Overview:

The ability to assess a patient's mobility, risk of acquiring hospital-acquired pressure injuries, and sleep quality are all valuable clinical indicators. As a result, precise estimate of sleeping positions during sleep is critical for monitoring individual well-being. So, in order to monitor sleep position, we have acquired the IMU sensor data that replicates various lying positions like supine, prone, and side (either right side or left side), sitting using the IMU sensor unit with a sampling frequency of 119HZ integrated in the Arduino Nano 33 SENSE. Here, the data collected from the IMU sensor is hard to interpret, so in this project, we have calculated "Roll", "Pitch", "Yaw" from accelerometer readings and use these values to determine the sleeping posture of the subject. The accelerometer, gyroscope, magnetometer readings corresponding to these posture are then written to a CSV file, also we built an deep learning algorithm that can detect the lying posture. We used the 2 D convolution neural networks as the deep learning algorithm with 144,613 training parameters. Later, we have converted the model for microcontroller using Tensorflow Lite with quantization.

Function Description: We have trained the model for 75 epochs, the training accuracy is 96.26% and test accuracy is 94.41%. For this we have used 86043 observations.

Deliverables: Initially, we have collected the data and written to "output.csv" file. Then, we have trained the model using "EML_Assignment.ipynb" file and converted the model using Tensorflow Lite and we got "model_quant.tflite" and hex file "model.h". Later, we have deployed the model on Arduino using 'Assignment3.ino" file.

Experiment:

In this project, we have considered 3 parameters "Roll", "Pitch", "Yaw" to determine the sleeping posture. The Roll, Pitch and Yaw are the measurement of the angle of orientation across the Longitudinal, Lateral and the Vertical Axis, respectively. To determine the human posture, the orientation across the Longitudinal, Lateral and the Vertical is to be measured. In this project, as we are considering sleeping position, the orientation of a person across the lateral and vertical axis is constant during his/her sleep, so we are using "Roll" and "Pitch" to determine the sleeping postures.

When the Roll and Pitch are 0 degrees, it means the person (here Microcontroller board) is facing upwards and resting on a bed which would be considered as the supine position. When the person (i.e, microcontroller) is facing down which can be considered as the prone position, then the Roll is 180 degrees and Pitch is 0 degrees. Apart from these two values of Roll (i.e R !={0,180}), remaining positions across the Longitudinal Axis and Pitch equal 0 degrees can be considered as the side position. If the person is sitting then pitch is 90 degrees. Other than the above specified angles we have considered the position as unknown. However, due to errors in calibration of the board we have considered sleeping position of a person based on roll and pitch as follows:

$$\text{Position} \begin{cases} \text{Supine}, -5 \le \text{roll} \le 5 \text{ and } -5 \le \text{pitch} \le 5 \\ \text{Prone}, -180 \le \text{roll} \le -170 \text{ or } 170 \le \text{roll} \le 180 \text{ and } -5 \le \text{pitch} \le 5 \\ \text{Side}, -170 \le \text{roll} \le -50 \text{ or } 5 \le \text{roll} \le 170 \text{ and } -5 \le \text{pitch} \le 5 \\ \text{Sitting}, 80 \le \text{pitch} \le 100 \end{cases}$$

Initially, we used the “MadgwichAHRS” library by Arduino to calculate the Roll, Pitch and Yaw. However, the values got deviated from the original values when the delay is added. So, we have calculated these parameters by using accelerometer reading from IMU sensor as follows:

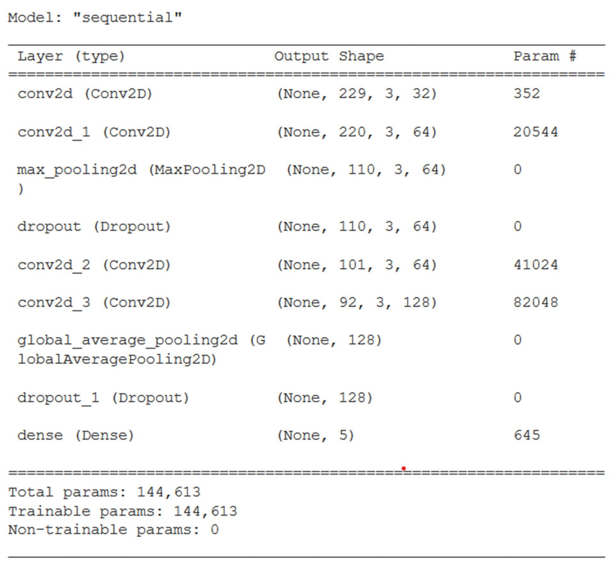
Roll = atan2f(ay,az), Pitch= asinf(x), Yaw = atan2f(ax,ay)

Algorithm:

1. Algorithm to detect Sleeping Position and creating dataset:

```
1. Import the required Libraries
2. Initialize the serial communication and IMU Sensor
3. Check the status of the Serial Communication and IMU Sensor
4. If accelerometer is accesible, then:
5.   Calculate Roll, Pitch, Yaw
6.   if $-5 \le$ pitch $\le 5$:
7.      // to detect if a person is sleeping or not, if a person is sleeping
8.      // then pitch = 0 but range is considered due to errors in sensor
9.      if $-5 \le$ roll $\le 5$:
10.          Print("Supine")
11.     else if $-180 \le$ roll $\le -170$ or $170 \le$ roll $\le 180$ :
12.          Print ( "Prone" )
13.     else:
14.          Print("Side")
15.  else if $80 \le$ pitch $\le 100$:
16       Print(" Sitting")
17.  else:
18.      Print("Unknown")
19.Delay 10 milli seconds
20.Jump to 4.
```

2. Convolution Neural Network Architecture:



```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 229, 3, 32)        352

 conv2d_1 (Conv2D)           (None, 220, 3, 64)        20544

 max_pooling2d (MaxPooling2D  (None, 110, 3, 64)       0
 )

 dropout (Dropout)           (None, 110, 3, 64)        0

 conv2d_2 (Conv2D)           (None, 101, 3, 64)        41024

 conv2d_3 (Conv2D)           (None, 92, 3, 128)        82048

 global_average_pooling2d (G  (None, 128)              0
 lobalAveragePooling2D)

 dropout_1 (Dropout)         (None, 128)               0

 dense (Dense)               (None, 5)                 645

=================================================================
Total params: 144,613
Trainable params: 144,613
Non-trainable params: 0
_________________________________________________________________
```

In our model, we have used 4 convolution 2D operations, 1 Max pooling operation, 1 Global Average Pooling operations. The input layer has the shape of 283*3 (3 represents Roll, Yaw, Pitch values and 238 represents the segment of data points taken in a sliding window of 2s. For the training process, we have used 80 % of observations in a dataset and remaining 20 % as test dataset. For validation we have used 20 % of observations in a training dataset. For our model we have used 144,613 training parameters and we have trained the model for 75 epochs.

Results:

For training the neural network, we have used 86043 observations. As, the sliding window size is 2 seconds and the sampling rate of the IMU sensor is 119HZ , so for each window, single data segment contains 2*119 = 238 of time steps and we have used step size of 40, hence the total number of data segments is 712. We have split these 712 data segments into training set with size of 569 observations and test set with size of 143 observations. Also, we have used 20% of training dataset as validation set.

We have trained the model for 50 epochs, the training accuracy is 96.26% and training loss is 13.03% while the validation accuracy is 95.61% and training loss is 13.29%. The plots of training loss vs validation loss and training accuracy vs validation accuracy is shown in Figure 1 and 2 respectively. From the graph, it is observed that the training and validation sets converges at epoch = 25. Finally, we have evaluated our model on the test set and it has test accuracy is 94.41% and test loss is 18.7%.

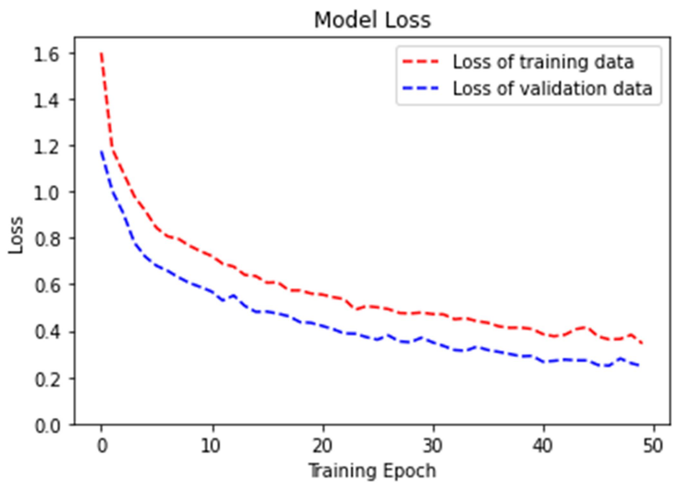


Figure 3: Loss using RELU activation function

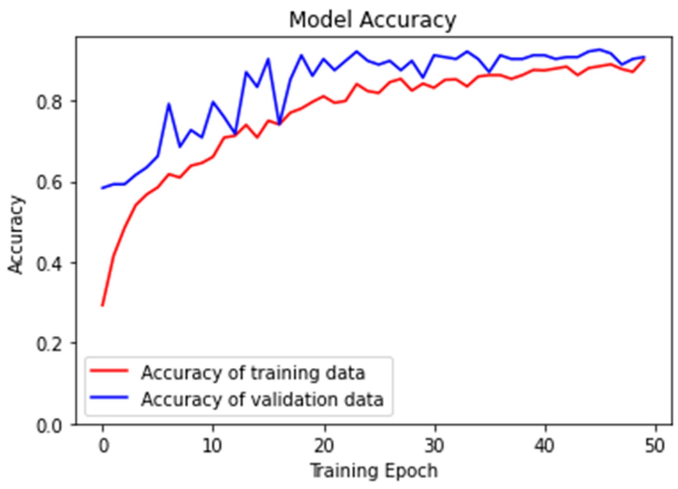


Figure 4: Accuracy using RELU activation function

Discussions:

In this project, we successfully implemented a deep learning algorithm that detects the sleeping position from dataset created using IMU sensor in Arduino Nano 33 BLE. We have performed our experiment on 86043 observations with RELU activation function. Finally, we have achieved the training accuracy of nearly 96.26 % and test accuracy of 94.41 % with ReLU activation function and our model doesn’t undergo overfitting or under fitting. However, while we are performing our experiment we have experienced problem while deploying the model on the Arduino board. Initially, we have experienced problem with memory allocation, we have made necessary changes required for memory allocation. After making required changes the code executed and uploaded successfully, but the Arduino didn’t get recognised by any USB port.