KULLIYYAH OF INFORMATION & COMMUNICATION TECHNOLOGY

CSC 3304 MACHINE LEARNING

SECTION 01

Group 2 Project Report

Title:

"Human Activity Recognition using RNN and LSTM"

PREPARED BY:

AHMAD (1526703)

MAHFUZEALAHI NOMAN (1515803)

ABID ENBA SAIF UTSHA (1433527)

SHEIKH MD HANIF HOSSAIN (1534721)

JIHAD ROBA JOBO (1614857)

LECTURER

DR. AMELIA RITAHANI BT. ISMAIL

DUE

6th May 2019

Contents

1.0 Abstract

Human physical activity recognition based on sensor data has applications relevant to our daily life such as healthcare, eldercare, security, etc. Instead of exploring handcrafted features from time-series sensor signals, signal sequences of accelerometers and gyroscopes are collected into a novel activity image, which enables Deep Convolutional Neural Networks (DCNN) to learn from within the optimal features using the activity image for recognising human activity. The proposed approach is to implement Recurrent Neural Network (RNN) and its improved Long Short-Term Memory (LSTM) on smartphone data in order to identify some specific human activities. The experimental setup shows that LSTM performs better in terms of accuracy and recognising activity than RNN. RNN provides an accuracy of approximately 60 percent, while on the other hand, LSTM gives an accuracy of 86 percent.

2.0 Introduction

Human Activity Recognition is a broad area which focuses on identifying specific human movements or actions based on sensor data. Movements would be typical activities like walking, sitting, laying on a bed. The sensor data can be collected in many ways like videos, radar, etc. The purpose of this project is to identify human behaviour using smartphone dataset. The smartphone is attached to the wrist and based on accelerometer and gyroscopes It will help to recognize the type of activity that the user is doing, for example - walking in a lobby or laying on a bed. Human activity analysis has been getting a remarkable amount of attention lately in the tech community. Its uses can range from intelligent surveillance systems to enhancing human-computer interaction (HCI). Furthermore, human activity can vary in action and in the number of people involved. Hence more information is required other than the motion of individuals. This paper focuses on activity recognition using sensor data on one individual using both an RNN and LSTM model to identify and compare the results and effectiveness of these two networks.

3.0 Background of Study

In human activity recognition, different types of sensor technologies have been explored to improve the recognition rate of human activities. Generally, those can be broadly categorized in three approaches: vision-based approach, environment interactive sensor based approach and wearable sensor based approach. Vision based approach mainly employ a camera or video to monitor and recognise different activity. This approach provide best recognition rate but there are some concern about the use of camera and video in indoor activity due to privacy concern. Also, the cost of monitoring using video or camera is pretty high. In addition, vision based approach often suffer from background change which limits their actual use. For environment interactive sensor based approach, it recognizes human activities and by capturing the interactions between subject and the object. For example, if a sensor imbedded in chair is triggered, it can be inferred that a person is sitting on the chair. Another example would be, if a sensor under the bed is triggered for a long time, it can be inferred that the person is sleeping. These schemes can often help to identify daily activities such as eating, sleeping, sitting and others. But the problem is this approach is costly to implement and often limited to indoor activities. Additionally, the sensor placement and distribution is critical to recognition rate. Unlike above two mentioned approach, wearable sensor based approach is suitable for both indoor and outdoor settings. It can be worn in both upper part of the body (arm, wrist) and lower part of the body (leg, ankle, waist). Activity recognition based on smartphone is promising and attractive due to its non-intrusiveness, high acceptance and adherence in daily life. In addition, smartphones contain built in sensor such as GPS, accelerometer, gyroscope which can be used to gather device related information.

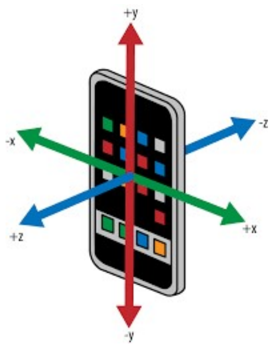

Figure 1: Accelerometer for smartphone

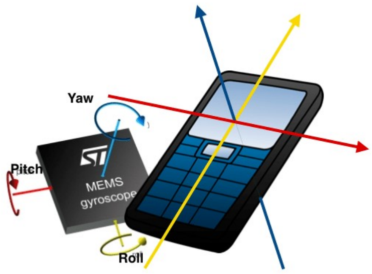

Figure 2: Gyroscope for smartphones

Among the smartphone sensors, GPS that are commonly used in location-based services, can locate one’s current position and track the trace. The triaxial accelerometer measures the proper acceleration and can be used to obtain the acceleration of three orthogonal axes: forward acceleration in y-axis, horizontal movement acceleration in x-axis and vertical acceleration in z-axis. It contains information to recognize different activities, for example, discriminating upstairs walking from downstairs walking with the acceleration values of z-axis in the two different cases. Accelerometer sensors sense the acceleration event of smartphones. The reading includes three axes whose directions are predefined in Figure 1. A time stamp can also be returned with three axes readings. Accelerometer is heavily used in smartphone sensors based activity recognition. Accelerometer directly measures the subject's physical motion status. For instance, if a user changes his activity from walking to jogging, it will reflect on the signal shape of the acceleration reading along the vertical axis and there will be an abrupt change in the amplitude. Furthermore, the acceleration data could indicate the motion pattern within a time limit which is helpful in the complex activity recognition. Gyroscope measures the phone's rotation rate by detecting the roll, pitch and yaw motion of the smartphones along the x, y and z axis. Gyroscope is helpful in navigation system and mobile gaming which use the rotation data. Gyroscope also used to assist the mobile orientation detection in activity recognition. Consequently, smartphones provide an alternative way for human activity recognition.

Sequence prediction challenges have a been around us for quite a while and are considered by some, as one of the hardest problems to deal with in the data science field. This is where Recurrent Neural Networks (RNN) and Long Short-Term Memory (LSTM) networks come into the picture, as it has the potential of providing an effective solution.

The idea behind RNNs is to make use of sequential information. In a traditional neural network, we assume that all inputs (and outputs) are independent of each other. But for many tasks that's a very bad idea. If someone wants to predict the next word in a sentence he or she better know which words came before it. RNNs are called recurrent because they perform the same task for every element of a sequence, with the output being dependent on the previous computations. Another way to think about RNNs is that they have a "memory" which captures information about what has been calculated so far (Britz, 2015). In theory RNNs can make use of information in arbitrarily long sequences, but in practice they are actually limited to looking

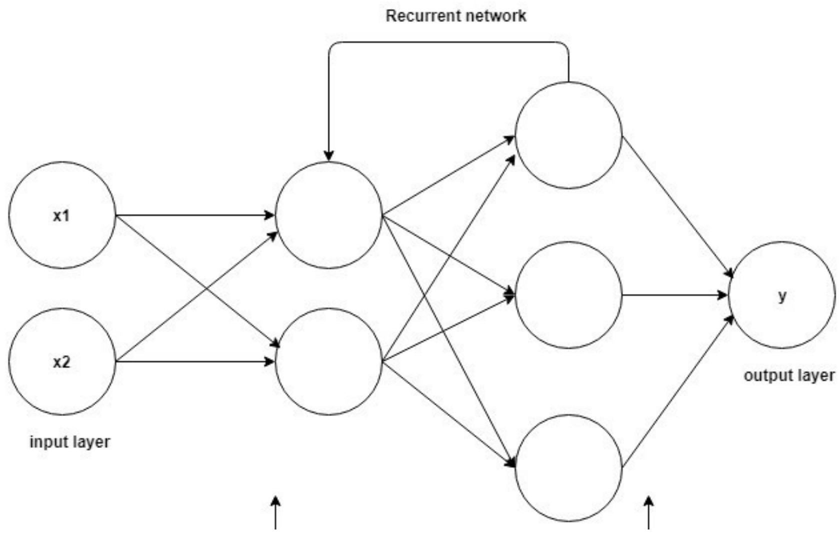

Figure 3: Simple RNN architecture

back only a few steps and that is why RNNs can produce limitations for networks that need to look back at several previous sequences (time steps). This issue is well known as the exploding and vanishing gradient problem.

An LSTM is said to be an improved version of RNN as it is capable of learning long-term dependencies. It is usually considered to be more complex but easier to train due to its ability to avoid the vanishing and exploding gradient problem. The concept of LSTM networks was introduced by Hochreiter and Schmidhuber in 1997 and was further refined by many people that work in this related field. LSTMs help preserve the error that can be backpropagated through time and layers. By maintaining a more constant error, they allow recurrent nets to continue to learn over many time steps (A.I Wiki, 2018).

The information in LSTMs are contained in a gated cell. Data can be stored in, written to, or read from that cell, much like data in a computer's memory. The cell makes decisions about what to store, and when to allow reads, writes and removals, via three gates that open and close (Kang, 2017). These gates are known as the input, output and forget gate. The input gate is in charge of controlling how much previous data can enter the cell, while the forget gate manages how much previous input data should be kept while the remaining data is dropped from the cell and finally the output gate provides the information that should be going to the next layer. Unlike the digital storage on computers, however, these gates are analog, implemented with element-wise multiplication by sigmoids, which are all in the range of 0-1. The gates block or pass on information based on its strength and import, which they filter with their own sets of weights. These weights are adjusted by the recurrent network learning process through a gradient descent. Due to its capability of working with various types of problems, LSTMs are widely used by many industries and organizations for various projects.

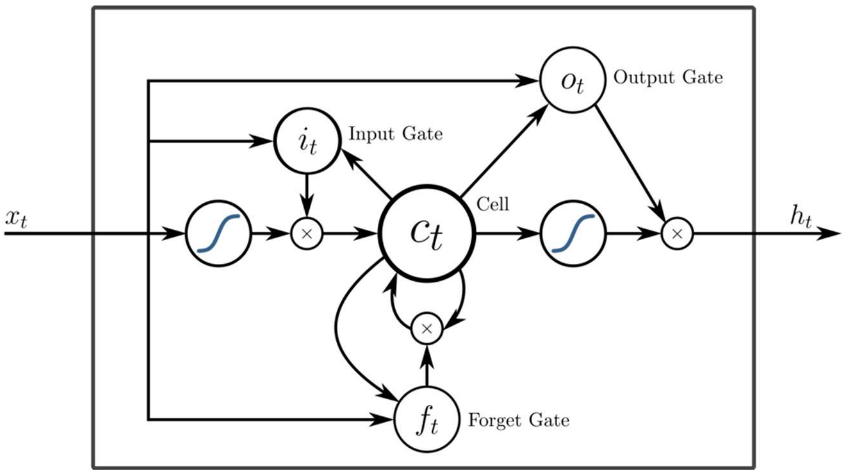

Figure 3: Simple LSTM architecture

Our study will be focused on Human Activity Recognition (HAR) using a smartphones dataset. This research is extended work of Guillaume Chevalier (2016) which is available in his GitHub repository.

4.0 Research Problem

Firstly, one of the challenges of this project is to predict human activity with a small number of sensor data. Generally, this problem is called as univariate or multivariate time series classification task. Additionally, different people perform the same activity in different way, sometimes, even the same person does the same activity slightly different manner in different time. This type of issue can be solved using deep learning. The next concern is whether LSTM would indeed perform better than an RNN model. Due to RNN having the vanishing and exploding gradient problem, they are supposedly not very effective when it comes to the model requiring to make use of information from several preceding time steps (sequences). This is partially because the information flowing through neural nets passes through many stages of multiplication, which can lead to the gradients becoming too small (vanishing) or extremely large (exploding) causing the weight to become biased. LSTMs are supposed to solve this issue and we would like to confirm whether that is the case. In this project, we are going to implement a Recurrent Neural Network as well as an LSTM(Long-Short Term Memory) to compare the performances of both models and identify whether LSTM does indeed work better than RNN.

5.0 Research Question

Do LSTM networks perform better than RNN networks for human activity recognition?

6.0 Research Objectives

- Identify human activity which includes walking, walking upstairs, walking downstairs, sitting, standing, laying.
- In-depth learning of Recurrent Neural Networks and LSTM
- Implement RNN and LSTM models with keras
- Achieve high recognition accuracy with low computational cost
- Compare results from the RNN and LSTM models

7.0 Research Significances

Even though it is challenging, human activity recognition is a very significant and important research topic and technology. This is because it can be used in many real-life human-centric problems such as – healthcare, eldercare, security. One of the prospect would be Alzheimer's disease characterized by long-time sitting or lying down. Another one would be identifying sleeping disorder by identifying activities such as sitting and lying down. This research also helped us to identify how RNN and LSTM functions. Furthermore, it helped us to get a better understanding about deep learning techniques and why LSTM manages to perform much better than RNN.

8.0 Literature Review

A large number of researchers have done considerable amount of work in exploring different sensing technologies and proposed a number of methods to model and recognize human activities. Bao and Intille (2004) used five biaxial accelerometer which were worn on the user's right hip, dominant wrist, non-dominant upper arm, dominant ankle and non-dominate thigh to monitor twenty types of activities, trained using twenty users. Experimental results showed that a decision tree classifier obtained the best performance with an accuracy of 84 percent.

Tapia et al. (2007) proposed to implement a real – time system that can recognize physical activities as well as their intensities using heart rate monitor and five triaxial accelerometers placed on the right arm, right leg and the waist. A Decision Tree and Naive Bayes were used as the activity recognition algorithm. The authors applied the system to recognise thirty physical gymnasium activities and obtained a 94.6 percent recognition rate and 56.3 percent subject independent accuracy and managed to obtain an 80.6 percent accuracy without differentiating the activity intensities. Multiple accelerometers attached to the body simultaneously leads to better recognition rates. But this type of sensors often discomforts users, hence a single accelerometer was used in many researches in the community.

Ravi, Dandekar, Mysore and Littman (2005) carried out a study to explore the possibility of activity recognition using a single triaxial accelerometer. They used an accelerometer mounted onto the hip of an individual to collect accelerometer data for about eight activities, including standing, walking, running, upstairs, downstairs, sitting, vacuuming and brushing teeth. Different settings were considered for the dataset which includes: data collected for a single subject over different days, data collected for multiple subjects over different days, data collected for a single subject on one day as training data and further data collected for the same subject on another day as testing data, data collected for a single subject on one day as training data and further data collected on another subject on another day as testing data. The experimental setup results find the plurality voting classifier as the best for the first three settings. It was also found difficult to distinguish the upstairs activity from downstairs or running and impossible to distinguish brushing teeth versus downstairs activities. Their classification accuracy for the forth setting was 73.33 percent.

Kwapisz, Weiss and Moore (2011) collected data from twenty-nine users, each carrying an Android phone in their pocket and performed six activities: walking, jogging, upstairs, downstairs, sitting and standing. Three learning algorithms were used: Logistic regression, J48 and Multilayer perceptron. The overall accuracy is above 90 percent, except for the case upstairs versus downstairs which posed greater difficulty. They also found that neural network classification can best detect jogging and upstairs while J48 effectively detects the other activities.

Akram Bayat, Pomplun and Tran (2014) collected data from multiple subjects using single triaxial accelerometer under real-world conditions for two most common phone positions: smartphone in hand and in pants pocket. They evaluated the performance of the following classifiers using Weka toolkit: Multilayer Perceptron, Random Forrest, Logistic Model Tree(LMT), Support Vector Machine(SVM), Simple Logistic and LogitBoost. Classifiers were trained and tested using a 10-fold cross validation method on the set of extracted features. A recognition accuracy of up to 91 percent was obtained on various everyday activities.

Casale, Pujol and Raveda (2011) used a wearable device for gathering acceleration fata for human activity recognition, obtaining a 94 percent of accuracy. In comparison with multiple accelerometers, a single accelerometer may be unable to discriminate similar activities such as upstairs and downstairs for activity recognition.

Huynh, Nguyen, Irazabal, Ghassemian and Tran (2015) used an accelerometer and a gyroscope to construct a wireless and wearable fall detection system and proposed a critical threshold based activity recognition algorithm to differentiate falls from non-falls. The sensing unit was attached to the chest of the participants to collect motion data under different daily activities such as standing, walking, sitting, running and four fall scenarios (forward, backward, left and right sideway).

Dernbach, Das, Krishnan, Thomas and Cook (2012) demonstrated the possibility of using the inertial sensor data collected from android-based smart phones to recognize simple activities such as biking, climbing, sitting, walking, running and standing as well as complex activities such as cleaning, cooking, washing and watering.

Anjum and Ilyas (2013) built a smartphone application to detect seven human physical activities and further to estimate calories consumption. The researchers collected 510 activity traces and evaluated the effectiveness with k-nearest neighbour, naïve bayes, support vector machine and decision tree classifiers.

Mohammad Mehedi Hasan, Md. Zia Uddin, Amr Mohamed and Ahmad Almogren (2018) used deep learning to get a successful activity recognition and compared it with traditional machine learning algorithms. The activity the authors tried to recognize were sitting, standing, lying down, walking, walking upstairs, walking downstairs, stand-to-sit, sit-to-stand, sit-to-lie, lie-to-sit, stand-to-lie, lie-to-stand. The mean recognition rate of the traditional ANN based approach was 65.31 percent while the traditional SVM approach achieved 82.02 percent. The accuracy was 94.12 percent using SVM. The proposed Deep Belief Network had a mean recognition rate of 89.61 percent with an accuracy of 95.85 percent.

Tran and Phan (2016) applied a support vector machine to recognize human activity using android smartphones. The six actions selected was sitting, standing, lying down, upstairs walking and downstairs walking. Ten participants carried out the activities aged between 11 to 26. The dataset was split into 70 and 30 percent for the training and testing dataset respectively. The first version was tested with 561 features and gave an accuracy above 93 percent. On the other hand, the second version was tested with 248 features and gave an accuracy above 89 percent. However, the researchers got poor recognition for the lying down activity in the second version.

Jian, Minh, Phyo, Xiao and Krishnaswami (2015) used a deep convolutional neural network for human activity recognition. The authors considered two datasets for human activity recognition: whole body's movement and only hand's movement. Their experimental setup demonstrated that even with and without smoothing, CNN performs better than DBN, SVM and others.

Jindong, Yiqiang, Shuji, Xiaohui and Lisha (2018) surveyed several deep learning techniques on human activity recognition. Jindong et al., concluded that RNN and LSTM are better to recognize short activities while CNN is better at inferring long-term repetitive activities. RNN performs better at time order relationship, on the other hand CNN is capable of learning deep features contained in recursive patterns.

Hammerla, Halloran and Plotz (2016) used deep, convolutional and recurrent models for human activity recognition using wearables. Hammerla et al., found that LSTM works better with the large dataset by a considerable margin than other state-of-the art algorithms. Just like Jindong et al., Hammerla et al., found that RNN works better with activities that are short in duration and on the other hand CNN works better with repetitive activities that are longer in duration. Additionally, Deep Neural Networks perform better but requires a lot more parameter optimization than RNN and CNN.

From the literatures above, it can be understood that many researchers are moving towards the deep learning techniques to identify human activity recognition. DBN, CNN and RNN are the most common approaches chosen by the researchers based on the dataset and project requirements. For this project, both RNN and LSTM will be implemented to compare and identify which model would perform better.

9.0 Methodology

This project was conducted using python from Google Colab. The dataset was collected from UCI Machine Learning Repositories named 'Human Activity Recognition using a Smartphones Data Set. The experiments have been carried out with a group of 30 volunteers within an age bracket of 19-48 years. Each person performed six activities (WALKING, WALKING_UPSTAIRS, WALKING_DOWNSTAIRS, SITTING, STANDING, LAYING) wearing a smartphone (Samsung Galaxy S II) on the waist. Using its embedded accelerometer and gyroscope, 3-axial linear accelerations and 3-axial angular velocities at a constant rate of 50Hz was captured. The experiments have been video-recorded to label the data manually. The obtained dataset has been randomly partitioned into two sets, where 70% of the volunteers was selected for generating the training data and 30% as the test data. The dataset was pre-processed to check for any missing values or inconsistencies. Next, Exploratory Data Analysis (EDA) was performed and data visualization was done to check whether it was possible to identify those activities.

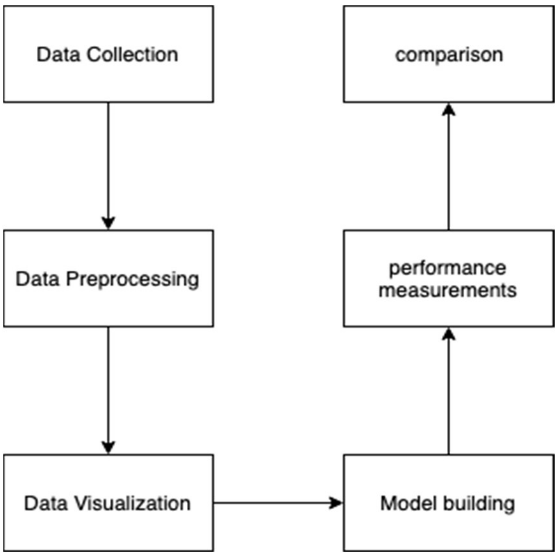

Figure 4: Diagram of Methodology

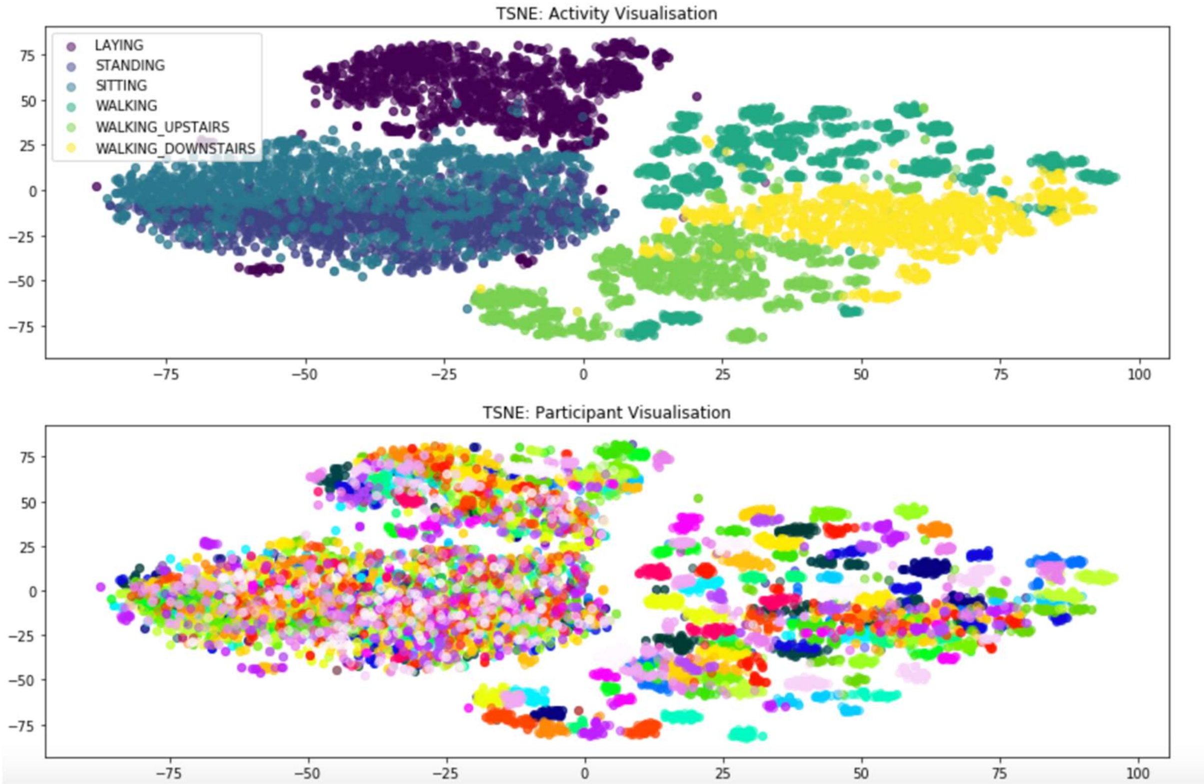

Figure 5: Visualization

From figure 6, it can be seen that those attributes are unique and are identifiable. Next, two deep learning technique Recurrent Neural Network (RNN) and Long-Short Term Memory was used to build the model. Accuracy was used as performance evaluation measurements. The library 'keras' was used to build both of the models. The output activities were converted into categories to get better results.

For both RNN and LSTM, the number of epoch and batch sizes were 15 and 64 respectively. The epoch is the number of iterations related with each input sample in the dataset. Passing the full dataset through the network only once is not a good option. Instead, through each epoch, the full dataset was passed multiple times. Multiple epochs help to update weights and biases as it progresses through each epoch. As the number of epochs increases, the model goes from underfitting to optimal and then to overfitting. 15 epochs were considered best with our dataset. The batch size is the number of samples that will be passed through the network. For this project the batch size was considered 64, which means, first 64 samples were trained from training dataset and next, 64 samples were trained again in the next batch. For both of the models, total acceleration, body acceleration and body gyroscope were passed as input and categorical activity was the output. Then, the dropout 0.5 was applied to both models. Dropout is a regularization technique used to prevent overfitting. Using dropout, the number of iterations required to converge almost doubles and the training time for each epoch is less. Furthermore, dropout increases the validation accuracy and decreases the loss.

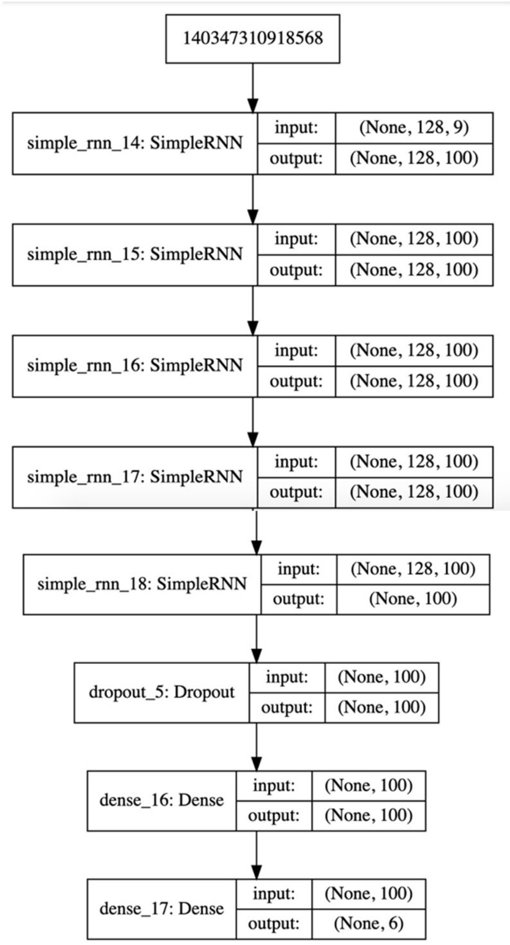

Figure 6: RNN architecture

After that, five hidden layer with 100 nodes was added and the activation function used for each node was ‘relu’ which stands for Rectified Linear Unit (ReLU). ReLU is a non-linear function which helps to backpropagate the errors easily. Additionally, the ReLU function does not activate all the neurons at the same time. Next, in the output layer, a ‘softmax’ activation function was used. The Softmax function helps to give categorical output for more than two categories. In addition, a ‘categorical_crossentropy’ loss was applied in the model.

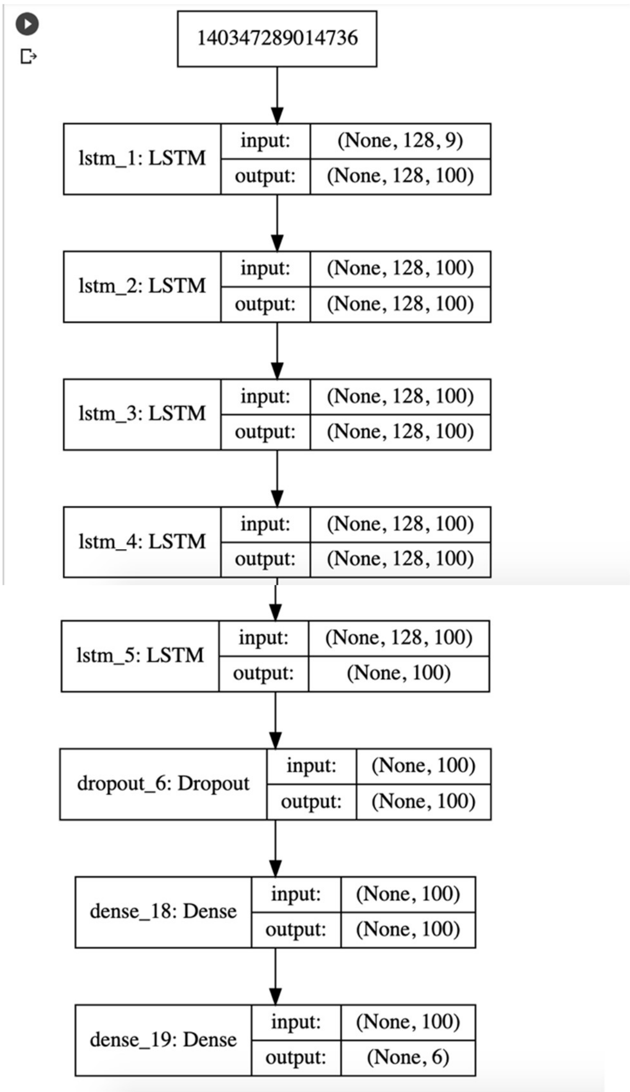

Figure 7: LSTM architecture

The ‘categorical_crossentropy’ is a cross_entropy applied in cases where there are many classes or categories and among those classes only one is true. It is used for multi-class classification. Next, the ‘Adam’ optimizer was used in the model. Adaptive Moment Estimation (Adam) optimizer is a combination of two other extensions of the stochastic gradient descent, specifically, Adaptive Gradient Algorithm (AdaGrad) and Root Mean Square Propagation (RMSProp). Stochastic gradient descent maintains a single learning rate whereas both AdaGrad and RMSprop maintains a per-parameter learning rate. For performance evaluation metrics, the ‘accuracy’ for each model was evaluated.

10.0 Results

10.1 RNN Results

The model is trained and evaluated using RNN first. The result is provided in the figure below.

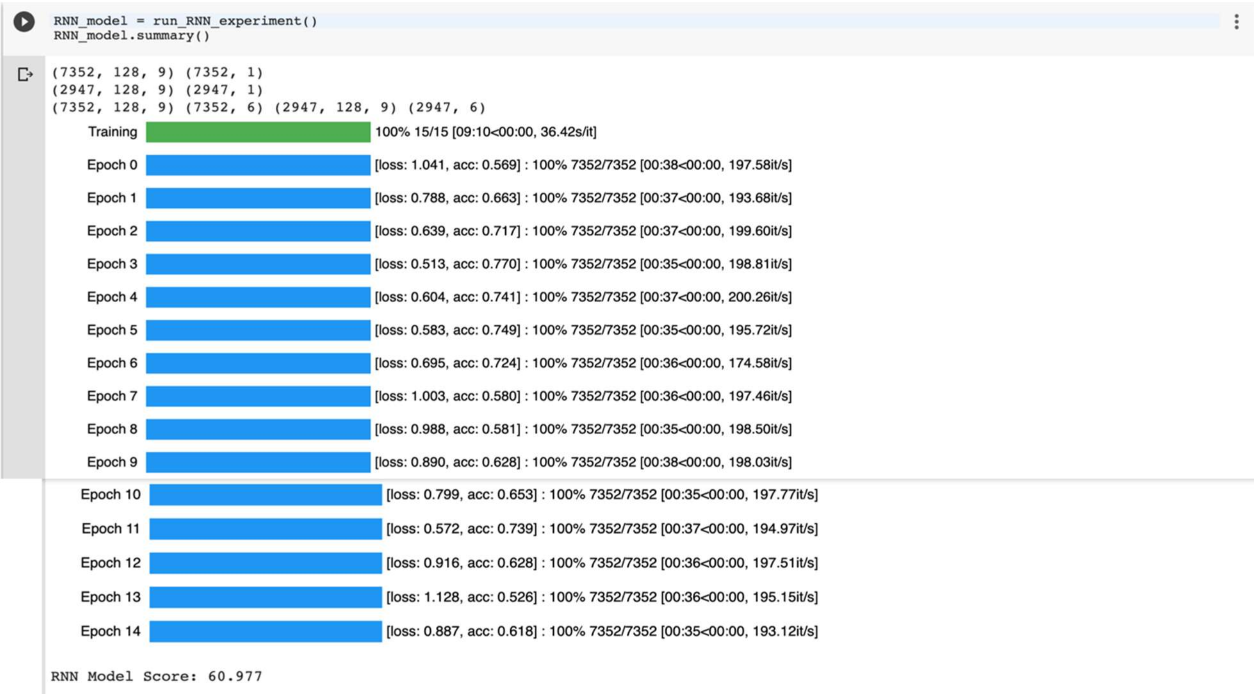

Figure 8: RNN experiment result

A total of 15 epochs were executed. From the diagram above we can see that the first two epochs (Epoch 0 and 1) had a low accuracy of 56.9% and 66.3%. Starting from epoch 2, the accuracy started improving to a much better rate until epoch 7 where it dropped back to 58%. The remaining epochs improved a little bit in accuracy and mostly fluctuating at around 60%. After running 15 epochs, the RNN model gave an average accuracy score of 60.977% which can be considered as a poor model.

```
RNN Model Score: 60.977
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
simple_rnn_14 (SimpleRNN)    (None, 128, 100)          11000
_________________________________________________________________
simple_rnn_15 (SimpleRNN)    (None, 128, 100)          20100
_________________________________________________________________
simple_rnn_16 (SimpleRNN)    (None, 128, 100)          20100
_________________________________________________________________
simple_rnn_17 (SimpleRNN)    (None, 128, 100)          20100
_________________________________________________________________
simple_rnn_18 (SimpleRNN)    (None, 100)               20100
_________________________________________________________________
dropout_5 (Dropout)          (None, 100)               0
_________________________________________________________________
dense_16 (Dense)             (None, 100)               10100
_________________________________________________________________
dense_17 (Dense)             (None, 6)                 606
=================================================================
Total params: 102,106
Trainable params: 102,106
Non-trainable params: 0
_________________________________________________________________
```

Figure 10: RNN Result

10.2 LSTM Results

Next, the LSTM model is implemented, and the accuracy results obtained by the model is provided in the figure below.

```
LSTM_model = run_LSTM_experiment()
LSTM_model.summary()

(7352, 128, 9) (7352, 1)
(2947, 128, 9) (2947, 1)
(7352, 128, 9) (7352, 6) (2947, 128, 9) (2947, 6)
Training  100% 15/15 [28:55<00:00, 116.05s/it]
Epoch 0   [loss: 1.168, acc: 0.495] : 100% 7352/7352 [01:57<00:00, 63.39it/s]
Epoch 1   [loss: 0.835, acc: 0.646] : 100% 7352/7352 [01:53<00:00, 62.32it/s]
Epoch 2   [loss: 0.773, acc: 0.666] : 100% 7352/7352 [01:53<00:00, 63.09it/s]
Epoch 3   [loss: 1.240, acc: 0.420] : 100% 7352/7352 [01:56<00:00, 62.23it/s]
Epoch 4   [loss: 1.021, acc: 0.510] : 100% 7352/7352 [01:55<00:00, 60.69it/s]
Epoch 5   [loss: 0.877, acc: 0.569] : 100% 7352/7352 [01:54<00:00, 61.05it/s]
Epoch 6   [loss: 0.726, acc: 0.624] : 100% 7352/7352 [01:56<00:00, 55.39it/s]
Epoch 7   [loss: 0.742, acc: 0.640] : 100% 7352/7352 [01:55<00:00, 61.82it/s]
Epoch 8   [loss: 0.618, acc: 0.698] : 100% 7352/7352 [01:55<00:00, 61.75it/s]
Epoch 9   [loss: 0.528, acc: 0.766] : 100% 7352/7352 [01:56<00:00, 62.06it/s]
Epoch 10  [loss: 0.434, acc: 0.819] : 100% 7352/7352 [01:54<00:00, 61.34it/s]
Epoch 11  [loss: 0.340, acc: 0.870] : 100% 7352/7352 [01:57<00:00, 62.57it/s]
Epoch 12  [loss: 0.278, acc: 0.899] : 100% 7352/7352 [01:55<00:00, 61.98it/s]
Epoch 13  [loss: 0.256, acc: 0.907] : 100% 7352/7352 [01:57<00:00, 61.39it/s]
Epoch 14  [loss: 0.273, acc: 0.903] : 100% 7352/7352 [01:55<00:00, 61.97it/s]

LSTM Model Score: 86.359
```

Figure 9: LSTM experiment result

For the LSTM, a total of 15 epochs were executed as well to ensure fairness between both models. Epoch 0 started very poorly with an accuracy of only 49.5%. Epoch 1 and 2 had an improvement in accuracy of above 60 percent but in epoch 3,4 and 5, the accuracy dropped below 50 percent and fluctuated around that. Starting from epoch 6 and onwards the LSTM model started improving and once it reached epoch 10 and onwards, the model's accuracy improved drastically and reached an all time high on epoch 13, achieving an accuracy of 90.7%. After the execution of all 15 epochs, the LSTM model managed to get an average accuracy of 86.359%.

```
LSTM Model Score: 86.359

Layer (type)                 Output Shape              Param #
=================================================================
lstm_1 (LSTM)                (None, 128, 100)          44000
lstm_2 (LSTM)                (None, 128, 100)          80400
lstm_3 (LSTM)                (None, 128, 100)          80400
lstm_4 (LSTM)                (None, 128, 100)          80400
lstm_5 (LSTM)                (None, 100)               80400
dropout_6 (Dropout)          (None, 100)               0
dense_18 (Dense)             (None, 100)               10100
dense_19 (Dense)             (None, 6)                 606
=================================================================
Total params: 376,306
Trainable params: 376,306
Non-trainable params: 0
```

Figure 12: LSTM Result

11.0 Performance Evaluation

From the results section, it is clear that the LSTM model performed much better than RNN.

Pred / True	LAYING	SITTING	STANDING	WALKING	WALKING_UPSTAIRS
LAYING	510	0	26	0	1
SITTING	0	241	248	1	1
STANDING	0	20	506	0	6
WALKING	0	0	31	146	319
WALKING_DOWNSTAIRS	0	0	13	104	303
WALKING_UPSTAIRS	0	4	26	47	394

Figure 10: Confusion Matrix of RNN model

Looking into the confusion matrix for RNN, it successfully predicts the laying activity without any issues while for the sitting activity it did very well, managing to predict 241/265 (90.9%) correctly. Although RNN managed well for the first two activities, it performed very poorly for the other four activities which is standing, walking, walking upstairs and walking downstairs. The model managed to only predict 506/850 (59.5%) standing activity and 146/298 (49%) for walking. For walking upstairs, the model only managed to classify only 394/1024 (38.4%) correctly. This indicates that the similarity of those four activities caused the model to have difficulty in predicting them accurately given the fact that it can't make use of several preceding time steps.

Pred / True	LAYING	SITTING	STANDING	WALKING	WALKING_DOWNSTAIRS \
LAYING	513	24	0	0	0
SITTING	0	246	242	3	0
STANDING	0	19	512	1	0
WALKING	0	0	4	453	15
WALKING_DOWNSTAIRS	0	0	0	0	402
WALKING_UPSTAIRS	0	0	1	26	25

Pred / True	WALKING_UPSTAIRS
LAYING	0
SITTING	0
STANDING	0
WALKING	24
WALKING_DOWNSTAIRS	18
WALKING_UPSTAIRS	419

Figure 11: Confusion Matrix for LSTM

On the other hand, the LSTM model accurately classified all six activities with minimal errors. Laying was predicted with no problems, while sitting was predicted correctly for 246/289 (85%). The standing activity was the only one that had a moderate accuracy prediction of 512/759 (67.5%). As for walking it managed to predict correctly 453/483 (93.8%). The walking downstairs activity was also predicted well by 402/442 (91%) while walking upstairs got 419/461 (90.9%).

12.0 Conclusion

From the observations of both model performances and results, it is very clear that the LSTM model outperformed the RNN model by a huge margin due to the last four activities being undistinguishable for the RNN model. Thus, we can conclude that an LSTM model is definitely the better option compared to an RNN model for detecting certain human activities due to its ability of storing memory of several preceding time steps which helps to eliminate the vanishing and exploding gradient problem which was existent in the RNN model.

13.0 Future Works

For the future, more data for different activities could be added to the model. Apart from that, a similar approach could be implemented on different types of sensor data such as – vision-based sensors and environment interactive sensors. Lastly, different models other than RNN and LSTM could be implemented, for instance, the Convolutional Neural Network, Deep Belief Network and others.

14.0 References

Anjum, A., & Ilyas, M. U. (2013, January). Activity recognition using smartphone sensors. In *2013 ieee 10th consumer communications and networking conference (ccnc)* (pp. 914-919). IEEE.

Bayat, A., Pomplun, M., & Tran, D. A. (2014). A study on human activity recognition using accelerometer data from smartphones. *Procedia Computer Science*, *34*, 450-457.

Budhiraja, A., & Budhiraja, A. (2016, December 15). Learning Less to Learn Better-Dropout in (Deep) Machine learning. Retrieved from https://medium.com/@amarbudhiraja/https-medium-com-amarbudhiraja-learning-less-to-learn-better-dropout-in-deep-machine-learning-74334da4bfc5

Bao, L., & Intille, S. S. (2004, April). Activity recognition from user-annotated acceleration data. In *International conference on pervasive computing* (pp. 1-17). Springer, Berlin, Heidelberg.

Brownlee, J. (2019, April 20). Gentle Introduction to the Adam Optimization Algorithm for Deep Learning. Retrieved from https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/

Casale P., Pujol O., Radeva P. (2011) Human Activity Recognition from Accelerometer Data Using a Wearable Device. In: Vitrià J., Sanches J.M., Hernández M. (eds) Pattern Recognition and Image Analysis. IbPRIA 2011. Lecture Notes in Computer Science, vol 6669. Springer, Berlin, Heidelberg

Chevalier, G. (2019, April 16). Guillaume-chevalier/LSTM-Human-Activity-Recognition. Retrieved from https://github.com/guillaume-chevalier/LSTM-Human-Activity-Recognition

Dernbach, S., Das, B., Krishnan, N. C., Thomas, B. L., & Cook, D. J. (2012, June). Simple and complex activity recognition through smart phones. In *2012 Eighth International Conference on Intelligent Environments* (pp. 214-221). IEEE.

Gupta, D., & Dishashree. (2019, February 22). Fundamentals of Deep Learning - Activation Functions and their use. Retrieved from https://www.analyticsvidhya.com/blog/2017/10/fundamentals-deep-learning-activation-functions-when-to-use-them/

Gomez, R. (2018, May 23). Understanding Categorical Cross-Entropy Loss, Binary Cross-Entropy Loss, Softmax Loss, Logistic Loss, Focal Loss and all those confusing names. Retrieved from https://gombru.github.io/2018/05/23/cross_entropy_loss/

Huynh, Q. T., Nguyen, U. D., Irazabal, L. B., Ghassemian, N., & Tran, B. Q. (2015). Optimization of an accelerometer and gyroscope-based fall detection algorithm. *Journal of Sensors*, *2015*.

Kulbear. (2017). Kulbear/deep-learning-nano-foundation. Retrieved from https://github.com/Kulbear/deep-learning-nano-foundation/wiki/ReLU-and-Softmax-Activation-Functions

Kwapisz, J. R., Weiss, G. M., & Moore, S. A. (2011). Activity recognition using cell phone accelerometers. *ACM SigKDD Explorations Newsletter*, *12*(2), 74-82.

Keras LSTM tutorial - How to easily build a powerful deep learning language model. (2018, February 03). Retrieved from https://adventuresinmachinelearning.com/keras-lstm-tutorial/

Ravi, N., Dandekar, N., Mysore, P., & Littman, M. L. (2005, July). Activity recognition from accelerometer data. In *Aaai*(Vol. 5, No. 2005, pp. 1541-1546).

Su, X., Tong, H., & Ji, P. (2014). Activity recognition with smartphone sensors. *Tsinghua science and technology*, *19*(3), 235-249.

Sharma, S. (2017, September 23). Epoch vs Batch Size vs Iterations. Retrieved from https://towardsdatascience.com/epoch-vs-iterations-vs-batch-size-4dfb9c7ce9c9

Tapia, E. M., Intille, S. S., Haskell, W., Larson, K., Wright, J., King, A., & Friedman, R. (2007, October). Real-time recognition of physical activities and their intensities using wireless accelerometers and a heart rate monitor. In *2007 11th IEEE international symposium on wearable computers* (pp. 37-40). IEEE.

Wang, J., Chen, Y., Hao, S., Peng, X., & Hu, L. (2018). Deep learning for sensor-based activity recognition: A survey. *Pattern Recognition Letters*, *119*, 3-11.

Yang, J., Nguyen, M. N., San, P. P., Li, X. L., & Krishnaswamy, S. (2015, June). Deep convolutional neural networks on multichannel time series for human activity recognition. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.