Republic of Iraq Ministry of Higher Education and Scientific Research Babylon University College of Engineering

Arduino-Based Implementation of Machine Learning Algorithm

A Project

Submitted to the College of Engineering / University

Babylon in partial fulfillment of the requirements for the degree of

Higher Diploma in Engineering/Electrical Engineering/Electronics and Communications

Prepared by

Huda Raheem Mohammed Abd

Supervised by

Asst. Prof. Dr. Hilal Al-Libawy

2022 A.D **1443 A.H**

بسم الله الرحمن الرحيم

﴿ قَالُواْ سُبْحَانَكَ لاَ عِلْمَ لَنَا إِلاَّ مَا
عَلَّمْتَنَا إِنَّكَ أَنتَ الْعَلِيمُ الْحَكِيمُ ﴾

صَدَقَ اللهُ العَلِيُّ العَظِيمُ

(سورة البقرة)

٣٢

Dedication

For your bravery, I dedicate this project to God Almighty my creator, my strong pillar, my source of inspiration, wisdom, knowledge and understanding. He has been the source of my strength throughout this research and on His wings only have I soared. Secondly, I would like to dedicate my work to:

My support and strength in life …… My father.

To

The foundation of love and tenderness … My mother.

To

My every day smile, My daughter and sister.

And lastly To

My Secret of success …. My dearest friends.

Huda Raheem Mohammed

2022

Acknowledgements

In the name of Allah, the Most Gracious the Most Merciful

Alhamdulillah, all praises and thanks be to Allah for blessing me to accomplish this work despite all the hardships.

I would like to express my deepest sincere thanks of gratitude to my supervisor whom I'm very proud to work with Dr. Hilal Al-Libawy for their great ideas and valuable suggestion in accomplishing this project, and for their continuous Scientific, moral and financial support to finalize this work with in the limited time frame.

Also, I wish to convey my heartfelt thanks to my family and my husband and my friends for their elevating inspiration and kind encouragement along the period of my project.

Last but not least I spread my deepest thanks to all the teachers who helped me and I would like to respect the immortal favor they offered me.

Huda Raheem Mohammed

2022

Abstract

The spread of the Internet of Things (IoT) devices and the need for efficient and smart applications raise the necessity for low-cost and efficient power-consuming devices. Usually, these devices have limited resources, and it's not very easy for them to handle many applications that need intelligent solutions such as artificial intelligence, including deep learning.

One of the challenges to implement a smart algorithm on limited-resources devices is the difficulty of implementing highly demanded resources algorithms such as Convolutional Neural Networks, which is one of the types of deep programmed used in python, learning on the Arduino BLE 33 Sense because it is limited in resources. In this work, A CNN network is proposed to respond to voice commands. This network is trained and tested on the cloud server using our own audio dataset that includes 400 samples and four classes. However, this network demands too many computational resources. To overcome this challenge, we implemented it on a Tiny Machine Learning (TinyML) framework because the Arduino is limited in resources.

The modified network, using TinyML, was built and installed on two edge devices. The first one was implemented on the Arduino device, thus obtaining an accuracy of (98.8%). In contrast, the second one was implemented on the mobile phone, with an accuracy of (100%), Where was the time spent on training is (6m 50s) as for processing time is (328ms) where most of the sounds were classified correctly.

Table of Contents

List of Figures

List of Tables

List of abbreviation

Abbreviation	Definition
ML	Machine Learning
AI	Artificial Intelligence
ASR	Speech Recognition Software
DNN	Deep Neural Networks
DL	Deep Learning
DSSM	Deep Structure Semantic Models
QSAR	Quantitative Structure Analysis relationship
QSPR	Quantitative Structure Prediction relationship
CRM	Customer Relationships Management
CLV	customer lifetime value
CNN	Convolutional Neural Network
ReLU	Rectified Linear Unit
MFCC	Mel Frequency Cepstral Coefficient
DFT	Discrete Fourier transform
DCT	Discrete Cosine Transform

SVM	Support Vector Machine
KNN	K-Nearest Neighbor
ANNs	Artificial Neural Networks
ROC	Receiver operating Characteristic
DSPs	Delayed Sleep Phase
FPGAs	Field Programmable Gate Arrays
GPU	Graphics Processing Unit
ROM	Read Only Memory
IMU	Inertial Measurement Unit
IDE	Integrated development environment
BLE	Bluetooth Low Energy
RGB	Read, Green, Blue
NPN	Negative-Positive-Negative
LED	Light emitting diode

Table of symbol

Item	Definition
X_t	Discrete Fourier Transform
N	The number of DFT sample
X_n	Input speech signal
f_{Mel}	The frequency of Mel
f	The frequency of linear
C_n	Cepstral coefficients
C	The number of MFCC
K	No. of trials
n	Learning rate
$\Delta_{Cm}(n)$	Derivatives cepstral coefficients
n^{th}	Time frame
m^{th}	Time feature
k_i	Signifies
i^{th}	The weight
T	The number of frames
z^l	Pre-activation output of layer L
h^l	Activation of layer L
W	Learnable parameters

Chapter One
Introduction

Chapter One: Introduction

1.1 Introduction

The process of developing computer systems that can learn from data is known as machine learning (ML). Early Artificial Intelligence (AI) techniques dealt primarily with deductive reasoning. The derivation of theorems from axioms can be compared to Inductive inference is a feature of machine learning, i.e., generalizations from a collection of observed examples. ML is a subfield of AI that overlaps with various scientific fields, including statistics, cognitive science, and information theory. Joining ML can be distinguished by impersonation as well as adaptability. A system for machine learning must store the acquired ability of multiple structures to represent information, which are referred to as (inductive) hypotheses and often take the form of a sample. According to the principle of the Ockham Code, the training data should propagate a hypothesis with the best simplification of the hypothesis; To achieve a good generalization, the hypothesis must be simpler than the training data [1]. As shown in Figure 1.1, the learning algorithm determines how to perform an update in relation to the acquired hypothesis for the recent experience (i.e., the training data), in order to maximize the execution measurement with relation to the mission.

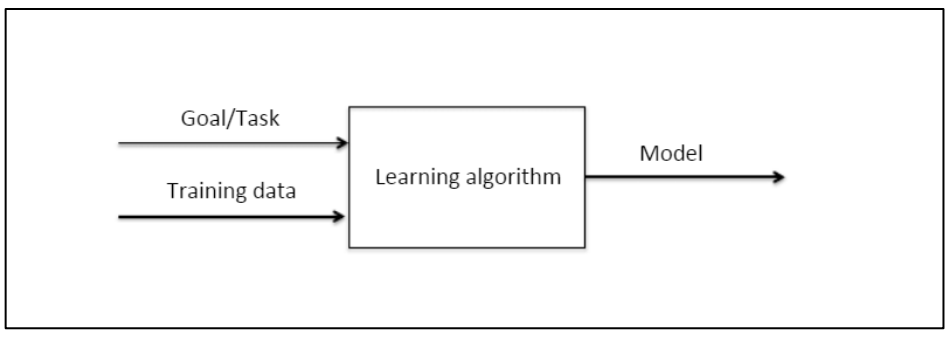

Figure 1.1: An approach of machine learning that is general

Machine learning approaches have been used to tackle a variety of real-world issues throughout the years, including speech recognition, fraud detection, customer connection department, gene position prediction, and so on. Consider the job of classifying email messages as spam or non-spam, wherever the machine learning method's success is measured by the proportion of emails properly classified. The training expertise in this subject might come in the shape of a database of emails that people have categorized as spam or no-spam.

The number of devices linked to the Internet is growing, allowing huge volumes of data to be exchanged and transforming the Internet into a 21st-century data silk road. This path has led to new applications for machine learning. Machine learning models, on the other hand, are not yet regarded as complicated systems that require expensive computers to execute (i.e., Cloud). These models are being incorporated into increasingly computationally limited devices as technology, methods, and algorithms progress. One of the applications is human programs a computer including the microcontroller on Arduino board by telling it what to do. ML is distinct in that the Arduino board is taught via the use of examples. Images, sounds, or any other form of sensor data might be used as examples .

After some learning, the Arduino can decide if it hears a term, detects movement, sees a person, or performs a variety of other tasks. Rather than transmitting all data to the cloud, performing machine learning on a microcontroller has privacy and efficiency advantages. With additional tools, approaches, and examples, machine learning is becoming easier on Arduino boards. Embedded ML or Tiny ML refers to the trend of executing machine learning on tiny devices. Machine learning algorithms exist in an assortment of shapes and sizes, each with its own set of benefits and drawbacks. Because of

the recent success of deep learning techniques, neural networks have sparked a lot of interest. However, there are traditional machine learning methods that are easy to comprehend, perform well with little amounts of data, and are ideal for embedded devices [2]. Deep Learning is a more sophisticated machine learning technique based on brain cell activity (neurons). It can extract hidden characteristics and deal with more complicated problem statement.

1.2 Problem Statement

1. The difficulty of machine learning and its implementation on hardware, so we are looking for how to implement machine learning or deep learning on Arduino devices with limited resources.
2. The micro-computer was relied upon because of the high cost.
3. Training the training and testing data in the cloud server because it contains ready-made blocks without the need for programming from the beginning.

1.3 Thesis aim

1. It was directly relied on the microcomputer instead of using the computer because of the high cost.
2. The time of data was processed took 328ms.
3. Use the benefit of cloud in processing of deep learning, to decrease the complexity.
4. Change the idea of implementing CNN from complex devices to simple devices.
5. Implement a voice recognition device with high accuracy.

1.4 Literature review

In 2019 , Sérgio Branco al. [3] shows that number of devices connected to the Internet is growing, allowing vast volumes of data to be exchanged. The authors examine the optimizations, techniques, and platforms used to integrate such models at the network's end, where extremely resource-constrained microcontroller units (MCUs) reside. Designed with neural networks, MCU sets a record ImageNet accuracy on MCU (71.8%), and achieves >90% accuracy on the visual wake words dataset under only 32kB SRAM.

In 2020, Aditya Priyadarshi1 al . [4] because it uses the support vector machine (SVM) technique, the one in this work gives a significantly more accurate output as well as a faster computing speed. This motion recognition system tracks a device's motion in mid-air in a 3D space, logs its speed, angular velocity, distance traveled, and then converts the device's motion data into English alphabet characters. This paper's device proposes a solution based on support vector methods and analyzes some of the issues produced by the device.

In 2020, Aybuke Kececi al. [5] hugaDB, an open source database, was used in this research. Running, sitting, walking, and standing are just a few of the human behaviors that are combined in this database. The data was collected using body sensor networks with six inertial sensors that can be worn on the right and left thighs, calves and feet. In a set, 211,962 samples were collected for all 18 subjects, yielding a total of 10 h for the data. The Random Forest classifier is very accurate, with an overall accuracy of over 95%.

In 2021 Vyacheslav Lyashenko al. [6] in this article, the authors consider how phonics condition setting, learning rates, and training set value influence the accuracy of vocal command pronunciation models. Speech recognition systems based on neural networks have been proposed.

The comparison is made for the two systems developed by Google Speech Recognition and Pocket Sphinx. The proposed system can recognize spoken commands with an accuracy of 84.4 percent.

1.5 Contribution

1- The application of artificial intelligence techniques (deep learning) on the Arduino device has been successful with limited resources because it is cheap and its power consumption is very low.

2- That's why in the framework I resorted to the use of Tiny ML, which was placed on the microcontroller, which made it possible to implement a convolutional neural network.

3- As a result, the work of the Arduino was to download the network inside it, and the network, in turn, classified the device itself.

1.6 Thesis organization

The rest of this paper is organized as:

Chapter Two: includes an overview of the most utilized hardware and software. A comprehensive overview of AI has been discussed additionally to machine learning and machine learning algorithms. It moreover has an explanation of ML applications and deep learning.

Chapter Three: The third chapter discusses the issue description as well as the proposed solution. Which includes block diagrams and algorithms. It shows

implementation steps of machine learning on Arduino kit. In addition, it contains an explanation of steps to design a trained automated model.

Chapter Four: Chapter four shows project implementation steps and results. How ML implemented into Arduino and basically how Arduino programmed.

Chapter Five: In this chapter, the conclusion and future work are proposed.

Chapter Two
Theoretical Background

Chapter Two: Theoretical Background

2.1 Introduction

The following chapter contains the main overview of concepts contributed in this project. It shows artificial intelligent introduction, machine learning introduction, deep learning introduction with some examples, limitations and applications; Moreover, it presents an overview of neural network which takes part into the project.

2.1.1 Artificial Intelligent (AI) Overview

Human curiosity prompts him to wonder, "Can a machine think and act like humans?" while leveraging the computing power of computers as a result, AI was developed with the objective of reaching the same degree of intelligence in computers as we appreciate in humans. This day, AI is vastly employed while implemented in practice to facilitate organizations in business better resolution and improving person tests. AI entails acting in an identical manner to human being, benefit algorithms, however in a more justice, effective, and faster manner. In addition to the information obtained by running a series of algorithms with small or no human being interaction, humans can improve their general thinking skills, use situational actions, and make more informed judgments [7].

The first step in applying AI and automation is to define a use case. From unlocking our phones with face recognition to self-driving automobiles, artificial intelligence is all around us. Siri, Alexa, and Google Assistant are AI-enabled devices used in this life today.

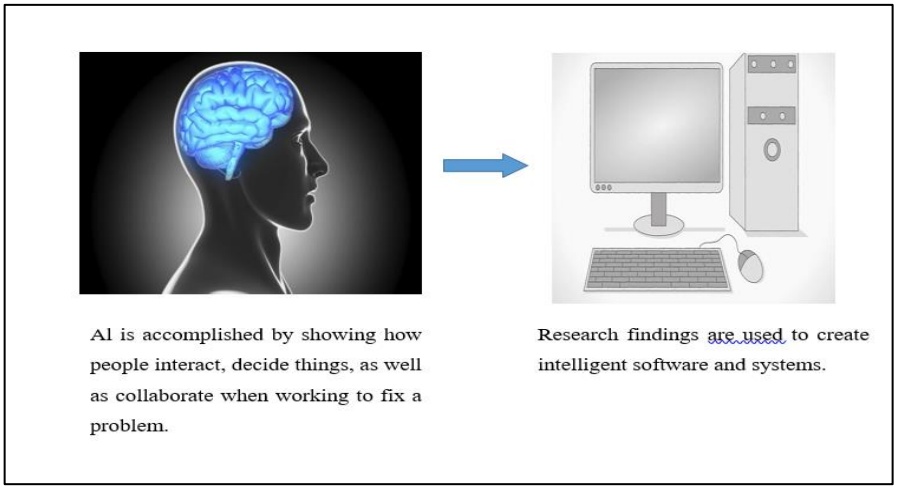

Figure 2.1: AI in Human and Computer [7]

2.1.2 Application of Artificial Intelligence (AL)

Artificial Intelligence has controlled a wide range of fields, such as:

1- Gaming Applications

AI plays a crucial part in strategic games such as chess, poker, tic-tac-toe, and others since the computer can think of a huge range of possible places based on theoretical information [8].

2- Processing of Natural Language

It is possible to communicate with a computer that understands human natural language.

3- Systems of Expertise

To give reasoning and recommendations, several applications combine machine, software, and special data. They assist consumers by explaining things to them and offering advice [9].

4- Vision Systems

These systems understand, analyze, and understand visual information on the computer [10].

- A spy plane, for example, captures photos that are used to deduce spatial information or a map of the region.
- To diagnose the patient, doctors employ a clinical expert system.
- Police employ computer software that compares a criminal's face to a forensic artist's stored portrait.

5- Recognized Speech

While a person speaks to it, certain smart systems have the ability to hear and comprehend language in terms of sentences and meanings. It can handle a wide range of languages, slang words, background noise, temperature-related variations in human sounds [11].

6- Recognition of Handwriting

Handwritten tracking technology interprets version written for a liner and ink or a slab on a computer. It can recognize letterforms and transform them to text that may be edited.

7- Smart Robots

Robots are appropriate of doing human-such jobs. They are armed for sensors that sense corporal data from the active environment, such light, heat, motion, noise, vibration, and pressure are all factors to consider. To explain intellect, they have potent computers, many sensors, while a great memory. They also have the ability to learn from their errors and adapt to new conditions [12].

2.2 Machine learning

Machine learning (ML) is a type of technology that has been able to change the rules in the competitive world of companies, study computer algorithms that improve automatically through experiment and study data, and spread these algorithms in organizations reliably and well, as well as companies build machine learning on a large scale. (ML) has become so widespread that it has become the main way for companies to solve a set of problems, which is considered a branch of artificial intelligence because education requires intelligence to make critical decisions, as it is an important application of artificial intelligence [13].

2.2.1 Example of Machine learning

1- Identifying and distinguishing items (faces, sound, pictures, letters…).

2- Assisting in the development of medicines to treat incurable illnesses [14].

3- Internet search engines, data security, and marketing.

4- Providing answers to questions (through the data bank).

5- Simulating the human brain's network of neurons to make judgments and suggestions [15].

2.2.2 Machine learning algorithms

It is a collection of programs that explain general principles of data processing's and finding correlations and patterns in the data by using statistical and mathematical calculations. Each algorithm has its own set of features and outputs, that allowing it to represent data in a variety of ways or anticipate new data outputs depending on the correlations. You can see the ML algorithm in

figure (2.2). There are many different machine learning algorithms, and we'll go through a few of them [16]. Are discussed as follows:

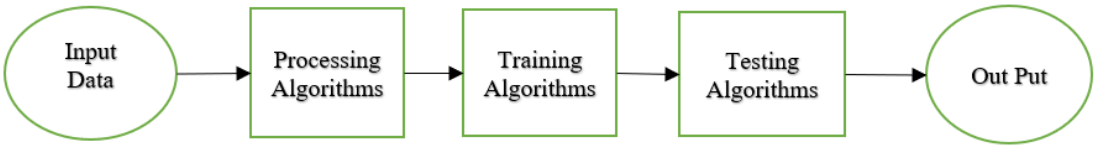

Figure 2.2: Machine learning algorithms

2.2.3 Type of Machine Learning

1- Supervised Learning

This is the most popular type of machine learning. In which the machine is educated by the use of data and data output, this is referred to as supervision, and it is oversee knowledge by providing data outputs figure (2.3) show the general about supervised learning. Although, because need to renaming data is needed to operate properly, this type provides a collection of data for the machine learning algorithm to work with. The capacity to find new patterns by training them to fresh data will be available [17].

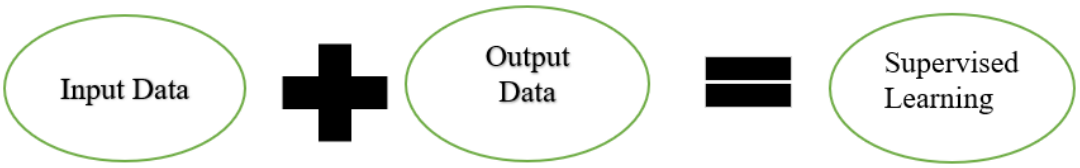

Figure 2.3: Supervised Machine learning

2- Unsupervised machine Learning

Unsupervised learning can work with data that isn't labeled. This means that the computer is only taught by supplying it with data and so it learns to detect correlations and patterns through training. Because of this data output prediction, no human interaction is necessary to create a collection of machine-readable data, allowing computers to work with considerably bigger data sets. Here, the training process, on the other hand, does not have labels to cope with, which leads to confusion in hidden structures [18]. The ability of unsupervised learning algorithms to create these hidden structures is what makes them so flexible. Unsupervised learning algorithms may adapt to the data by modifying the underlying structures dynamically, rather than using a predefined and precise problem statement [19].

3- Learning through Reinforcement

Reinforcement learning is influenced by how individuals learn from data in many aspects of life since it uses self-developing algorithms and a trial-and-error technique to understand when negative output is amplified and discontinuous outcomes are really not [19]. This sort of algorithm works by putting algorithms in a certain environment with the presence of a compiler and a reward system, with the output result being supplied to the compiler every time the algorithms are repeated. This determines whether or not this outcome is sequential, and offer a reward to the algorithms, but if the outcome is unsatisfactory, they will iterate until they achieve a better result, so the program is trained to provide the greatest possible result for the best possible reward [20]. Figure (2.4) shows the Reinforcement learning.

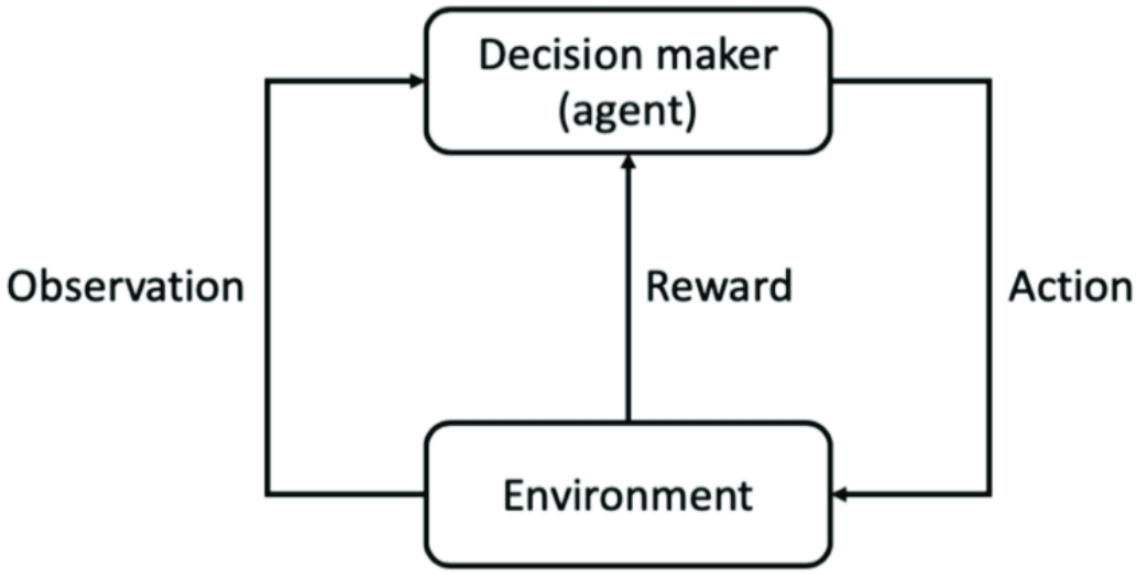

Figure 2.4: Reinforcement Learning [20]

2.2.4 Limitation of machine learning

Feature extraction is one of the most difficult aspects of conventional Machine Learning models. -This is a major difficulty for complicated problems like object identification or handwriting recognition[21] .

2.2.5 Applications of Machine Learning

Machine learning is applied in a wide range of fields, including health, banking, social media, travel, email spam and malware screening, online customer service, search engine result refinement, product suggestions, and more. Some of these are described in further detail below [22].

1. Health:

2. Finance:

3. Social Media

4. Travel:

5. Email Spam and Malware Filtering:

6. Online Customer Support:

7. Product Recommendations:

8. Natural Language Processing:

9. Picture processing and computer vision:

10. Automotive Industry

2.3 Deep learning

Deep learning is a type of machine learning that involves the use of large amounts of data. Entails a more advanced level of learning. In difficult tasks, deep learning models are extremely successful. The deep learning model may reach high levels of accuracy that are sometimes superior to human ability [23].

2.3.1 Examples of deep learning

1- Devices like as phones, TVs, and speakers may be controlled with voice commands [24].

2- Carry out categorization tasks using pictures, text, or audio.

3- Autonomous driving: Automotive researchers utilize deep learning to automatically recognize items like stop signs and traffic lights.

4- Electronics: used in automated speech and hearing translation.

5- Games and competitions: finding answers to a game with the fewest moves, for example [25].

2.3.2 Limitations of deep learning

- Deep learning requires a large amount of data to educate [25], and neural networks are easily fooled.
- Deep learning's accomplishments are entirely experimental, and learning algorithms have been dubbed "sensor arrays" for their incomprehensibility.
- Deep learning hasn't been adequately connected with prior knowledge up to this time [26].

2.3.3 An application for deep learning

This section lays the groundwork for this analysis by examining different domains where the deep learning algorithm has been used.

1. Speech Recognition Software (ASR)
2. Recognition of images
3. Processing of Natural Language
4. Toxicology and Drug Discovery
5. Management of Customer Relationships
6. Bioinformatics

2.4 Convolutional Neural Network (CNN)

Because biological processes in the human brain’s visual cortex inspire it, a convolutional neural network (CNN, or ConvNet) is a type of deep neural network that is most often used in image processing [27].

Convolutional neural networks are made up of several layers of artificial neurons. Natural neurons with mathematical functions that add several inputs and output the activation value are known as artificial neurons. They're in that layer when you summon them. Horizontal edges or a queen are extracted from

normal initial models. This output is sent to the outside from the output. As we move deeper into the network, we may choose more sophisticated like items, etc., as seen on the figure (2.5) [28]. The layer that makes up CNN.

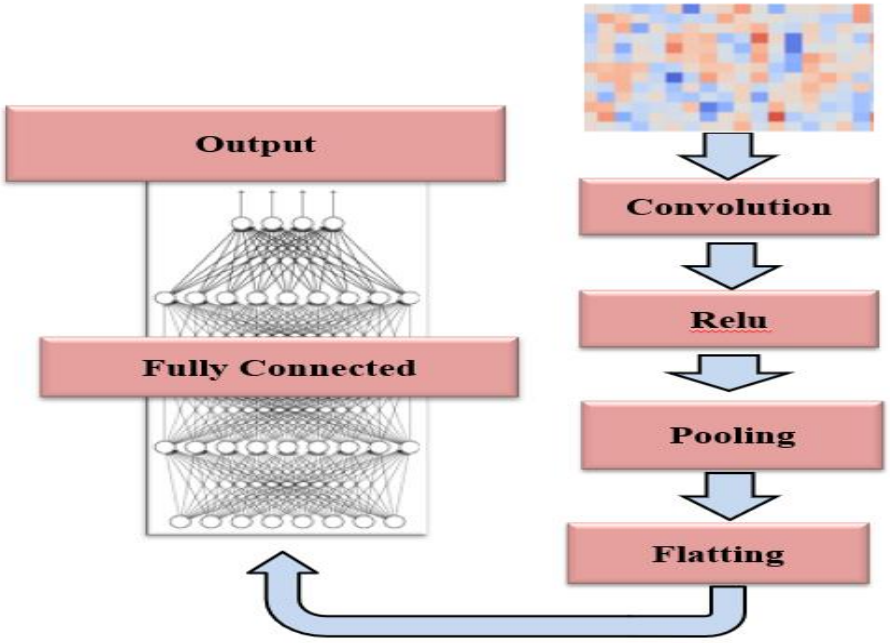

Figure 2.5: Convolutional Neural Network (CNN) [28]

1- The first layer utilized to extract features from an input data is the convolution layer [29].

2- Pooling Layer: When the data are too huge, pooling layers refers to the number of parameters [30]. Pooling decreases the size of each map while preserving the essential information. Pooling can be done in a variety of ways:

a. Average Pooling

b. Maximum Pooling

c. Pooling of sums

3- Flattening: Flattening is the simplest stage, which involves arranging each row of a feature map into a single column. As input-to-input layer, this column value will be used [31].

4- Activation Function: Any neural network's output end includes an activation function. It aids in the mapping of values from 0 to 1 or -1 to 1 depending on the function. Linear and non-linear activation functions are the two types of activation functions [32].

2.5 Mel Frequency Cepstral Coefficients (MFCC)

It is recommended that you employ MFCC features since they are common capabilities in speech processing and may be found in a variety of software packages. Various items are included [33].

1. Pre emphasis: where concentration mentions to refinement that accentuates the height of the release, with the goal of balancing the spectrum of high-frequency voice sounds with a steep slope [34].

2. Blocking and windowing of frames: to achieve sound parameters that are relatively steady, the speech signal is a variable signal that changes slowly or practically continuously. Speech must be evaluated in a relatively short amount of time. This is accomplished through the use of continuous speech analysis on clips. The voice signal is meant to be fixed across it, therefore it's short [35].

3. DFT: Each window frame is repurposed a size spectrum by using the DFT technique according to the mentioned equation as

$$x(k) = \sum_{n=0}^{N-1} x(n) e^{\frac{-j2\pi nk}{N}} ; 0 \leq K \leq N-1 \qquad (2.1)$$

where N is the number of DFT sample [36].

4. Mel spectrum: With the aid of the well-known interference window, we assign the spectra to a slope scale in order to obtain an approximation of the energy in each location [37].

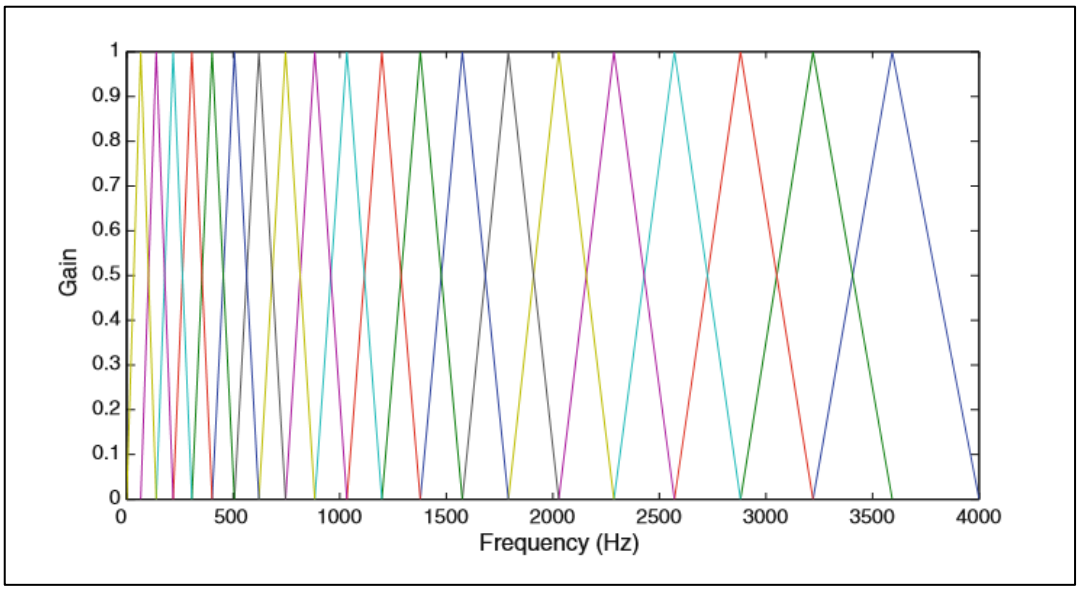

Figure 2.6: Mel scaling mapping [37]

When it comes to Mel scaling map, the scaling of the real particular frequency (Hz) and the measurement of perceived frequency (Mel) must be done at the same time. As indicated in figure (2.6), logarithmically. And the slope approximation of the frequency can be expressed as free, where the formula is given as follows [38].

$$\mathcal{F}_{\text{Mel}} = 2595 \log_{10}(1 + \frac{f}{700}) \qquad (2.2)$$

It is simple to obtain a sufficient estimation of the energy in a location by using a filter bank with the a proper spacing. To store the buffer and calculate the first 13 coefficients using DCT, the register of energies known as a mile spectrum can be used. These increased the numbers that represent a faster estimate of the estimated energies and thus have less information to be used in the classification of data where the first 13 coefficients are calculated using DCT and disposal. [39].

5. Discrete Cosine Transform (DCT)

DCT is applied into the MFCC after they have been converted. Before computing the DCT, it generates a large number of Cepstral coefficients.

$$C(n) = \sum_{m=0}^{M-1} \log_{10}(s(m)) \cos(\frac{\pi n(m-0.5)}{M}) \qquad (2.3)$$

The cepstral coefficients are c(n), and the number of MFCC is C. Only the septal coefficients 8–13 are used in MFCC systems, and the zero parameter is frequently ignored because it reflects the rate log power from the input indicative, which only contains a few Speakers information [40].

6. Dynamic MFCC features

The cepstral coefficients are usually referred to as static features, since they only contain information from a given frame. The extra information about the temporal dynamics of the signal is obtained by computing first and second derivatives of cepstral coefficients [7–9]. The first-order derivative is called delta coefficients, and the second-order derivative is called delta–delta coefficients. Delta coefficients talk about the speech rate, and delta–delta coefficients provide information similar to acceleration of speech. The commonly used definition for computing dynamic parameter is [41].

$$\Delta_{Cm}(n) = \frac{\sum_{i=-T}^{T} k_i C_m(n+i)}{\sum_{i=-T}^{T} |i|} \quad (2.4)$$

where $\boldsymbol{Cm(n)}$ indicates the $\mathbf{n^{th}}$ time frame's $\boldsymbol{m^{th}}$ feature, $\boldsymbol{k_i}$ signifies the $\boldsymbol{i^{th}}$ weight, and T is the number of frames utilized for computation. T is usually considered to be 2. The delta–delta coefficients are computed by taking the first-order derivative of the delta coefficients [42].

2.6 Tiny Machine Learning (ML)

Tiny ML is a project that tries to make machine learning (ML) applications run on low-power devices like microcontrollers. In order to operate ML applications, edge devices usually need to be linked to data centers. Tiny ML's three most promising approaches these approaches have been successfully used in a variety of applications that have been commercialized in a variety of goods. These products can be found in domestic, office or industrial scenarios, you can find it in the following application:[43].

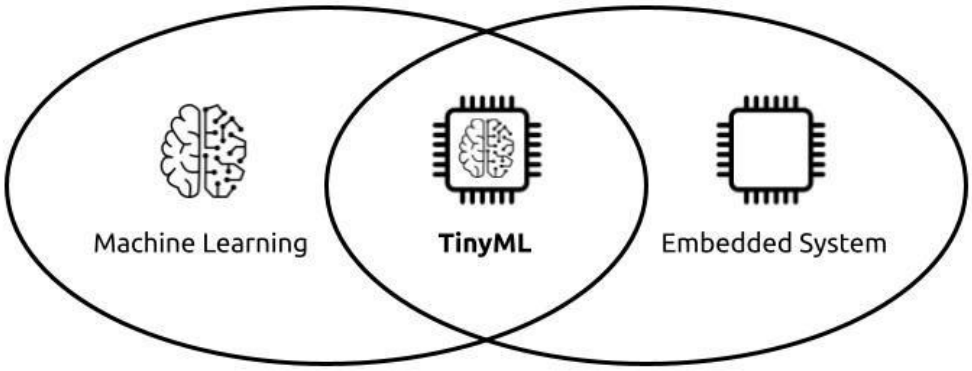

Figure 2.7: Tiny Machine Learning (ML)[43]

1. Keyword Spotting

Keyword spotting (KWS) is a speech recognition approach that uses a brief sound clip to identify certain words.

2. Visual Wake Word

The extension of keyword detection for visuals is the visual wake word approach. The application process follows a similar pattern: A camera continuously captures photos from the surroundings in order to determine if a certain object or person is in the image (using a machine learning model) [43].

3. Anomaly Detection

Anomaly detection is a method for detecting unusual occurrences. Anomaly detection, unlike prior systems, uses an unsupervised learning approach, in which the model must uncover patterns in unlabeled data [44].

2.7 Classification

When it comes to studying statistical problems, classification is a crucial technique. Classification is the challenge of identifying whether an item belongs to a specific class based on a previously learnt model in machine learning or statistics [44]. Based on a previously established set of training data, this model was statistically trained. Supervised learning is the term for this approach. Classification and regression are the two elementary types from supervised methods. In degradation, it produces variable takes continuous rate, but during classification, the production variable takes class labels. In machine learning, classification is crucial, especially for future planning. During the training and classification phases, it's possible that there are missing values in the data set cause issues [45]. Preparers can ignore omitted data, swap the complete omission values with a single global constant, swap the omission value with the mean of the chosen class, and manually monitor samples with omitted values and insertion of a potential or likely value to overcome the

problem of missing data. Bayesian characterization and decision trees are the most basic classification methods, whereas sophisticated ordering approaches include Support Vector Machine (SVM), K-Nearest Neighbor (KNN), and Artificial Neural Networks (ANNs). One of the most frequent diagnostics for measuring planned probability is the ROC curve. There are generally four different sorts of categorization jobs that you may encounter [46].

- Classification with several categories
- Ratings on multiple stickers
- Unbalanced Rating.

2.7.1 Classification performance metrics

The accuracy with which a classifier can predict is a critical factor to consider when selecting a classifier. The accuracy and other performance metrics can be used to compare different types of classifiers and determine which one is the best**.**

Four expressions are used in numerous performance measures: [47]

- **True positive (TP):** refers to the number of times the classifier properly identifies the positive class as positive.
- **True Negative (TN):** refers to the number of times the classifier properly predicts the negative class as negative.
- **False Positive (FP):** This term refers to the number of times a classifier wrongly predicts a negative class as a positive.
- **False Negative (FN):** This is the number of times the classifier predicts the positive class as negative. From the confusion matrix, here are some of the most frequent performance measures you can utilize.

Table 2.1: Confusion matrix in its most general form

Predicted class	**Actual class**				
	Class	Left	Noise	Right	Stop
	Left	TP	FP	FP	FP
	Noise	FN	TN	TN	TN
	Right	FN	TN	TN	TN
	Stop	FN	TN	TN	TN

The accuracy the percentage of total samples correctly identified by the classifier [48]. Use the following formula to calculate accuracy:

$$ACC = \frac{(TP+TN)}{(TP+TN+FP+FN)} \quad (2.5)$$

In order to calculate the loss (L) between the predicted and the actual class, the Mean Squared Error is used as shown in the equation 2.6.

$$L = (Predicted\ class - Acutal\ class)^2 \quad (2.6)$$

Performance can also be evaluated using other metrics. True Positive Rate (TPR) or sensitivity, which is equivalent to recall, is the ratio of correctly classified positive examples, whereas True Negative Rate (TNR) or Specificity informs you what fraction of all negative samples the classifier correctly predicts as negative. True Negative Rate is another name for it (TNR).

These measurescan be calculated as shown in equation (2.7) and (2.8).

$$\text{sensitivity} = \frac{TP}{(TP+FN)} \quad (2.7)$$

$$\text{Specificity} = \frac{TN}{(TN+FP)} \quad (2.8)$$

The most common criteria for measuring the performance of classification

systems are accuracy, recall or sensitivity, specificity, precision, and recall. Precision is yet another classification metric that is frequently used. It informs you what percentage of positive forecasts were truly positive [48].

$$\text{Precision} = \frac{TP}{(TP+FP)} \tag{2.9}$$

The F measure (also known as F1) is a new method for combining precision and recall into a single metric that may be calculated as shown in equation (2.10).

$$F1 = 2 \times \frac{\text{precision} \times \text{recall}}{precision+recall} \tag{2.10}$$

In an ideal scenario, we'd like a model with a precision of one and a recall of one. That translates to an F1-score of 1, which indicates 100 percent accuracy, which is rarely the case with machine learning models. So, we should aim for a higher precision while maintaining a higher recall value [49].

2.7.2 Arduino Nano 33 BLE Sense

Arduino is an open and free electronics platform with simple hardware and software. Small, powerful, Bluetooth-enabled, and equipped with all the sensors you'll need to create cutting-edge apps. Mid-July 2019 is the estimated ship date. The NINA B306 module for BLE communications is at the heart of this small and dependable Nano board. A Nordic NRF 52840 CPU with a powerful Cortex M4F and a set of links run the device, allowing for highly innovative and dynamic designs [50].

Wearable devices are now possible and projects that rely on gestures to interact with more close-range gadgets to be created. The BT 5.0 multi-protocol radio on the BLE Sense Arduino Nano 33 makes it perfect for projects using interactive automation 45 x 18 mm are the dimensions. Sensor interfaces should be ready to use, the computer data connection is similarly simple.

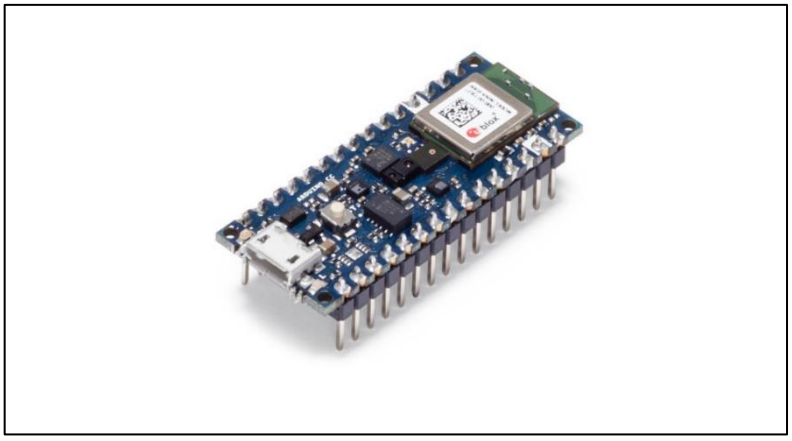

Figure 2.8: Arduino Nano 33 BLE Sense [50]

You can see the Arduino BLE 33 Sense in figure (2.8), it's a small and dependable Nano board based on the NINA B306 module for Bluetooth 5 and BLE communication; the module is based on the Nordic nRF 52840 processor, which has a strong Cortex M4F, and the board features a variety of sensors that enable the construction of unique and engaging designs [51]. The specifications of it are:

- 64MHz clock
- 1MB flash
- 256KB RAM

Chapter Three

Propose system Analysis and Implementation

Chapter Three: Proposed system Analysis and Implementation

3.1 Introduction

In this chapter, the steps of how to create a trained machine-learning model that can recognize words from our training data proposes in cloud server. The procedures for creating an MFCC block and a neural network are illustrated in Figure (3.1).

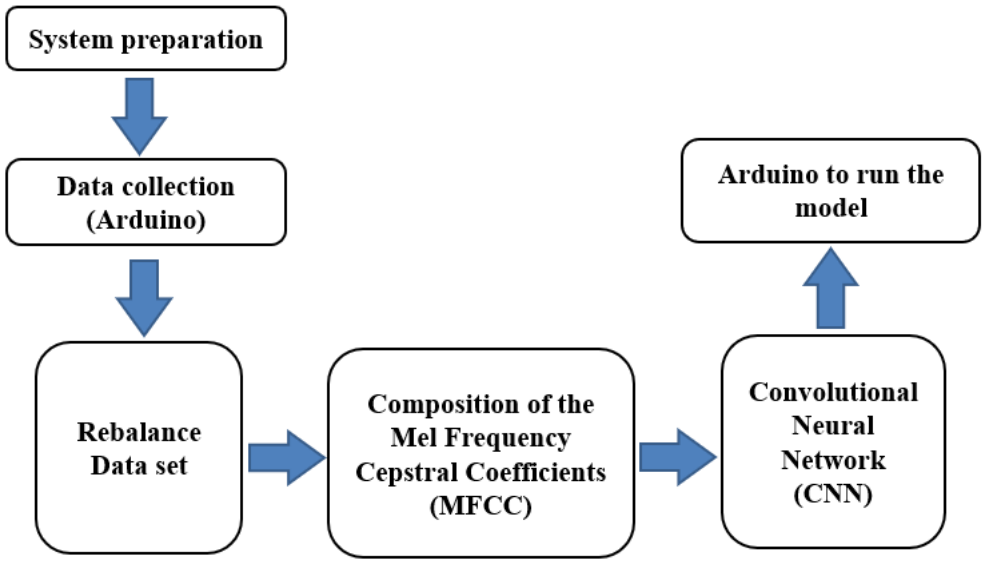

Figure 3.1: Steps to design trained data

3.1.1 System preparation

The sampling is started with raw data, then show how to build and deploy a trained machine learning model. The following applications needed by the Arduino must be installed before installing the samples, then the path must be added to the computer so that the Arduino can be found. The desks running the Arduino board firmware library, Node.js and NPN are downloaded in step two.

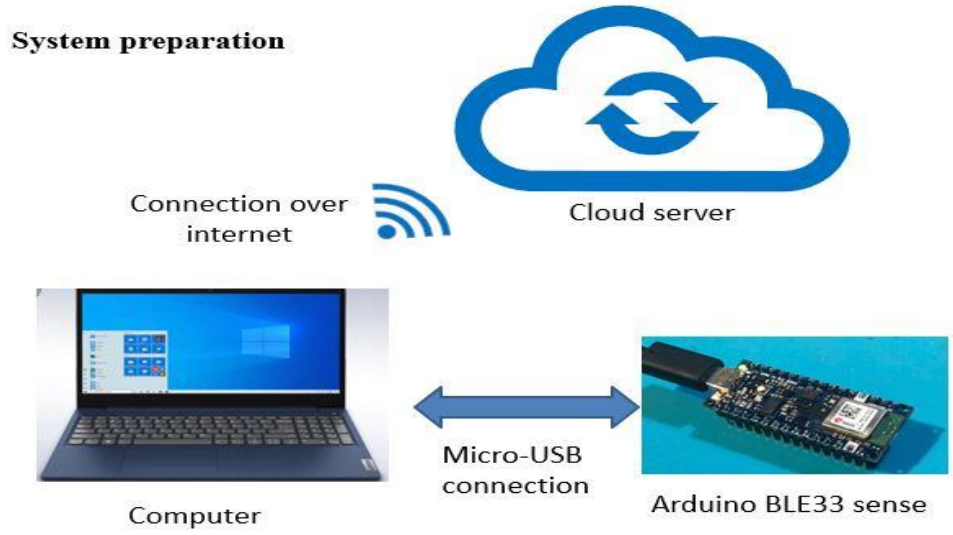

Figure 3.2: Pre-processing step results

3.1.2 Data collection

Typically create the own keyword recognition systems, such as right, left, stop and noise in the processing of this stage. To gather this data, we'll go to the data collection interface and change our terms, such as Right, and set the sample duration to one second. Here, the sensor was a microphone with a 16000 sample/sec. Then, presses the start sample button and said the word correctly 100 times. It's been found that when we utilized the cell phone is utilized as a sensor, the data gathering was quick. Then, it's been discovered that when we input the data is input, so a file is got with the word corrected in it. Now, it's going to collect the data by repeating the previous procedure on the left word and so on for other words. The data collection for all words are seen in figure (3.3, 3.4, 3.5, 3.6).

The data acquired by repeating the previous procedure on the stop word. We used the word stop and set the sample length to 1 second, and our sensor was a microphone with a frequency of 16000 sample/sec and said the word

stop 100 times, and when entered the data, so noticed that a file appeared that clearly contained the word stop. Now we'll collect the data by repeating the previous procedure on the left word, utilized the left word here and set the sample length to 1 second, utilized sensor to a microphone, and the frequency to 16000 sample/sec, it collects samples and say the word "left" 100 times. As for the word stop, it is indicated in the case of incomprehensible speech then it's observed that when enter the data, a file opens with the word left plainly written in it.

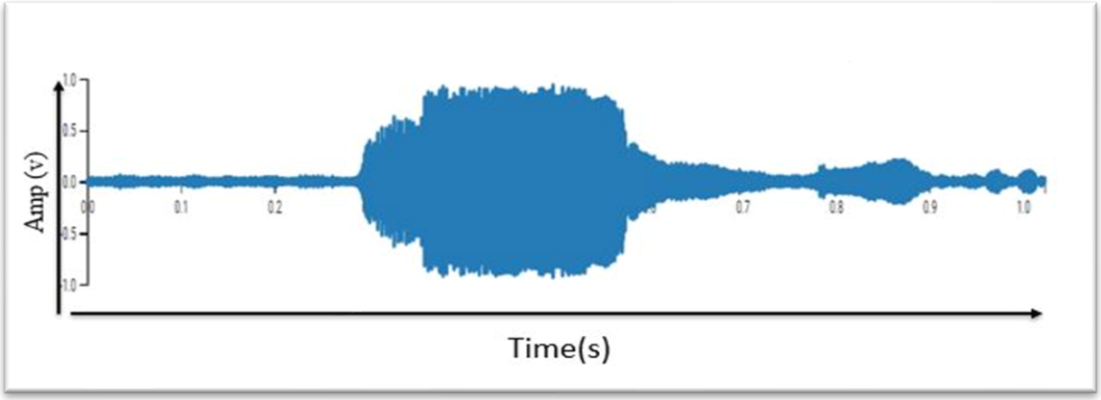

Figure 3.3: Data collected-right

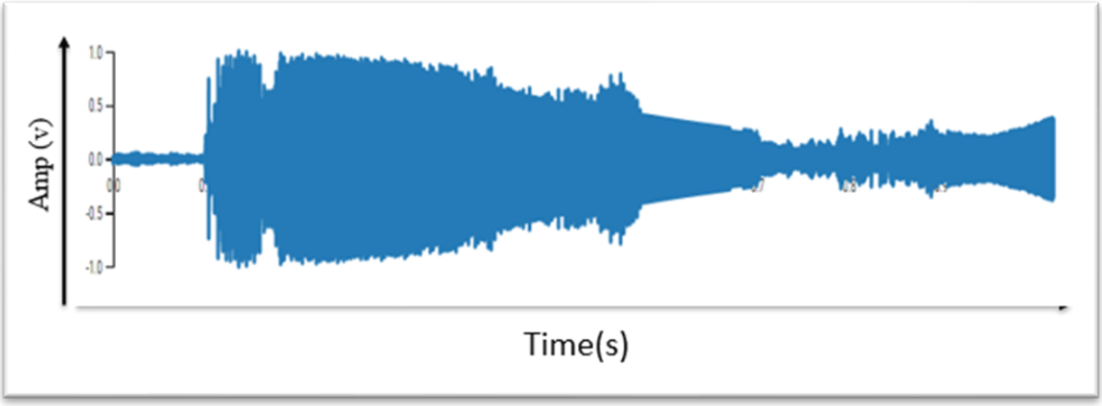

Figure 3.4: Data collected-left

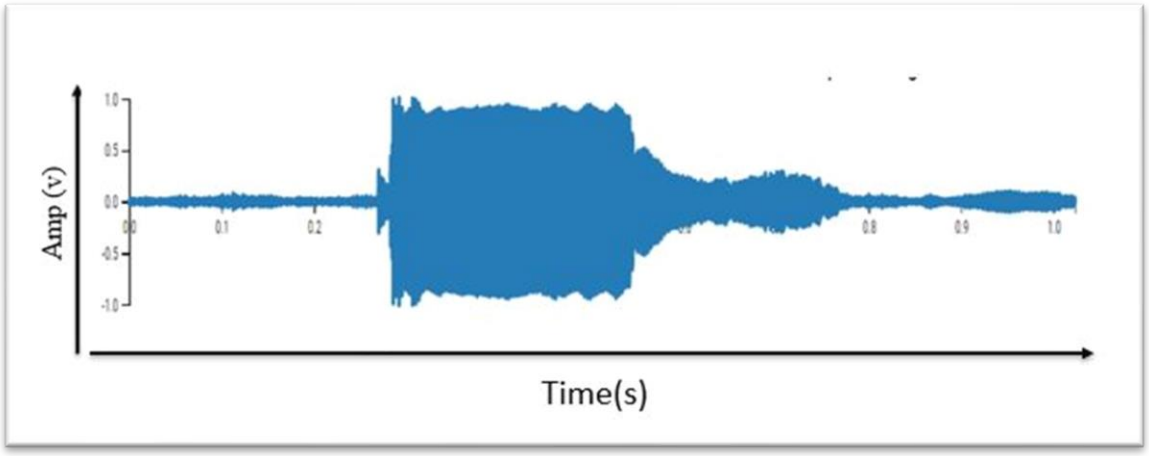

Figure 3.5: Data collected-stop

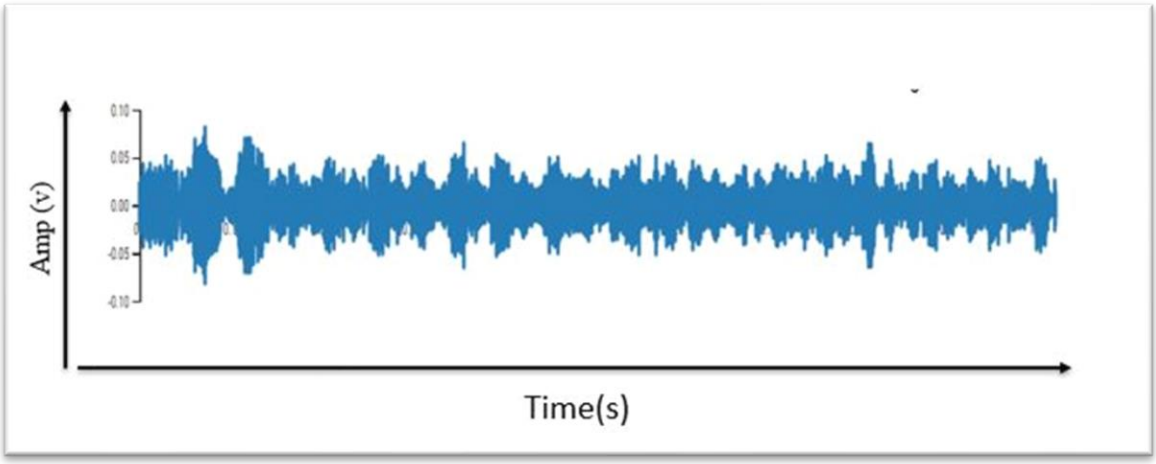

Figure 3.6: Data collected-noise

3.1.3 Rebalance the data set

A portion of the data is obtained after creating a machine learning model and trained the algorithm will be utilized to verify the quality of the training. This implies that the data is separated into two parts: data for testing and data for training Although the test and training data are not the same, they are both required for automatic learning. The test data is a subset of the data that used to evaluate the trained types. Performance and accuracy after it were built with the training data set. Because it's needed to save 78% of the training data for

the model and 22% to validate the trained model, where rebalancing our dataset used here and automatically split it between the training and test groups. The trained and tested dataset is depicted in table (3.1) and table (3.2) respectively.

Table 3.1: Training data result from model

Type	Data	Time
Data Collection Total 6m 50s	Stop	1m 42s
	Left	1m 42s
	Right	1m 42s
	Noise	1m 42s
Training 78 % & Test 22 %	Training	6m 50s
	Testing	1m 54s

Table 3.2: Test data result from model

Type	Data	Time
Data Collection Total 1m 54s	Stop	24s
	Left	31s
	Right	27s
	Noise	33s
Training 78 % & Test 22 %	Training	6m 50s
	Testing	1m 54s

3.1.4 Composition of the Mel Frequency Cepstral Coefficients (MFCC) block

Passed to the MFCC menu once impulse built, as show of figure (3.7)

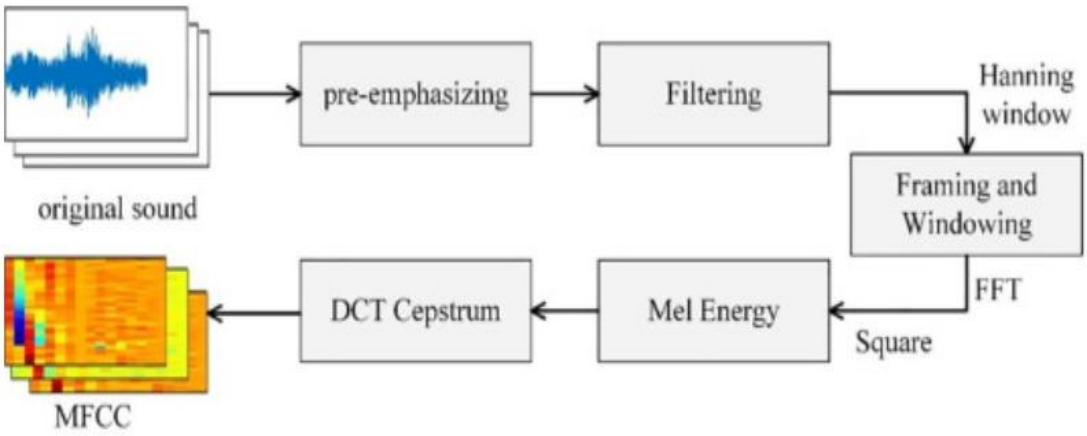

Figure 3.7: Block diagram of MFCC

And this page shows us how to transform the data. The passage of time as the frequencies increases on the vertical axis (where the number of frequencies can be controlled by parameters) and time on the horizontal axis (controlled by frame lines and frame length), because these visual patterns in the diagram shown contain information about how the types of sounds it makes. Whether it's right, left, stop, or noise, the spectrogram difference is cleaned from one image to the next depending on the sort of sound we have. Basically, data can be quickly acquired and searched on different types of spectrograms to select the audio sample for the purpose of display as shown in the following diagram: And the Table (3.3) shows the parameters of form MFCC setting.

Table 3.3: The set parameters of the MFCC

Parameter name	**Value**
Number of coefficients	13
Frame length	0.02sec
Frame stride	0.01sec
No. of filter	32 Filter
FFT length	256 Points
Normalization window size	101 msec
Low frequency Gain	300
Pre-emphasis	
Parameter name	Value
Coefficient	0.98
Shift	1 Roll over the input signal

As notice in the figures (3.8), (3.9), (3.10), (3.11), it can be noticed the differences between the time and intensity, as it is used to define our keywords Right, left, stop, noise. Also note the differences between them.

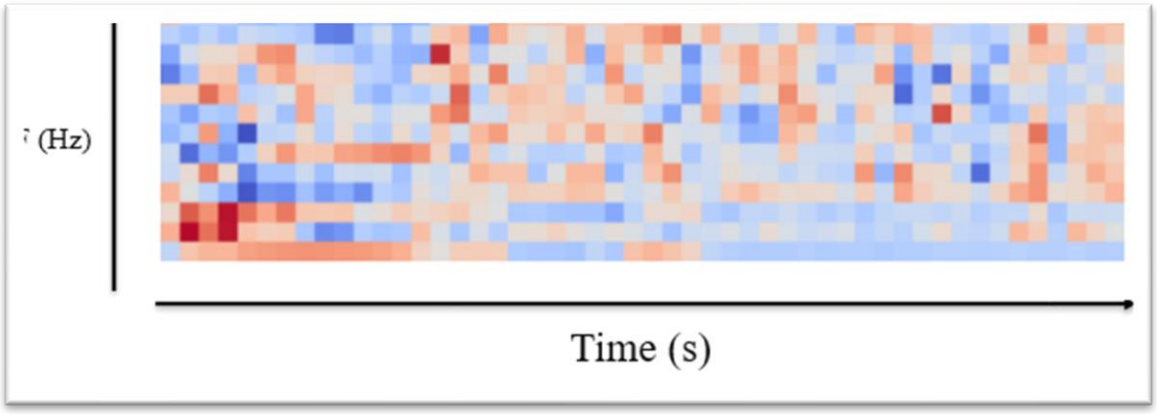

Figure 3.8: Spectrogram from data collection (right)

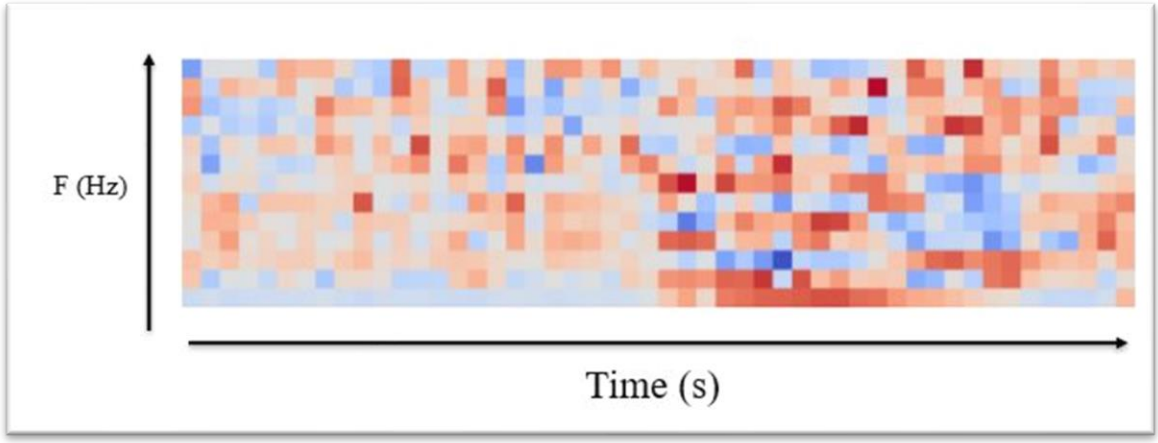

Figure 3.9: Spectrogram from data collection (left)

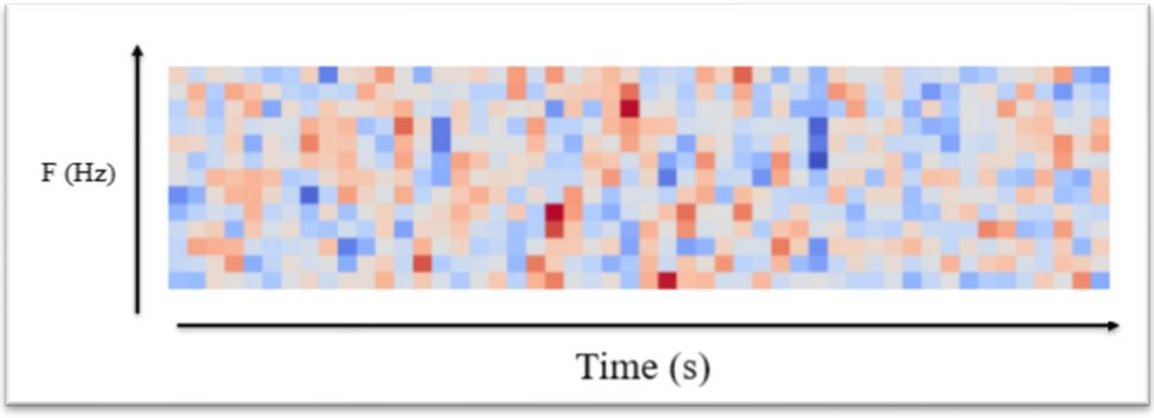

Figure 3.10: Spectrogram from data collection (stop)

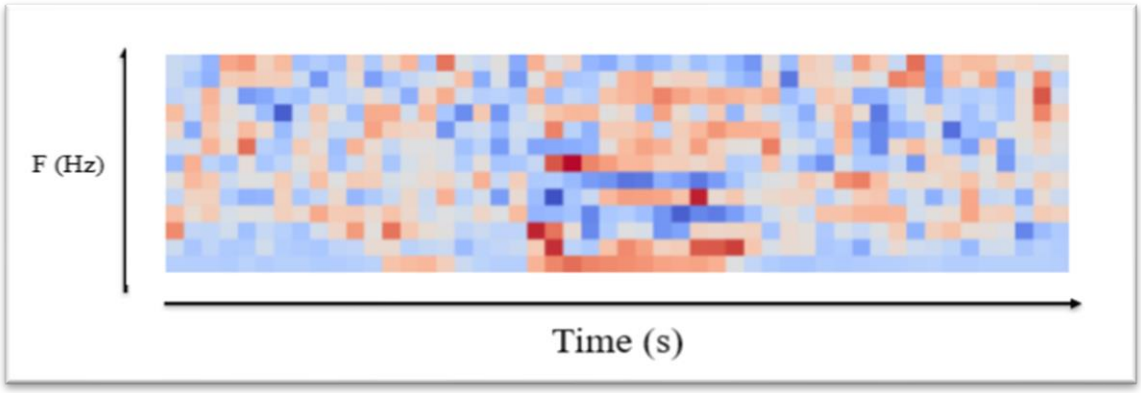

Figure 3.11: Spectrogram from data collection (noise)

As it can be seen the performance of the microcontroller as the total time consumed by the microcontroller is 328ms, and 26KB of RAM was utilized.

3.1.5 Feature Explorer

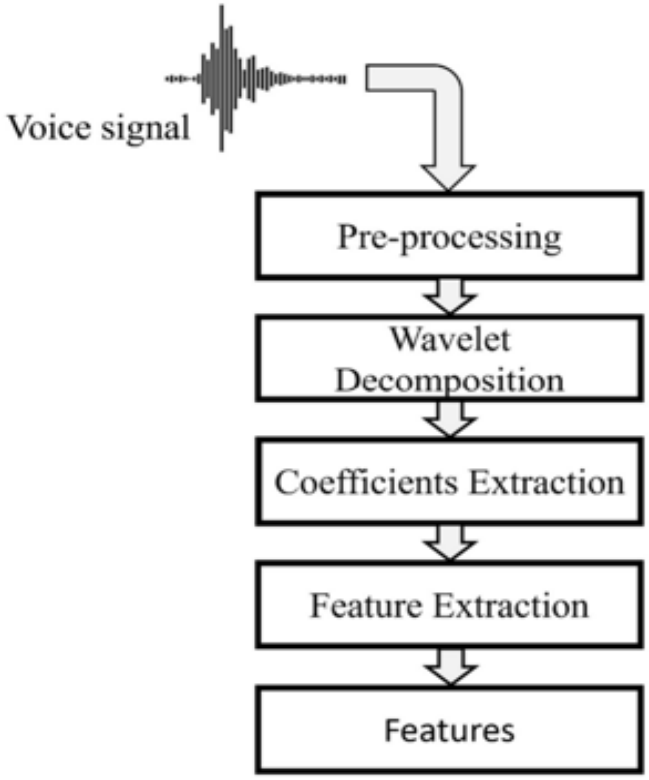

Figure 3.12: Block diagram of feature explorer

After the MFCC block creation and spectrum plots, the data will be passed to a neural network architecture to learn patterns in this sort of data.

However; before it the data must be train for all audio windows. Displays the entire data with each element color-coded to its term, allowing us to click on each element to see the sample and zoom in on the element to see whether there is an error element in the collection. This is an excellent technique to see if the data set has any false components as for machine learning, if it is acceptable. Figure (3.12) show the display of feature explorer.

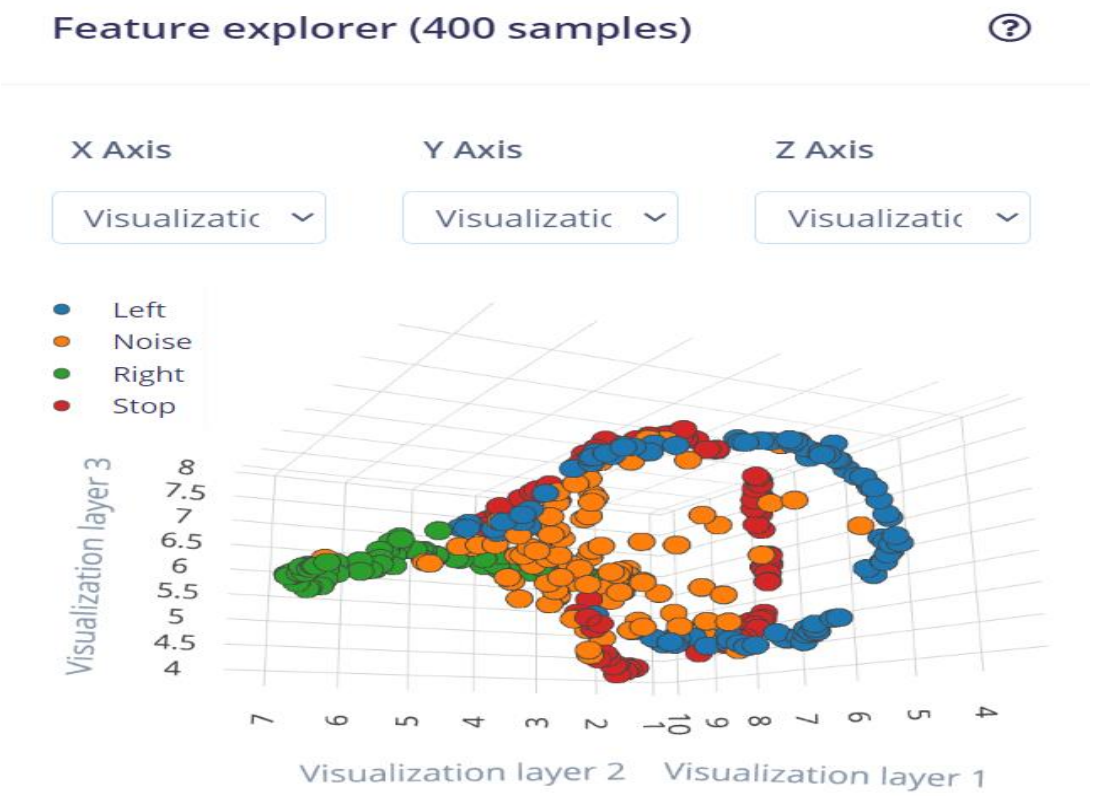

Figure 3.13: Feature explorer display input data

Data visualization for data and information is called graphic visualization which facilitates the detection and recognition of trends, data patterns and outliers. Charts, graphs, and maps are examples of visual components, that

known as data visualization. Data visualization methods and technologies are equally crucial, which analyzes a high category of data and takes action for the patterns as for well as the colors. So that the differences between red, blue, yellow and green can be quickly noticed.

3.1.6 Convolutional Neural Network (CNN)

After processing all the data, the next stage was to train a neural network, which are artificial intelligence systems based on the human brain and capable of identifying and detecting the sounds in the training data.

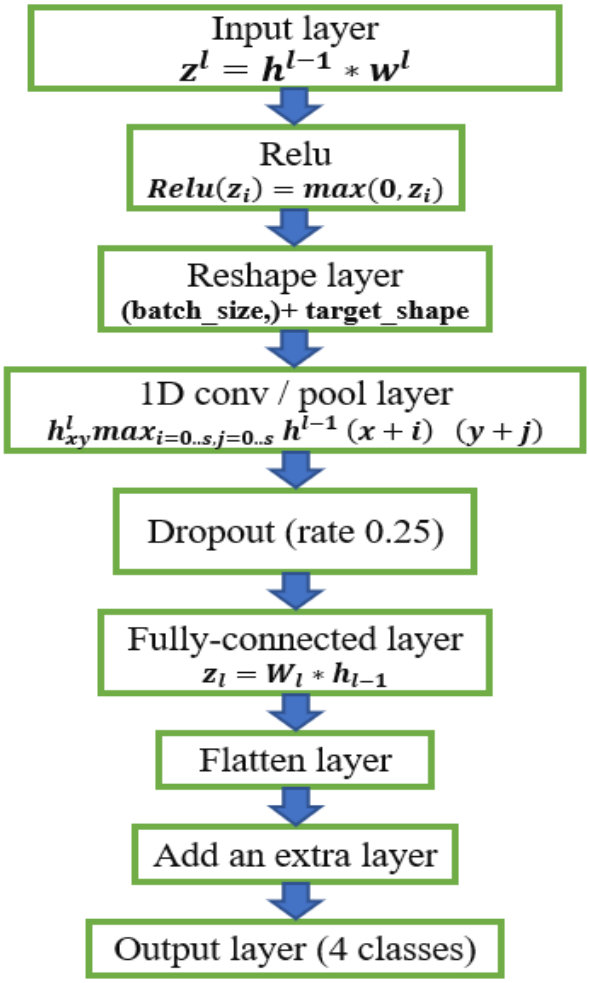

Figure 3.14: Convolutional Neural Network configuration

Virtual "neurons" from the neural network have several levels, as can be seen in the layer diagram, the first layer shows spectrum graphs, which are filtered and modified based on the internal position of each neuron, while the second layer repeats the process by adding and outputting the first layer. And, in order to improve it, the network's inputs must be adjusted appropriately in order to provide accurate outputs. A trained network is created when this method is several times layers provide structure and depending on how the structures differ, they may be utilized for a wide range of tasks. The training setting is show in figure (3.15).

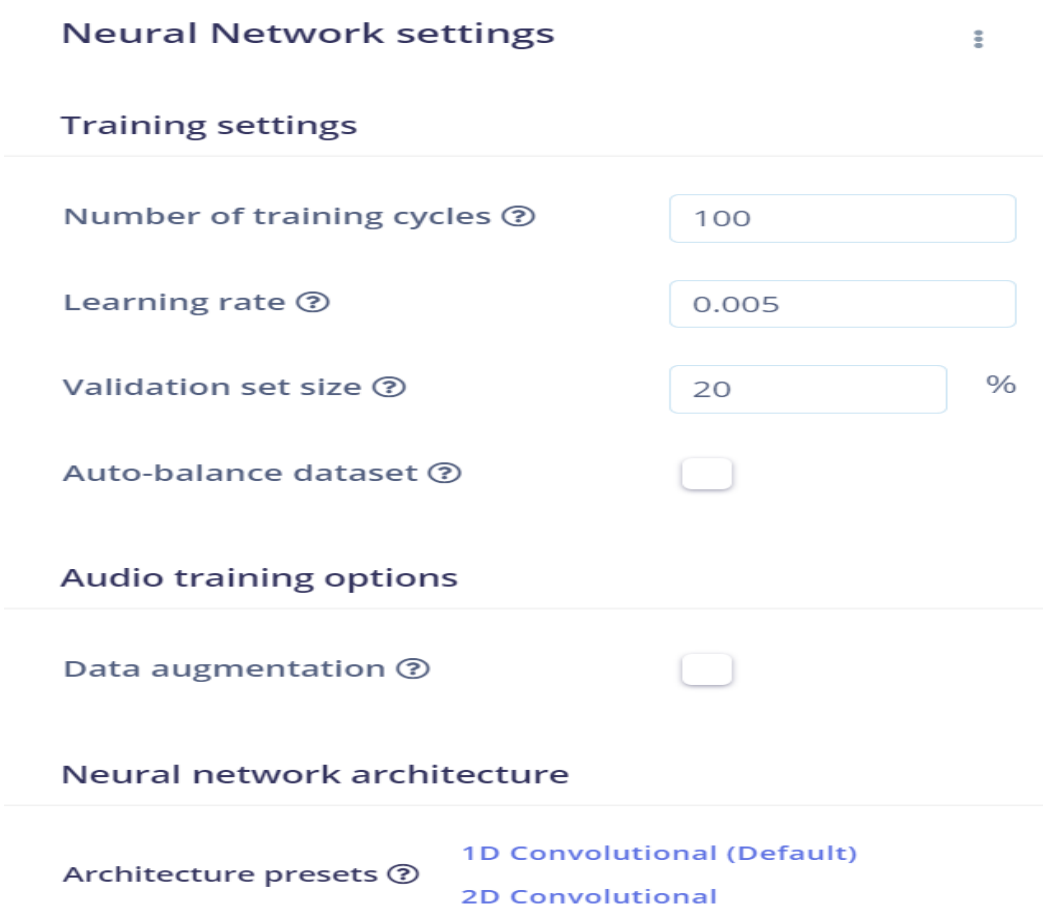

Figure 3.15: Neural network Setting

Then, after selecting the number of training cycles and training rate a sample of training and validation data is supplied. Layers give structure and may be utilized for a number of activities depending on how the structures differ after selecting the value of 100 for the number of training sessions. The larger the rise in the number of training sessions, the lower the accuracy, and vice versa. The training rate was 0.005, and the coefficient value, which is the sound reliability ratio, varied from 0.95 to 0.98, that categorized After a few minutes of training.

Chapter Four
Results and Discussion

Chapter Four: Results and discussion

4.1 Introduction

In this chapter the results are obtained by using two systems, the first is by using the deep convolutional neural networks learning (CNN) with Arduino Nano BLE Sense, and the second system is by using a mobile device with a computer. In both systems different audio data values were entered using Cepstral coefficients for frequency-slope, which extracts features from audio signals and is great for the human voice. Also using Classification where it learns patterns from data, and can apply them to new data. In both systems, the accuracy of the values of the input audio data was obtained, and the losses were also calculated.

4.2 Results of Arduino and Mobile devices

After the data set was tested by a CNN with an Arduino and a mobile device, the results are obtained as follows:

Part One: Implementation of CNN with Arduino

The accuracy and loss in the proposed system can be show in table (4.1).

Table 4.1: Accuracy and loss in percentage

Accuracy	98.8%
Loss	0.04

It should be noted that the accuracy indicates the percentage of the given sound that are correctly classified, and increasing the accuracy will make the system better. The performance of the proposed system is show in Table (4.2).

Table 4.2: The conclusion from NN Classifier at each class

Class	Sensitivity	Precision	Specificity	accuracy	F-Score
Right	1	1	1	1	1
Left	1	1	1	1	1
Stop	0.95	1	1	0.98	0.98
Noise	1	0.95	0.98	0.98	0.97

Table (4.3) show the confusion matrix of the system.

Table 4.3: The performance from the given data

	Left	Noise	Right	Stop
Left	100%	0%	0%	0%
Noise	0%	95%	0%	5%
Right	0%	0%	100%	0%
Stop	0%	0%	0%	100%
F1_ SCORE	1.00	0.97	1.00	0.98

The confusion matrix explains the balance of the classified sound source in spite of the unclassified sound source in order to be understood. Which can classify the sound source of a word named (right) as it is word right and a word named (left) as a word left, etc. Which allows us to note that there is a loss in the result obtained but a small percentage that guarantees the accuracy given. Figure (4.1) state the three-dimensional feature explorer, and it can be seen that

the values of the input data got a small error rate as shown in the red circle, so there were data losses.

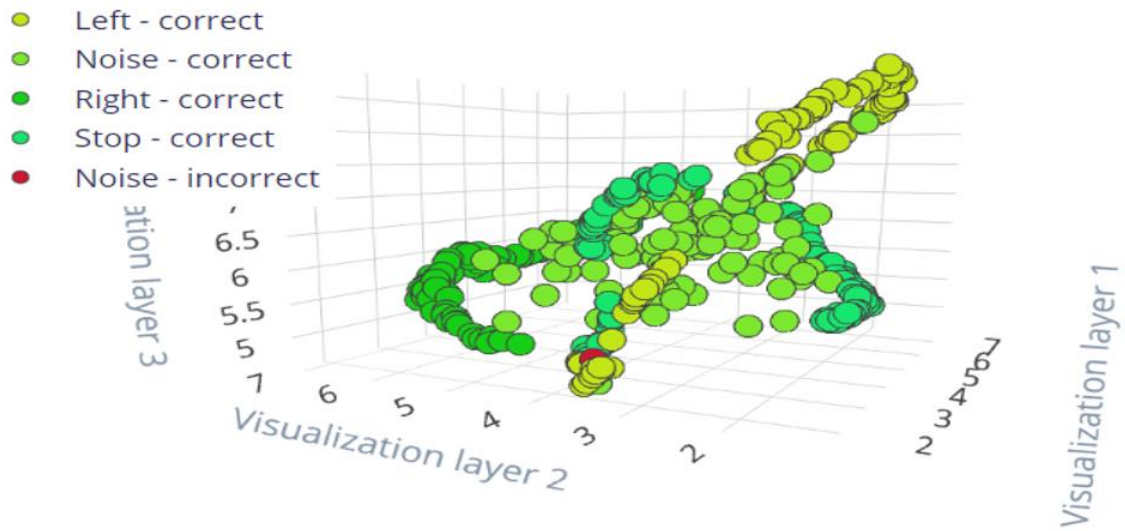

Figure 4.1: The Feature Explorer

Platform performance is stated in Table (4.4), which displays decile for how much model is executed during the system, such as time out, details and the least memory usage, which shows us how much RAM is required to be performed.

Table 4.4: On-device performance

Parameter name	**Value**
Inferencing time	4ms
Peak RAM usage	5.0k
Flash usage	33.5k

Platform performance displays decile for how much model is executed during the system, such as time out, which is a rating from how extended it might take the model into test one second from data in representative console, and the least memory usage, which shows us how much more RAM is required to be performed.

Part Two: Implementation the mobile device with the computer

The accuracy and loss can be show in Table (4.5).

Table 4.5: Accuracy and loss in percentage

Accuracy	100%
Loss	0.00

In this part, the model is trained using the mobile device with the computer. The value here will differ from those in the training program using the Arduino board with the computer. Here the accuracy is 100%, although it is noted that its accuracy is often evidence that our model exceeded data training. It will be revealed if it is correct in the next stage of testing the model. Which allows to note that there is no loss in the result obtained, so that it guarantees the accuracy given. The performance of the proposed system is given in table (4.6) and the confusion matrix is show in Table (4.7).

Table 4.6: The conclusion from NN Classifier at each class

Class	Sensitivity	Precision	Specificity	accuracy	F-Score
Right	1	1	1	1	1
Left	1	1	1	1	1
Stop	1	1	1	1	1
Noise	1	1	1	1	1

Table 4.7: Last training performance of the given data

Class	Left	Noise	Right	Stop
Left	100%	0%	0%	0%
Noise	0%	100%	0%	0%
Right	0%	0%	100%	0%
Stop	0%	0%	0%	100%
F1 SCORE	1.00	1.00	1.00	1.00

For this confusion matrix, the comparison is done with number of correct categorized windows to the number of errors a categorized window, in which the sound overture of the word Right are classed as Right, the word left as left, and the word stop as stop. Also, for the classification of noise windows, no losses are observed in the construction of the model as depicts in Figure (4.2).

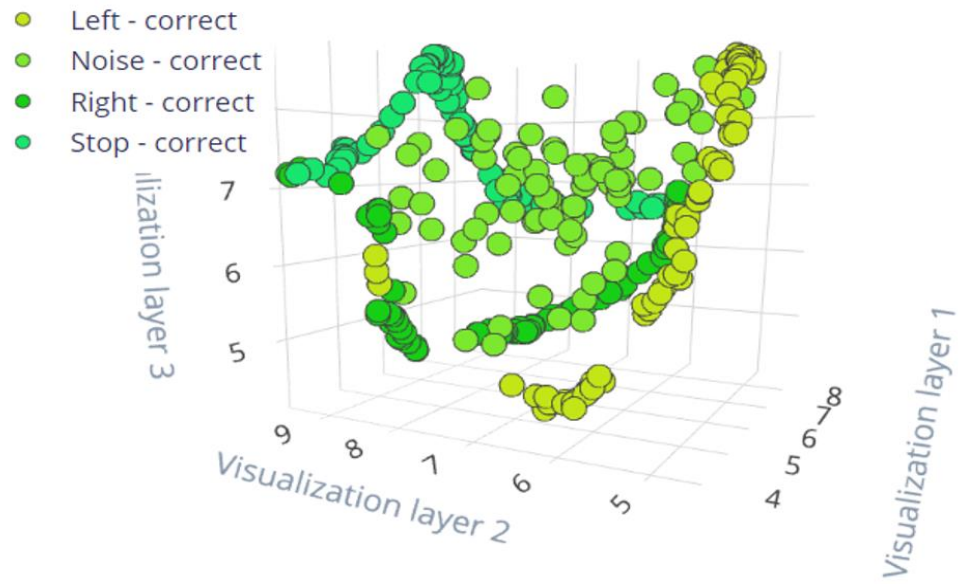

Figure 4.2: Feature explore

From figure (4.2), it is noted that the values of the input data did not have any error rate. Where no losses are obtained as mentioned in the first section. This means that the accuracy and implementation of deep learning using this technique was good.

The statistics that encode how the device model is expected to work are shown in Table (4.8) which taken form the standard microcontroller, while the time inference represents an assessment from how much long it will occupy for the model into study one more from data. As for peak memory, it refers to the amount of RAM for how the device works.

Table 4.8: On-device performance

Parameter name	Value
Inferencing time	4ms
Peak RAM usage	4.7kB
Flash usage	33.5kB

4.3 Classifying of data

From the previous stage, it was found that the results of the model period performed well for the training data. It is important to test it on new data that has not been previously selected. This step shows useful features like the ability into log alive data of system while classify this instantly. Figure (4.3) shows a new data for classification and figure (4.4) states the Results of it.

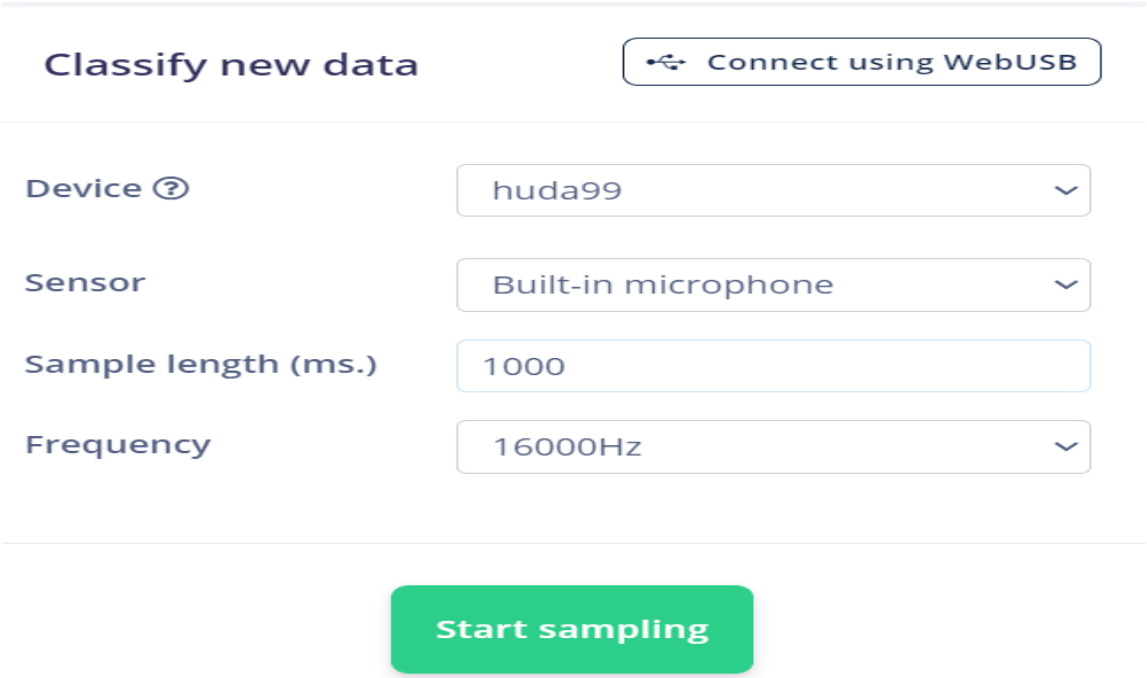

Figure 4.3: Classified new data

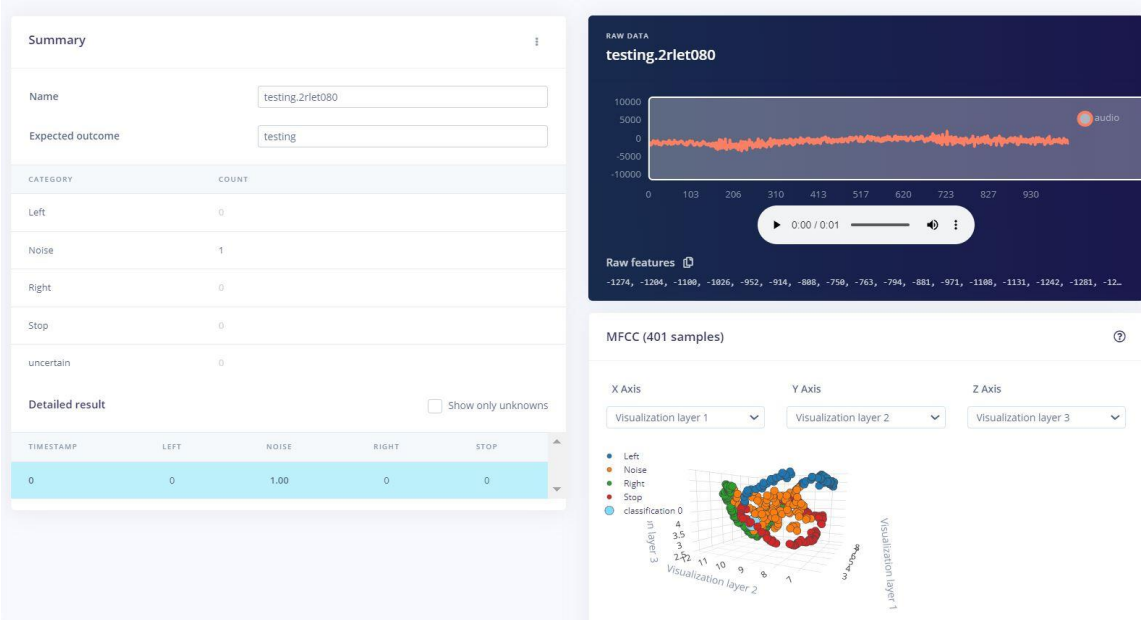

Figure 4.4: Result of classifying data

After loading the sample, it is segmented into several windows. Then the windows are classified. As indicated, our model's 400 captured acoustic windows were rated as being Noise, Right, Left, and stop. The accuracy of the proposed system is 98.8% based on the investigation data. It was expected that at least 2% of the windows would incorrectly classify - and more likely if the model didn't work perfectly.

4.4 Model testing

The form can be quickly tested using the live evaluation page to see how it works. However, more comprehensive testing is needed to ensure that it is working properly. The form Test tab is launched, the sample just taken will be listed in the test dashboard stated in Figure (4.5).

Figure 4.5: The Test data panel

Each word contains a test data set as well as its own training data. The test data set is immediately saved with the samples taken in the live classification, and the form test page displays all the test data. The expected result of the sample obtained must be accurately specified before it is used for testing. The accuracy of the model was estimated based on the test results.

Ideally, it may be desirable to collect a test set that containing at least 25% of the data from the model of training set. If the training data is collected about 10 minutes, then it should be less than the test data about 2.5 minutes. It must be ensured that these test data cover a wide range of conceivable scenarios when the model can be evaluated on the basis of different inputs. Figure (4.6) shows the test result for a record from testing data of words right, left, stop and noise.

	LEFT	NOISE	RIGHT	STOP	UNCERTA
LEFT	100%	0%	0%	0%	0%
NOISE	0%	93.8%	0%	0%	6.3%
RIGHT	0%	0%	100%	0%	0%
STOP	0%	4.3%	0%	91.3%	4.3%
F1 SCORE	1.00	0.95	1.00	0.95	

Figure 4.6: Test result at a lot from samples

Classification reveals that the work of proposed model was with an accuracy of 98.88%, 2% lower than the performance of the validation data. Since it is common for a model to perform quite poorly for new data, it is a good result, and as a working strategy.

Samples that frequently contain the wrong number of classifications are useful because they contain examples of the sound class that the model does not fit. It is a good idea to add it to the training data after completing the steps, and the test data should be replaced with new data to make up for the loss.

The distinction between visually labeled data from training data should be observed by looking at the feature explorer on the X, Y, and Z axes. So it can take the a new (second) and trained it. A sample could be identified as anomalies and the lack of confidence in the neural network is because the distance from the block too large.

Testing the model in the real world helps ensure that it works, which is something that should be done after every change. By making frequent changes to the model in order to further improve test data set, however, the model may

begin to adapt the test data set and lose its usefulness in the procedure. To prevent this, add a number of new test data set.

4.5 Model program on the device

The data from the sensors and the execution on of the signal processing code and the categorization of the data is as follows:

When the data for right is tested, it will appear on the Arduino interface after a network has been downloaded in which the percentage of data for right on the screen is higher than the percentage of data for left, stop and noise.

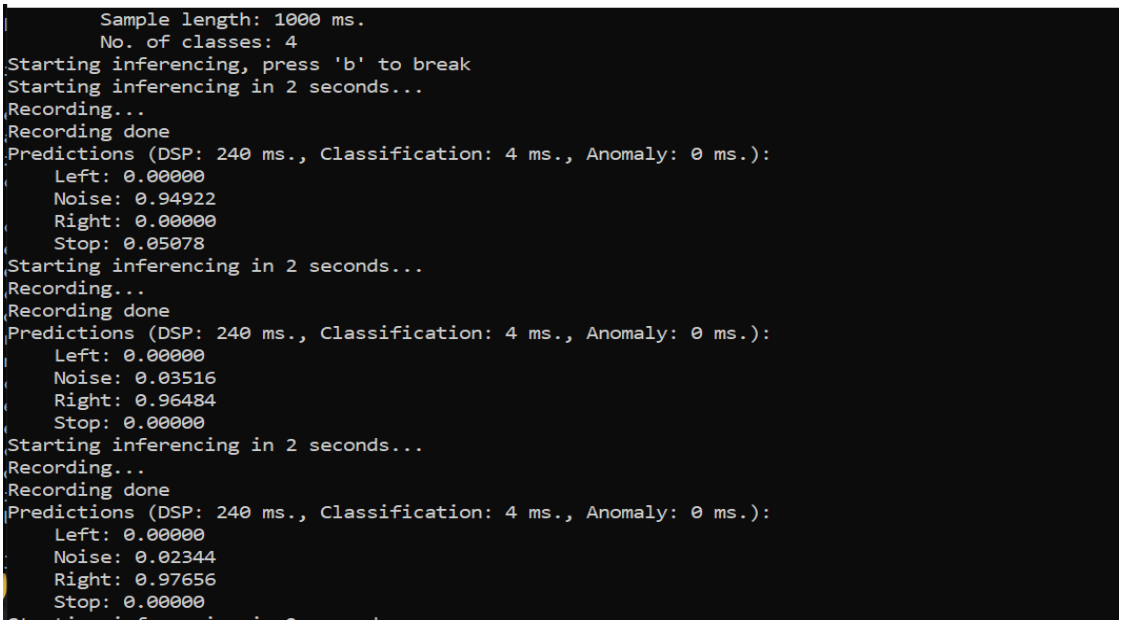

Figure 4.7: Test word data right

Also, for the data of the word left, when performing the test, the percentage of data for the word left will appear on the screen higher than the percentage of data for the words right, stop and noise.

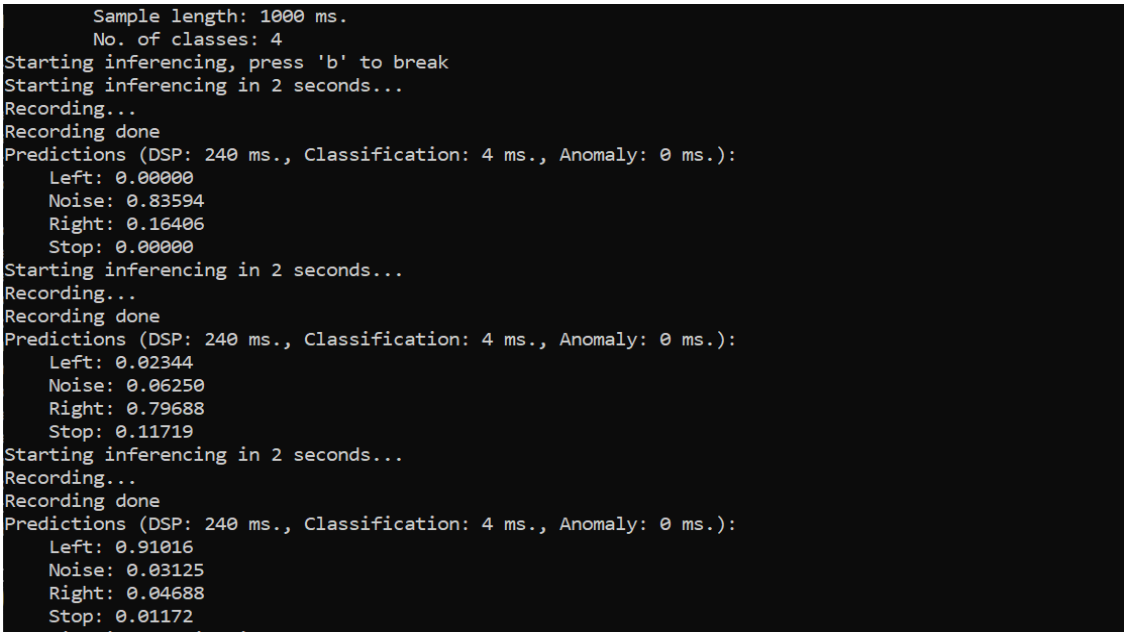

Figure 4.8: Test word data left

So, for the data of the word stop when testing, the percentage of data for the word stop will appear on the screen higher than the percentage of data for the words right, left and stop.

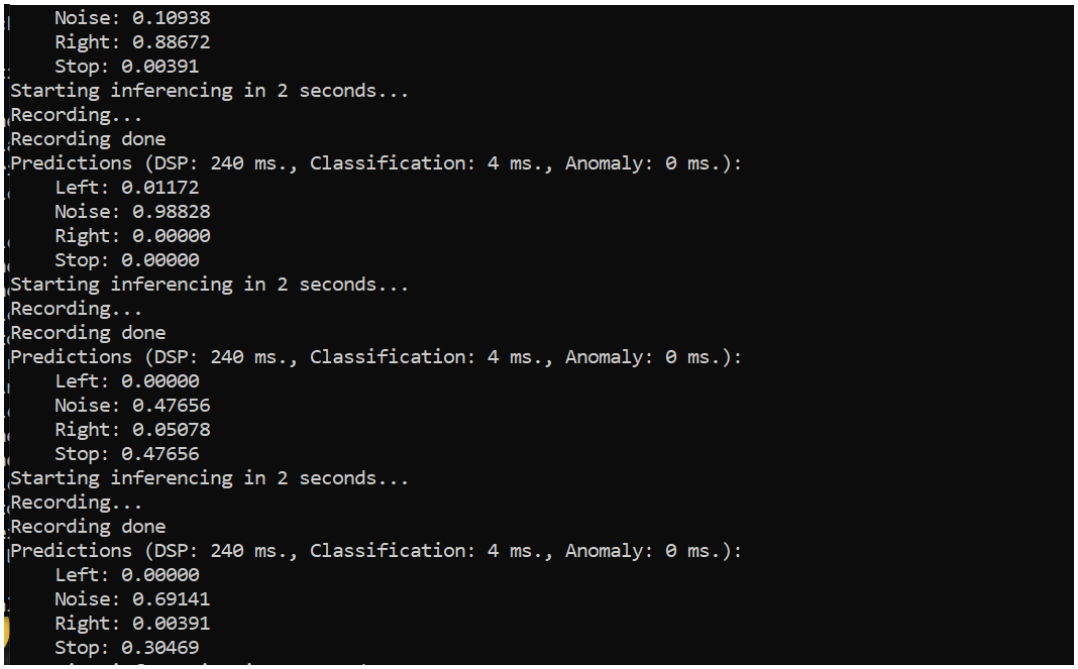

Figure 4.9: Test word data stop

When you test the data for the word noise, the percentage of data for the word noise will appear on the screen higher than the percentage of data for the word right, left and stop.

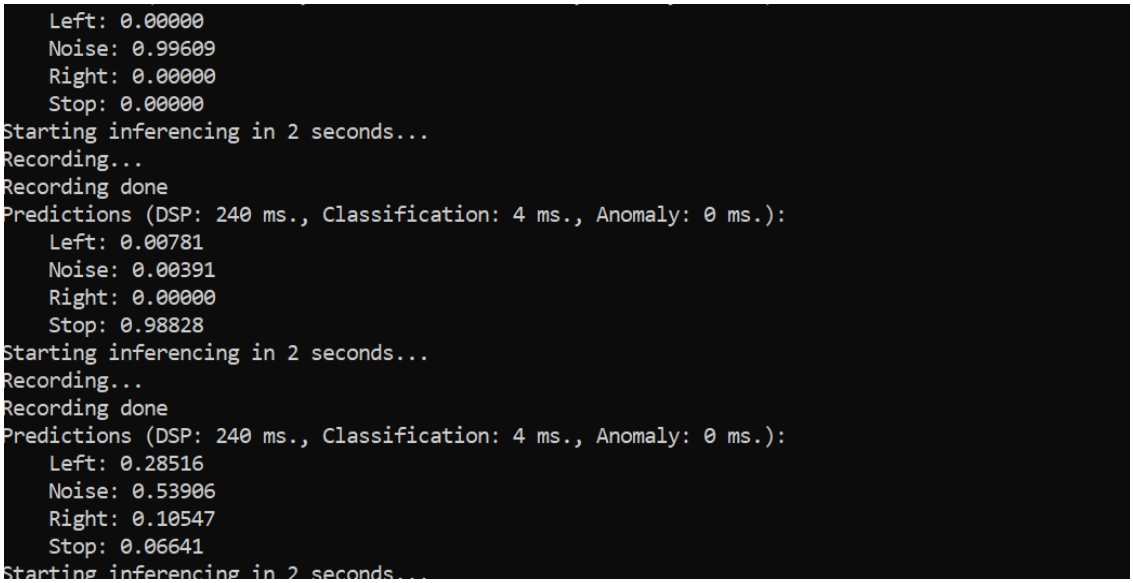

Figure 4.10: Test word data noise

Thus, a deep learning network (CNN) was made and trained on 400 sounds, after which it was converted in a way that was tested on the Arduino device.

Chapter Five
Conclusion and Future Work

Chapter Five: Conclusion and Future Work

5.1 Conclusion

In this work, deep learning algorithms have been applied in speech analysis and from the Results, the following conclusion can be depicted:

- CNN are implemented on an Arduino of limited resources with very good results.
- Mel Frequency Cepstral Coefficients (MFCC) for slope frequency are used to extract selected audio features to get the highest possible accuracy and reduce data sample and speed learning.
- Because Arduino is limited in resources, Tiny Machine Learning (ML) was used to transform the network, it is less efficient but operational and bears the resources of deep learning. And it was implemented on Arduino with minimal resources and simplicity of use.
- Convolution neural network (CNN) has been successfully built in Arduino for training and classification.
- Good results were obtained in both directions. It is expected that the results of the mobile device will be higher due to the absence of losses and the speed of entering voice data through it and the record is very clearly.
- The accuracy of the Arduino result was (98.8%) while the result of the mobile was (100%).

5.2 Suggestions for future work:

1. In our future work, a servo motor will be used to represent the given data. The implementation will be executed on the motor itself and the Arduino instead of being show on the execution screen, when any of the audio data is heard in the microphone of the Arduino.
2. Or implement the project using the method FPGA.

References

[1] M. Z. IQBAL, "MFCC and Machine Learning Based Speech Emotion Recognition on," *Found. Univ. J. Eng. Appl. Sci.*, vol. 1, no. 1, pp. 1–6, 2020.

[2] M. M. Oo, "Comparative Study of MFCC Feature with Different Machine Learning Techniques in Acoustic Scene Classification," *Int. J. Res. Eng.*, vol. 5, no. 7, pp. 439–444, 2018, doi: 10.21276/ijre.2018.5.7.1.

[3] F. Carlos, A. C., C. Y., P. Pupin, and A. Alaniz, "An Analysis of Visual Speech Features for Recognition of Non-articulatory Sounds using Machine Learning," *Int. J. Comput. Appl.*, vol. 177, no. 16, pp. 1–9, 2019, doi: 10.5120/ijca2019919393.

[4] K. Okabe, "Arduino programing of ML-style in ATS Arduino programing of ML-style in ATS," no. September, pp. 3–5, 2015.

[5] S. Branco, A. G. Ferreira, and J. Cabral, "Machine learning in resource-scarce embedded systems, FPGAs, and end-devices: A survey," *Electron.*, vol. 8, no. 11, 2019, doi: 10.3390/electronics8111289.

[6] A. Priyadarshi, V. Bharath, A. Ks, and S. Sanjit, "Arduino Based Motion Tracking Keyboard Using Machine Learning," vol. 4, no. October, pp. 9–14, 2020.

[7] V. Lyashenko, F. Laariedh, S. Sotnik, and M. Ayaz Ahmad, "Recognition of Voice Commands Based on Neural Network," *TEM J.*, vol. 10, no. 2, pp. 583–591, 2021, doi: 10.18421/TEM102-13.

[8] M. Bakouri *et al.*, "Steering a Robotic Wheelchair Based on Voice Recognition System Using Convolutional Neural Networks," pp. 1–17, 2022.

[9] S. F. Ahmad, M. K. Rahmat, M. S. Mubarik, M. M. Alam, and S. I. Hyder, "Artificial intelligence and its role in education," *Sustain.*, vol. 13, no. 22, 2021, doi: 10.3390/su132212902.

[10] M. Matheny, S. T. Israni, and M. Ahmed, "Artifical Intellingence in Health Care," *Natl. Acad. Med.*, pp. 1–269, 2018.

[11] O. F.Y, A. J.E.T, A. O, H. J. O, O. O, and A. J, "Supervised Machine Learning Algorithms: Classification and Comparison," *Int. J. Comput. Trends Technol.*, vol. 48, no. 3, pp. 128–138, 2017, doi:

10.14445/22312803/ijctt-v48p126.
[12] M. Kupi, "The possibilities of AI applicability in tourism," no. November, 2021.
[13] M. Westerlund, "An ethical framework for smart robots," *Technol. Innov. Manag. Rev.*, vol. 10, no. 1, pp. 35–44, 2020, doi: 10.22215/timreview/1312.
[14] W. Apt, "INTRODUCTION TO MACHINE LEARNING," *Demogr. Res. Monogr.*, pp. 1–13, 2014, doi: 10.1007/978-94-007-6964-9_1.
[15] F. Yucalar, "Python Programming with Example Applications & Machine Learning," no. November, 2021.
[16] A. Ali, F. Yangyu, and S. Liu, "Automatic modulation classification of digital modulation signals with stacked autoencoders," *Digit. Signal Process. A Rev. J.*, vol. 71, pp. 108–116, 2017, doi: 10.1016/j.dsp.2017.09.005.
[17] M. Adamiak, "Quantum-assisted supervised machine learning Adam Mickiewicz University in Pozna « Faculty of Mathematics and Computer Science Quantum-assisted supervised machine learning," no. October, 2019, doi: 10.13140/RG.2.2.26778.59840.
[18] S. Dolgikh, "Identifying explosive epidemiological cases with unsupervised machine learning," *CEUR Workshop Proc.*, vol. 2753, pp. 1–10, 2020.
[19] T. Song *et al.*, "Unsupervised Machine Learning for Improved Delaunay Triangulation," 2021.
[20] E. F. Morales and J. H. Zaragoza, "An introduction to reinforcement learning," *Decis. Theory Model. Appl. Artif. Intell. Concepts Solut.*, pp. 63–80, 2011, doi: 10.4018/978-1-60960-165-2.ch004.
[21] K. N. Hampton and W. Chen, "On Social Media: Studying Social Media from an Egocentric Perspective," *Pers. Networks*, no. October, pp. 718–733, 2021, doi: 10.1017/9781108878296.056.
[22] K. Monika, "Product Recommendation using Machine Learning Model," no. June, pp. 0–4, 2017.
[23] A. Niimi, "Natural Language Processing: A Promising Research Tool of Chronic Cough for the Big Data Era," *Chest*, vol. 159, no. 6, pp. 2149–2150, 2021, doi: 10.1016/j.chest.2021.01.045.
[24] S. and Y. H. M. Abu Ghurah1, 2, M. K. A. Kamarudin1,*, N. A. Wahab1, R. Umar1, N. A. F. Nik Wan1, H. Juahir1, M. B. Gasim1, A.

R. Hassan1, F. Lananan1, A. F. Ireana Yusra1, “DETECTION OF ASPHYXIA IN INFANTS USING DEEP LEARNING CONVOLUTIONAL NEURAL NETWORK (CNN) TRAINED ON MEL FREQUENCY CEPSTRUM COEFFICIENT (MFCC) FEATURES EXTRACTED FROM CRY,” *J. Fundam. Appl. Sci.*, vol. 4, no. 1, pp. 9–10, 2018, [Online]. Available: http://dx.doi.org/10.4314/jfas.v10i1s.7.

[25] K. Chen, H. Zhu, L. Yan, and J. Wang, “A Survey on Adversarial Examples in Deep Learning,” *J. Big Data*, vol. 2, no. 2, pp. 71–84, 2020, doi: 10.32604/jbd.2020.012294.

[26] E. U. Moya-sanchez, “Introduction and Limitations of Deep Learning II Introduction and Limitations of Deep Learning II 2 v1 . 1,” no. March, 2018, doi: 10.13140/RG.2.2.34561.56169.

[27] U. Shafique, “A Comprehensive Study on Natural Language Processing and Natural Language Interface to Databases,” no. SEPTEMBER 2014, 2015.

[28] H. Hofbauer, F. Autrusseau, and A. Uhl, “Low Quality and Recognition of Image Content,” *IEEE Trans. Multimed.*, vol. PP, pp. 1–1, 2021, doi: 10.1109/tmm.2021.3103394.

[29] A. M. Hamandi, H. Bahjat, and A. K. Abdul Hassan, “Natural Language Processing Using Natural Language Toolkit, p. 70, 2016, doi: 10.34279/0923-007-002-010.

[30] M. Visser and M. Fokkema, “Customer relationship management,” *Digit. Mark. Fundam. From Strateg. to ROI*, no. December, pp. 427–470, 2021.

[31] A. A. Kankam Boadu, “Customer Relationship Management and Customer Retention,” *SSRN Electron. J.*, no. October, pp. 0–76, 2019, doi: 10.2139/ssrn.3472492.

[32] M. Sharma, R. Mittal, A. Bharati, D. Saxena, and A. Kumar, “A Survey and Classification on Recommendation Systems,” no. January, 2022.

[33] H. Bhowmick, “Comprehensive Movie Recommendation System,” no. December, 2021.

[34] H. Ding, Y. Ma, A. Deoras, Y. Wang, and H. Wang, *Zero-Shot Recommender Systems*, vol. 1, no. 1. Association for Computing Machinery, 2021.

[35] W. Is, Y. Diagnosis, P. I. Practice, and B. Reviews, “Recomandation

System," no. January, pp. 620–621, 2009, doi: 10.13140/RG.2.2.29607.88484.
[36] J. XIONG, *Essential Bioinformatics*. 2006.
[37] B. Lahasan and H. Samma, "Convolutional Neural Network for Skull Recognition," no. December, 2021.
[38] I. Aziz, "Deep Learning : An Overview of Convolutional Neural Network (CNN) Irfan Aziz : Deep Learning : An Overview of Convolutional Neural Network," no. April, 2020.
[39] J. Martinez, H. Perez, E. Escamilla, and M. M. Suzuki, "Speaker recognition using Mel Frequency Cepstral Coefficients (MFCC) and Vector quantization (VQ) techniques," *CONIELECOMP 2012 - 22nd Int. Conf. Electron. Commun. Comput.*, no. February, pp. 248–251, 2012, doi: 10.1109/CONIELECOMP.2012.6189918.
[40] O. K. Hamid, "Frame Blocking and Windowing Speech Signal," *J. Information, Commun. Intell. Syst.*, vol. 4, no. 5, pp. 87–94, 2018.
[41] S. Patients *et al.*, "We are IntechOpen , the world ' s leading publisher of Open Access books Built by scientists , for scientists TOP 1 %," *Intech*, p. 13, 2012, [Online]. Available: http://dx.doi.org/10.1039/C7RA00172J%0Ahttps://www.intechopen.com/books/advanced-biometric-technologies/liveness-detection-in-biometrics%0Ahttp://dx.doi.org/10.1016/j.colsurfa.2011.12.014.
[42] P. He, Y. Li, S. Chen, H. Xu, L. Zhu, and L. Wang, "Core looseness fault identification model based on Mel spectrogram-CNN," *J. Phys. Conf. Ser.*, vol. 2137, no. 1, p. 012060, 2021, doi: 10.1088/1742-6596/2137/1/012060.
[43] T. Kamm, H. Hermansky, and A. G. Andreou, "Learning the Mel-scale and Optimal VTN Mapping," no. October 2011, pp. 1–8, 2016.
[44] S. Umesh, L. Cohen, and D. Nelson, "Fitting the Mel scale," *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, vol. 1, no. September 2014, pp. 217–220, 1999, doi: 10.1109/icassp.1999.758101.
[45] S. A. Khayam, "The Discrete Cosine Transform (DCT): Theory and Application1," no. January 2003, 2021.
[46] S. Besbes and Z. Lachiri, "Multitaper MFCC Features for Acoustic Stress Recognition from Speech," *Int. J. Adv. Comput. Sci. Appl.*, vol. 8, no. 3, 2017, doi: 10.14569/ijacsa.2017.080361.
[47] S. H. Lim and M. M. Nojiri, "Morphology for jet classification," *Phys.*

Rev. D, vol. 105, no. 1, p. 14004, 2022, doi: 10.1103/physrevd.105.014004.

[48] A. Tubaishat, "Connect Alexa with Arduino Using Artificial Intelligence to Reduce Infectious Diseases Transmission in Lifts Ahmad Al-Nassr Eng . Ahmad Tubaishat E-Learning Coordinator Echo : Lift for Covid-19 Hands-Free Operation," no. September, 2021.

[49] A. C. Gheorghe and C. I. Stoica, "Wireless Weather Station Using Arduino Mega and Arduino Nano," *Sci. Bull. Electr. Eng. Fac.*, vol. 21, no. 1, pp. 35–38, 2021, doi: 10.2478/sbeef-2021-0008.

[50] M. E. Alabasy, "Conveyor Belt for Barcode Reader Based on Arduino Republic of Iraq Conveyor Belt for Barcode Reader Based on Arduino " no. December, 2021.

[51] "TinyML : From Basic to Advanced Applications," 2021.

الخلاصة

إن انتشار أجهزة إنترنت الأشياء (IoT) واحتياجات التطبيقات الفعالة والذكية يزيد من الحاجة إلى أجهزة منخفضة التكلفة وفعالة تستهلك الطاقة. عادةً ما يكون لهذه الأجهزة موارد محدودة وليس من السهل جدًا عليها التعامل مع العديد من التطبيقات التي تحتاج إلى حلول ذكية مثل الذكاء الاصطناعي ، بما في ذلك التعلم العميق. يبحث الباحثون والمطورون عن خوارزميات ذكاء اصطناعي قابلة للتطبيق ليتم تنفيذها على أجهزة محدودة الموارد مثل وحدات التحكم الدقيقة ، بما في ذلك Arduino.

تتمثل إحدى التحديات التي تواجه تنفيذ الخوارزمية الذكية على الأجهزة ذات الموارد المحدودة في صعوبة تنفيذ خوارزميات الموارد المطلوبة بشدة مثل الشبكات العصبية التلافيفية ، والتي تعد أحد أنواع التعلم العميق على Arduino نظرًا لمحدودية الموارد. في هذا العمل ، تم اقتراح شبكة CNN للرد على الأوامر الصوتية. يتم تدريب هذه الشبكة واختبارها على الخادم السحابي باستخدام مجموعة البيانات الصوتية الخاصة بنا والتي تتضمن 400 عينة وأربع فئات. ومع ذلك ، تتطلب هذه الشبكة الكثير من الموارد الحسابية. للتغلب على هذا التحدي ، قمنا بتطبيقه على إطار عمل Tiny Machine Learning (TinyML) لأن Arduino محدود في الموارد.

تم بناء وتركيب الشبكة المعدلة باستخدام TinyML على جهازي حافة ، الأول تم تنفيذه على جهاز Arduino ، وبذلك حصل على دقة (98.8٪) ، حيث كان الوقت المستغرق في التدريب(6m 50s) أما زمن المعالجة فهو (328ms)
والثاني تم تنفيذه على الهاتف المحمول ، ودقة تبلغ (100٪) تم الحصول عليها. أي أن معظم الأصوات تم تصنيفها بشكل صحيح.

جمهورية العراق
وزارة التعليم العالي والبحث العلمي
جامعة بابل كلية الهندسة / قسم الهندسة الكهربائية

تنفيذ خوارزمية التعلم الآلي المرتكز على الأردوينو

رسالة
مقدمة الى كلية الهندسة
كجزء من متطلبات نيل درجة الدبلوم العالي في جامعة بابل
في كلية الهندسة/ قسم الهندسة الكهربائية/الكترونيك واتصالات

من قبل:
هدى رحيم محمد عبد

إشراف:
أ.م.د. هلال اليباوي

1443هـ 2022 م