科技部補助
大專學生研究計畫研究成果報告

計 畫 名 稱 ：	Contactless Method for Apnea and Snoring Detection Using Deep Learning

執行計畫學生： 鄧立霞

學生計畫編號： NSTC 111-2813-C-346-007-E

研究期間： 111 年 07 月 01 日至 112 年 02 月 28 日止，計 8 個月

指導教授： 黃有評 教授

處理方式： 本計畫可公開查詢

執行單位： 國立臺北科技大學電機工程系

中華民國　112 年 02 月 25 日

ABSTRACT

Sleep Apnea diagnoses are complicated, time-consuming, and expensive. Polysomnography (PSG) test monitoring oxygen level in a patient throughout the night, thus requiring hospital staff and heavy machinery. With these extensive efforts, many would find PSG to be discouraging. In addition, the most common sleep apnea is daylight tiredness and snoring, sleep apnea goes undiagnosed sleep apnea despite health and safety risk it poses. Hence, it created a new demand for a more simplified and less demanding form of PSG. The new framework needs to be capable of recording and identifying events during sleep while maintaining the sleep quality of patients.

With the development of deep learning models, we aspired to make sleep apnea and snoring detection systems more accurate and accessible. We can rely on the accuracy of the complex computation of machine learning paired with widely spread audio recording devices to help fulfill these goals.

The framework utilized a hybrid between CNN (Convolutional Neural Network) and LSTM (Long Short-Term Memory) where the output of CNN is fed into the LSTM model to recognize and predict apnea episode time. The audio is recorded using Arduino Due audio recording and smartphone devices. Using Arduino, we will record sleep events throughout the night and feed the audio to CNN-LSTM Keras model for classification. The three classifications are Snore, Cough, and Noise. The result from machine learning can help in the identification of sleep apnea episodes throughout the night.

Total pages : 25 pages

School : National Taipei University of Technology

Advisor : Professor Huang, Yo-Ping

Keywords : CNN-LSTM, Deep Learning, Arduino, Audio processing, Audio Classification, Snore, Sleep Apnea episode, Audio recording.

TABLE OF CONTENTS

ABSTRACT..i

TABLE OF CONTENTS.. ii

LIST OF FIGURES iii

LIST OF TABLES iv

CHAPTER 1 INTRODUCTION 1

1.1 Background..1

1.2 Objective of Study 2

CHAPTER 2 LITERATURE REVIEW 3

CHAPTER 3 METHOD PURPOSED..4

3.1 CNN-LSTM Model..4

3.1.1 Audio Processing..5

3.1.2 Training and Validation...8

3.2 Hardware 16

3.2.1 Arduino 16

3.2.2 Smartphone Devices 17

CHAPTER 4 RESULT AND DISCUSSION 18

4.1 Results 18

4.2 Discussion 22

CHAPTER 5 CONCLUSIONS 23

REFERENCES..24

LIST OF FIGURES

Figure 1. Obstructive Sleep Apnea airway 1

Figure 2. Audio preprocessing flow 5

Figure 3. An audio from a dataset calculated MFCC form 7

Figure 4. Structure of 2D CNN used in this project 8

Figure 5. Max pooling on left and Average pooling on the right 9

Figure 6. Dropout applied to a Standard Neural Network 10

Figure 7. Model Loss of 2D CNN 11

Figure 8. Model Accuracy of 2D CNN 11

Figure 9. Illustration and Equation for input-output relationship in LSTM model 12

Figure 10. Inner of LSTM block 12

Figure 11. Overall model structure of CCN-LSTM Hybrid 13

Figure 12. CNN-LSTM accuracy and loss 14

Figure. 13. One epoch leads to underfitting of the curve in the graph 14

Figure 14. Hardware connection 16

Figure 15. Waveform Audio File Format 16

Figure 16. Model prediction of new audio files using iPad and iPhone 18

Figure 17. Audio recording using Arduino Due 19

Figure 18. Controlled test on Arduino Due microphone 19

Figure 19. Visualization of Arduino audio classification result 20

Figure 20. Audio classification mapping the classification with respect to time. 20

Figure 21. Result comparison to audio reference 21

Figure 22. Trial for snoring and sleep apnea episodes.. 21

LIST OF TABLES

Table 1. Dataset sources 5

Table 2. CNN-LSTM model detailed summary 13

Table 3. Comparison of all models by validation progress 15

Table 4. Precision, Sensitivity and Misclassification of each audio class 15

CHAPTER 1. INTRODUCTION

1.1 Background

The most prevalent symptoms of sleep apnea are typically loud snoring and excessive daytime fatigue. The primary cause of this condition is the collapse of the soft tissue located at the back of the throat during sleep. The relaxed soft tissues obstruct the airway, leading to snoring and instances of suffocation for the patient [1]. Snoring noises usually occur during inhalation when the body struggles to acquire sufficient oxygen.

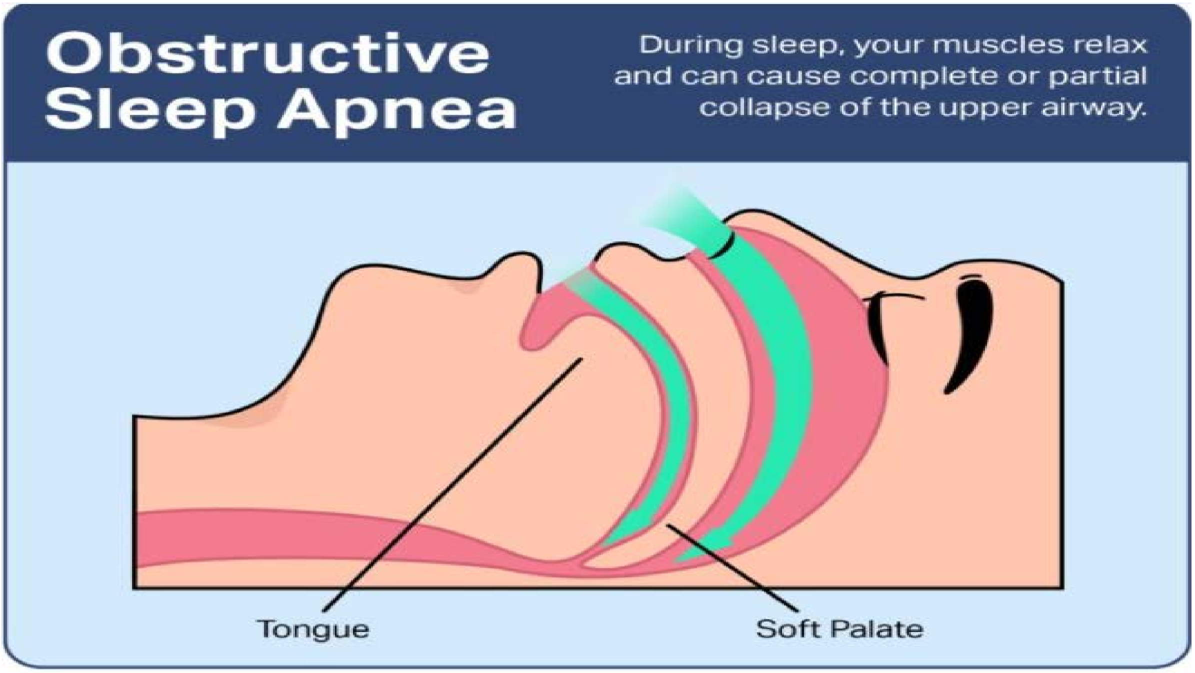

Obstructive Sleep Apnea
During sleep, your muscles relax and can cause complete or partial collapse of the upper airway.
Tongue
Soft Palate

Fig 2. Obstructive Sleep Apnea airway. [1]

According to a paper authored by Worley [2], there is evidence of a potential link between insufficient sleep and a variety of disorders, such as hypertension, obesity, type-2 diabetes, impaired immune functioning, cardiovascular disease, arrhythmias, mood disorders, neurodegeneration, and dementia. The same paper also revealed that up to 80% of sleep disorders may go undetected or undiagnosed, and this number could be even higher in more densely populated countries.

The high incidence of undiagnosed sleep disorders can be attributed to inadequate awareness of sleep apnea risks and the complexity of the diagnostic process. However, with the advancement of technology, we anticipate more cost-effective and user-friendly diagnostic methods for sleep apnea, which could be accessible to a wider population. This development would significantly improve overall public health and individual quality of life.

1.2 Objective of Study

This project is aimed at providing a readily accessible method for the early evaluation of sleep apnea. Our approach involves the development of a machine learning model that can effectively detect various night events, such as snoring, coughing, and noises. Our primary objective is to identify any breathing patterns that may be interrupted due to airway blockage.

To ensure the accuracy and reliability of our model, we will utilize three types of microphones, each equipped with a mono channel. In addition, we will be using an Arduino Due and a mobile phone, both of which will be placed in the bed headboard. All microphones will be positioned within a range of 30cm to 60cm from the subject.

Our goal is to create a system that is both efficient and effective in detecting potential sleep apnea symptoms, thereby encouraging individuals to seek professional medical assistance at the earliest possible stage.

CHAPTER 2. LITERATUR REVIEW

A. Impacts of undiagnosed sleep apnea.

In sleep apnea, breathing disruptions can be caused by various factors, resulting in several types of the condition. Of these types, Obstructive Sleep Apnea (OSA)is the most prevalent, while Central Sleep Apnea (CSA)is less common, estimated to affect less than 1% of individuals. Therefore, in this paper, we will refer to sleep apnea as OSA instead of CSA. [1]

Polysomnography (PSG) is a sleep study that measures various physiological parameters during sleep, including brain waves (EEG), eye movements (EOG), muscle activity (EMG), heart rate (ECG), and breathing patterns [1]. With the development of our technology, sleep studies can now be conducted using portable monitoring or an out-of-center sleep test called a Home Sleep Apnea Test (HSAT), which are diagnostic tests used by sleep professionals and doctors to diagnose obstructive sleep apnea. Typically, these devices are capable of measuring various body functions during sleep, including breathing, movement, heart rate, and the duration of different sleep stages. However, in spite of the increased options and convenience of HSAT, a survey conducted in 2019 by Benjafield [3] The study found that Obstructive Sleep Apnea (OSA) is highly prevalent globally, with an estimated 936 million adults aged 30-69 years affected, corresponding to a global prevalence of 37%.

A study conducted by Wickwire [4], which investigated the economic burden of undiagnosed sleep apnea on the US economy. The researchers note the result of compared healthcare utilization (HCU) and costs in beneficiaries with OSA and matched control patients without the condition for the 12 months leading up to the initiation of treatment. Beneficiaries with OSA incurred greater incremental expenses for hospitalization costs when compared to the control group. Wickwire [4] also highlighted the significance of identifying and treating disorders such as OSA, particularly in older adults who are already at greater risk of developing other severe illnesses. They specifically noted that OSA is linked to a higher likelihood of experiencing hypertension, diabetes, depression, and other related conditions. The high economic and societal burden associated with obstructive sleep apnea is attributed to its multifactorial and social consequences.

After a thorough review by Benjafield [3] and Wickwire [4] on the diagnostic process of obstructive sleep apnea (OSA), it is apparent that there is still room for optimization. With the continuous development of artificial intelligence and advancements in computing power, there is a need for the implementation of an enhanced system that can minimize the percentage of undiagnosed cases and improve the overall diagnostic process.

B. Implementation of Deep Learning for Snoring and Sleep Apnea Detection

A paper by Pevernagie [5] examines the acoustics of snoring and its clinical relevance in the diagnosis and treatment of sleep disorders. They highlight the importance of using acoustic analysis to identify snoring patterns and distinguish between simple snoring and more serious forms of sleep apnea. Consequently, several researchers have utilized the acoustic properties of snoring as a foundation for further investigation, leading to the development of wireless devices for snore detection. We aim to explore the potential limits of digital audio recording for snoring detection, given the growing availability and affordability of tools such as smartphone devices. A study by Kim [7] evaluated the diagnostic value of similar proposals and found that the mean difference between smartphone devices and polysomnography tests was minimal. The accuracy of this proposals is around 88% with 10% of false positives, this combination indicates the high applicable for sleep apnea diagnosis using smartphone devices. Although Kim's study [7] demonstrated that modern smartphone devices are capable of performing audio analysis for sleep apnea diagnosis, many existing techniques have not incorporated deep learning into their models. Incorporating deep-learning into sleep apnea diagnostics has been limited to well-known polysomnography techniques, such as ECG [8] and respiratory sound analysis [7] [12]. Therefore, there is a need to explore new ideas that combine deep learning with more generalized audio recording to diagnose high-risk sleep apnea patients.

Convolutional neural networks are widely used in image analysis due to their ability to effectively extract crucial information from an image and categorize it into labels. The neural network's flexibility allows for endless possibilities for object classification models that can be taught to the network [12]. For example, Mitilineos [20] applied neural networks to audio classification for feature detection in snore detection, while Nakano [6] used audio features such as frequency range, audio gain, and time distribution. Mel-spectrogram can be used to convert different formats of audio into an image format, which is superior in retaining audio information. This unique audio information contained in each Mel-spectrogram image can then be used as input. Given the importance of apnea episode moments highlighted by Nakano [6], we propose using long short-term memory (LSTM) to maintain internal memory and utilize feedback connections to learn temporal information from sequences of inputs. This approach is supported by Zhang's article [8], which suggests that LSTM can help identify future events and enhance the model's precision and overall performance.

CHAPTER 3. METHOD PURPOSED

3.1. CNN-LSTM Model

3.1.1 Audio Preprocessing

The data gathered are all open sources as detailed in Table 1.

Table 1. Dataset sources

Source	Snore	Bedroom Environment Noise	Cough
T. Khan [21]	358 files	244 files	309
UrbanSound8K [GitHub] [11]			379 files
Mitilineos [20]	642 files	756 files	269 files
Orlandic [22]			352 files
Augmentation	1000 files	1000 files	1000 files
Audio files in total			6000 files (Audio Len = 1 sec) Sample: 16k Hertz, 16bits

All files are available as 1-second-long audio tracks in ".wav" file format. Some audio files will undergo preprocessing, ensuring that all audio files have the same length and sample rate. The audio preprocessing will follow the structure shown in (Fig 2) and will be discussed in detail below.

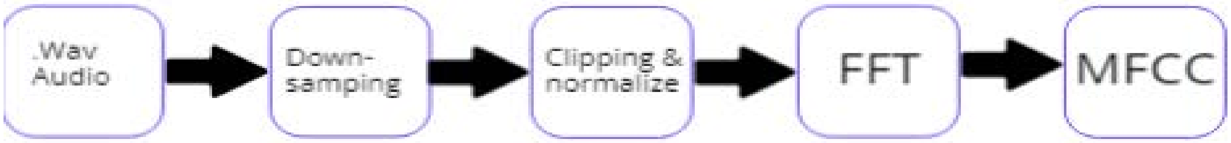


Fig. 2. Audio preprocessing flow.

Resampling (Down sampling)

Down sampling is also associated with compression or bandwidth reduction (bandwidth filter). When the process is performed on a sequence of samples of a signal it produces an approximation of the sequence that would have been obtained by sampling the signal at a lower rate.

The down-sampling can be imagined as a two-step operation. In the first step, the original signal {x[n]}is multiplied with the sampling function {s M[n]} defined by,

$$s_M[n]=\begin{cases}1, & n=0,\pm M,\pm 2M,\ldots\\ 0, & \text{otherwise}\end{cases} \qquad (1)$$

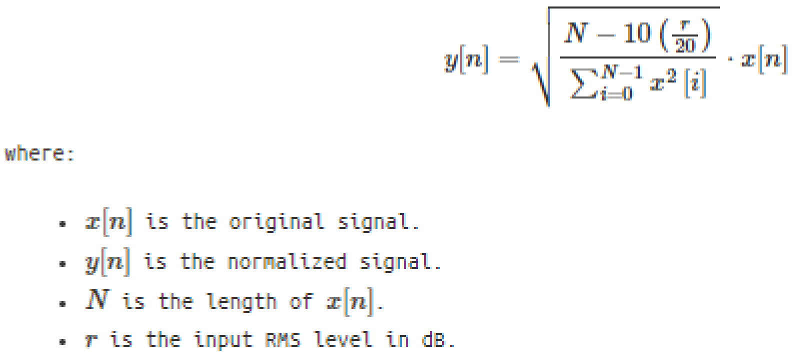
Multiplying the sequence {x[n]} by the sampling function {s M[n]} results in the intermediate signal {y s[m]},

$$y_s[n]=x[n]s_M[n]=\begin{cases}x[n], & n=0,\pm M,\pm 2M,\ldots\\ 0, & \text{otherwise}\end{cases} \qquad (2).$$

For our project we down-sample all audio to 16kHz to reduce the computing power needed to sample and extract the value of the audio.

Audio Clipping and Audio Normalize

Audio that gathers all in different lengths, to avoid heavy computing and weigh down GPU we decided to clip the audio to 1 second/audio. We then use the volume normalization function from one of the Python libraries. The calculations used in the function are Root-Mean-Square (RMS). The approach to RMS normalization can be summarized in the following mathematical formula:



$$y[n]=\sqrt{\frac{N-10\left(\frac{r}{20}\right)}{\sum_{i=0}^{N-1}x^2[i]}}\cdot x[n]$$

where:

- $x[n]$ is the original signal.
- $y[n]$ is the normalized signal.
- N is the length of $x[n]$.
- r is the input RMS level in dB.

(3)

Fast Fourier Transformer (FFT)

Fast Fourier transform also known as Discrete Fourier Transform is a function that gets a signal in the time domain as input, and outputs its decomposition into frequencies. FFT mathematical representation is,

$$X[k] = X(e^{j\omega})\Big|_{\omega=2\pi k/N} = \sum_{n=0}^{L-1} x[n] e^{-j2\pi kn/N}, \quad 0 \le k \le N-1. \tag{4}$$

Fast Fourier transform is the sequence obtained by uniformly sampling the discrete-time Fourier transform X(e jω) on the ω-axis in the range 0 ≤ ω < 2π. While {x[n]} being the sequence of a finite length is zero valued outside the interval 0 ≤ n< L–1. The finite-length sequence {x[n]} is described in the frequency domain by the finite-length sequence {X[k]}.

Mel-frequency Cepstral Coefficients (MFCCs)

Mel-frequency cepstral coefficients represent a short-term power of a sound, this based on linear cosine transform from a log power spectrum on nonlinear Mel-scale of frequency. The Mel-frequency scale is defined as:

$$f_{mel} = 2595 \times log_{10}(1 + \frac{Hz}{700}) \tag{5}$$

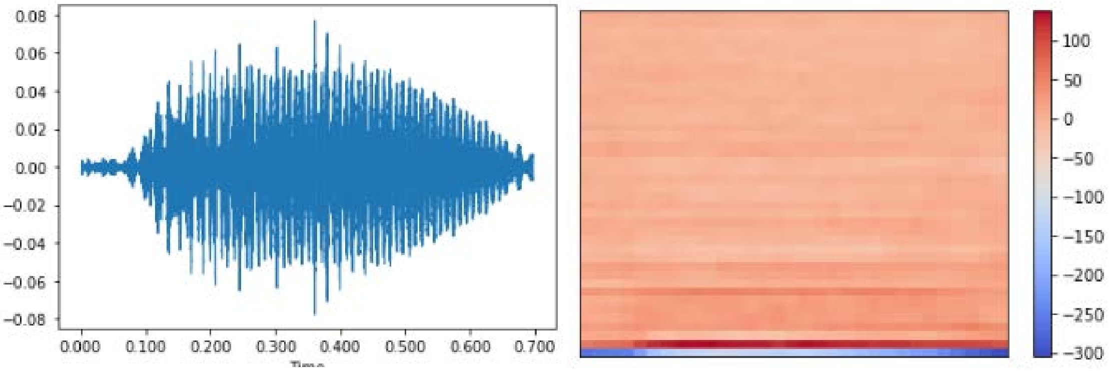


Fig. 3. An audio from a dataset calculated into MFCC form.

As shown in (Fig. 3), the spectrogram form can be used to depict signal frequency which describe how humans would perceive a sound and help our model learn features. In this project, we used filters with 128x128 shape. 128 are filter banks used and time steps per clip designating that data will go through simplification and refine 128 times. The image data then turned into an array as input.

3.1.2 Training and Validation

Convolution Neural Network

In mathematics, our understanding of Convolution is two functions represented as x and h, which will produces the third function of y.

$$y[n] = x[n] * h[n] = \sum_{k=-\infty}^{\infty} x[k] \cdot h[n-k] \quad (6)$$

Above is a mathematical equation that represents the convolution of 2 functions or also known as Convolution Neural Network in Machine Learning.

In this project we will use 2-dimensional convolution which can be represented in mathematical (Eq. 7). The original concept of convolution two functions is present, however instead we add one more variable to each function.

$$y[m,n] = x[m,n] * h[m,n] = \sum_{j=-\infty}^{\infty} \sum_{i=-\infty}^{\infty} x[i,j] \cdot h[m-i, n-j] \quad (7)$$

As can be observed from (Eq. 6), the convolution process in 1-dimensional space implies that each function only has one variable. In our case, this variable is a time series. Conversely, 2-dimensional convolution involves two variables, namely height, and width, as shown in (Eq. 7).

In our approach, we represent audio as an MFCC image, which is then converted from a three-channel (red, green, blue) RGB image to a grayscale image (black and white). The pixel values of the grayscale image are stored in an array that is convoluted to produce values that can be classified. The use of a grayscale image, represented as a 2D array of numbers proportional to pixel brightness, reduces the computing power required for each layer. The computing steps in our model are illustrated in (Fig. 4) and explained in greater detail below.

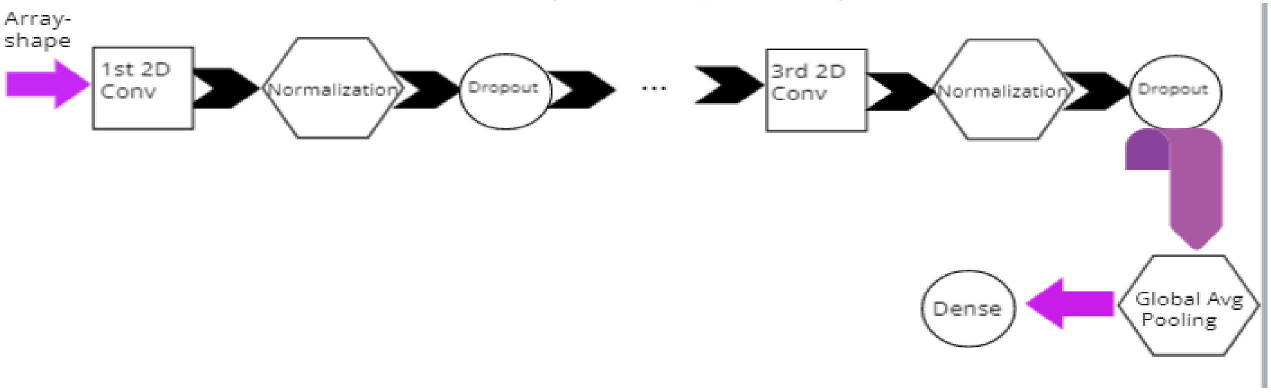


Fig 4. Structure of 2D CNN used in this project.

- Convolution Layer: This layer we compute most of the features of the inputs, in mathematics it is seen as two sources of information are intertwined in orderly procedure. For 2D CNN the input has the shape of [BatchSize, Height, Width, channel] with kernel

3x3 as commonly used kernel and 32 filters before increasing the filters to 64 filters for boosting learning rate.

- Normalization: We normalize the output of a previous activation to optimize training time. This calculation is done with subtracting the batch mean with previous activation and dividing by the batch standard deviation.

- Max-Pooling: This is a common technique to down-sample our data reducing dimensionality. Feature maps that are used in convolution are sensitive to positions of each feature thus using down-sampling before another convolution layer can help with the sensitivity. We only use this once in this model after the first convolution layer as shown in (Fig. 5).

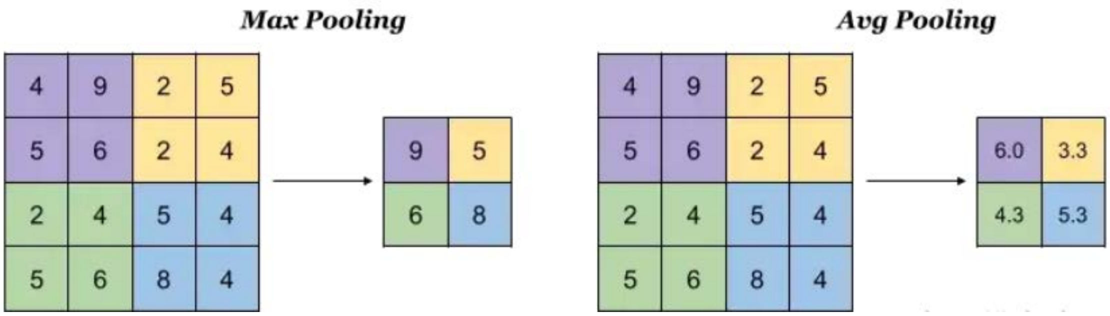
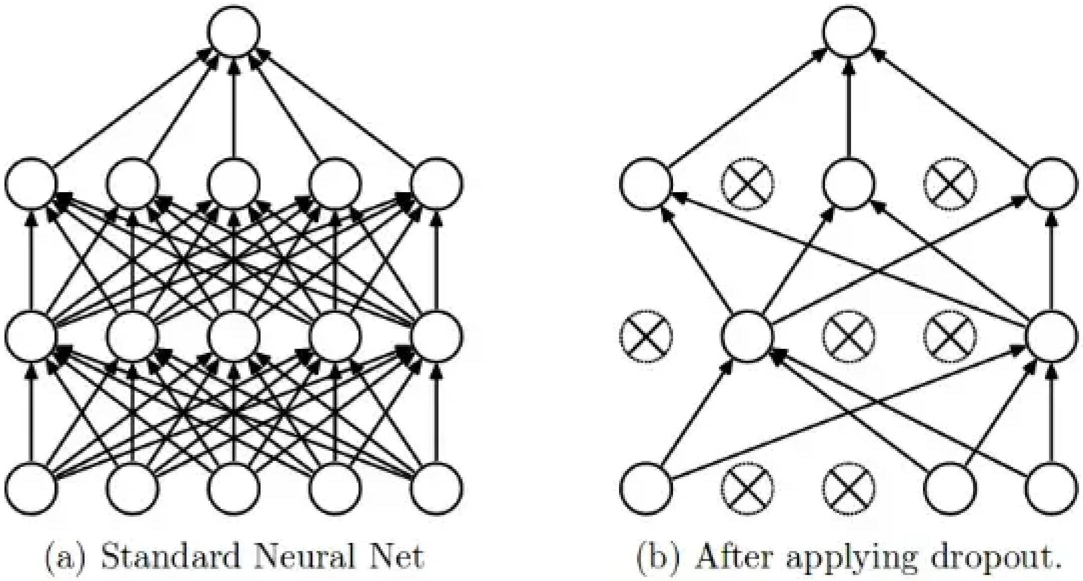


Fig. 5. Max pooling on left and average pooling on the right. [15]

- Dropout: This layer is used to drop out entire feature maps by a given rate, preventing activations from becoming strongly correlated. Using dropout regularization usually in small value can reduce overfitting and improve the generalization of deep neural networks. Especially when we deal with multiple layers of neural network as shown in (Fig. 6).


(a) Standard Neural Net

(b) After applying dropout.

Fig. 6. Dropout applied to a Standard Neural Network. [15]

- Global Average Pooling: Global Average Pooling works the same as Max-Pooling but instead of just referencing the previous layer, Global Average Pooling calculates the average output of each feature map in the previous layer.

- Dense: As its name suggests this layer compresses all neural network layers to become our output that will provide classification outputs. Softmax is a generalization of the Logistic Function, and it makes sure that our prediction adds up to 1.

  $$f(x) = \frac{L}{1 + e^{-k(x-x_0)}}, \tag{8}$$

  Where we denote: L = the supremum of the values of the function, k = the logistic growth rate or steepness of the curve and x_0, the x value of the Sigmoid midpoint.

- Loss: Loss is a value that represents the summation of errors in our model. It measures how well the model performs. If the errors are high, the loss will be high, which means that the model does not do a good job. Meanwhile, accuracy is measuring how well our model predicts by comparing the model predictions with the true values in terms of percentage (Fig. 7).

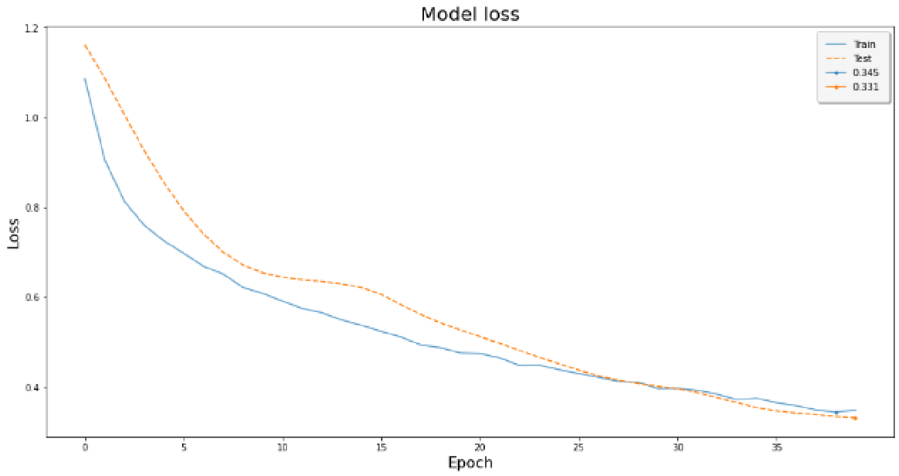


Fig. 7. Model Loss of 2D CNN.

- Accuracy: Model accuracy is a measure of how well a model predicts classifications relative to the number of predictions. This provides an overall perspective on how effective the model is on a given dataset. Using the plot and confusion matrix this will help us evaluate the model as shown in (Fig. 7) and (Fig. 8).

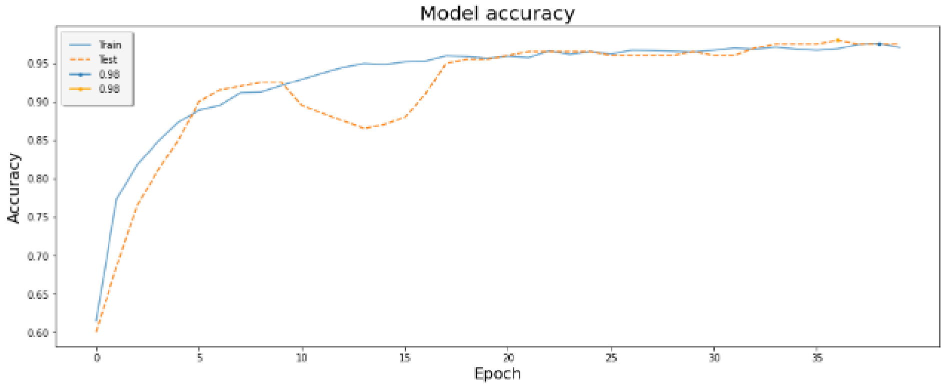
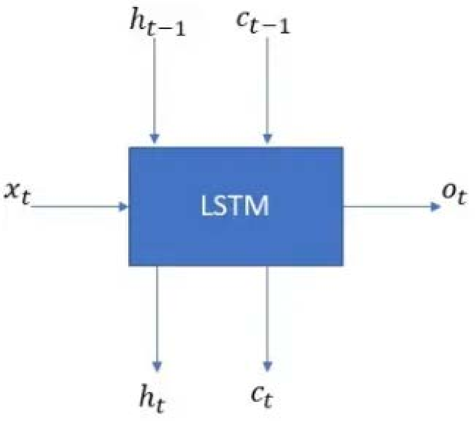


Fig 8. Model accuracy of 2D CNN.

Although the CNN model exhibits high accuracy, we aim to improve its overall performance by combining it with an LSTM.

LSTM (Long Short-Term Memory)

As previously mentioned, CNNs are effective neural networks for image classification, as they can represent multiple dimensions through internal computing without losing information. On the other hand, LSTMs are advanced recurrent neural networks that can compute a mapping from an input sequence, as illustrated in (Fig. 9). As a result, LSTMs are well-suited for processing long-term input sequences. By combining these two models, we aim to enhance the overall performance of our system.



$$f_t = \sigma_g\ (W_f \times\ x_t + U_f \times h_{t-1} + b_f)$$

$$i_t = \sigma_g\ (W_i \times\ x_t + U_i \times h_{t-1} + b_i)$$

$$o_t = \sigma_g\ (W_o \times\ x_t + U_o \times h_{t-1} + b_o)$$

$$c'_t = \sigma_c\ (W_c \times\ x_t + U_c \times h_{t-1} + b_c)$$

$$c_t = f_t \cdot c_{t-1}\ + i_t \cdot c'_t$$

$$h_t\ = o_t \cdot \sigma_c(c_t)$$

σ_g : *sigmoid*
σ_c : tanh
. : element wise multiplication

f_t *is the forget gate*
i_t *is the input gate*
o_t *is the output gate*
c_t *is the cell state*
h_t *is the hidden state*

Fig. 9. Illustration and equation for input-output relationship in LSTM model. [14]

As evident from both (Fig. 9) and (Fig. 10), the input-output connections reveal that the inputs are influenced by *Hidden state* **h$_t$** and *Cell state* **c$_t$** that are functions from previous gates. *Hidden state* contains an output weight of the activation layer while *Cell state* is a memory cell that stores previous output weight making a recurrent, this cell is adjusted by *Forget gate*. This cell is adjusted by the Forget gate. The internal workings of the LSTM layer are depicted in detail in (Fig 10).

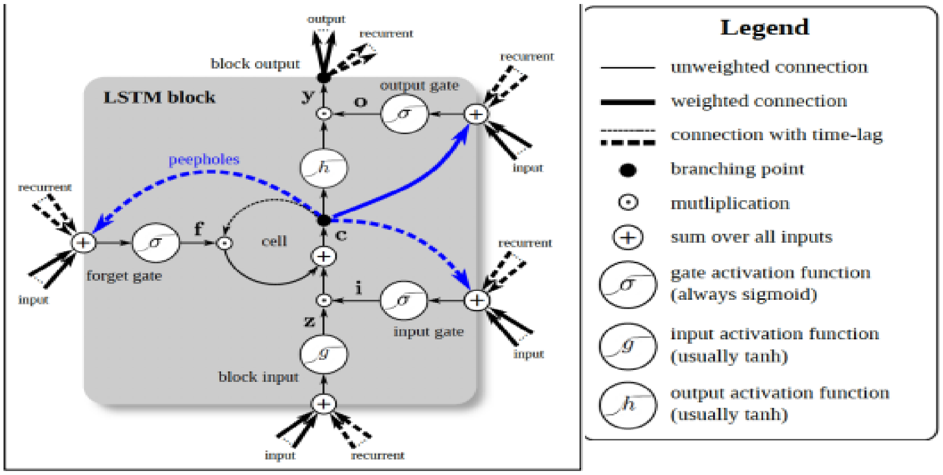


Fig. 10. Inner of LSTM block. [14]

CNN-LSTM

As previously discussed, the main variables in 2-dimensional CNN are width and length, while LSTM is based on sequences of input. This difference proves that each neural network has its own disadvantages, thus combining the two neural networks can address the issue. By using a wrapper method and adding time sequences into the original CNN layer, the results can be seen in the summary presented in Table 2.

Table 2. CNN-LSTM model detailed summary.

Layer	Output Shape	Parameters
Time distributed – CNN	(None, 1, 38, 443, 64)	640
Time distributed – Normalization	(None, 1, 38, 443, 64)	256
Time distributed – Pooling	(None, 1, 19, 221, 64)	0
Time distributed - 2nd CNN	(None, 1, 17, 219, 32)	18464
Time distributed - 2nd Pooling	(None, 1, 17, 219, 32)	128
Time distributed - 2nd Pooling	(None, 1, 8, 109, 32)	0
Time distributed – Flatten	(None, 1, 27904)	0
LSTM layer	(None, 60)	6711600
1st Dense layer	(None, 3)	183
2nd Dense layer	(None, 3)	12

In the new model the input shape will be [BatchSize, Time step, Height, Width, Channel] the time step is set to 1, this is to avoid any effect on the CNN layer. We can represent our model structure in (Fig. 11).

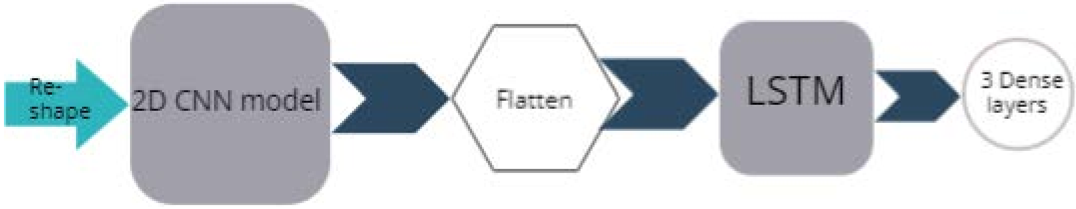


Fig. 11. Overall model structure of CCN-LSTM Hybrid.

After we trained and tested the model’s performance we evaluate their accuracy and losses in both training and testing period represented by (Fig. 12).

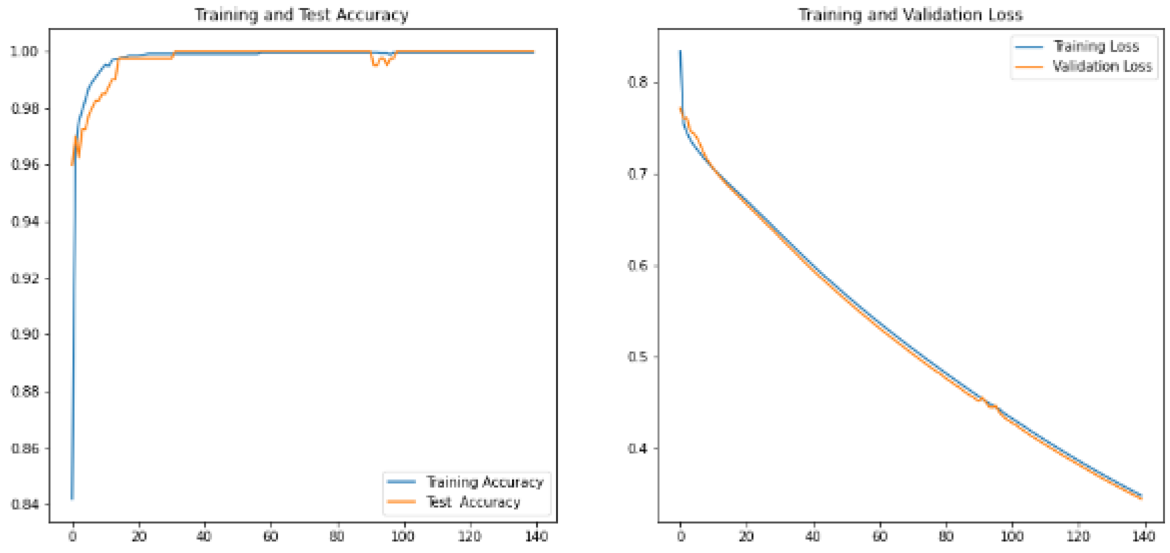


Fig 12. CNN-LSTM accuracy and loss

During the training of our hybrid model, we observed that adding 3 layers of convolution resulted in overfitting, while adding 2 layers did not. Overfitting occurs when a model has more parameters than can be justified by the data, leading to the extraction of residual variation (i.e., noise) as if it represented the underlying model structure [14].

In the context of model training, it is crucial to understand the concept of "epochs." An epoch is completed when the entire dataset is passed forward and backward through the neural network once. The number of epochs required for training is closely related to the batch and dataset size, as shown in (Fig. 13).

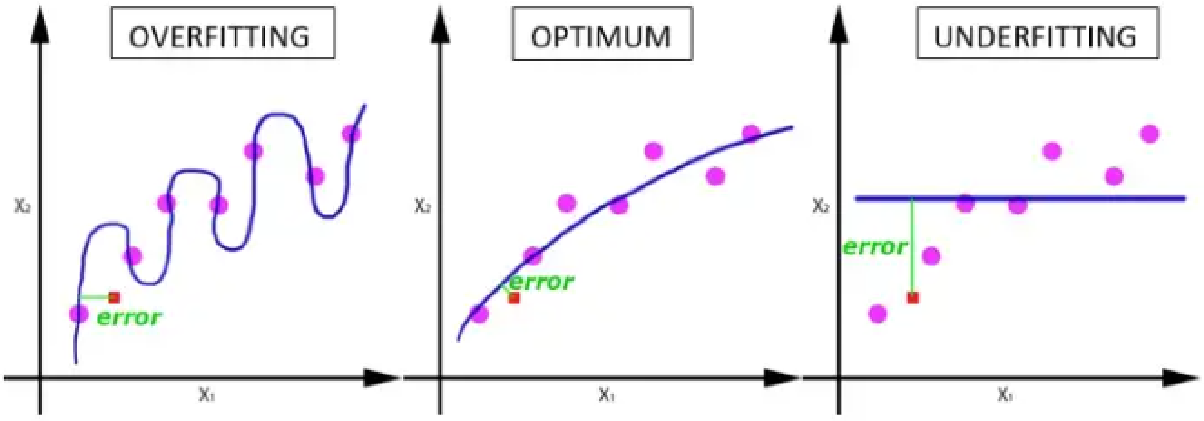


Fig. 13. One epoch leads to underfitting of the curve in the graph. [17] [15]

Thus, in the end we selected 2-layers of convolution for our newer model. For reference we have provided in Table. 3. for further performance comparison.

Table 3. Comparison by layers and Neural Network types.

Model	CNN		CNN-LSTM	
2 Layers	Loss = 28.19%	Accuracy= 98.33 % (Overfitting)	Loss = 35.16%	Accuracy = 99.58%
3 Layers	Loss = 27.58 %	Accuracy = 97.68%	Loss = 24.36%	Accuracy = 99.88% (Overfitting)

Using the bias matrix, we can calculate that the True Positive is high while the False Negative is low. With all the values calculated, we can arrange them into their respective classes, as shown below.

Table 4. Evaluation of each audio class on CNN-LSTM.

	Noise	Cough	Snore
Precision	99.4%	97.6%	99.5%
Sensitivity	94.6%	98.9%	97.7%
Accuracy	98.16%	99%	99%
Misclassification	1.8%	1%	1.2%

3.2 HARDWARE

Arduino

Hardware material used is Arduino due with Atmel SAM3X8E microprocessor, pre-amp microphone (MAX9814) and MicroSD card adapter as shown in (Fig. 14). The microphone in this circuit are set to float value to avoid any sudden spike in overall file.

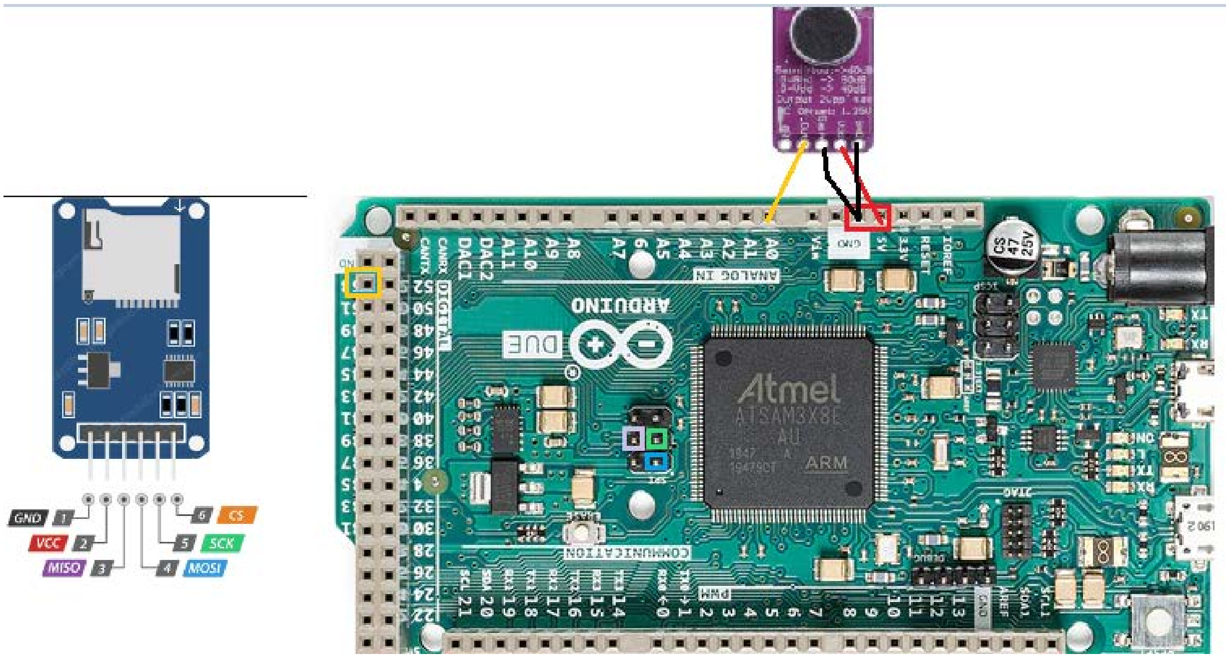
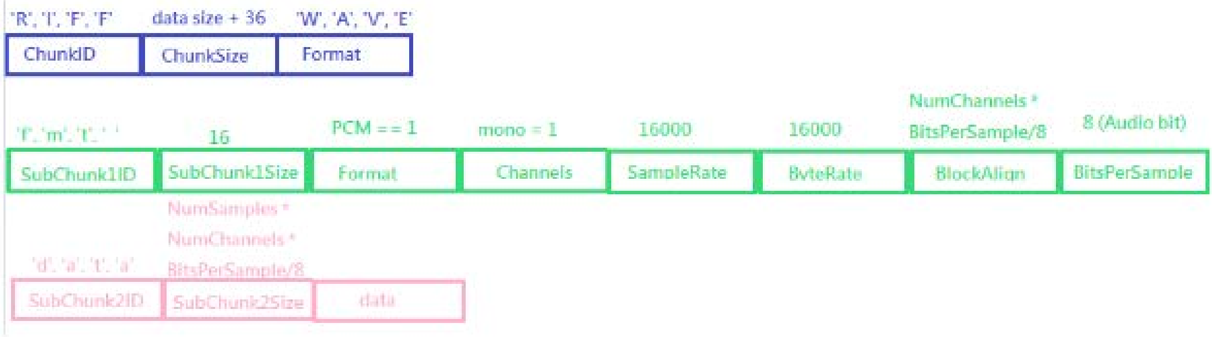


Fig. 14. Hardware connection.

To prevent any loss of information, we utilize the .wav format as an audio file, which has the advantage of preserving the original elements in the file. The audio signal is captured and converted from analog to digital values using the Analog-Digital-Converter of the Arduino Due. The collected and sampled values are packaged and arranged in the .wav file format, as depicted in (Fig. 15).



'R', 'I', 'F', 'F'	data size + 36	'W', 'A', 'V', 'E'
ChunkID	ChunkSize	Format

'f', 'm', 't', ' '	16	PCM == 1	mono = 1	16000	16000	NumChannels * BitsPerSample/8	8 (Audio bit)
SubChunk1ID	SubChunk1Size	Format	Channels	SampleRate	ByteRate	BlockAlign	BitsPerSample

'd', 'a', 't', 'a'	NumSamples * NumChannels * BitsPerSample/8	
SubChunk2ID	SubChunk2Size	data

Fig. 15. Waveform audio file format.

As shown in (Fig. 15), it is evident that the audio format of the .wav file requires a header file with a size of 44 bytes. Therefore, we have implemented the buffer-flush method when saving the file into MicroSD to avoid information losses. The MicroSD adapter is connect to the SPI (Serial Peripheral Interface), which creates new memory slots. The ADC and Timer counters connected to DMA (Direct Memory Access), which stores the data temporarily before flushing it into the MicroSD to be saved. Once the recording is complete, we input the corresponding keyword to store all the leftover bits into MicroSD before disconnecting. The Arduino is equipped with keyboard input that depends on the USB cable connected to the Arduino. Using power supply from the laptop, we can start and stop the recordings at our preferred time, which provides us with flexibility and a secure power source.

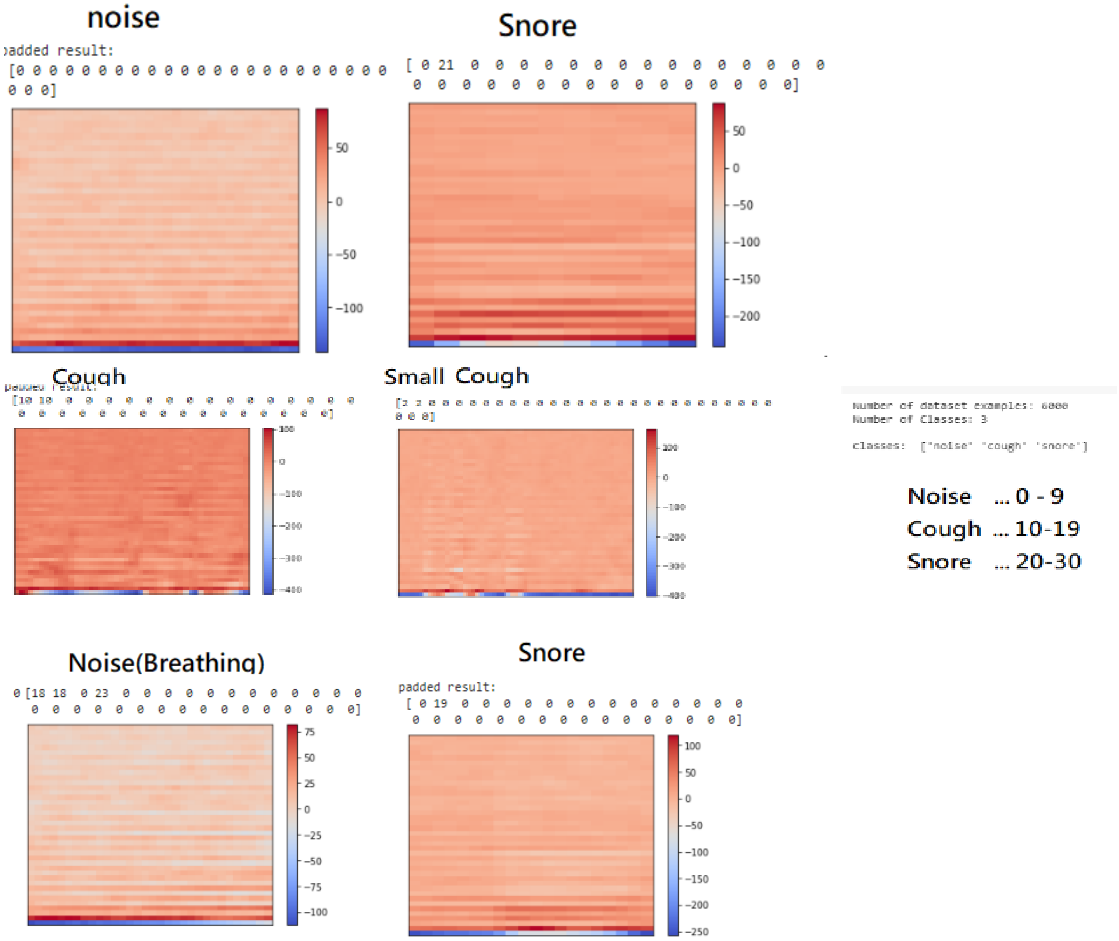
Smartphone Devices

Mobile phones used in this test are the iPad Pro 11 and iPhone SE 2020. Similar to other mobile smartphones the audio .wav has standard 16 bits / 44.1kHz. The audio recorded using devices then uploaded into Google Drive storage. The audio recording of iPhone is .m4a file type that are not compatible with our input, thus converting are additional step needed before pre-processing step.

CHAPTER 4. RESULTS AND DISCUSSION

4.1 Results

After discussing the preprocessing, processing, evaluation, and hardware, the predictions of the model are presented as a probability value. The values returned from the model are in the form of a 1D array of prediction values. These values are fitted into a range of 0-30, as shown in (Fig. 16), where values 0-9 are classified as noise, 10-19 as cough, and 20-30 as snore. As previously mentioned, the prediction values heavily depend on the previous input sequences. The test results in (Fig. 16) were recorded using smartphone devices and were preprocessed by cutting the audio into 1-second audio files before being fed into the deep learning model. It is worth noting that the manually assigned labels to each audio file in the figure are valid. The classification performance can be further studied by examining the test results in (Fig. 16) and the trial results in (Fig. 20).



```
padded result:
[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0]
```

```
[ 0 21  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
```

```
padded result:
[10 10  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
```

```
[2 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0]
```

```
Number of dataset examples: 6000
Number of Classes: 3

classes:  ['noise' 'cough' 'snore']
```

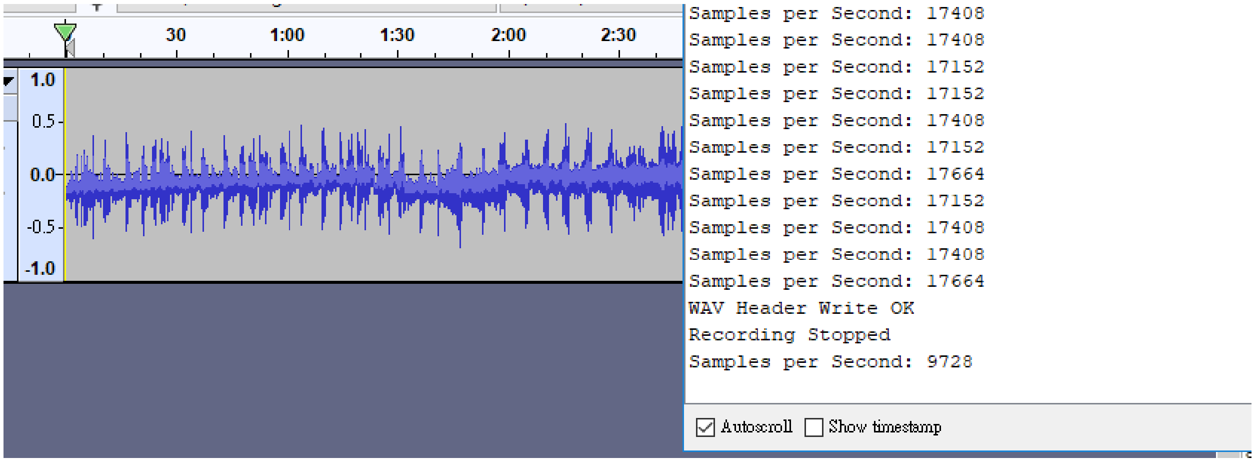
Noise ... 0 - 9
Cough ... 10-19
Snore ... 20-30

```
0 [18 18  0 23  0  0  0  0  0  0  0  0  0  0  0  0
  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
```

```
padded result:
 [ 0 19  0  0  0  0  0  0  0  0  0  0  0  0  0  0
  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0]
```

Fig. 16. Model prediction of new audio files using iPad and iPhone.

The second test is done using an Arduino microphone, with the same procedure we put the microphone approximately 30 cm away from the audio source. An article published by Xie [19] providing strong evidence on distant and audio quality correlations. In the paper [19], it concluded the placement of microphone at proximately 30cm-60cm have robust performance. We run the trial as shown (Fig. 17), as mention on Chapter 3, the recording is operating on float value since converting sample value into integer can cause information loss.



```
Samples per Second: 17408
Samples per Second: 17408
Samples per Second: 17152
Samples per Second: 17152
Samples per Second: 17408
Samples per Second: 17152
Samples per Second: 17664
Samples per Second: 17152
Samples per Second: 17408
Samples per Second: 17408
Samples per Second: 17664
WAV Header Write OK
Recording Stopped
Samples per Second: 9728
```

Fig. 17. Audio recording using Arduino Due.

We observed from both figures the static noises are prominent in overall audio despite being cohesive to a human brain. We run test using controlled audio as shown in (Fig. 18). The audio used in Arduino Due are augmentation from audio that are not used in the training. We synchronize the time stamp to our manually classified, as can be observed in (Fig. 18) the model mostly classified the audio as noises.

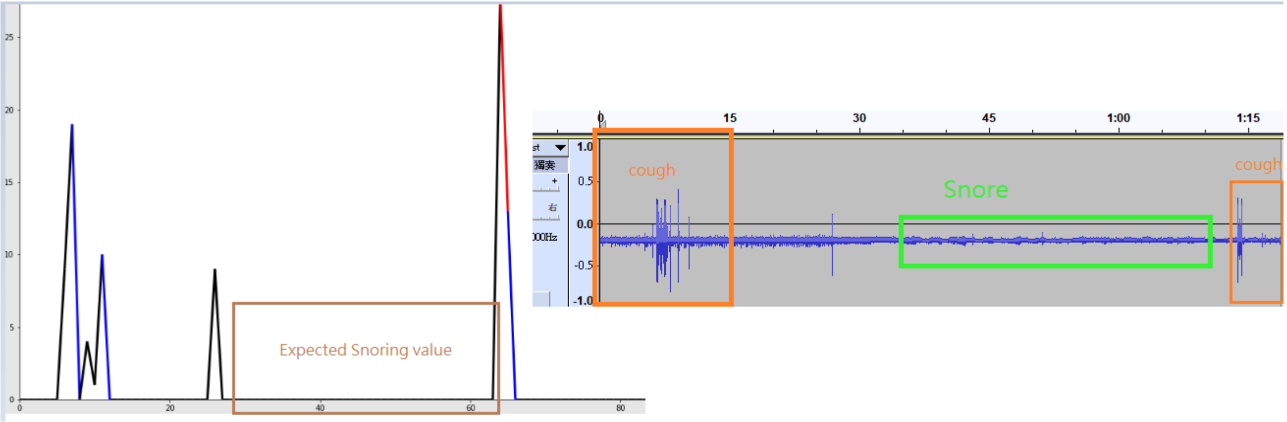


Fig. 18. Controlled test on Arduino Due microphone.

We conducted an additional test using the Arduino, with the audio source positioned at a closer distance and increased volume, we can observe the static noises still prominent through out the audio file. However, the system is able to classify the audio without the need for additional audio pre-processing beyond audio clipping. The result can be observed in (Fig. 19)

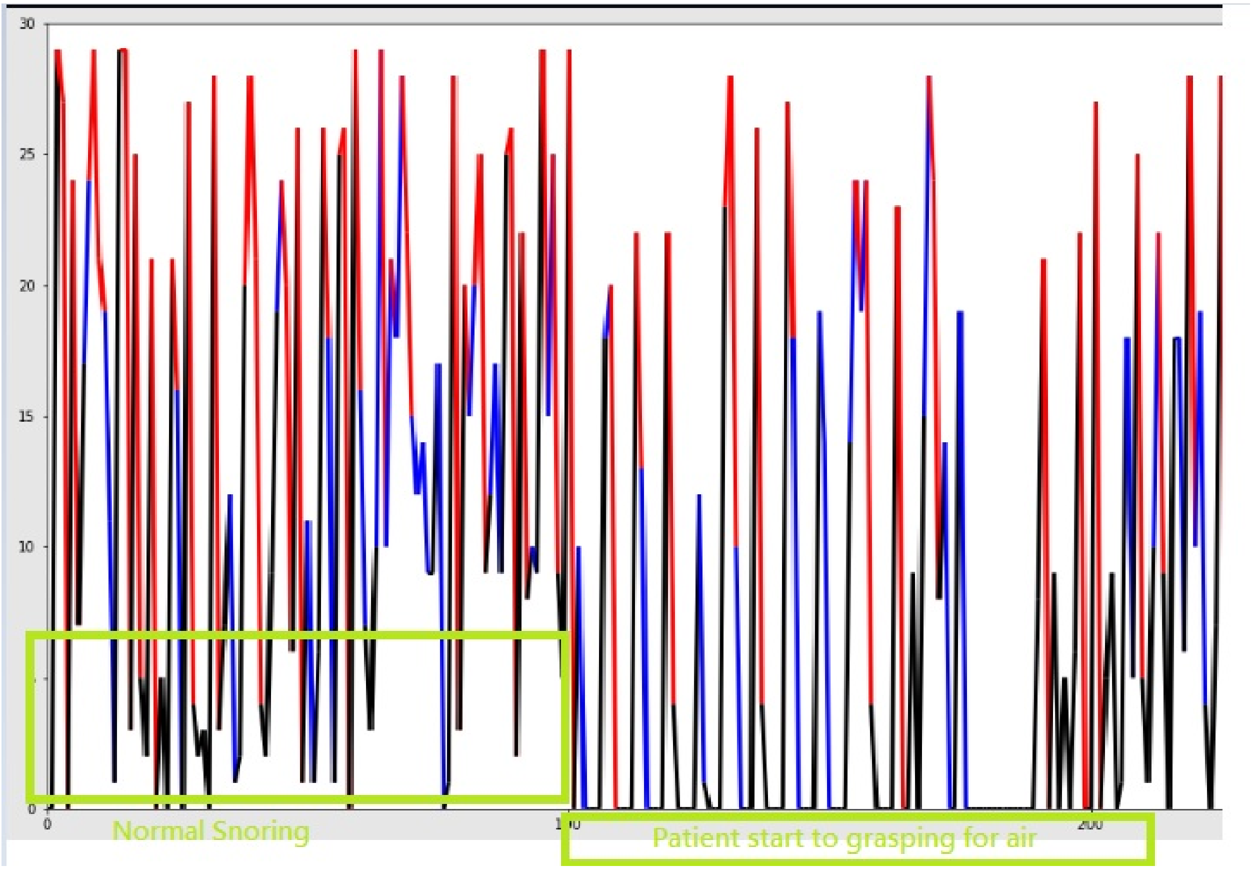


Fig. 19. Visualization of Arduino audio classification result

In (Fig. 19) we can observe a distinct difference between normal snoring and sleep apnea snores. The gap between each exhale, which is often identifies as snoring sound is more significant.

After the initial tests, we run a controlled trial using file gathered from an online open source video [18], the video was manually labeled to identify sleep apnea episode from a smartphone recording. We extract the audio from the video followed by preprocessing before the audio is processed through the model. The result of the audio can be observed in (Fig. 20).

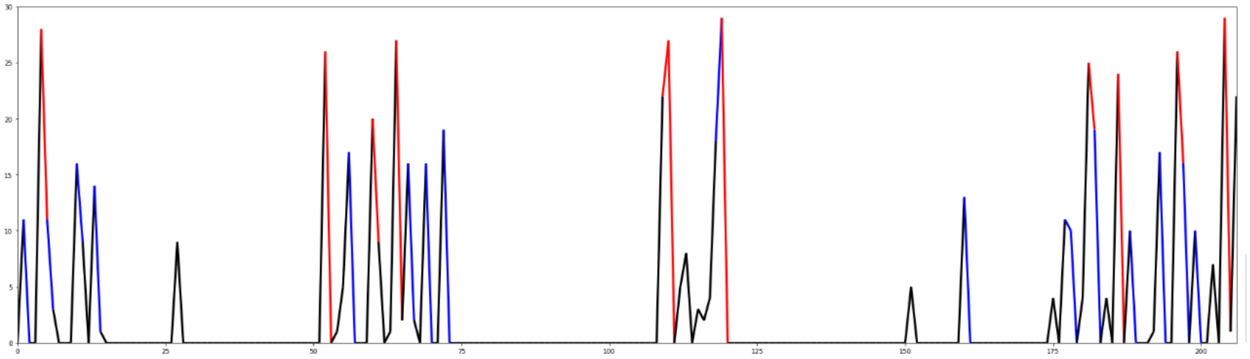


Fig. 20. Audio classification mapping the classification with respect to time.

We colored snore, cough and noise with red, blue and black in that respect, as the result of the trial in (Fig. 21) we can see the events that might be significant. The area with high occurrence of noise represented by black color (noise) after snoring event (red) is a high indication of the patient experiencing disruption in breathing pattern.

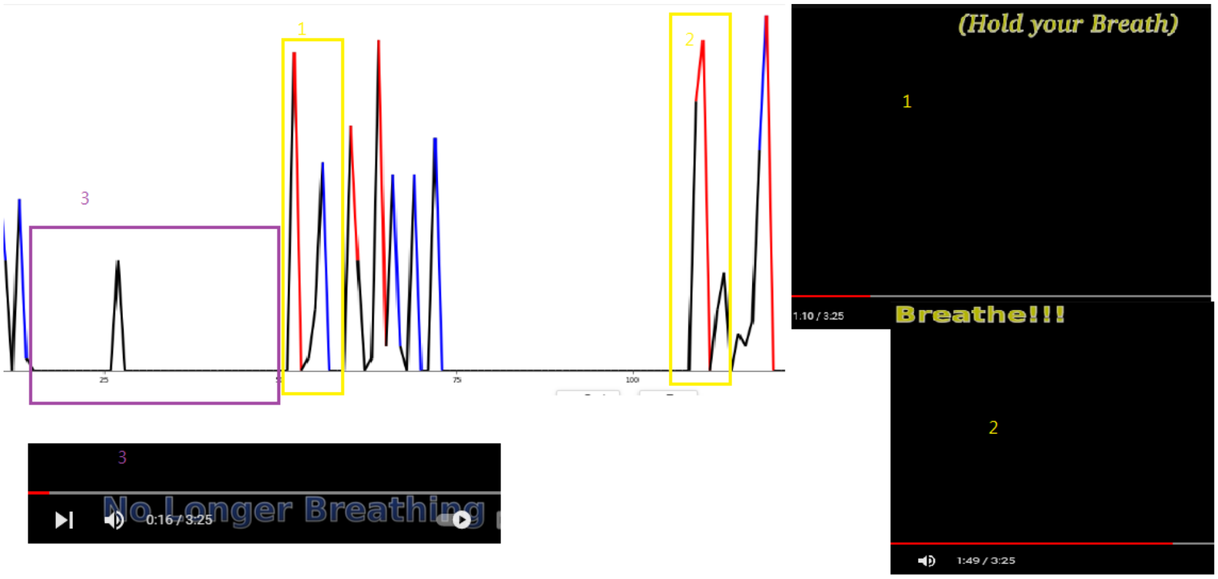


Fig. 21. Result comparison to audio reference

The results are manually evaluated by matching the timestamp of original video to the model result and the reference label. As observed in (Fig. 21) the evaluation of the sleep apnea from this model are robust in accuracy of the sleep apnea happened.

Second trial, the audio gathered from open source video platform [23]. The process is identical to the previous trial, the result of the audio classification is shown in (Fig. 22).

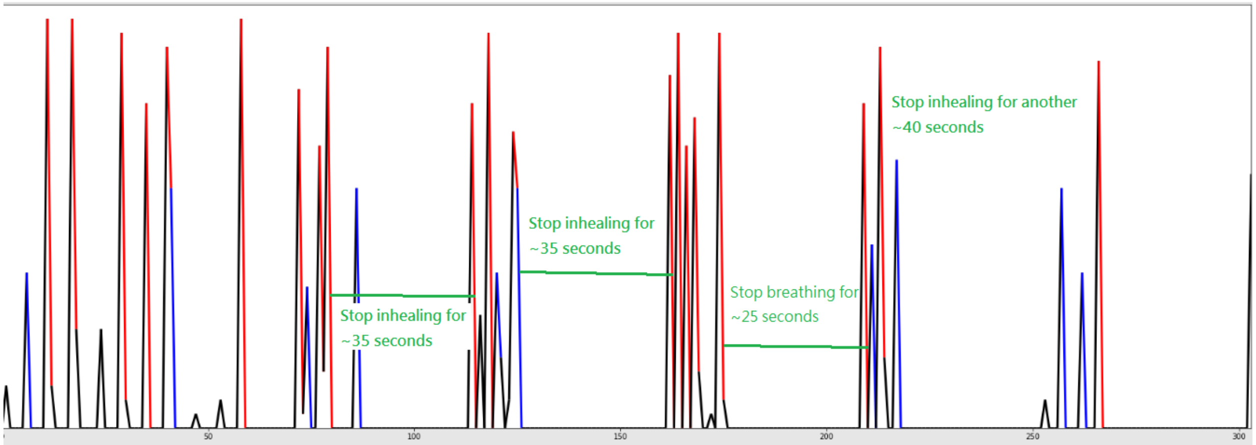


Fig. 22. Trial for snoring and sleep apnea episodes.

In this study, we observed a severe case of sleep apnea in the audio recordings. The patterns that emerged were distinct between snoring sounds and episodes of apnea. Our analysis revealed that sleep apnea can be identified by changes and gaps in color, with a shift from red to black indicating an exhalation followed by a pause before inhalation. The result provides a clear picture of when sleep apnea occurs during the night, which can be valuable in the diagnosis and improve individualized treatment of sleep disorders. The use of digital audio recordings has proven to be a promising tool in the detection of sleep apnea, particularly given the increasing accessibility of devices such as smartphones. The findings of our study contribute to a growing body of literature on the use of digital audio recordings for sleep monitoring and suggest that such technology has the potential for the identification and diagnosis of sleep disorders.

4.2 Discussion

We developed a model that combine the CNN and LSTM to enhance their respective advantages. CNN are often used for image classification due to the focus of its neural network variable are width and height while LSTM dependent on input sequence and memory activation. Therefore, combining the neural network can help us favoring one advantages over other. As the first model using pure CNN only display 97.68% accuracy, as we implement LSTM into our CNN model, it enhances the accuracy to 99.58% shown (in Table. 4). The classification of the audio label is in range model as shown (Fig. 16), this practice aims to visualize events during sleep, assist in identify and diagnose sleep apnea more effectively while still can allow for human correction.

With the test and trial result in previous section, we utilize Arduino Due to exert the computing power and the practicality of 8bits audio. 8bits audio are sufficient to be cohesive to human despite the muffled sound and static noise as can be observed (in Fig. 18). However, despite the muffled sound and static noises, Arduino might still be useful in severe cases of sleep apnea.

Smartphone devices and widely distributed microphone works in 16bits or up to 32bits thus have better quality and accessibility. Smartphone recording also has fewer issues with noise, although the placement of the device is still relevant. To ensure optimal recording quality, the smartphone should be positioned within 30-60cm from the bed, preferably on the headboard. Thus, smartphone recording is more suitable for moderate to severe cases of sleep apnea.

CHAPTER 5. CONCLUSION

We propose a deep learning method for detecting snoring and sleep apnea episodes. The neural network model was designed, trained and tested using open source audio. We used less conventional audio sample and lower bit depths to challenge the limits of our model. Our result showed that more economical audio recording can be used. The CNN-LSTM model using smartphones can make a functional system and despite static noise of the Arduino Due audio recording it can still perform for severe sleep apnea cases. According to our result in the previous chapter, the hybrid CNN-LSTM model exhibits positive results when paired with a more sufficient audio recording. In addition, we utilized advanced visualization techniques to facilitate more accurate and insightful analysis. Furthermore, our visualization technique can be used as an instrument in assisting medical professionals to refine the most effective approach to personalized medical treatment for patients with sleep apnea.

As our experiment result suggest, Arduino’s low quality (16kHz, 8bits) audio can still be used for sleep apnea identification. Arduino recording accuracy are heavily dependent on the level of ambient noise in the environment and only limited to severe case of snoring. In addition, the placement of the microphone also has a significant effect on the experiment. The audio samples used for the machine learning training mostly are either unrecognizable or too low in audio value. The placement of the recording devices in this experiment is all within 30-60 cm above the head of the patient to guarantee the comfort of the patient and reduce movement noises.

Future work could involve the development of a more user-friendly application that incorporates the proposed deep learning method. Additionally, the use of more advanced audio processing techniques, such as noise reduction and filtering, could further improve the performance of the model. Further research could explore the use of more powerful computing resources to improve the running time of the model.

Acknowledgment: The author expresses sincere gratitude to Professor Huang Yo-Ping for his invaluable guidance and support throughout the research process. The author also wishes to extend appreciation to all open-source providers for their generous contribution of data, which greatly aided in acquisition of crucial insights.

REFERENCES

1. Rob Newsom , “Obstructive Sleep Apnea,” , *Sleep Foundation*, Accessed: June 2022, Online: https://www.sleepfoundation.org/sleep-apnea/obstructive-sleep-apnea

2. S.L. Worley, “The extraordinary importance of sleep,” *Health and Pub. Safety Sleep Research,* pp.758-763, Dec. 2018.

3. A.V. Benjafield, N.T. Ayas, P.R. Eastwood, R. Heinzer, M.S.M. Ip, M.J. Morrell, C.M. Nunez, S.R. Patel, T. Penzel, J.L Pépin, P.E. Peppard, S. Sinha, S. Tufik, K. Valentine, and A. Malhotra. “Estimation of the global prevalence and burden of obstructive sleep apnoea: a literature-based analysis.”, Lancet Respir Med, Vol.7, pp.687-698, Aug 2019.

4. E.M. Wickwire, , S.E. Tom, A. Vadlamani, M. Diaz-Abad, L.M. Cooper, A.M. Johnson, S.M. Scharf, J.S. Albrecht, “Older adult US Medicare beneficiaries with untreated obstructive sleep apnea are heavier users of health care than matched control patients.”, *Journal of Clin. Sleep Med.*, Vol.16, pp.81-89, Jan 2020

5. D. Pevernagie , R.M Aarts , M De-Meyer. ,“The acoustics of snoring.”, *Sleep Med Rev.* Vol.14, pp.131-144, Aug 2009.

6. H. Nakano, K. Hirayama, Y. Sadamitsu, A. Toshimitsu, H. Fujita, S. Shin, and T. Tanigawa, “Monitoring sound to quantify snoring and sleep apnea severity using a smartphone: proof of concept,” *Clin Sleep Med*, vol. 10, pp.73-78, Jan. 2014.

7. Kim DH, Kim SW, Hwang SH. “Diagnostic value of smartphones in obstructive sleep apnea syndrome: A systematic review and meta-analysis,” *PLos ONE*, vol. 17, pp.1-12, May 2022.

8. J. Zhang, Z. Tang, J. Gao, L. Lin, Z. Liu, H. Wu, F. Liu, and R. Yao, “Automatic detection of obstructive sleep apnea events using a deep CNN-LSTM model,” *Comp. Intel. and Neurosci.*, vol. 2021, DOI: 10.1155/2021/5594733, Mar. 2021.

9. C. Park and D. Lee, “Classification of respiratory states using spectrogram with convolutional neural network,” *Appl. Sci*, vol. 12, pp.1-17, Dec. 2021.

10. A. Malek and H.M. Malek. “Pydiogment: A Python package for audio augmentation.,” Accessed: Nov 2022, Online: https://github.com/SuperKogito/pydiogment/blob/master/paper/paper.pdf.

11. E.G. Rajo. “Urban sounds classification with Convolutional Neural Networks,” Accessed Aug. 2022, Online: https://github.com/GorillaBus/urban-audio-classifier/

12. Y. Castillo-Escario, L. Werthen-Brabants, W. Groenendaal, D. Deschrijver, and R. Jane, "Convolutional neural networks for apnea detection from smartphone audio signals: effect of window size," in *Proc. of Annu. Int. Conf. IEEE Eng Med Biol Soc.*, Glasgow, Scotland, United Kingdom, Jul. 2022.

13. G. Namyal, "What is 2-Dimensional convolution?" Accessed: Aug 2022, Online*: https://medium.com/theleanprogrammer/2-dimensional-convolution-189abb174d92*,

14. N. Srivastava, G. Hinton, A.Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *Mac. Learning Research*, vol. 15, pp.1929-1958, Jun. 2014.

15. M. Rastogi, "Tutorial on LSTMs: a computational perspective," Accessed: Dec 2022, Online: https://towardsdatascience.com/tutorial-on-lstm-a-computational-perspective-

16. [Source: https://indoml.com/2018/03/07/student-notes-convolutional-neural-networks-cnn-introduction]

17. S. Sharma, "Epoch vs batch, size vs iterations," Accessed: Nov 2022, Online: https://towardsdatascience.com/epoch-vs-iterations-vs-batch-size-4dfb9c7ce9c9

18. "What does sleep apnea sound like?" (Nov. 5, 2012) YouTube video, added by Fluffyshotme, Accessed: Jan 2023, Online: https://www.youtube.com/watch?v=9bFTcmREtqQ

19. Xie J, Aubert X, Long X, van Dijk J, Arsenali B, Fonseca P, Overeem S., "Audio-based snore detection using deep neural networks," *Comput Methods Programs Biomed*, vol.200, pp.10-16, Mar 2021.

20. S.A. Mitilineos, N.-A. Tatlas, G. Korompili, L. Kokkalas, and S.M. Potirakis, "A real-time snore detector using neural networks and selected sound features," *Eng. Proceedings*, vol. 68, pp.8-15, Oct. 2021.

21. T. Khan, "Snoring," Accessed: Oct 2022, Online : https://www.kaggle.com/datasets/tareqkhanemu/snoring

22. L. Orlandic, T. Teijeiro, and D. Atienza, "The COUGHVID crowdsourcing dataset, a corpus for the study of large-scale cough analysis algorithms," *Sci Data*, vol. 158, pp.156-166, Jun. 2021.

23. "Snoring vs Sleep Apnea - What the difference sounds like," (Dec 23, 2016) YouTube video, added by DavidOleniaczVideos, Accessed: Jan 2023, Online: https://www.youtube.com/watch?v=rv3XEx8MsiI