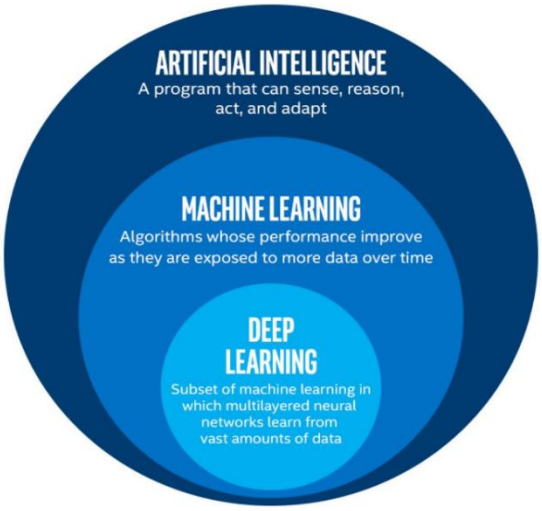


Figure 2: Artificial Intelligence vs Machine and Deep Learning.

Deep Learning for Audio:

- Deep learning models have been employed in voice recognition to convert spoken words into text, including recurrent neural networks (RNNs) and convolutional neural networks (CNNs). By analysing audio data as a series of characteristics, such as spectrograms or Mel-frequency spectral coefficients (MFCCs), these models may learn to detect patterns in speech.

- Deep learning models have been used to categorize music according to genre, artist, or mood. By decomposing audio data into a series of time-frequency representations, these models are able to distinguish musical elements including pace, rhythm, and melody.
- Deep learning models have been used in sound event identification to locate certain noises in an audio recording, such as a dog barking or an automobile honking. By analysing audio inputs as a spectrogram or a Mel-spectrogram, these models may learn to distinguish the acoustic features of various sound occurrences.

Challenges in deep learning:

1. Limited availability of labeled data: In order to attain high accuracy, deep learning models need a lot of labeled data. Large volumes of labeled data may be hard to come by for some audio identification applications, such as sound event detection in natural settings.

2. Audio recording variation: Deep learning models may have trouble generalizing to new data due to the considerable variation in background noise, speaker accent, and recording quality in audio recordings.

3. Computational needs: Deep learning models may be computationally demanding and call for strong hardware, such GPUs or TPUs, for deployment and training. Because of this, it may be challenging for academics and businesses with minimal funding to use deep learning for audio identification tasks. Interpretability: Deep learning models can be difficult to interpret, which can make it challenging to understand why a model is making certain predictions or to diagnose errors.

4. Multimodal integration: To increase accuracy, voice recognition frequently integrates various modalities, such as audio and visual input.

However, creating powerful multimodal deep learning models may be difficult and calls for multidisciplinary knowledge [13].

Techniques to improve speech recognition:

Several methods may be applied to increase voice recognition precision in noisy settings:

1. Noise reduction: Pre-processing the voice stream to reduce background noise is a typical approach. Numerous techniques exist to reduce noise, including spectral subtraction, Wiener filtering, and adaptive filtering.
2. Beamforming: Beamforming is a signal processing approach that boosts the spoken signal and reduces background noise by employing a variety of microphones. This is particularly useful in situations with several talkers.
3. Feature enhancement: Improving the speech features that the speech recognition system uses is another strategy. Techniques like spectrum subtraction, spectral mean subtraction, and feature warping can be used for this.
4. Speaker separation is a method for separating the speech of several speakers in a situation when there are multiple talkers. Techniques for blind source separation, such as independent component analysis (ICA) and non-negative matrix factorization (NMF), can be used for this.
5. Deep learning: Deep learning models may be specifically trained to deal with cluttered voice inputs. In order to develop noise-resistant voice representations, convolutional neural networks (CNNs) or recurrent neural networks (RNNs) are frequently used.
6. Data augmentation: Lastly, strategies for data augmentation may be utilized to enhance the volume of training data and strengthen the voice recognition system. This may entail introducing noise to recordings of clear speech or creating artificial acoustic settings.

In general, enhancing speech recognition accuracy in noisy contexts is a difficult challenge that frequently necessitates the combination of different strategies to get positive outcomes [12].

- Challenges in deep learning

Even though deep learning has made great strides in audio identification, there are still issues that still need to be resolved. Several of these difficulties include:

1. Limited availability of labeled data: In order to attain high accuracy, deep learning models need a lot of labeled data. Large volumes of labeled data may be hard to come by for some audio identification applications, such as sound event detection in natural settings.

2. Audio recording variation: Deep learning models may have trouble generalizing to new data due to the considerable variation in background noise, speaker accent, and recording quality in audio recordings.

3. Computational needs: Deep learning models may be computationally demanding and call for strong hardware, such GPUs or TPUs, for deployment and training. Because of this, it may be challenging for academics and businesses with minimal funding to use deep learning for audio identification tasks.

4. Interpretability: Deep learning models can be tricky to grasp, making it difficult to identify faults or comprehend why a model makes certain predictions.

5. Multimodal integration: To increase accuracy, voice recognition frequently integrates various modalities, such as audio and visual data. However, creating powerful multimodal deep learning models may be difficult and calls for multidisciplinary knowledge [13].

- Applications of deep learning

Deep learning has many applications across a wide range of fields, such as:

1. Applications in computer vision, such as object identification, picture categorization, and facial recognition, employ deep learning.
2. Deep learning is employed in natural language processing applications including sentiment analysis, speech recognition, and language translation.
3. Robotics: Deep learning is applied to robot control, autonomous navigation, and object identification in robotics.
4. Healthcare: Deep learning is applied in medical applications including illness diagnosis, medication discovery, and medical picture analysis.
5. Finance: Deep learning is employed in financial applications such as risk management, fraud detection, and stock forecasting.
6. Deep learning is utilized in autonomous vehicles to perform tasks including object identification, lane detection, and decision making.
7. Video games: Deep learning is applied to the creation of generative content, game AI, and game physics.

When compared to conventional techniques, audio recognition using deep learning has demonstrated considerable gains. The following are some significant variances between deep learning and conventional approaches [13]:

- Traditional methods for audio recognition

1. Feature extraction: Hand-crafted features, such Mel-frequency spectral coefficients (MFCCs), which are intended to capture the crucial acoustic aspects of speech and sound sources, are frequently used in traditional approaches for audio recognition. Instead of requiring manually created features, deep learning may learn features directly from the unprocessed audio data.
2. Model construction: Statistical models, such hidden Markov models (HMMs), or template matching techniques, like dynamic temporal warping (DTW), are frequently used in conventional approaches for audio identification. These models frequently need extensive subject knowledge and have limitations when it comes to capturing intricate data interactions. On the other hand, deep learning models can use recurrent and convolutional neural networks to capture complicated relationships in the data.
3. Training data: To obtain high accuracy, traditional algorithms for audio recognition frequently need a lot of labeled training data. On the other hand, deep learning models can benefit from unsupervised pre-training and transfer learning and can achieve high accuracy with less labeled training data.
4. Robustness: Compared to conventional techniques, deep learning models for audio recognition have proved to be more resistant to noise, distortions, and fluctuations in speech and sound signals. This is due to the ability of deep learning models to generalize to new cases and learn to extract pertinent characteristics from the input.

Overall, deep learning has demonstrated substantial advancements in audio identification against conventional approaches, reaching cutting-edge performance on a variety of tasks like voice recognition, speaker detection, and ambient sound categorization. Traditional approaches still have a role in some applications, such as those with limited resources or those requiring domain expertise [11].

- Building and training deep learning models

Although developing and training deep learning models might seem overwhelming at first, there are numerous tools available to guide you through the process. You can begin by doing the following actions:

1. Get to know the fundamentals of deep learning and machine learning: It's crucial to have a firm grasp of the fundamental ideas before beginning to develop and train models. Start by reading online books and tutorials, enrolling in online courses, or going to seminars.
2. Select a framework for deep learning: Deep learning frameworks like Tensor Flow, PyTorch, and Keras are among the many that are accessible. It's crucial to select a framework that meets your goals and ability level because each one has unique strengths and drawbacks.
3. Configure your development environment: Following your selection of a deep learning framework, you must configure your development environment. This might entail setting up your IDE, configuring your GPU (if you have one), and installing the required tools and libraries.
4. Select a dataset: A dataset containing labeled examples is necessary to train a deep learning model. There are several publicly accessible datasets, like the Common Voice dataset for voice recognition and Picture Net for picture categorization.
5. Create and train your model: Following the collection of data, you may begin creating and training your model. To make sure your model is learning properly, this may entail choosing the right neural network design, adjusting hyperparameters, and monitoring the training process.
6. Evaluate your model: Following training, you must assess your model's performance on a test dataset. You may use this to assess how effectively your model generalizes to fresh data.
7. Iterate and enhance: Deep learning is an iterative process, and to enhance the performance of your model, you will probably need to experiment with various neural network designs, hyperparameters, and training methods [12].

- A speech signal characteristic

A voice signal primarily has the following two qualities:

1. Frequency content: The sound waves that make up a voice signal travel through the air as time-varying fluctuations in air pressure. The distribution of energy across various frequencies is known as frequency content, and it may be used to examine these fluctuations in air pressure. For speech recognition, the frequency content of a voice signal is crucial since it provides details about the individual sounds or phonemes that make up spoken words.
2. The amplitude and length of each sound or phoneme, as well as the overall intonation and rhythm of the speech, are all time-varying elements that a speech signal carries in addition to its frequency content. The structure and meaning of the uttered words are conveyed by these time-varying properties, which are crucial for speech recognition.

- Pre-Processing Organization

The processes listed below make up the conventional structure of signal pre-processing for speech and audio signals, as seen in Figure 3.

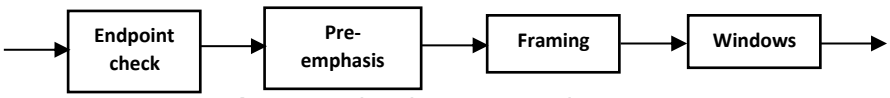


Figure 3: Signal Pre-processing structure.

1. Sampling: If the signal is analog, it is sampled using an analogue-to-digital converter at a set rate. The Nyquist-Shannon sampling theorem states that to prevent aliasing, the sampling rate should be at least twice the highest frequency component of the signal.
2. Filtering: To reduce noise or undesired frequencies, the signal may be filtered. To do this, a low-pass or high-pass filter may be used to exclude frequencies that are below or above a specific threshold, respectively. To exclude frequency ranges that won't be useful for the study, a band-pass filter can also be employed.
3. Normalization: To guarantee that the signal's amplitude range is constant, it may be normalized. As a result, next processing processes like feature extraction may be more accurate. By scaling the signal to have a maximum absolute value of 1, or by applying alternative normalizing methods, the signal can be normalized.
4. Frame-based processing: A tiny piece of the signal is included in each frame, which is how the signal is divided into segments. The frames are usually 10 to 30 milliseconds long, with a 50% overlap between each set of consecutive frames. This makes it possible to examine the signal across time and can aid in identifying the signal's time-varying characteristics.
5. Windowing: To minimize spectral leakage and increase the precision of the following processing stages, the frames are multiplied by a window function, such as the Hamming or Hanning window.

6. Feature extraction: Features, such as the Mel-frequency spectral coefficients (MFCCs) for speech signals, are recovered from the frames. The power spectrum, spectral centroid, and spectral flux are spectral properties that are additionally often utilized.
7. Post-processing: To enhance their quality and lower their dimensionality, the collected features may be further treated using methods like mean and variance normalization or principal component analysis (PCA) [12].

Speech recognition by Lab View program

The process of creating and implementing a program utilizing the LabVIEW software to detect speech and translate it into executable commands is known as speech recognition using LabVIEW program. Digital signal processing methods are used to analyze sound waves, and then a model for speech recognition is created using deep learning and artificial neural networks. The model is then implemented using the LabVIEW software, and the system's recognition accuracy is tested. The created technology enables voice management of smart home appliances, making user interaction simpler and more effective. Here is a broad description of how to use LabVIEW to do voice recognition:

1. Gather audio data: Gathering audio data from a microphone or other audio input device is the first step. For obtaining audio data from various sources, such as sound cards or USB microphones, LabVIEW includes built-in routines.
2. Preprocess the audio data: In order to extract useful characteristics for speech recognition, the audio data has to be pre-processed. Techniques including filtering, feature extraction, and normalizing may be used for this.
3. Train the speech recognition model: Following the preprocessing of the audio data, the speech recognition model must be trained. Machine learning methods, such deep neural networks (DNNs) and hidden Markov models (HMMs), can be used to do this.
4. Recognize speech: After being trained, the speech recognition model may be used to identify speech in real-time. Applying the model to the audio data that has already been processed and producing the identified speech as text are what this entails.
5. Assess performance: To make sure the speech recognition system is accurate and dependable; it is crucial to assess its performance. This may entail measuring metrics like word error rate (WER) or accuracy [2] and comparing the detected speech to the actual speech.
6. Improve the system: The speech recognition system may be improved by modifying the model's parameters or the preprocessing procedures in accordance with the assessment findings.

Overall, developing a speech recognition system using both knowledge of machine learning and signal processing are necessary for LabVIEW. However, it is feasible to develop a very accurate and trustworthy voice recognition system in LabVIEW given the correct tools and methods.

Every day, programmers create new software programs to boost effectiveness and productivity in numerous contexts. The computer language LABVIEW is a potent tool that may be utilized to assist in achieving these objectives. National Instruments created the visually based programming language LABVIEW (Laboratory Virtual Instrument Engineering Workbench). Because of its graphical character, it is perfect for applications involving automation, instrument control, data collecting, and data analysis. Compared to traditional programming languages, this leads to huge productivity gains. National Instruments has a strong understanding of creating LabVIEW because they concentrate on products for T&M [6].

Virtual instrumentation takes advantage of the ever-increasing performance of personal computers, for instance, engineers have used virtual instrumentation to downsize automated test equipment (ATE) in test, measurement, and control while experiencing user-defined solutions that meet their specific needs. This is a great alternative to proprietary, fixed-functionality traditional instruments. Program language history is depicted in Figure 4.

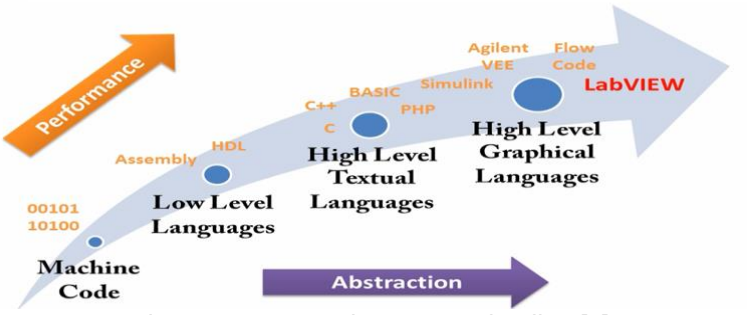


Figure 4: program languages timeline [3].

This paper will provide a brief introduction to LabVIEW. Some basic topics will be covered to understand LabVIEW.

- Virtual Instrument

LabVIEW's Virtual Instrument (VI) is a programming component. A front panel, a block diagram, and an icon that symbolizes the application make up a VI. While the block diagram houses the code for the VI, the front panel is utilized to show controls and indications for the user. The program inputs and outputs are connected to the icon, which is a picture of the VI. A VI's front panel controls the function inputs and outputs, while the code diagram executes the VI's instructions. Large-scale applications may be made using a variety of VIs; in fact, large-scale applications may have hundreds of VIs. A VI can be used in an application as a function or as the user interface. [6].

- The front panel

A LabVIEW VI's front panel is seen in Figure 5. It has controls for choosing the measurement type, the number of measurements per average, and the stop button. A digital indicator shows the output result. To act as an application's user interface, a complex front panel may be easily developed [6].

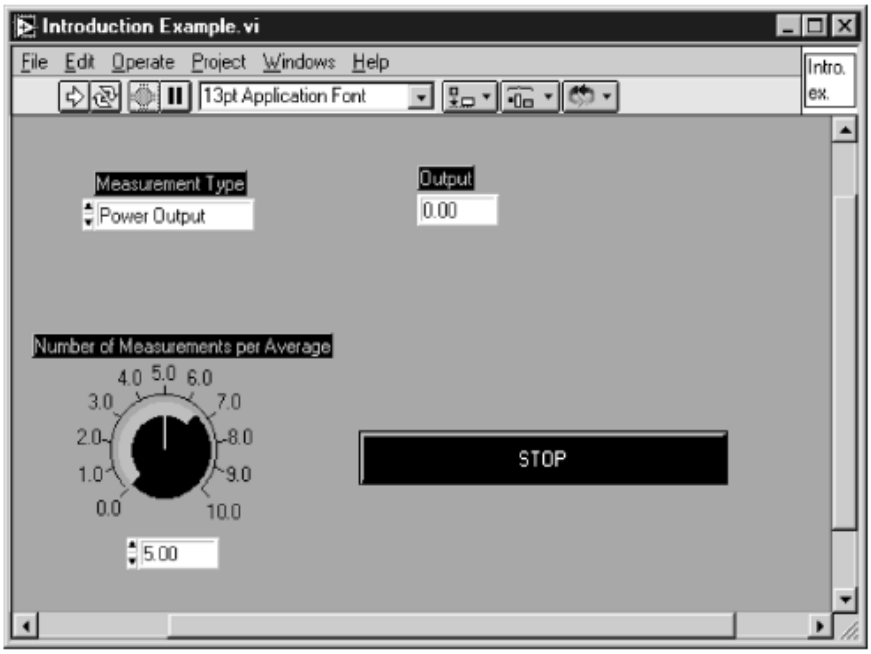


Figure 5: the front panel [4].

- The block diagram

The block diagram, or source code, that goes with the front panel in Figure 5 is shown in Figure 6. The inner rectangular structure is a case structure, while the outside rectangular structure is a while loop. The icon in the middle represents a VI, or subroutine, that receives as input the average number of measurements and outputs the frequency value. The data being transmitted from the control into the VI is represented by the orange line or wire. To choose which case is run, the measurement type choice is wired or connected to the case statement. The while loop comes to an end when the stop button is pressed. You get your first glimpse of a Virtual Instrument's front panel, block diagram, and icon in this example, which also highlights LabVIEW's emphasis on graphics. The execution engine for LabVIEW compiles LabVIEW code in the background; it is not an interpreted language. The LabVIEW execution engine handles the executable code that the VIs is compiled into, much like Java is done. LabVIEW creates a wire table for the VI whenever a modification is made to a VI. This wire table defines the block diagram parts that require inputs in order to function. Elements might be simple operations like addition, or they can be more complicated like a subVI. You are shown a solid arrow if LabVIEW successfully creates all the wire tables and the VIs may be run. If the wire table cannot be formed, a broken arrow is displayed for the problematic VIs as well as for any loaded VI that depends on the problematic VI for execution. Various subsystems run LabVIEW [6].

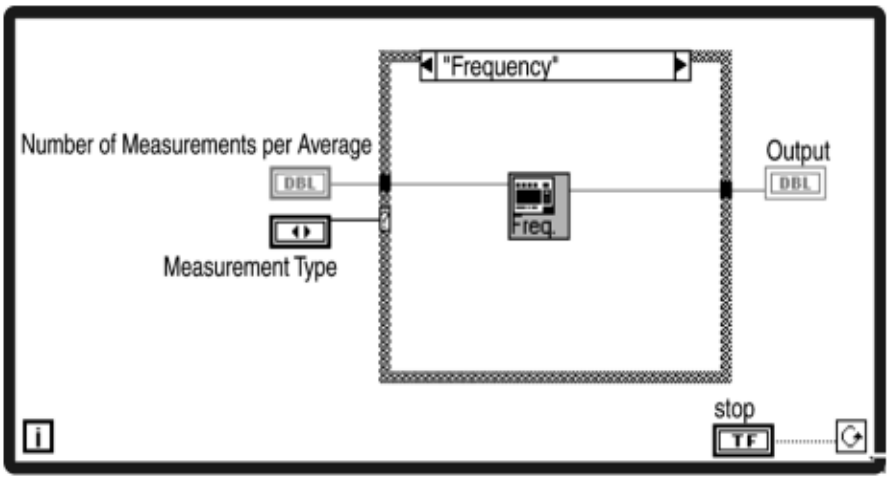


Figure 6: the bock diagram [4].

Speech recognition and Speech synthesis in this paper, it will discuss an overall overview of Speech recognition and Speech synthesis using LABVIEW how they work and their applications.

- A speaker verification system has two primary stages, which are represented in Figure 7's block diagram of a simple speaker verification system: the training phase, during which target speakers are enrolled, and the testing phase, during which a determination regarding the speaker's identification is made. Speaker models may be divided into generative and discriminative types from the perspective of training. Feature distributions within each speaker are estimated via generative models like the Gaussian Mixture Model (GMM). In contrast, discriminative models that simulate the border between speakers include Support Vector Machine and Deep Neural Network (DNN).
- The performance of speaker verification systems is degraded by the variability in channels and sessions between enrolment and verification speech signals. Factors which affect channel/session variability include:
  - Channel mismatch between enrolment and verification speech signals such as using different microphones in enrolment and verification speech signals.
  - Environmental noise and reverberation conditions.
  - The differences in speaker voice such as ageing, health, speaking style and emotional state.
  - Transmission channel such as landline, mobile phone, microphone and voice over Internet protocol (VoIP) [15].

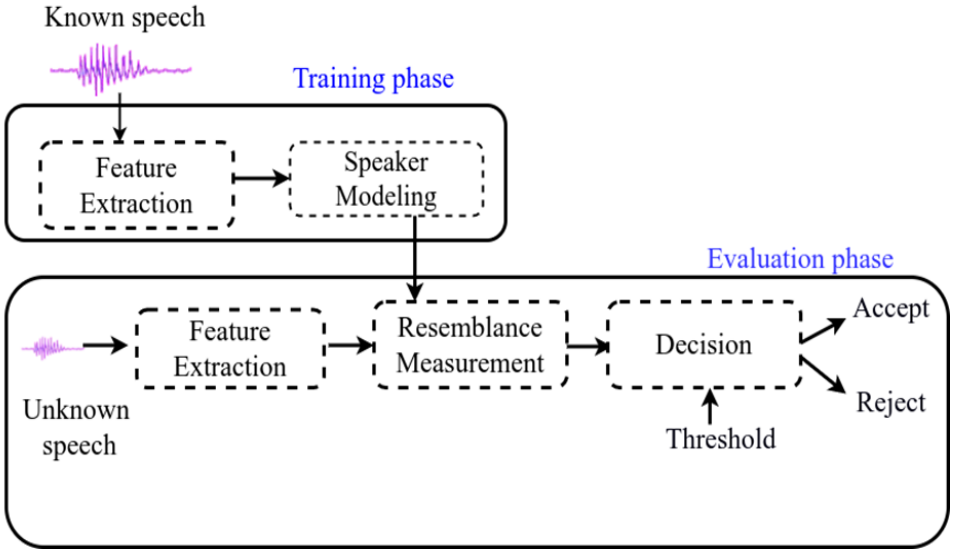


Figure 7: Block diagram of a basic speaker verification system.

Block diagram of TTL processor on LABVIEW

The block design for the LabVIEW-based text-to-speech converter is shown in Figure 6. A typical speech recognizer's block diagram includes a number of essential elements that cooperate to analyze and decipher spoken input. An abbreviated version of the block diagram is shown below:

1. Audio Input: A speech recognizer's input audio can be recorded from a microphone or any other audio source.
2. Preprocessing: To improve the audio signal's quality and get it ready for more analysis, the audio signal passes through preprocessing phases. Activities like noise reduction, filtering, and normalization may fall under this category.
3. Feature Extraction: The pre-processed audio stream is used to extract pertinent features at this stage. Mel-frequency spectral Coefficients (MFCCs), which record the spectral properties of the spoken stream, are often employed features.
4. Acoustic Model: An acoustic model is fed the feature vectors that were created in the previous step. A deep neural network that has been trained on a significant quantity of tagged speech data often serves as the acoustic model. In order to calculate the likelihood of each unit given the input features, it translates the input features to phonetic or sub-phonetic units.
5. Language Model: A language model is coupled with the acoustic model's output probabilities. The language model incorporates knowledge of word sequences and grammatical rules to control the recognition process while capturing the statistical characteristics of the language being spoken.
6. Decoding: The combination of the acoustic model and language model is used to perform decoding, where the most likely sequence of words or phrases is determined based on the input audio. Various algorithms, such as Hidden Markov Models (HMMs) or Connectionist Temporal Classification (CTC), can be used for decoding [16].
7. Post-processing: The recognized speech output may undergo post-processing to refine the results. This can involve techniques like language model rescoring, confidence estimation, or error correction.
8. Output: The final recognized speech output is generated, which could be in the form of text or specific commands that can be used to control smart home devices or perform other tasks.

- Improve the acoustic representation of the PCM digital audio: The input for the speech recognizer is a stream of amplitudes that are sampled at a rate of around 16,000 times per second. But the recognizer cannot utilize audio in this manner. As a result, graphs of the frequency components that describe the sound heard for 1/100th of a second are produced using Fast-Fourier transformations. Any sound is then recognized by being compared to the nearest item in the database of such graphs, yielding a number known as the "feature number" that serves as the sound's description [7].
- The likelihoods of each sequence of speech recognition units matching the input speech are provided by the unit matching system. These units might be syllables, phones, diaphones, or derivatives like fen and acoustic ones. They might also be whole words or units that relate to a cluster of two or more words. A Hidden Markov Model (HMM) that has parameters that are computed using a training set of speech data characterizes each such unit [7].
  The unit matching system is constrained to only pursue search path sequences whose speech units are contained in a word dictionary [7] by lexical decoding.
  Use a "grammar" to tell the voice recognition software what phonemes to anticipate. This significantly restricts the unit matching system's search sequence. From a context-free grammar to full-fledged English, a grammar may be anything [7].
  Identifying the phonemes that are uttered is a tricky undertaking since different words sound differently when pronounced by various people. Additionally, the microphone's ambient sounds cause the recognizer to detect a different vector. So, while recognizing, a probability analysis is performed. On the basis of this study, a hypothesis is developed. A voice recognizer operates by simultaneously speculating on a variety of distinct "states". A phoneme with a history of prior phonemes may be found in each state. The final recognition result uses the hypothesized state with the greatest score. The highest score is used as the final recognition result [7].
- Speech recognition (SR) is a multidisciplinary branch of computational linguistics that combines expertise from the fields of linguistics, computer science, and electrical engineering to create methodologies and technologies that allow computers and other computerized devices, such as robots and Smart Technologies, to recognize and translate spoken language into text. It is also known as "speech to text" (STT), "automatic speech recognition" (ASR), or "computer speech recognition"[7].

It is used in this research to demonstrate how to use LABVIEW software to control robot movement.

- History development:

A lot of speech aware applications are already there in the market. Dragon, IBM, and Philip have each produced a variety of dictation software. Microsoft created the interactive voice recognition program Genie. Users may manage their computer with voice commands, just like they can browse the Internet, thanks to a variety of voice navigation programs, including one created by AT&T. This type of application is becoming more prevalent every day. The acoustic and linguistic models required for recognition are provided by the CMU SPHINX speech recognizer. The Hidden Markov Models (HMM) serve as its foundation. One of them is the SONIC recognizer, created by the University of Colorado. Other recognizers, including Voice for Linux, use IBM's Via Voice, which is currently only available for Windows. The worst aspect of voice recognition is background noise. The recognizer becomes perplexed and is unable to hear what it should. Robots now have a recognizer like this that, despite the unavoidable motor sounds, allows for effective communication with people. A noise-type-dependent acoustic model matching to a robot doing motion is used to achieve this. On an HP Smart Badge IV embedded system, voice recognition optimizations have been proposed. By merging DSR (Distributed Speech Recognition) with scalable compression, another such scalable system has been presented, lowering the computational burden and bandwidth need on the server. Voice Banking and Directory Assistance are only two examples of the many features that modern speech recognizers in the telecoms industry currently include.

Description:

To manage the house appliances, my major objective is to combine an Arduino board with the SR system. The following block diagram, Figure 8, shows how it operates:

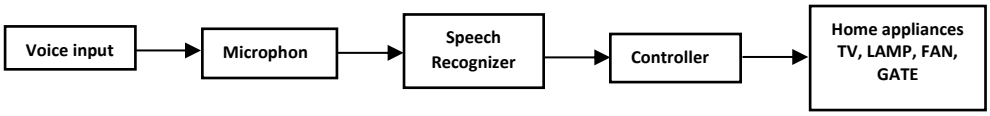


Figure 8: Block diagram for the SRS functioning integrated with controller [15].

The HP Pavilion PC is the intended software platform for this project. A quarter-VGA color LCD screen, 8GB of RAM, and an Intel(R) Core(TM) i7-2670QM CPU@2.20GHz processor are all features of the HP Pavilion computer. I picked this gadget because it features a voice recognition system and the Windows 7 operating system, allowing me to utilize it and transmit commands to an Arduino board. The following is a description of each part of our system:

- Voice Input: Human voice is input and is sampled at a rate of 16,000 samples per second.
- Microphone: The microphone that I am using for recognition is built onto the Hp platform itself. It has got its own advantages and disadvantages:

Advantages:

- Nothing to plug in.
- User's hands are free.

Disadvantages:

- Low accuracy unless the user is close to the monitor.
- Not good in a noisy environment

• Speech Recognizer: Platform speed directly affected our choice of a speech recognition system for our work. The entire computer which works on windows operation system has speech recognizer [7].

- Speech synthesis (text to voice)

Speech synthesis is the artificial production of human speech. A system used for this purpose is termed a speech synthesizer, and can be implemented in software or
hardware. Speech synthesis systems are often called text-to-speech (TTS) systems in reference to their ability to convert text into speech. However, systems exist that instead render symbolic linguistic representations like phonetic transcriptions into speech.

The text written in everyday language is translated into voice using a text-to-speech program. By joining the segments of the recorded speech that are saved in the database, the synthesized speech may be created. The vocal tract model on the synthesizer may be used to produce synthetic voice output with human voice characteristics. The vocal tract is a cavity found in both humans and animals that filters sound from a sound source. The capacity of a synthesizer to be totally comprehended by the user and a comparison to human voice are used to assess its quality. In a text-to-speech converter, there are two components. There are two main functions for the front section. First, it converts the words and numbers in the text into the equivalent of written words. Text normalization or tokenization is the term

used for this. Each phrase is given a phonetics transcription in the front section, which is a visual depiction of the speed sounds. Text-to-phoneme conversion is the process of giving phonetic transcriptions to words. The second component, referred known as a synthesizer, creates sound out of words and numbers. The text-to-speech processor's fundamental block diagram is displayed in Figure 9.

The text-to-speech conversion utilizing LABVIEW is the main topic of this article. Additionally, this paper provides a brief overview of the TTS processor's fundamental block diagram and highlights some of the many ways it is used in both this paper and the real world.

- The block diagram of TTL processor on LABVIEW

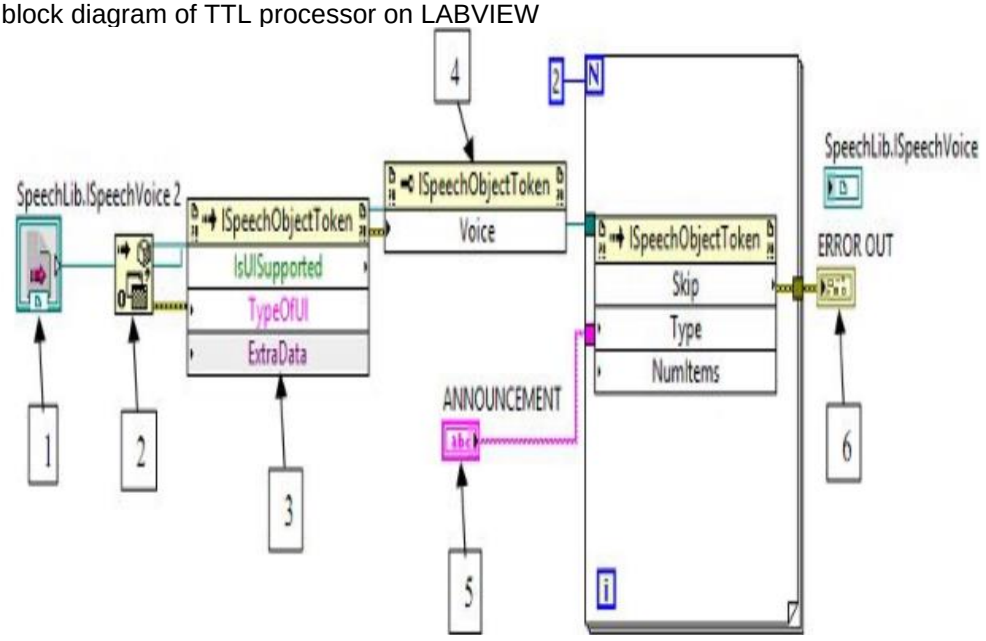


Figure 9: shows the block diagram of the LabVIEW bases text-to-speech processor.

Figure 10 Block diagram of LabVIEW based TTS Processor. The working of each block is explained as:

1. Speech lib.1speech voice2
2. Automation open

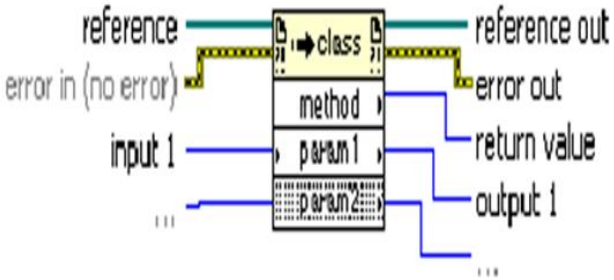


Figure 10: Automation open

Figure 10 shows Automation open. It returns automation Regnum, which points to a specific ActiveX object.

- Automation Regnum provides the object type for the Automation Regnum output.
- Machine name indicates on which machine the VI should open the Automation Regnum. If no machine name is given, the object is opened on the local machine.
- If open new instance is TRUE, LabVIEW creates a new instance of the Automation Regnum. If FALSE (default), LabVIEW tries to connect to an instance of the regnum that is already open. If the attempt is unsuccessful, LabVIEW opens a new instance.
- Error in describes error conditions that occur before this node runs. This input provides standard error in functionality.
- Automation Regnum is the regnum associated with an ActiveX object.

- Error out contains error information. This output provides standard error out functionality.

3. Invoke node

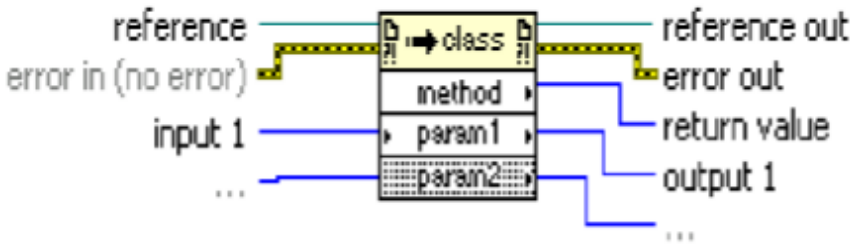


Figure 11: Invoke node

Figure 11 shows Invoke node. It invokes a method or an action on a reference. If the node is configured for VI Server Application class or Virtual Instrument class and reference is unwired, reference defaults to the current Application or VI.

LabVIEW includes Invoke Nodes preconfigured to access XML methods, NET methods and ActiveX methods.

- Reference is the refnum associated with the object on which you want to invoke a method or perform an action. If the Invoke Node class is Application or VI, you do not have to wire a refnum to this input. For the Application class, the default is the current application instance. For the VI class, the default is the VI containing the Invoke Node.
- Return value is an example return value of a method.

4. Property node

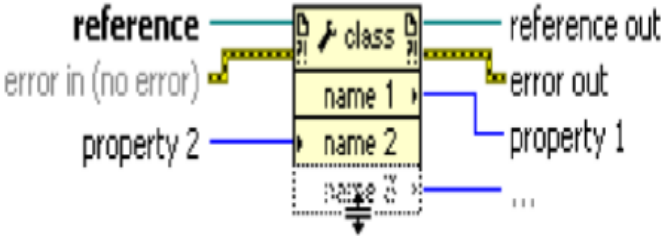


Figure 12: Property node

Figure 12 shows Invoke node. It reads and/or writes the properties of the reference. The Property Node automatically adapts to the class of the object that you reference. Property 1…..n, property 2…..n are the examples of the property you want to get.

5. Announcement

It gives the platform to write the text which is going to be converted to speech.

6. Error out

It is the cluster of three elements:

1. Status (Boolean TRUE or FALSE).
2. Code (long 32bit integer).
3. Source (string).

- Applications

The TTS processor has a wide variety of applications. It will soon benefit society and become advantageous. Because it can be utilized by a variety of impaired persons and can help them become more like regular people, it removes the barrier of disability for those who face discrimination in society. Below are some of the most significant applications:

The TTS converter may be installed anywhere, including on computers and mobile devices, for persons with dyslexia who have trouble reading text.

- It is quite helpful for those who are less clever. They have a lot of difficulty getting the other person to comprehend their needs. Therefore, they may quickly enter in the TTS processor's announcement window and speak.
- It can be used to teach the pre-illiterate children. As it can teach them how to pronounce the words in their basic stage of learning.
- It can be used in gaming and animations.

- Evaluate the performance of speech recognition system

To evaluate the performance of a speech recognition system, there are several metrics that can be used. Here are some commonly used metrics:

1. Word Error Rate (WER): WER is the most commonly used metric for evaluating speech recognition systems. It measures the percentage of words that are incorrectly recognized by the system. WER is calculated as the number of word substitutions, deletions, and insertions divided by the total number of words in the reference transcript.
2. Accuracy: Accuracy is the percentage of correctly recognized words. It is calculated as the number of correctly recognized words divided by the total number of words in the reference transcript.
3. Precision and Recall: Precision measures the proportion of correctly recognized words out of all recognized words, while recall measures the proportion of correctly recognized words out of all reference words. These metrics are useful for evaluating the trade-off between precision and recall in the system.
4. F1 Score: F1 score is the harmonic mean of precision and recall. It provides a balanced measure of the system's performance, taking into account both precision and recall.
5. Confusion Matrix: A confusion matrix shows the number of correctly and incorrectly recognized words for each word in the reference transcript. It can provide insight into which words are more difficult for the system to recognize.
6. Processing Time: Processing time measures the time taken by the system to recognize speech. This metric is important for real-time applications, where processing time can affect the system's usability.

Overall, a combination of these metrics can be used to evaluate the performance of a speech recognition system. It is important to choose the most appropriate metrics based on the application and the goals of the system [16].

Practical Side:

Figure 13 shows examples of the devices used in the research to be controlled in terms of lock or open voice by mentioning the name of the device, and we have allocated four devices (TV, LAMP, FAN, and Gate) and can add an unspecified number of devices.

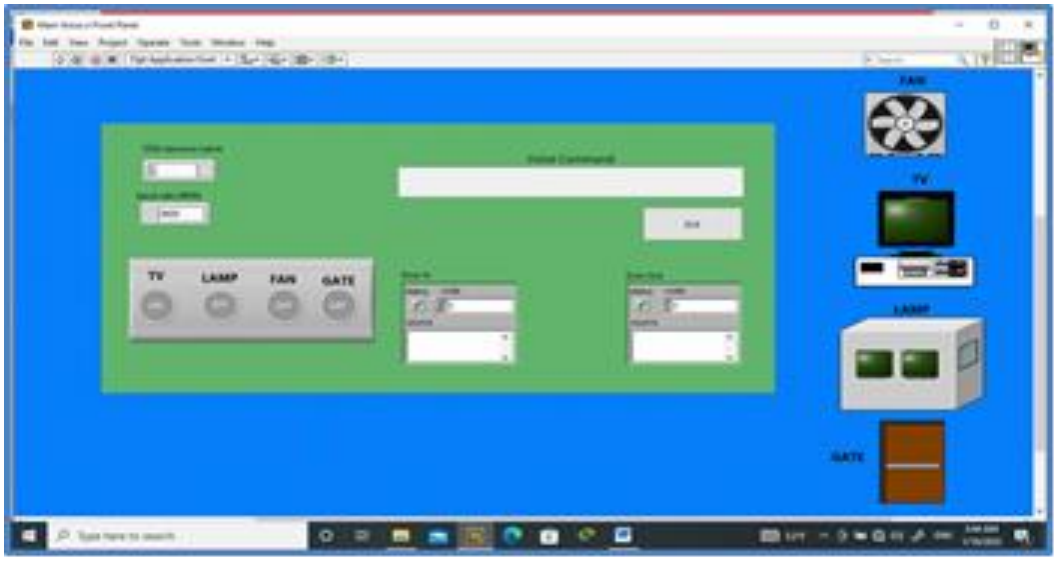


Figure 13: Lab View Applied Interface.

The idea of the work of the paper

The idea of painting the research is summarized in the following points

1. The operation time of the devices can be controlled by setting time in advance and when operating more than one device can determine the time of each device separately.
2. Before running the interface of the research (RUN) is preparing to connect the Arduino with the LabVIEW through the computer by the so-called (COM) and the relevant part is on the front interface on the image of VISA.

3. When operating the system and giving the voice command, we notice on the front of the research the lighting of the indicators of the device that has been turned on or locked, which is the LED lights.
4. Operating the practical part of the circuit, which is the electrical devices, must be synchronized with the indicators (LED) on the front of the paper.

Block Diagram interface (code):

Figure 14 shows the main parts of the program code in the language of the view, the first part (1) of the code relates to the sound (a sub-vi which is stored by the words stored in words for voice commands such as TV, fan and others, Part II) 2) is an event structure. It represents two or two cases. The first case represents the exit from the program. The second case contains the third part (3), which is several case cases. Each case is assigned to a specific voice command to operate the device or two or more devices on demand. (4) With respect to the voice response, for example, when the voice command is given to operate a device, a voice message is issued N device has been run and so for other cases.

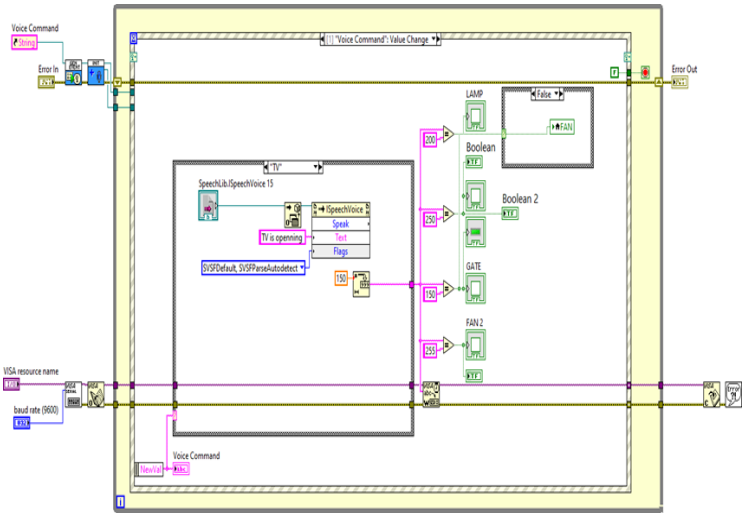
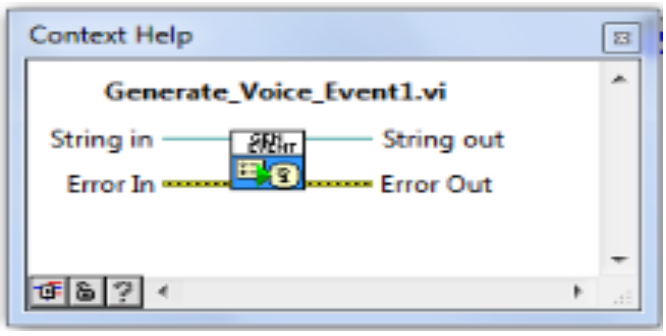


Figure 14: the box layout (lab view code).

- Speech recognition

As we explained earlier in figure 15, Speech recognition consists of two tools:

1. Generate-Voice-Event tool: Figure 15 shows the code for the tool.



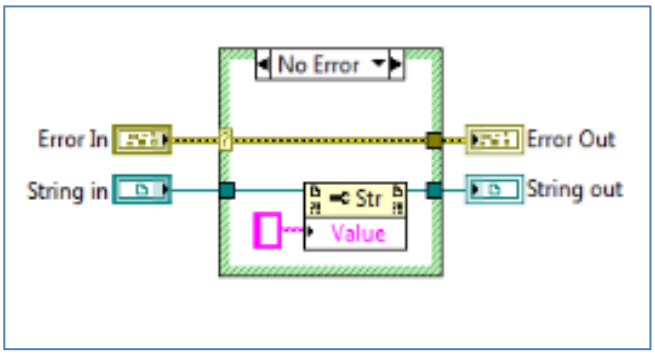


Figure 15: Generate-Voice-Event.

2. Voice-Init tool: Figure 16 shows the code for the tool.

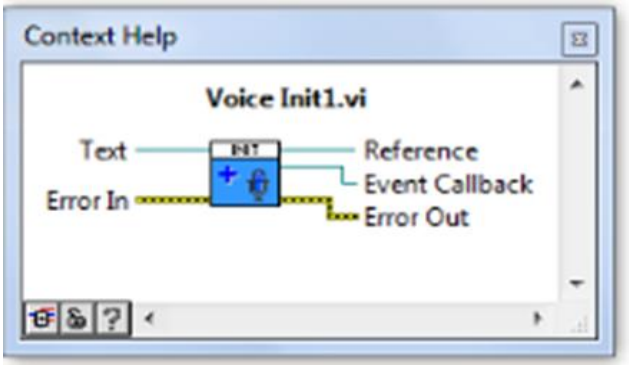


Figure 16: Generate-Voice-Events.

3. The interface SpeechLib. ISpeechVoice is a part of the Microsoft Speech API (SAPI), which provides functionality for text-to-speech synthesis. The ISpeechVoice interface represents a speech synthesis voice, allowing control over various aspects of voice output.

Some of the key methods and properties provided by the ISpeechVoice interface include:

Methods:

- Speak: Allows the voice to speak a given text or a prepared SpeechLib. ISpeechStreamFile.
- SpeakStream: Allows the voice to speak from a SpeechLib. ISpeechBaseStream.
- Pause: Pauses the speech output.
- Resume: Resumes speech output after it was paused.
- Skip: Skips a specified number of items in the speech output.
- SkipStream: Skips a specified number of items in the speech output stream.
- WaitUntilDone: Blocks the execution until all queued speech is completed.

Properties:

- Voice: Gets or sets the speech synthesis voice used for output.
- Volume: Gets or sets the volume level of the voice output.
- Rate: Gets or sets the speaking rate of the voice output.
- Status: Gets the current status of the voice, such as speaking, paused, or idle.
- AudioOutput: Gets or sets the audio output device for the voice.

Figure 17, 18 shows the control interface during operation where it is evident in the LED lighting indication that the device is ON.

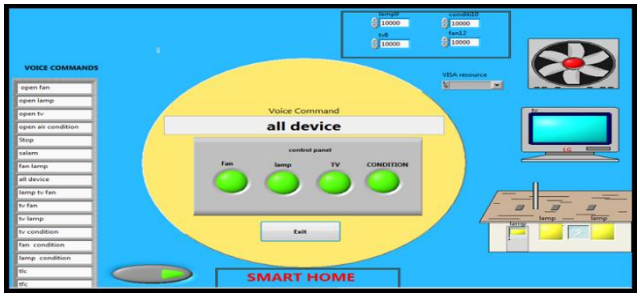


Figure 17: Control interface during the operation of the entire device.

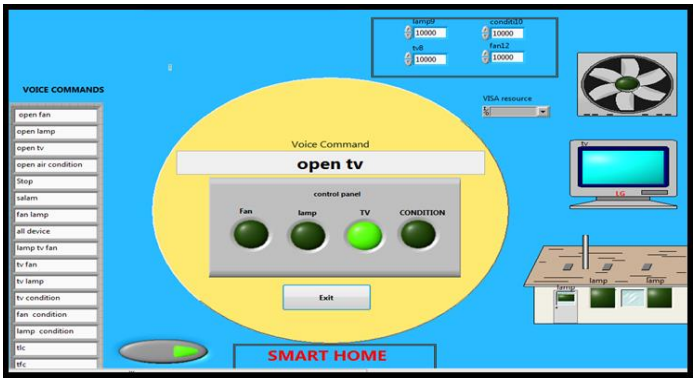


Figure 18: Control interface during Open Tv operation.

Conclusions

- The research demonstrates the potential of deep learning for speech recognition in smart home devices, aiming to enhance the user experience.
- Specific speech datasets were collected and pre-processed for training deep learning models.
- Various deep learning techniques were employed to achieve accurate real-time speech recognition, even in challenging environments.
- Techniques such as data augmentation and transfer learning were explored to improve model performance.
- Privacy and security implications of speech recognition in smart home devices were addressed, with considerations for protecting user data.
- The research contributes to the field of speech recognition in smart homes and provides insights into enhancing user experience and ensuring privacy and security.
- Future work could involve further optimization of models, exploring additional techniques for data protection, and enhancing the security of smart home devices.

References

[1] M. Haleem, “Voice Controlled Automation System”, IEEE International Multitopic Conference, 12th pp.508-512, December 2008.

[2] R. Larsen, “LabVIEW for Engineer ”, Pearson, 1st edition.

[3] M. El-Hawary, “Introduction to Electrical Power Systems”, Wiley-IEEE Press, 1st edition.

[4] R. Bitter, T. Mohiuddin, and M. Nawrocki, “Introduction to LabVIEW", Boca Raton: CRC Press LLC,ch.1, 2007, pp.1-10, 2nd edition,.

[5] J. Kring, "LABVIEW for everyone", Prentice Hall, 3rd edition, July 2006.

[6] P. Felzenszwalb, Daniel. Huttenlocher, “Pictorial Structures for Object Recognition,” International Journal of Computer Vision, vol. 61, no. 1, Jan 2005, pp. 55-79,.

[7] G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, and A. R., Jaitly. “Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups”, IEEE Signal Processing Magazine, 29(6), October 2012, pp.82-97.

[8] A. Graves, A. Mohamed, and G. Hinton, “Speech recognition with deep recurrent neural networks”, In Acoustics, speech and signal processing (ICASSP), IEEE International Conference on Acoustics, May 2013, pp. 26-31.

[9] D. Amodei, S. Ananthanarayanan, R. Anubhai, R., and et.al, “Deep speech 2: End-to-end speech recognition in English and Mandarin”. In Proceedings of the 33rd International Conference on Machine Learning, Vol 48, 2016, pp. 173-182.

[10] A. Mehrish, N. Majumder, R. Bhardwaj, R. Mihalcea, S.Poria, “ A Review of Deep Learning Techniques for Speech Processing”, arXiv: 2305.00359v3, May 2023, 181(30), pp.1-7.

[11] S. Irfan, N. Anjum, N.Masood, A. Khattak, N. Ramzan, “A novel hybrid deep learning model for speech recognition in smart home devices”, Journal of Ambient Intelligence and Humanized Computing, December 2021, 21(24), pp.2647-2658.

[12] L. Silva, C. Morikawa, I.Petra, (2020). Speech recognition in smart homes: A review of the state-of-the-art and future directions. IEEE Internet of Things Journal, 25(7), 2020, pp.3692-3705.

[13] D. Bouchabou, S. Nguyen, C. Lohr, B. Leduc, and L. Kanellos, “A Survey of Human Activity Recognition in Smart Homes Based on IoT Sensors Algorithms: Taxonomies, Challenges, and Opportunities with Deep Learning”, Journal of Ambient Intelligence and Humanized Computing, 21(18), September 2021.

[14] B. Kowalski, T. Kajdanowicz, “Privacy and Security of Smart Home Systems: A Review”, IEEE Access, 7, 2019, pp. 121-131.

[15] H. Chen, & H. Hu, “A Survey of Privacy-Preserving Techniques in Speech Recognition Systems”, IEEE Transactions on Information Forensics and Security, Vol 16, 2021, pp.1463-1481.

[16] S. Kim, S. Lee, J. Kim, “Lightweight and Secure Speech Recognition for Smart Home Devices”. IEEE Transactions on Consumer Electronics, 66(4), 2020, pp.335-342.