Indonesian Journal of Computer Science
ISSN 2302-4364 (*print*) and 2549-7286 (*online*)
Jln. Khatib Sulaiman Dalam, No. 1, Padang, Indonesia, Telp. (0751) 7056199, 7058325
Website: ijcs.stmikindonesia.ac.id | E-mail: ijcs@stmikindonesia.ac.id

Portable Cough Classification System Based on Sound Feature Extraction Using Tiny Machine Learning

Lathifah Arief, Mutiah Risky, Derisma, Werman Kasoep, Nefy Puteri
lathifah.arief@it.unand.ac.id, mutiah.risky@gmail.com, derisma@it.unand.ac.id, werman.kasoep@it.unand.ac.id, nefyputeri@it.unand.ac.id
Univesitas Andalas

Information

Submitted : 1 Oct 2021
Reviewed : 15 Oct 2021
Accepted : 27 Oct 2021

Keywords

Cough, MFCC, Arduino Nano 33 BLE Sense, Neural Network Classifier, Tiny Machine Learning

Abstract

Cough is one of the most common markers that can provide information in diagnosing a disease. More specifically, cough is a common symptom of many respiratory infections. There are several types of cough, including: dry cough, wet cough (cough with phlegm), croup cough and whooping cough. This study aims to create a system that can classify the sounds of coughing up phlegm, dry cough, whooping cough and croup cough. The system development uses the concept of tiny machine learning. In the system built, Arduino Nano 33 BLE Sense is used as a control device and LED is used as an output device.

In this study, the classification of dry cough, wet cough, croup cough and whooping cough was performed using the MFCC voice feature extraction. In the process of classifying coughing sounds using the Neural Network Classifier, the system has a percentage of dataset training accuracy from a total of 5 classes (croup, dry, noise, wet, whooping) of 97.1% by applying an epoch value of 500, window size 3000ms and a window increase of 500ms.

A. Introduction

Based on data from Google until January 2021, there were more than 96.2 million confirmed positive cases of COVID-19 worldwide. This has led to an increasing need for screening and early diagnosis tools to reach a very large and dispersed population. For diagnostic purposes, the World Health Organization (WHO) describes the main symptoms of COVID-19 as high temperature, difficulty breathing, and cough [1]. Cough is one of the most common markers that can provide information in diagnosing a disease. More specifically, cough is a common symptom of many respiratory infections. There are several types of cough, including: dry cough, wet cough (cough with phlegm), croup cough and whooping cough.

In a pandemic condition where disease diagnosis is expected to be carried out without direct contact between patients and doctors, the question of the type of cough is one that is difficult for patients to answer. Usually doctors or health experts distinguish the sound of coughing manually. This method is done by listening to the patient's cough directly by utilizing the sense of hearing, namely the ear. This way of listening directly is certainly not what is expected in the conditions of the COVID-19 pandemic because it presents risks for doctors or other health workers who examine them.

Even if it is circumvented by playing a recorded coughing sound to the doctor, this method has shortcomings in several factors. Not all doctors or health workers can easily distinguish the types of coughs from sound recordings. Human factors that present a high level of possible misdiagnosis include age factors that cause hearing loss, stress or fatigue factors that cause inaccuracy, and also the influence of work experience factors [2]. If a patient with a cough, for example, is diagnosed with lung cancer on his first doctor's visit and later medical records show that the patient really does have the flu, that would be considered a misdiagnosis. Therefore, in this pandemic situation, it is clear that a system is needed that can help classify the type of cough based on the coughing sound produced. by patients automatically, accurately and consistently.

Several studies have been conducted previously to provide convenience for medical personnel in diagnosing cough disease quickly and computerized. In research [3] an automatic cough detection system was built with an approach to monitoring the frequency of coughing sounds designed using the Digital Signal Processing (DSP) algorithm. This DSP algorithm extracts features such as Linear Predictive Coding (LPC) coefficients, Mel Frequency Cepstral Coefficients (MFCC) and spectral characteristics of a sound signal to detect whether the sound is categorized as coughing or not. In another study [4], a classification system for coughing sounds was made with feature extraction using Fast Fourier Transform (FFT) and Power Spectral Density (PSD) as well as applying the Artificial Neural Network-Back Propagation (ANN-PB) method.

To differentiate between types of cough based on the characteristics of the sound, a separate study [5] examined the sound waveforms and spectrograms of dry and wet coughs to extract characteristic features of the sound associated with the presence of mucus in the airways as a marker of wet cough. From other literature, it is known that dry cough is usually distinguished by a loud coughing sound [5]. Whooping cough can be characterized by a series of loud coughing

sounds that occur continuously [6]. Meanwhile, croup cough is characterized by infection in the upper respiratory tract which causes a characteristic coughing sound such as barking [7].

These studies generally still use computing devices in the form of computers with computing and simulation software (e.g Matlab) so that they still lack the portability aspect of the system to be used flexibly in different locations. Fortunately, the current development of edge computing technology has enabled the machine learning or deep learning inference processes needed for cough detection and classification to be carried out in real-time on relatively compact and portable embedded devices. Taking that fact into consideration, this research proposes a new approach by developing a portable system for classifying cough types based on audio feature extraction using MFCC, Tiny Machine Learning and Arduino Nano 33 BLE Sense.

B. Theory

Cough

The nature of cough is important in pathological studies for diagnostic purposes. A typical cough sound is usually divided into three phases: (1) explosive expiration due to the sudden opening of the glottis, (2) an intermediate phase with attenuation of coughing sounds, and (3) a voiced phase due to closure of the vocal cords. In fact, there are various patterns of cough that occur; For example, some coughing sounds have only two phases (intermediate phase and voiced phase) and the blast phase is usually prolonged due to several diseases [9].

Wet Cough (Cough with phlegm)

Usually wet cough is caused by the presence of foreign objects (such as bacteria, viruses) that cause inflammation and secretions in the lower respiratory tract: bronchitis, asthma, pneumonia. Sometimes it can also be triggered by the upper respiratory tract.

Dry Cough

Dry Cough is an indication name for a dry type of cough without phlegm. A typical cough sound signal, consists of three phases. Phase 1: initial opening blast, Phase 2: noisy airflow and Phase 3: glottal closure [8]. There are cases where Stage 3 is not seen in the cough signal [9].

Whooping Cough (Whooping Cough)

Whooping cough is a highly contagious respiratory infection. In many people, this is characterized by a severe cough followed by high-pitched breathing that sounds like a scream [10].

Whooping cough, also known as pertussis, is a highly contagious respiratory infection caused by the bacterium Bordetella pertussis. In 2018, there were more than 151,000 cases of pertussis globally. Pertussis spreads easily from person to person mainly through droplets produced by coughing or sneezing. This disease is most dangerous in infants, and is the leading cause of illness and death in this age group. [6]

Deaths associated with whooping cough are rare but occur most often in infants. That's why it's so important for pregnant women - and anyone else who will be in close contact with the baby - to be vaccinated against whooping cough. When an infected person coughs or sneezes, tiny droplets containing germs are sprayed into the air and inhaled into the lungs of anyone who happens to be nearby.

Croup Cough

Croup is a common respiratory infection in children, croup can cause inflammation of the upper airways that restricts normal breathing and produces a coughing sound that is usually described as a "barking cough".

A 2008-2009 study in Australia on children aged 0-14 years found croup in 1.2% of subjects or about 154,000 times per year [7]. It is prescribed to be most common in children aged 1-4 years. Upper airway inflammation caused by infection restricts normal breathing leading to a 'croupy' or 'barking' cough sometimes accompanied by hoarseness, and respiratory distress [11].

A croup cough can be life-threatening if severe. It is the most common airway obstruction in children between the ages of 6 months and 6 years, peaking between the ages of 1 and 2 years. A characteristic coughing sound is the main clinical feature used in clinical practice to diagnose croup. Doctors make subjective judgments on coughing sounds such as barking after listening to them. Therefore the diagnosis of croup, is limited to human perception and depends on the skill of the doctor [12].

Tiny Machine Learning

Tiny Machine Learning or often known as TinyML is a rapidly growing field of machine learning technology and applications including algorithms, hardware, and software capable of analyzing sensor data on devices (vision, audio, IMU, biomedical, etc.) very low, typically in the mW range, enabling a variety of always-on use cases targeting battery-operated devices [13]. Machine Learning is one aspect of Artificial Intelligence (AI) where computing systems are able to learn from data and make decisions. Machine Learning has become one of the most important areas in development organizations looking for innovative ways to understand data assets to help businesses reach new levels of understanding [14]. One of the most studied research topics on supervised machine learning is case classification [15].

Arduino Nano 33 BLE Sense

The Arduino Nano 33 BLE Sense board has been designed to offer a power-efficient and cost-effective solution for builders who wish to have energy-efficient bluetooth connectivity. The Nano 33 BLE Sense is the same as the Arduino Nano 33 BLE with the addition of a set of sensors [17].

Cough

The nature of cough is important in pathological studies for diagnostic purposes. A typical cough sound is usually divided into three phases, namely: (1) explosive expiration due to the sudden opening of the glottis, (2) an intermediate

phase with attenuation of the coughing sound, and (3) a voiced phase due to coughing.

Figure 1. Arduino Nano 33 BLE Sense

The Arduino Nano 33 BLE Sense is an evolution of the traditional Arduino Nano, but features a much more powerful processor. This allows for larger programs than with the Arduino Uno (which has only 1MB of program memory, 32 times more), and with more variables (128 times more RAM). The main processor includes other amazing features like bluetooth pairing via NFC and a very low power consumption mode.

Mel Frequency Cepstral Coefficient (MFCC)

One of the most important types of parametric representation used in speech recognition is the Mel Frequency Cepstral Coefficient (MFCC). MFCC describes the characteristics of the audio data frame in the cepstral domain. This shows the short-term power spectrum of a sound.

The MFCC feature extraction technique basically involves windowing the signal, applying DFT, taking a log of magnitude, and then bending the frequency on the Mel scale, followed by applying DCT (Discrete Cosine Transform) in reverse [21].

Edge Impulse

Edge impulse provides services that enable machine learning for all developers with open source tools. Edge Impulse enables easy collection of real sensor data, signal processing directly from raw data to neural network, testing and deployment to any target device. This open source allows collecting data from or deploying code to any device.

Using Edge Impulse, it is easy to collect sensor data, train a machine learning (ML) model on this data in the cloud, and then apply the model back to Arduino devices [29]. In this way, it is possible to integrate the model into the Arduino sketch with a single function call. Sensors are then much smarter and able to understand complex events. Time series data is a type of data that is collected according to the order of time in a certain time span. then on the edge impulse it is necessary to set the window size and window increase time.

Neural Network Classification

Neural Network Classification (NN) is an electronic network of neurons that is relatively rough based on the neural structure of the brain. Neural Network Classification processes the recordings one by one, and learns by comparing their record classification with the actual known record classification.

Neural Network takes inspiration from the learning process that occurs in the human brain. NNs consist of artificial network functions, called parameters, that allow a computer to learn, and improve itself, by analyzing new data. Each parameter, sometimes also referred to as a neuron, is a function that produces an output, after receiving one or more inputs. The output is then passed on to the next layer of neurons, which uses it as the input of its own function, and produces further output. The output is then passed on to the next layer of neurons, and continues until each layer of neurons has been considered, and the terminal neuron has received its input. The terminal neurons then output the final results for the model.

C. Research Methodology

System General Design

The following is a general design of the system that will be made.

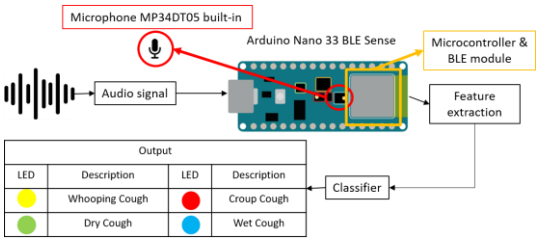

Figure 2. General Design

Based on Figure 2 above, the microphone on the Arduino Nano 33 BLE Sense captures the sound in the surrounding environment. Then the microphone on the microcontroller detects the captured sound and will be classified using the Neural Network Classifier on the Arduino Nano 33 BLE Sense.

Furthermore, the results of the system classification are displayed in the form of an output in the form of LEDs, where each LED color represents a different type of coughing sound, namely: yellow LED to describe coughing up phlegm; Orange LED to represent dry cough; Red LED to represent whooping cough; and a blue LED to represent croup cough.

Process Design

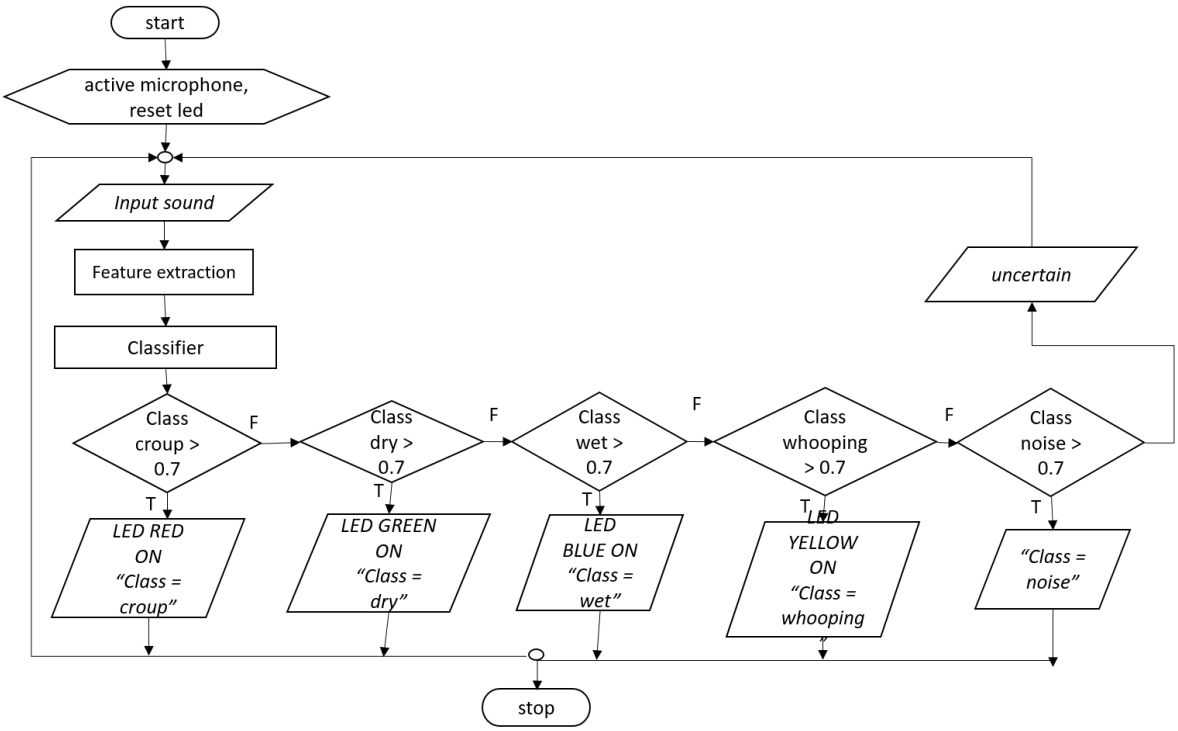

Figure 3. Process Flowchart

Based on Figure 3 above, this system is divided into several process stages, namely: The first process is the microphone is active and receives signals or sound waves from the surrounding environment. Then the system will classify the voice signal. Furthermore, the results of the system classification will be displayed in the form of an output in the form of a lit LED.

Hardware Design

The overall hardware design can be seen in Figure 4.

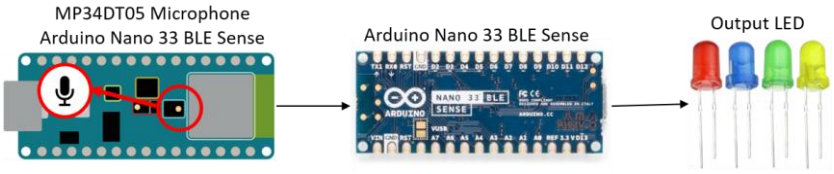

Figure 4. Hardware Design

Based on Figure 4. above, the working principle of each device is as follows.

1. The built-in Arduino Nano 33 BLE Sense microphone sensor MP34DT05, is used as a tool or component used to record audio signals and then convert them into an analog signal electrical waveform.
2. Arduino Nano 33 BLE Sense, acts as a controller and processing for systems with machine learning and processing for reading voice signal data and then performs the classification process for voice signals.
3. Output LED, serves to display the results of the system classification in the form of LEDs where each LED color represents the type of system classification for different cough sounds.

Software Design

The software used in this system is the Arduino Program and the Edge Impulse Application, the program that will be used is divided into three, namely the system training program, the system testing program, and the program displaying the output of the system classification results in the form of LEDs.

System Training Design

In designing this training system, it is useful to conduct training on the system in order to be able to get the results of the classification of the type of coughing sound according to what has been determined. This training process begins by labeling each audio dataset and then proceeds to create an impulse design. The impulse design is organized into 4 blocks: time series data, processing, learning, and output features.

Then perform feature extraction using MFCC, then the system will take the dataset randomly and then classify it using Neural Network (Hard Library). The purpose of conducting system training is to get the data model as desired. This software will be designed based on the flowchart below:

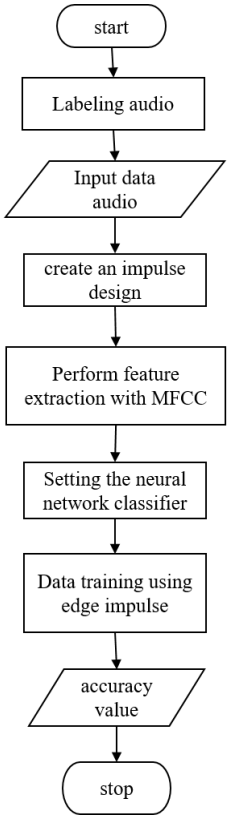

Figure 5. System Training Flowchart

System Testing Design

At the design stage, system testing is carried out by classifying cough sounds using live testing on Edge Impulse which is connected to a mobile phone. Then testing was also carried out on the Arduino Nano 33 BLE Sense system. For more details on how this software is designed, it can be seen in the following algorithm.

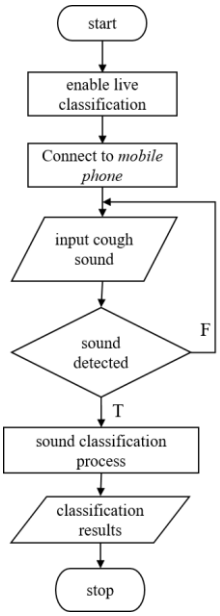

Figure 6. System Testing Flowchart

D. Result and Discussion

The classification system for the type of cough based on voice feature extraction is implemented based on the tools and materials described in the previous chapter. So that the implementation and testing process is structured, the implementation and testing is divided into two parts of discussion, namely hardware implementation and software implementation.

Hardware Implementation

Hardware implementation is the process of assembling (wiring) hardware components that have been adapted to the hardware requirements of the system. To build a cough classification system based on this sound feature extraction, the hardware used is Arduino Nano 33 BLE Sense, LED, built-in microphone on Arduino Nano 33 BLE Sense and jumper cables. These components are arranged as shown in Figure 7.

Figure 7. Hardware Implementation

In Figure 7. these hardware components have the following functions:

- Arduino Nano 33 BLE Sense, functions as the main system and as a controller that will process data, process the work of the system and run a cough classification program based on sound feature extraction.
- MP34DT05 microphone sensor built-in Arduino Nano 33 BLE Sense, serves to receive and capture sound signals from the environment.
- LEDs here serve as output media for displaying the classification result, i.e the type of coughing, identified from the sound provided.

Software Implementation

Software implementation is needed so that the hardware can work according to system requirements. The process of implementing the software into the system is as follows:

Collecting Dataset

In this study, data collection was carried out on the Edge Impulse application. Before collecting the dataset on Edge Impulse, the dataset is selected. The process of selecting and sorting datasets is done using the Audacity application. The selection process carried out is cutting and eliminating datasets that have noise or a lot of idle time. The process of removing the noise and silent parts is done so that the model does not study this noise/silent part as a cough dataset. The process of selecting and cutting the noise/silent part of the coughing sound can be seen in Figures 8 and 9.

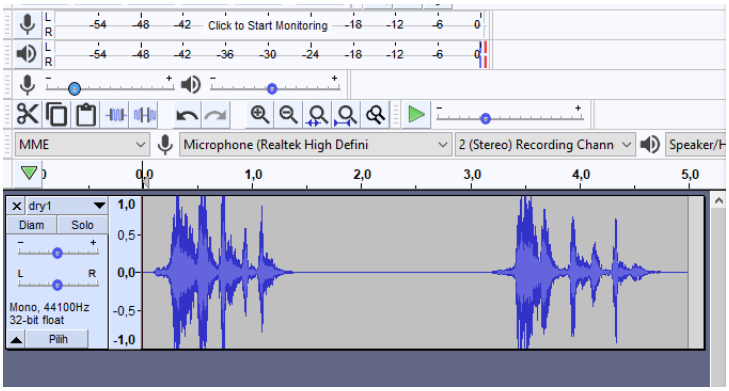

Figure 8. Cough audio sample before cutting the noise/silent part

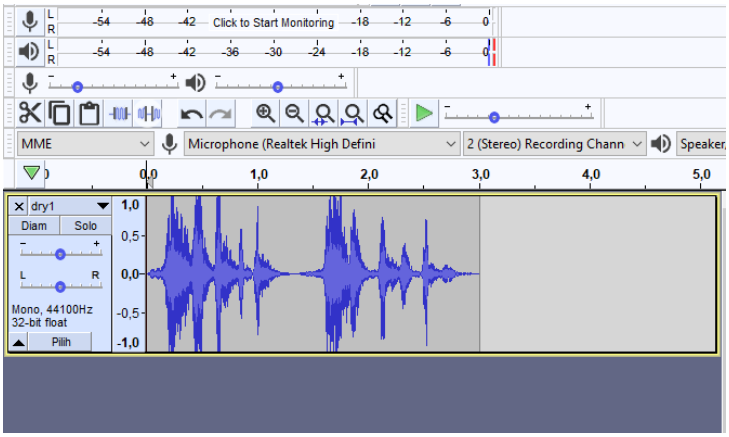

Figure 9. Cough audio sample after cutting the noise/silent part

The dataset collected in this study must have a frequency of 16000 Hz so that the data matches the required frequency on the Arduino Nano 33 BLE Sense. The process of collecting datasets on Edge Impulse is done using a file uploader. Labeling is adjusted to each file name according to the format. The number of datasets collected is 1877 seconds. The dataset that has been collected is then divided into two group:

A. Training Dataset

The total dataset for training collected was 1501 seconds (25 minutes 1 second) consisting of:

Table 1. Training Dataset

Label	Time (ms)
Croup	342
Dry	245
Noise	323
Wet	268
Whooping	323
Total	**1501**

B. Testing Dataset

The total dataset for testing collected was 375 seconds (6 minutes 15 seconds) consisting of:

Table 2. Testing Dataset

Label	Time (ms)
Croup	66
Dry	75
Noise	72
Wet	67
Whooping	94
Total	**375**

Impulse Design

Impulse design is done to take a collection of data and then divide it into smaller windows. Then use signal processing blocks to perform voice feature extraction and classify new data using learning blocks. Impulse design includes 4 processes, namely:

- *Time Series Data*, this block aims to set the duration of the window size, increase the window and add zero-pad data if needed. window size is used to determine the length of duration in one audio sample that will be processed to get a good accuracy value. The increase window is used to set the increase in the duration of the sample that will be used as the next sample. The next step is to add zero-pad data so that the data sample that is shorter than the window size can still be processed by adding a zero value to the data sample that is shorter than the window size so that the system gets better performance.

- *Processing Block*, the signal processing block to extract features from the audio signal used in this research is Mel Frequency Cepstral Coefficients (MFCC) with audio input.
- *Learning Block*, the learning block used in this research is the Neural Network with the addition of the Keras library which learns patterns from data, and applies them to new data.
- *Output Features*, based on the dataset that has been collected and labeled, this system has 5 outputs, namely croup, dry, noise, whooping and wet.

MFCC Block Configuration

In this section, the configuration of the MFCC block is performed, and shows a visualization of the MFCC audio output known as a spectrogram. The pattern seen in the spectrogram contains information about the type of sound it represents.

The spectrogram generated by the MFCC block will then be forwarded to the Neural Network architecture which will recognize the pattern of each type of sound. Before being forwarded to the Neural Network to conduct dataset training, it is necessary to create MFCC blocks on all audio windows through the Generate Features.

After doing Generate Features, the results of the 3D representation show the distribution of the sample dataset. In the previous impulse design, the number of window size and window increase settings have been made.

Neural Network Configuration

The network that will be trained in this study uses MFCC as input, then mapped into one of five types of classification croup, dry, whooping, wet and noise as the output form.

Training Process

The Neural Network training process in this study is a sequential model. In this training program, a learning rate of 0.005 is entered. In this study, the training process was carried out by choosing the epoch values of 100, 200, 300, 400 and 500. The following are the results of the dataset training process in Table 3.

Table 3. Training Result

			Accuration per Classification (%)				
Epoch	**Accuracy (%)**	**Loss**	**Croup**	**Dry**	**Noise**	**Wet**	**Whooping**
100	94.07.00	00.22	98.02	87.02.00	97.05.00	89.02.00	92.01.00
200	95.05.00	00.27	100.00	87.02.00	99.02.00	89.02.00	90.05.00
300	96	00.18	100.00	87.02.00	100.00	83.08.00	95.02.00
400	96	00.18	100.00	87.02.00	100.00	83.08.00	95.02.00
500	97.01.00	00.16	100.00	91.05.00	100.00	89.02.00	95.02.00

Based on Table 3 which contains the training results, the epoch 500 value gives the best accuracy value compared to the use of epoch 100, 200, 300 and 400. In addition, epoch 500 also gives the lowest loss value.

Software Testing

Testing on Model

The results of the training process that has been carried out show that the model works well on the training data. Using the testing dataset that has been prepared previously, the testing process is done to ensure that the model also achieve a good result on new data. In this study, the process was carried out by testing the model achieved at each of the epoch values namely 100, 200, 300, 400 and 500. The testing process carried out on the different epoch values had different accuracy results. The following is Table 4 that contains the results of the model testing process.

Table 4. Testing Result

Epoch	Accuracy (%)
100	76.47
200	77.12
300	78.65
400	78.65
500	80.83

Based on the test results using the epoch values of 100, 200, 300, 400 and 500, the best model testing result is the use of the 500 epoch values. The following are the results of the testing model on the 500 epoch:

Table 5. Testing Model

No	Sample Name	Label	Duration	Accuracy	Classification Result
		Croup			
01.00	croup.t1.wav	croup	9s	100%	12 croup
02.00	croup.t2.wav	croup	8s	100%	11 croup
03.00	croup.t3.wav	croup	14s	100%	22 croup
04.00	croup.t4.wav	croup	16s	100%	26 croup
05.00	croup.t5.wav	croup	10s	26%	5 noise, 4 croup, 3 wet, 3 uncertain
06.00	croup.t6.wav	croup	10s	100%	14 croup
		Dry			
07.00	dry.t1.wav	dry	4s	100%	2 dry
08.00	dry.t2.wav	dry	3s	100%	1 dry
09.00	dry.t3.wav	dry	4s	50%	1 dry, 1 whooping

10.00	dry.t4.wav	dry	4s	100%	3 dry
11.00	dry.t5.wav	dry	5s	100%	5 dry
12.00	dry.t6.wav	dry	17s	72%	21 dry, 7 wet, 1 uncertain
13.00	dry.t7.wav	dry	9s	16%	4 wet, 4 uncertain, 2 dry, 2 whooping
14.00	dry.t8.wav	dry	3s	0%	1 uncertain
15.00	dry.t9.wav	dry	5s	80%	4 dry, 1 wet
16.00	dry.t10.wav	dry	6s	100%	7 dry
17.00	dry.t11.wav	dry	18s	48%	15 dry, 12 wet, 4 uncertain
18.00	dry.t12.wav	dry	4s	0%	2 wet, 1 whooping
		Noise			
19.00	noise.t1.wav	noise	22s	56%	22 noise, 7 wet, 4 dry, 4 uncertain
20.00	noise.t2.wav	noise	20s	100%	35 noise
21.00	noise.t3.wav	noise	27s	100%	49 noise
22.00	noise.t4.wav	noise	3s	100%	1 noise
		Wet			
23.00	wet.t1.wav	wet	6s	66%	4 wet, 2 whooping
24.00.00	wet.t2.wav	wet	4s	100%	2 wet
25.00.00	wet.t3.wav	wet	4s	100%	2 wet
26.00.00	wet.t4.wav	wet	3s	100%	1 wet
27.00.00	wet.t5.wav	wet	4s	0%	3 whooping
28.00.00	wet.t6.wav	wet	5s	60%	3 wet, 2 uncertain
29.00.00	wet.t7.wav	wet	6s	100%	7 wet
30.00.00	wet.t8.wav	wet	4s	100%	3 wet
31.00.00	wet.t9.wav	wet	5s	100%	4 wet
32.00.00	wet.t10.wav	wet	4s	66%	2 wet, 1 whooping
33.00.00	wet.t11.wav	wet	5s	100%	4 wet
34.00.00	wet.t12.wav	wet	4s	100%	3 wet
35.00.00	wet.t13.wav	wet	5s	75%	3 wet, 1 whooping
36.00.00	wet.t14.wav	wet	4s	100%	2 wet

37.00.00	wet.t15.wav	wet	4s	100%	2 wet
		Whooping			
38.00.00	whooping.t1.wav	whooping	4s	100%	3 whooping
39.00.00	whooping.t2.wav	whooping	6s	100%	6 whooping
40.00.00	whooping.t3.wav	whooping	8s	100%	11 whooping
41.00.00	whooping.t4.wav	whooping	5s	100%	5 whooping
42.00.00	whooping.t5.wav	whooping	5s	100%	5 whooping
43.00.00	whooping.t6.wav	whooping	4s	100%	3 whooping
44.00.00	whooping.t7.wav	whooping	4s	100%	3 whooping
45.00.00	whooping.t8.wav	whooping	5s	25%	1 croup, 1 dry, 1 noise, 1 whooping
46.00.00	whooping.t9.wav	whooping	5s	80%	4 whooping, 1 uncertain
47.00.00	whooping.t10.wav	whooping	5s	100%	5 whooping
48.00.00	whooping.t11.wav	whooping	5s	0%	4 wet, 1 uncertain
49.00.00	whooping.t12.wav	whooping	5s	60%	3 whooping, 2 uncertain
50.00.00	whooping.t13.wav	whooping	5s	100%	5 whooping
51.00.00	whooping.t14.wav	whooping	5s	100%	4 whooping
52.00.00	whooping.t15.wav	whooping	5s	100%	4 whooping
53.00.00	whooping.t16.wav	whooping	5s	100%	5 whooping
54.00.00	whooping.t17.wav	whooping	5s	100%	5 whooping

The testing results presented in a confusion matrix as seen in Table 6 below.

Table 6. Confusion Matrix

		predicted labels					
		croup	**dry**	**noise**	**wet**	**whooping**	**uncertain**
actual values	croup	89	0	5	3	0	3
	dry	0	61	0	26	4	10
	noise	2	4	107	7	0	4

wet	0	0	0	42	7	2
whooping	1	1	1	4	72	4

To evaluate the performance of the classifier, the following steps are used:

True Positive (TP), subjects with class 'x' correctly classified the cough as class 'x'.

False Positive (FP), the subject with the condition class '~x' but the classifier classifies the cough as class 'x'.

True Negative (TN), subjects with class '~x' and classifier classified the cough as class '~x'.

False Negative (FN), the subject with cough class 'x' and classifier classified the sound as class '~x'.

Sensitivity, refers to the classifier's ability to correctly identify subjects with class 'x'.

Sensitivity = TP /(TP + FN)

Classification Accuracy of the Classifier, this is the ratio of the total number of correct assessments to the total number of assessments.

Accuracy = (TN + TP)/(TN + TP + FN + FP)

Table 7. Results of Data Performance Evaluation for Class Classification of Croup Cough

Classification Result	***Value***
Total Amount of Data	459
True Positive (TP)	89
True Negative (TN)	356
False Positive (FP)	3
False Negative (FN)	11
Sensitivity / Recall	89%
Accuracy	97%

Based on table 7. above, the results of the performance evaluation of the data used for the classification of croup cough class obtained an accuracy value of 97%. Classification is able to separate different data by correctly classifying subject data in the croup class. It can be seen from 100 cough validation data features, obtained True Positive (TP) as many as 89 data features.

Table 8. Results of Data Performance Evaluation for Class Classification of Dry

Classification Result	***Value***
Total Amount of Data	459
True Positive (TP)	61
True Negative (TN)	353
False Positive (FP)	5
False Negative (FN)	40
Sensitivity / Recall	60%
Accuracy	90%

Based on table 8. above, the results of the evaluation of the performance of the validation data used for the dry cough class classification obtained an accuracy value of 90%. Classification is able to separate different data by correctly classifying subject data in the dry class. It can be seen from 101 cough validation data features, obtained True Positive (TP) as many as 61 data features.

Table 9. Results of Data Performance Evaluation for Class Classification of Noise Cough

Classification Result	*Value*
Total Amount of Data	459
True Positive (TP)	107
True Negative (TN)	329
False Positive (FP)	6
False Negative (FN)	17
Sensitivity / Recall	86%
Accuracy	95%

Based on table 9. above, the results of the evaluation of the performance of the validation data used for class classification of noise obtained an accuracy value of 95%. Classification is able to separate different data by correctly classifying the subject data in the noise class. Seen from 124 samples of noise class data, obtained True Positive (TP) as many as 107 data samples.

Table 10. Results of Data Performance Evaluation for Class Classification of Wet Cough

Classification Result	*Value*
Total Amount of Data	459
True Positive (TP)	42
True Negative (TN)	368
False Positive (FP)	40
False Negative (FN)	9
Sensitivity / Recall	82%
Accuracy	89%

Based on table 10 above, the results of the evaluation of the performance of the data used for the classification of the wet class obtained an accuracy value of 89%. Classification is able to separate different data by correctly classifying subject data in the wet class. It can be seen from 51 samples of wet class data, obtained True Positive (TP) as many as 42 data samples.

Table 11. Results of Data Performance Evaluation for Class Classification of Whooping Cough

Classification Result	*Value*
Total Amount of Data	459
True Positive (TP)	72
True Negative (TN)	365
False Positive (FP)	11
False Negative (FN)	11

Sensitivity / Recall	87%
Accuracy	95%

Based on table 11. above, the results of the evaluation of the performance of the validation data used for the classification of the whooping class obtained an accuracy value of 95%. Classification is able to separate different data by correctly classifying subject data in the whooping class. It can be seen from 83 samples of whooping class data, obtained True Positive (TP) as many as 72 data samples.

Hardware Testing and Analysis

Arduino Nano 33 BLE Sense Test and Analysis

The Arduino Nano 33 BLE Sense test and analysis was carried out to find out how the performance and memory usage was when the program was run on the Arduino Nano 33 BLE Sense.

the program uses 259624 bytes or 26% of the total maximum storage space of 983040. And global variables used are 50032 bytes or 19% of dynamic memory with a maximum number of 262144 bytes. This means that the program that is run on the Arduino Nano 33 BLE Sense can run well.

Testing and Analysis of LED Components for classification with Confidence Rating > 0.70

The testing of the cough classification indicator program aims to test the accuracy of the indicators in classifying the system according to predetermined conditions. In this test, LED is used as an indicator of cough classification. There are 4 LEDs that serve as indicators, namely, a red LED as an output for the classification of croup cough, a green LED as a dry cough classification output, a blue LED as a wet cough classification output (phlegm) and a yellow LED as a whooping cough classification output (whooping).

Percentage of Success = Total Percentage of Success Trials
Percentage of Success = 19/25 x 100%
Percentage of Success = 76%

Table 12. Cough Classification Indicator Test

No	Name	Duration (Frequency)	***R*** **(Croup)**	**G (Dry)**	**B (Wet)**	**Y (Whooping)**	Notes
1	Croup-1	45s (5x)	4x	-	-	1x	Fit
2	Croup-2	10s (2x)	2x	-	-	-	Fit
3	Croup-3	92s (6x)	5x	-	-	-	Fit
4	Croup-4	34s (6x)	6x	-	-	-	Fit
5	Croup-5	25s (4x)	4x	-	-	-	Fit
6	Dry-1	11s (3x)	-	2x	-	-	Fit
7	Dry-2	6s (2x)	-	1x	-	-	Fit
8	Dry-3	10s (3x)	-	1x	-	1x	Not Fit
9	Dry-4	9s (2x)	-	1x	-	-	Fit
10	Dry-5	5s (1x)	-	1x	-	-	Fit
11	Wet-1	8s (2x)	-	-	1x	-	Fit
12	Wet-2	6s (2x)	-	-	-	1x	Not Fit
13	Wet-3	5s (1x)	-	-	-	-	Not Fit
14	Wet-4	5s (1x)	-	-	-	-	Not Fit

15	Wet-5	6s (2x)	-	1x	1x	-	Not Fit
16	Whooping-1	26s (3x)	1x	-	-	2x	Fit
17	Whooping-2	21s (2x)	-	1x	-	1x	Not Fit
18	Whooping-3	22s (2x)	-	-	-	1x	Fit
19	Whooping-4	54s (4x)	-	1x	-	2x	Fit
20	Whooping-5	72s (7x)	-	-	-	6x	Fit
21	Noise-1	5s (1x)	-	-	-	-	Fit
22	Noise-2	5s (1x)	-	-	-	-	Fit
23	Noise-3	5s (1x)	-	-	-	-	Fit
24	Noise-4	5s (1x)	-	-	-	-	Fit
25	Noise-5	5s (1x)	-	-	-	-	Fit

Based on Table 12 above, it can be seen that from 25 experiments conducted with 5 classes of croup cough sounds, 5 classes of dry cough sounds, 5 types of wet cough sound classes, 5 classes of whooping cough sounds and 5 classes of noise sounds with different amounts of sound duration. The difference and frequency of episodes of different coughs according to the table above resulted in a percentage of successful cough classification of 76%.

Overall System Testing and Analysis

Overall system testing is carried out to determine the ability of the system that has been built in solving the given problem, namely determining the type of cough sound based on voice feature extraction and indicating it via LED according to each type of cough.

Testing this system is done by looking at how the system gives an indication of the sound given. The results of system testing can be seen in Table 13 below. Overall system testing result in the table shows there are 5 incorrect classification results and 20 correct classification results according to the actual label using epoch 500. The percentage of the system's success in indicating the type of coughing sound is:

Percentage of Success = Number of System Success Number of Trials

Percentage of Success = (20/25) x 100%

Success Percentage = 80%

Of the 25 data that have gone through the cough sound classification process in the system, the percentage of success is 80%.

Table 13. Overall System Test

No	Class	LED				Serial Monitor	Time(s)	Notes
		R	G	B	Y			
1	Croup	1	0	0	0	croup: 0.97656 **class : croup**	3.476	Fit
2	Croup	1	0	0	0	croup : 0.78516 **class : croup**	3.476	Fit
3	Croup	1	0	0	0	croup: 0.86719 **class : croup**	3.476	Fit
4	Croup	1	0	0	0	croup:0.94922 **class : croup**	3.476	Fit
5	Croup	1	0	0	0	croup:0.98828 **class : croup**	3.477	Fit
6	Dry	0	1	0	0	dry:0.96484 **class : dry**	3.476	Fit
7	Dry	0	0	0	0	**uncertain**	3.476	Not Fit
8	Dry	0	1	0	0	dry:0.79297 **class : dry**	3.476	Fit

9	Dry	0	1	0	0	dry: 0.76562 **class : dry**	3.476	Fit
10	Dry	0	1	0	0	dry:0.75000 **class : dry**	3.476	Fit
11	Wet	0	0	1	0	wet: 0.81641 **class : wet**	3.476	Fit
12	Wet	0	0	0	1	whooping: 0.98828 **class : whooping**	3.476	Not Fit
13	Wet	0	0	0	0	**class : uncertain**	3.476	Not Fit
14	Wet	0	1	0	0	dry: 0.78906 **class : dry**	3.477	Not Fit
15	Wet	0	0	1	0	wet: 0.75000 **class : wet**	3.476	Fit
16	Whooping	0	0	0	1	whooping: 0.86719 **class : whooping**	3.476	Fit
17	Whooping	0	0	0	1	whooping: 0.76562 **class : whooping**	3.477	Fit
18	Whooping	0	0	0	1	whooping: 0.89844 **class : whooping**	3.476	Fit
19	Whooping	0	0	0	1	whooping: 0.80078 **class : whooping**	3.476	Fit
20	Whooping	0	0	0	1	whooping: 0.99609 **class : whooping**	3.476	Fit
21	Noise	0	0	0	0	noise: 0.72656 wet: **class : noise**	3.476	Fit
22	Noise	0	0	0	0	noise: 0.99609 **class : noise**	3.476	Fit
23	Noise	0	0	0	0	noise: 0.98438 **class : noise**	3.477	Fit
24	Noise	0	0	0	0	noise: 0.99609 **class : noise**	3.476	Fit
25	Noise	0	0	0	0	noise: 0.81250 **class : noise**	3.476	Fit

E. Conclusion

Based on testing and analysis of the whole system on a portable system for classifying cough types based on voice feature extraction using tiny machine learning, the conclusions are as follows:

The training process on the classification of the Neural Network algorithm using epoch 500 can increase the accuracy value to produce an accuracy percentage of 97.1% and a loss of 0.16.

The testing process on the Neural Network algorithm classification carried out on Model Testing using epoch 500 produces an accuracy percentage of 80.83%.

The system testing environment affects the classification accuracy value on the system when in a noisy environment, the sensor will detect every incoming sound and affect sensor readings.

The use of a window size of 3000 ms has described a more realistic feature space for concept learning. Spatial distance is quite compatible with conceptual distance. However it is not possible to define a perfect characteristic function based on this feature due to imperfect separation, but this realistic feature space can be used to study useful approximations for characteristic functions with small errors.

The use of zero-pad data helps data samples that are shorter than the window size can still be processed by adding a zero value to data samples that are shorter than the window size so that the system gets better performance.

F. References

[1] G. Deshpande and B. W. Schuller, “An Overview on Audio, Signal, Speech, & Language Processing for COVID-19,” arXiv, pp. 1–5, 2020.

[2] Prihastomo, Ibnu Hafid, “Optimasi Fitur Suara pada Klasifikasi Suara Batuk Basah/Kering Anak-Anak dengan Algoritme Genetika” Universitas Islam Yogyakarta.2018.

[3] M. Mlynczak, K. Pariaszewska, and G. Cybulski, “Automatic cough episode detection using a vibroacoustic sensor,” Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. EMBS, vol. 2015-Novem, no. September, pp. 2808–2811, 2015, doi: 10.1109/EMBC.2015.7318975.

[4] N. Afifah, “Klasifikasi Penyakit Batuk Berdasarkan Sinyal Data Suara Menggunakan Ekstraksi Ciri Fast Fourier Transform Dan Power Spectral Density Dengan Algoritma Jaringan Saraf Tiruan- Propagasi Balik,” vol. 2, no. 2, pp. 2841–2846, 2012.

[5] H. Chatrzarrin, A. Arcelus, R. Goubran, and F. Knoefel, “Feature extraction for the differentiation of dry and wet cough sounds,” MeMeA 2011 - 2011 IEEE Int. Symp. Med. Meas. Appl. Proc., pp. 162–166, 2011, doi: 10.1109/MeMeA.2011.5966670.

[6] World Health Organozation. 2018. "Pertussis". https://www.who.int/health-topics/pertussis#tab=tab_1 . accessed at6 Februari 2021 , 16.20 .

[7] J. Charles, H. Britt, and S. Fahridin, "Croup," Australian Family Physician, vol. 39, no. 5, pp. 269-269, 2010.

[8] J. Liu, Z. Wang, G. Li, X. Xu, and Z. Qiu, “Cough Detection Using Deep Neural Networks,” 2014.
[9] Y. Shi, H. Liu, Y. Wang, M. Cai, and W. Xu, “Theory and application of audio-based assessment of cough,” J. Sensors, vol. 2018, 2018, doi: 10.1155/2018/9845321.
[10] Mayo Clinic Staff. "Whooping Cough". https://www.mayoclinic.org/diseases-conditions/whooping-cough/symptoms-causes/syc-20378973. accessed at 6 Februari 2021 , 16.00 .
[11] C. L. Bjornson and D. W. Johnson, "Croup in children," CMAJ: Canadian Medical Association Journal, vol. 185, no. 15, pp. 1317-1323, 2013.
[12] R. V. Sharan, U. R. Abeyratne, V. R. Swarnkar, and P. Porter, “Automatic croup diagnosis using cough sound recognition,” IEEE Trans. Biomed. Eng., vol. 66, no. 2, pp. 485–495, 2019, doi: 10.1109/TBME.2018.2849502.
[13] Janapa Reddi, Vijay. " The Future of ML is Tiny and Bright ".https://learning.edx.org/course/course-v1:HarvardX+TinyML1+3T2020/block-v1:HarvardX+TinyML1+3T2020+type@sequential+block@31122d81c8364930883bf446d457bb3e/block-v1:HarvardX+TinyML1+3T2020+type@vertical+block@61c2efbfd16445b1a342f1926ab8d47a , accessed at22 Januari 2021, 13.40 .
[14] Oyedeji, A., Salami, A., Folorunsho, O., & Abolade, O. (2020, March 30). Analysis and Prediction of Student Academic Performance Using Machine Learning. JITCE (Journal of Information Technology and Computer Engineering), 4(01), 10-15. https://doi.org/https://doi.org/10.25077/jitce.4.01.10-15.2020
[15] Nofriani, N. (2020, September 30). Machine Learning Application for Classification Prediction of Household’s Welfare Status. JITCE (Journal of Information Technology and Computer Engineering),4(02),72-82. https://doi.org/https://doi.org/10.25077/jitce.4.02.72-82.2020
[16] Lutkevich, Ben. " Definition Microcontroller (MCU) " . https://internetofthingsagenda.techtarget.com/definition/microcontroller, accessed at22 Januari 2021 , 13.23 .
[17] Arduino Team. 2018. " Getting started with the Arduino NANO 33 BLE Sense ". https://www.arduino.cc/en/Guide/NANO33BLESense/ . accessed at23 Januari 2021 , 20.23 .
[18] Tensorflow.2021."Get started with TensorFlow Lite". https://www.tensorflow.org/lite/guide/get_started . accessed at1 Februari 2021 , 13.30 .
[19] Tanwar, Sanchit. 2019."Building our first neural network in keras" . https://towardsdatascience.com/building-our-first-neural-network-in-keras-bdc8abbc17f5#:~:text=Keras%20is%20a%20simple%20tool,output%20is%20of%204%20values.&text=In%20our%20neural%20network%2C%20we,of%2016%20and%2012%20dimension . accessed at1 Februari 2021 , 15.52 .
[20] Keras. "Keras". https://keras.io/ . accessed at29 Januari 2021 , 11.30 .
[21] P. Works and S. Recognition, “MFCC Features,” pp. 7–16, 1959, doi: 10.1007/978-3-319-49220-9.

[22] Edge Impulse .2020. " TinyML for All Developers with Edge Impulse" . https://www.hackster.io/news/tinyml-for-all-developers-with-edge-impulse-2cfbbcc14b90 . accessed at5 Februari 2021 , 15.42 .

[23] Hackster .2020. " Edge Impulse Brings TinyML to Millions of Arduino Developers" . https://www.hackster.io/news/edge-impulse-brings-tinyml-to-millions-of-arduino-developers-91cec576dc99 . accessed at1 Februari 2021 , 13.00 .

[24] W. Thorpe, M. Kurver, G. King, and C. Salome, “Acoustic analysis of cough,” ANZIIS 2001 - Proc. 7th Aust. New Zeal. Intell. Inf. Syst. Conf., no. November, pp. 391–394, 2001, doi: 10.1109/ANZIIS.2001.974110.

[25] C. W. Thorpe, L. J. Toop, and K. P. Dawson, “Towards a quantitative description of asthmatic cough sounds,” Eur. Respir. J., vol. 5, no. 6, pp. 685–692, 1992.

[26] Arduino Team. 2020. " Arduino CLI: An introduction ". https://blog.arduino.cc/2020/03/13/arduino-cli-an-introduction/ . accessed at23 Januari 2021 , 20.10 .

[27] National Center for Immunization and Respiratory Diseases, Division of Bacterial Diseases .2017. " Pertussis (Whooping Cough)". https://www.cdc.gov/pertussis/about/signs-symptoms.html . accessed at6 Februari 2021 , 15.30 .

[28] Khandelwal, Renu. 2020 . " A Basic Introduction to TensorFlow Lite " . https://towardsdatascience.com/a-basic-introduction-to-tensorflow-lite-59e480c57292 . accessed at1 Februari 2021 , 13.20 .

[29] Frontline Solvers ."Neural Network Classification" . https://www.solver.com/xlminer/help/neural-networks-classification-intro . accessed at24 Januari 2021 , 20.52 .