12-2018

SENSOR-BASED HUMAN ACTIVITY RECOGNITION USING BIDIRECTIONAL LSTM FOR CLOSELY RELATED ACTIVITIES

Arumugam Thendramil Pavai
005777794@coyote.csusb.edu

SENSOR-BASED HUMAN ACTIVITY RECOGNITION USING BIDIRECTIONAL LSTM FOR CLOSELY RELATED ACTIVITIES

A Project

Presented to the

Faculty of

California State University,

San Bernardino

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

in

Computer Science

by

Arumugam Thendramil Pavai

December 2018

SENSOR-BASED HUMAN ACTIVITY RECOGNITION USING BIDIRECTIONAL LSTM FOR CLOSELY RELATED ACTIVITIES

A Project

Presented to the

Faculty of

California State University,

San Bernardino

by

Arumugam Thendramil Pavai

December 2018

Approved by:

Dr. Qingquang Sun, Advisor, Computer Science and Engineering

Dr. Owen Murphy, Committee Member

Dr. Kirsten Voigt, Committee Member

ABSTRACT

Recognizing human activities using deep learning methods has significance in many fields such as sports, motion tracking, surveillance, healthcare and robotics. Inertial sensors comprising of accelerometers and gyroscopes are commonly used for sensor based HAR. In this study, a Bidirectional Long Short-Term Memory (BLSTM) approach is explored for human activity recognition and classification for closely related activities on a body worn inertial sensor data that is provided by the UTD-MHAD dataset. The BLSTM model of this study could achieve an overall accuracy of 98.05% for 15 different activities and 90.87% for 27 different activities performed by 8 persons with 4 trials per activity per person. A comparison of this BLSTM model is made with the Unidirectional LSTM model. It is observed that there is a significant improvement in the accuracy for recognition of all 27 activities in the case of BLSTM than LSTM.

ACKNOWLEDGEMENTS

I would like to express my sincere thanks to my advisor, Dr. Qingquan Sun, for his support and guidance. I thank my committee members, Dr. Owen Murphy and Dr. Kirsten Voigt for their valuable time and guidance. I would also like to thank my parents whose constant support and encouragement helped me accomplish the tasks of this project.

DEDICATION

To my parents for their valuable guidance and support right from my childhood. Thank you for showing me the right direction and inspiring me to work hard and achieve my goals.

To my grandparents for their unconditional love and cherished memories.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER ONE

INTRODUCTION

Purpose

Human activity recognition (HAR) has significance in various fields such as healthcare, robotics, motion monitoring and tracking, surveillance, and sports. Sensor based human activity recognition is in practice since several years. With the advancement in technology, sensor based HAR is constantly growing in terms of efficiency and accuracy. There is a scope for tremendous improvement in sensor based HAR using deep learning models. Deep learning models reduce the need for hand crafted feature extraction and can recognize complex and closely related activities more efficiently than traditional methods.

Sensor data primarily used for HAR purposes are those of body worn inertial sensors comprising of triaxial accelerometer (accelerations in x, y and z axes) and triaxial gyroscope sensors (angular velocities in x, y and z directions). These sensors can capture a sequence of motion data generated in a time series. For this project, the body worn inertial sensor data provided by the UTD-MHAD dataset [5] is used. The dataset consists of groups of closely related activities such as baseball swing and tennis swing, which is obtained by wearing the sensor in the right arm and activities such as lunge and squat which is obtained by wearing the sensor in the right thigh. A Bidirectional Long Short-

Term memory model is trained, and the results are analyzed and compared with a unidirectional Long Short-Term model.

Motivation

The importance of human activity recognition related applications has brought in the demand for cost cutting and efficient methods. The applications range from taking care of elderly patients to analyzing player movements in sports. Simple and basic activities have been recognized successfully in the past using deep learning models. However, recognizing complex and closely related activities is a challenging task. Body worn sensor based HAR is a time series classification task. Many of the sensor based HAR using deep learning have used a set of basic activities on deep learning models. The contribution of this project is to use a variant of LSTM model which is a Bidirectional LSTM model for human activity recognition on activities which are closely related to each other, meaning activities which either have similarity in their sequence of sensor data or appear similar to human eye and on a larger pool of activities, in this case 27 activities. Figure 1 provides with an overview of the proposed model for this HAR.

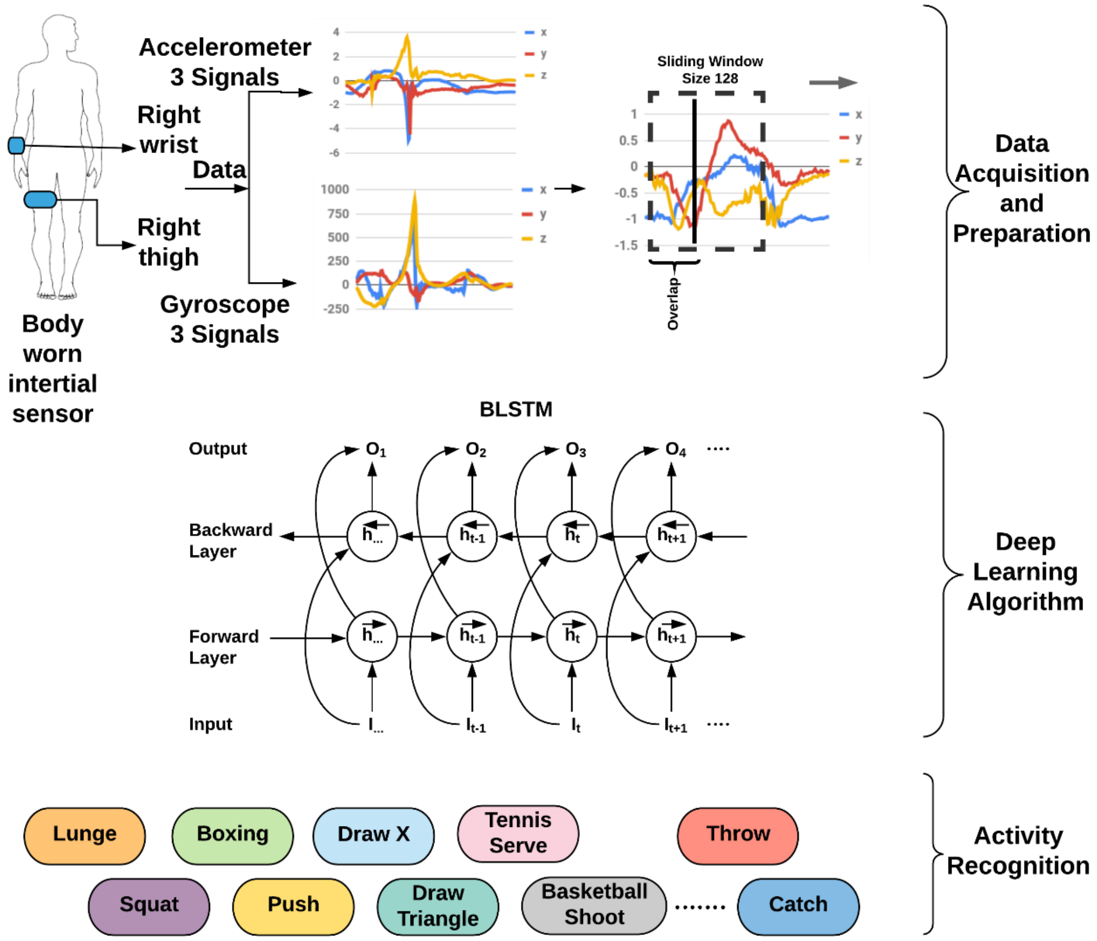


Figure 1. Overview of the proposed model for HAR

CHAPTER TWO

SENSOR BASED HUMAN ACTIVITY RECOGNITION

Types of Sensors for HAR

Most of the human activity recognition can be broadly classified into two categories, video-based activity recognition which is also called as vision-based activity recognition [4,6] and sensor-based activity recognition. Video based approach utilizes images or videos captured by video cameras. The data generated from this approach are in the form of video sequences at a specific frame rate and/or depth image.

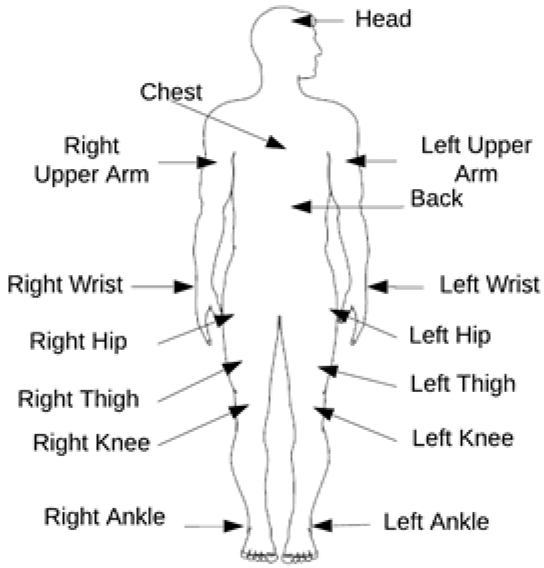


Figure 2. Typical positions of body worn inertial sensors

Sensor-based activity recognition utilizes the motion data captured by sensors such as accelerometer, gyroscope, magnetometer, bluetooth, GPS etc.

The generated data is in time series which can be in time domain or frequency domain. These sensors can again be broadly classified as body worn or wearable sensors which are attached to the human body, object sensors which are attached to the object in the environment, ambient sensors for the environment and hybrid sensors [4].

Body Worn Inertial Sensors

Body worn inertial sensor is the most common sensor used for HAR because of the advancement in wearable computing and availability of low cost and small sized inertial sensors. These sensors comprising of accelerometers and gyroscopes, sometimes magnetometer as well, are attached to specific parts of the body such as, hands and waists to record human motions. The portability and compactness of these sensors makes it suitable for attaching to the body parts for capturing the motion data. Figure 2 provides the typical body positions for these sensors. In this project, the body worn triaxial inertial sensor data (accelerometer and gyroscope) attached to the right wrist and the right thigh of 8 persons, is used. Figure 3 and 4 show the sensor readings of a baseball swing and a tennis swing of the UTD-MHAD dataset.

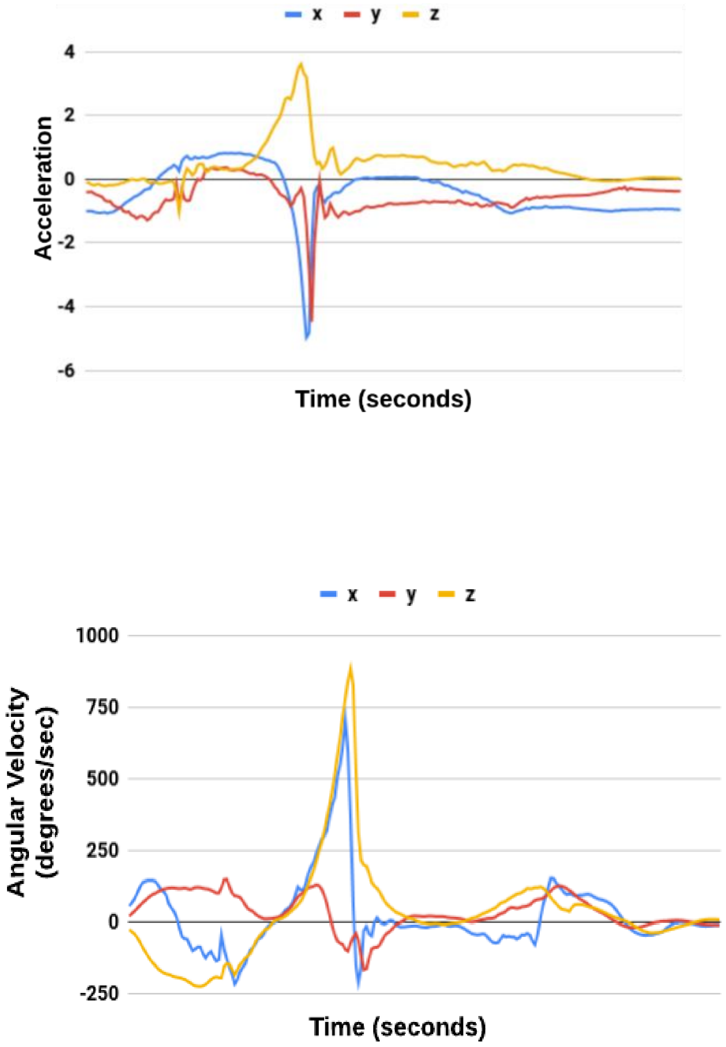


Figure 3. Accelerometer and gyroscope readings for a baseball swing

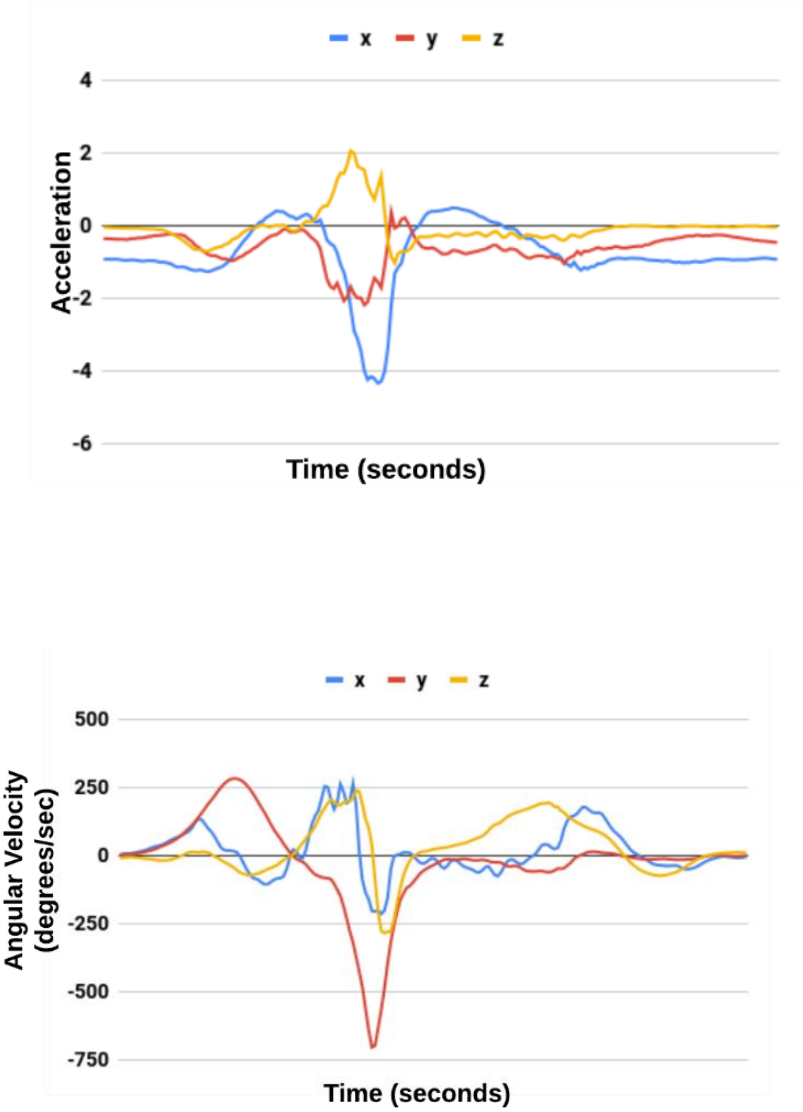


Figure 4. Accelerometer and gyroscope readings for a tennis swing

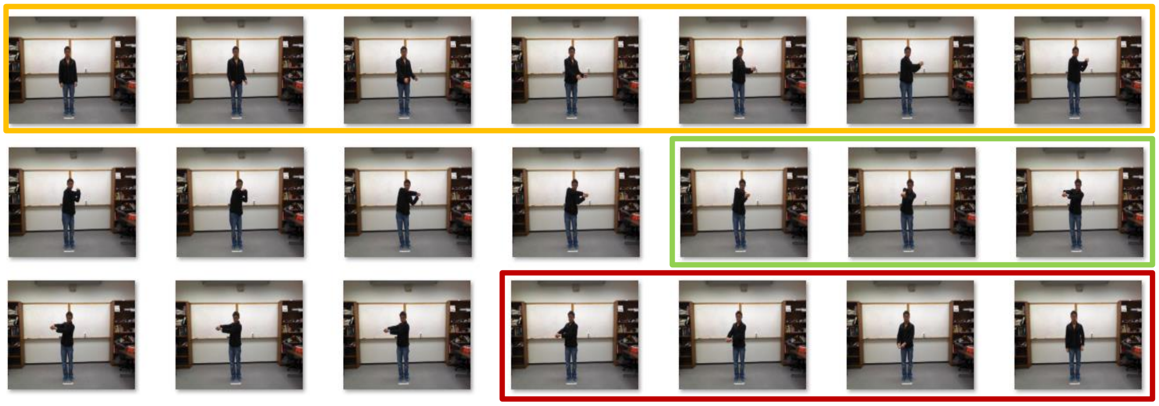


Figure 5. Images for sequence of motions from the dataset for a baseball swing

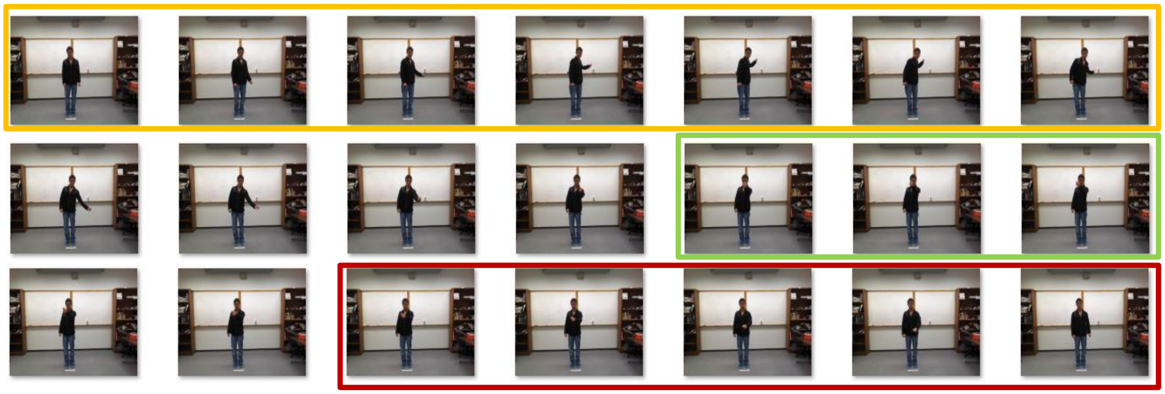


Figure 6. Images for sequence of motions from the dataset for a tennis swing

The purpose of Figure 5 and Figure 6 is to provide an understanding of the sequence of motions for closely related activities, a baseball swing and a tennis swing from the dataset [5]. The color coding for similar set of motion sequence is highlighted. It is to be noted that values of the acceleration and/or angular velocities are very close for these two activities at the above highlighted sequences.

CHAPTER THREE

DEEP LEARNING FOR HAR

Background

Traditional HAR methods require feature engineering and domain specific knowledge on the raw data before using it in machine learning or statistical models. These conventional techniques rely heavily on heuristic hand-crafted feature extraction. For example, for accelerometer data, the feature extraction could be based in time domain such as variance and mean or it could be in frequency domain such as the distribution of signal energy and amplitude. Identifying relevant features becomes time consuming and identifying complex activities becomes difficult [1] as the features extracted are based on mathematical operations rather than based on context. These methods put limitations on accuracy and require expertise in the respective field.

This is where deep learning based HAR has proved beneficial [13]. Deep learning models automatically learn the features required to make accurate predictions from the raw data directly. This enables new and large datasets to be used for HAR. Different types of sensor data can be used which results in efficient models and two or more sensor data can be combined as the model can adopt faster. These models are also capable of learning high level features which can be very well utilized in complex HAR.

Related Work

There have been several studies and methodologies adopted for human activity recognition. Some have used camera sensors for video-based analysis and some have used smartphone or body worn inertial sensors. Some of the machine learning and deep learning methods adopted in this direction are discussed in this section. Support Vector Machines (SVM) is a discriminative classifier which represents samples as points in space. The category of new points is determined based on the side of the optimal hyperplane. Hidden Markov Models (HMM) attempts to build a probabilistic description of the data space has been used in human activity recognition. Transitions among the states in the data space are governed by the transition probabilities. For a particular state, an outcome and not the state visible to an external observer is generated, according to the associated probability distribution. Some other methods include Stacked Autoencoders which consists of multiple layers of sparse autoencoders, Deep Belief Networks (DBN) which are a class of unsupervised pretrained networks which consists of hidden units connected between the layers but is disconnected with units within each layer, Restricted Boltzmann Machines (RBM) which are shallow, two-layer neural nets and the building blocks of deep belief networks. It is restricted because there is no intra-layer communication. Some other techniques used in researches include Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM) and models combining two or more methods such as DeepConvLSTM and combination of LSTM and CNN models.

Li et al. [14] used the sparse autoencoder by adding noise to the cost function and adding KL divergence, which improved the performance of HAR. Hammerla et al. [15] used CNN where they treated the 1D sensor data as an ID image for activity recognition. Anguita et al. [7] proposed a hardware friendly multiclass classification on smartphones using Support Vector Machines with fixed point classifications. Ravi et al. [13] adopted a 2D convolutional model using the smartphone sensor data. The concept of binary RBM was implemented by Lane et al. [17]. Kim et al. [20] used Hidden Markov Model to make a comparison analysis on concurrent and interleaved human activity recognition with the conditional random field approach for pattern discovery. Chen et al. [3] used online SVM and CT-PCA on smartphone sensor data where they designed a HAR system in terms of placement, orientation, and subject variations based on coordinate transformation. Kellokumpu et al. [2] implemented a discrete Hidden Markov Model on sequence of postures for activity recognition. Li et al. [8] adopted a hybrid model of CNN and LSTM for HAR and defined an evaluation framework to fix the stages of Activity Recognition Chain. Ordóñez et al. [1] further extended the HAR using a combination of deep convolutional and LSTM RNN networks.

CHAPTER FOUR

UNIDIRECTIONAL AND BIDIRECTIONAL LSTM

Unidirectional LSTM

Due to the time varying nature of actions, LSTM based models can capture the dynamic temporal variations for accurate sequence recognition and classifications. LSTMs can learn the context by themselves. This is highly suitable for HAR as the model can be trained to learn high level features and context on its own. The main advantage of LSTM over Recurrent Neural Network (RNN) is that it can remember the long-term time dependencies without the problem of vanishing or exploding gradients. LSTM is advantageous over HMMs, RNNs and other time series and/or sequence based learning models in various applications because of its insensitivity to gap length. In this project, the effectiveness of LSTM and BLSTM in applications involving recognition of closely related activities is explored and compared.

LSTM Cell

LSTM which is a variant of recurrent neural network has the capability to remember long term dependencies without the problem of vanishing gradients. LSTM was proposed by Sepp Hochreiter and Jürgen Schmidhuber in 1997 [9]. An LSTM layer consists of several memory blocks. These blocks are made up of internal gates (input, forget and output gates). LSTM cells which share the same input gate, forget gate [9,18] and output gate forms an LSTM block. The internal gates perform the read, write and erase operations for a block. The internal gates allow the model to be trained successfully using backpropagation through time which solves the problem of vanishing gradients.

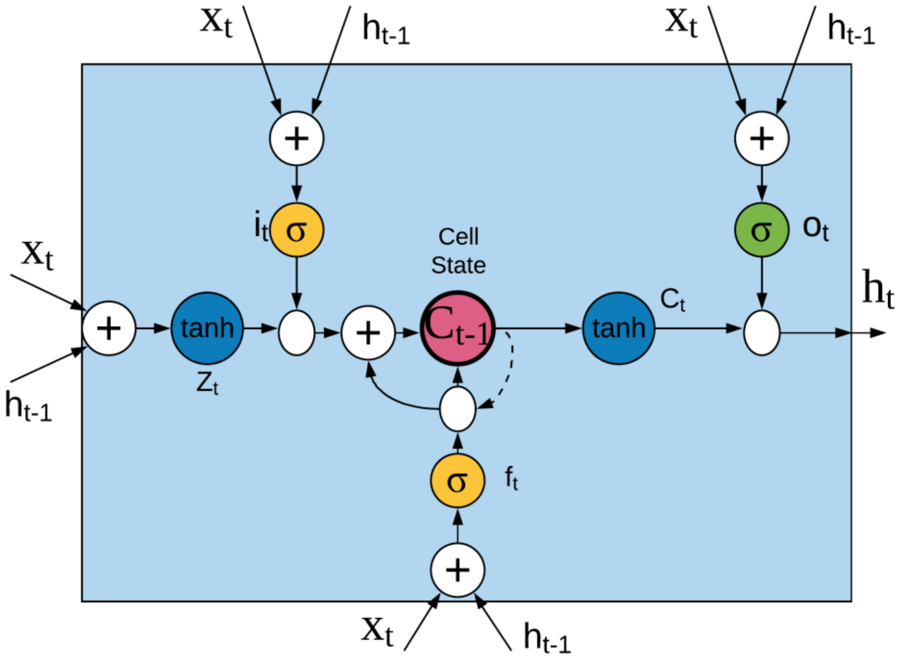


Figure 7. LSTM Cell

The equations (1) – (6) provide the working of the gates and memory state equations,

$$z_t = \tanh(W_z x_t + V_z h_{t-1} + b_z) \qquad (1)$$

$$i_t = \sigma(W_i x_t + V_i h_{t-1} + b_i) \qquad (2)$$

$$f_t = \sigma(W_f x_t + V_f h_{t-1} + b_f) \qquad (3)$$

$$o_t = \sigma\left(W_o x_t + V_o h_{t-1} + b_o\right) \quad (4)$$

$$c_t = c_{t-1} \circ f_t + i_t \circ z_t \quad (5)$$

$$h_t = o_t \circ \tanh\left(c_t\right) \quad (6)$$

$z_t, i_t, f_t, o_t, c_t, h_t$ are the input node, input gate, forget gate, output gate, memory state and hidden state at time t, respectively. W is the weight matrix for x which is the input, V is the weight matrix for hidden state of the previous cell. b denotes the bias for the corresponding cell state and gates and ∘ denotes the Hadamard product. σ and tanh are the sigmoid and hyperbolic tangent activation functions respectively.

Input node z_t is the new memory generated using the input x_t and the previous hidden state h_{t-1}.

Forget gate f_t holds the authority to determine the removal of information from the previous state after receiving it as input. It takes the decision of erasing the cell and is governed by a sigmoid function which keeps the input between 0 and 1.

Input gate i_t holds the authority to add new information from the current input to current cell state. These are governed by sigmoid and tanh functions. Input gate takes the decision of writing to the cell. Tanh layer creates a vector for new candidates to be added to the current cell state and the sigmoid layer decides which values are to be updated.

Memory state c_t is the final memory generated by taking the advice of the forget gate f_t and forgetting the past memory c_{t-1}. Also, it takes the advice of the input gate i_t and gates the input node z_t which is the new memory generated. The sum of these two results gives the memory state c_t.

Output gate o_t decides on what to output from the cell state which is done using the sigmoid function. The input lies between -1 and 1 because of the tanh function and this is multiplied with the output of sigmoid function. This allows to output only what is needed.

Bidirectional LSTM

The LSTM version of the bidirectional Recurrent Neural Network (BRNN) structure is called Bidirectional LSTM (BLSTM). BRNN was proposed by Mike Schuster and Kuldip K. Paliwal in 1997 [10] for eliminating various restrictions of RNNs. In BRNN, there are two different recurrent networks in forward and backward directions through the same input layer as shown in Figure 6. These two networks connect to the same output layer to generate output results. With this structure, both future and past information of sequential inputs in a time frame are evaluated without delay [10]. In this project, this concept is utilized for time series classification where the start of an activity and the end of an activity in reverse order are trained by receiving the information from the input layer.

A BRNN computes the backward hidden sequence h_f, the forward hidden sequence h_f and the output sequence y by repeating the backward layer from time = T to 1, the forward layer from time = 1 to T and then updates the output layer. H is the hidden layer function. For maintaining two hidden layers at any time, BRNN consumes twice as much memory space for its bias and weight parameters.

$$h_f = H\left(W_{xh_f} x_t + W_{h_f h_f} h_{f_{t+1}} + b_{h_f}\right) \qquad (7)$$

$$h_b = H\left(W_{xh_b} x_t + W_{h_b h_b} h_{b_{t+1}} + b_{h_b}\right) \qquad (8)$$

$$y_t = W_{y h_f} h_{f_t} + W_{y h_b} h_{b_t} + b_y \qquad (9)$$

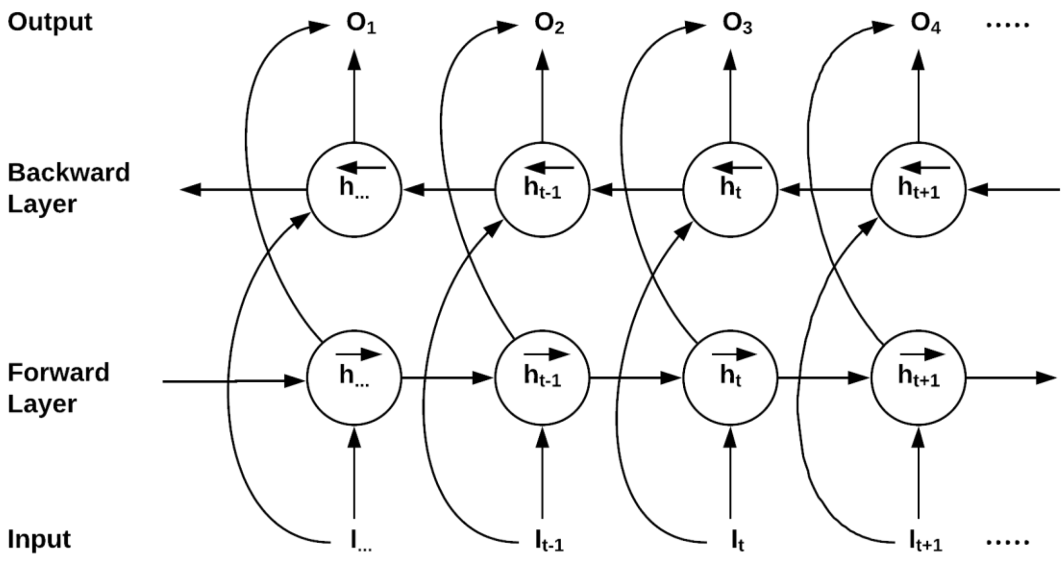


Figure 8. Bidirectional Recurrent Neural Network

1. Neurons in the forward state of BRNN are unidirectional. Training the network is same as of a regular RNN since both the networks are not connected to each other.

2. In the forward pass, for time t in 1≤t≤T, all of the input data is run via BRNNs and the outputs are predicted. Passes in forward (time from t =1 to t=T) and in backward (time from t =T to t=1) are finished. For the output neurons as well, a forward pass is finished.

3. In the backward pass, for time t in 1≤t≤T, the derivative of error function is calculated which is used in the forward pass. A backward pass is completed for both the forward states (from time t=T to t=1) and backward states (from time t=1 to t =T) and for the output neurons.

4. After this, all the weights are updated.

The LSTM version of this BRNN is BLSTM and it can show improvement over LSTM’s performance in classification processes. BLSTMs are capable of remembering the past and the future information as the model is trained in both forward and backward directions. In this project, this property of BLSTM is utilized and the BLSTM model is devised to access long-range context in both the directions. The experiment and evaluation are done on this model to show the improvement in the performance in recognizing closely related activities.

CHAPTER FIVE

METHODOLOGY

Data

The BLSTM model is trained and tested for recognizing closely related activities. The dataset used for this purpose is the publicly available UTD-MHAD [5] dataset, which provides wearable inertial sensor data (3-axis acceleration and 3-axis gyroscope for rotation signals) for 27 different activities in an indoor environment. The data recorded is from only one wearable inertial sensor data with a sampling rate of 50 Hz and a measuring range of ±8g for acceleration and ±1000 degrees/second for rotation.

The activities are draw triangle, bowling with right hand, swipe right, throw, arm cross, draw x, draw circle (clockwise), push, knock on door, jogging in place, sit to stand, stand to sit, forward lunge (left foot forward), squat (with two arms stretch out), walking in place, swipe left, right hand wave, two hand front clap, arm curl, basketball shoot, draw circle (anti-clockwise), front boxing, baseball swing from right, tennis forehand swing, tennis serve, catch an object and pick up an object. The first 15 activities were used for experimentation of 15 activities and the complete set of activities for experimentation of 27 activities. The activities are carried out by 8 persons (4 females and 4 males). The inertial sensor was worn on the person's right wrist and each action by a person has 4 trials. A total of 861 [(27 X 8 X 4) -3] sequences are derived from this, after removing 3 corrupt sequences.

Table 1. Set of 27 activities in the dataset [5]

Body worn inertial sensor on right wrist
Swipe left
Swipe right
Right hand wave
Two hand clap
Two hand push
Cross arms in the chest
Arm curl (two arms)
Draw x
Draw circle (clockwise)
Draw circle (anti-clockwise)
Draw triangle
Bowling
Front boxing
Baseball swing from right
Tennis forehand swing
Basketball shoot
Tennis serve
Throw
Knock on door
Catch an object
Pick up and throw
Body worn inertial sensor on right thigh
Jogging in place
Walking in place
Sit to stand
Stand to sit
Forward lunge (left foot forward)
Squat (with two arms stretched out)

Architecture

The application of the model is for activity recognition on a large pool of activities, 27 in this case and which are closely related to each other, meaning activities which either have similarity in their sequence of sensor data or appear similar to human eye. For example, swipe right and right-hand wave might appear similar to human eye and activities such as baseball swing and tennis swing have highly similar x, y and z axes accelerations. In order to classify such activities using deep learning approach, the model needs to be efficient to identify the minor differences in the motion of the activity.

In BLSTM, the data is trained in forward and backward directions in two separate hidden layers through the same input layer. For the model to accurately distinguish between similar activities, this property of BLSTM structure will provide with better results than other network structures. In this project, the BLSTM model has two layers, one Bidirectional LSTM layer and an output layer which is a dense layer, as shown in Figure 7. The first layer follows a many to many architecture. The output of all the cells in the first layer are used as the input to the dense layer. The dense layer for the output has a sigmoid activation.

The number of BLSTM cells in each layer is derived based on the number of data samples for each trial of the activity in the dataset. The number of cells is

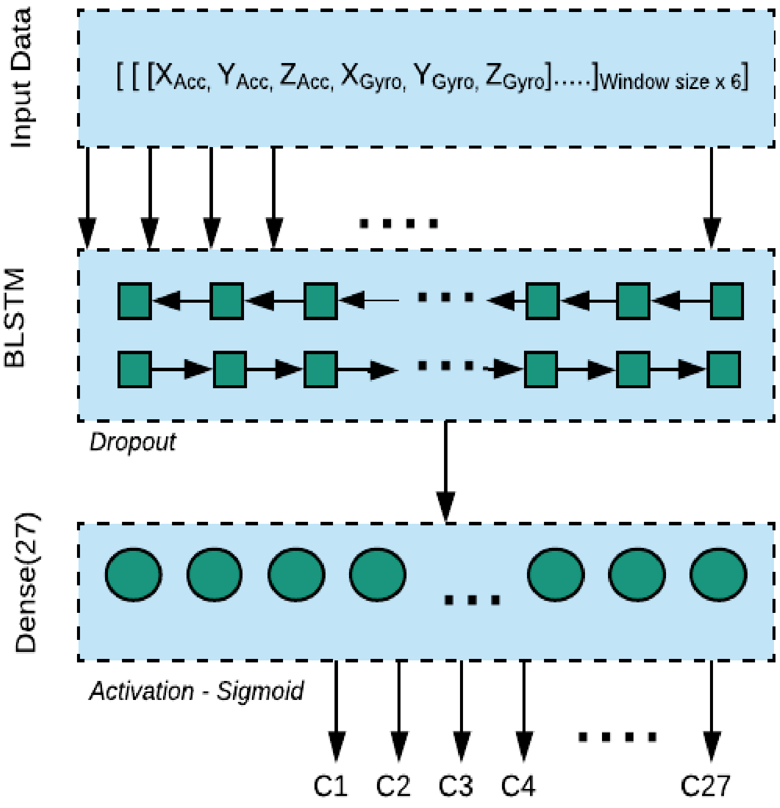

Input Data
[[[XAcc, YAcc, ZAcc, XGyro, YGyro, ZGyro].....]Window size x 6]
BLSTM
Dropout
Dense(27)
Activation - Sigmoid
C1
C2
C3
C4
C27

Figure 9. Architecture of the Bidirectional LSTM

kept fixed for both 15 set of activities and 27 set of activities. The number of sequences in the data increased to a larger number as the concept of sliding window is applied to the data. To prevent the model from overfitting, regularization technique of dropout was used.

The optimizations best suited in these cases were analyzed. RMSProp and Adam optimizations were used based on the size of the sample in consideration. RMSProp was efficient for sample size of 15 activities whereas Adam was suitable for a higher sample size of 27 activities of the dataset. It is observed that there is a need for change in the optimization as the size of the sample increases. Also, comparison of the final results of HAR is done between the BLSTM and LSTM model. Figure 8. gives the architecture of the LSTM model which is used for comparison with the BLSTM model.

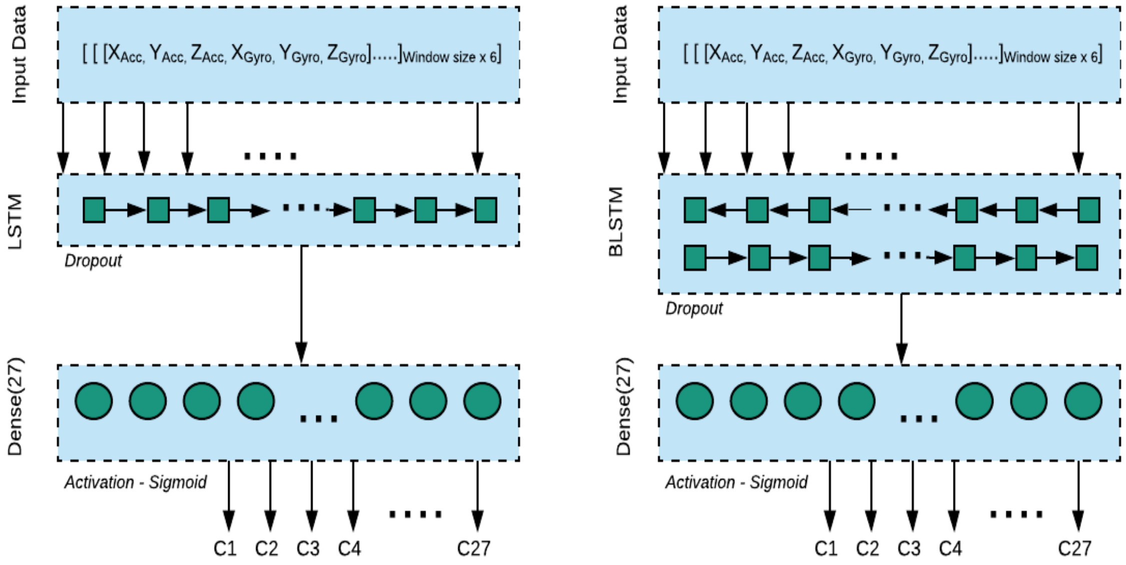

Input Data
[[[X_Acc, Y_Acc, Z_Acc, X_Gyro, Y_Gyro, Z_Gyro].....]Window size x 6]
LSTM
Dropout
Dense(27)
Activation - Sigmoid
C1
C2
C3
C4
C27
Input Data
[[[X_Acc, Y_Acc, Z_Acc, X_Gyro, Y_Gyro, Z_Gyro].....]Window size x 6]
BLSTM
Dropout
Dense(27)
Activation - Sigmoid
C1
C2
C3
C4
C27

Figure 10. BLSTM and LSTM models for comparison

Data Preparation

For the chosen dataset, training and experimentation is done for 15 activities and all of 27 activities. The reason for dividing the dataset as two different sets of activities is to enable deeper analysis of the BLSTM model. This allows for identifying the need for better methods and/or tuning of hyperparameters, if any with the increase in the complexities of the activities. Each of this data is divided into training (80%) and test set (20%). For test dataset, the dataset was tested based on subject specific and subject generic splitting of test data [21]. In subject specific test set, last two samples of each activity by each person was kept for testing. In this case, there is contribution of each person to the test data. In subject generic test dataset, the data of the last two persons was kept for testing. In this case, the contribution to the test data was by only last two persons whose entire activity samples were used for testing and the remaining six persons' entire activity samples were used for training.

Software Tools

The model was implemented in an Intel Core i7 machine with 16 GB memory using Tensor Flow 1.10.0 framework and the deep learning libraries Keras 2.2.2, Pandas 0.23.4, Numpy 1.14.5 and Scikit-Learn 0.19.2.

Sliding Window

One of the problems with the time series data is the unequal sequence length of each sample. In this case, the sequence length for each activity sample of a specific trial by each person are of unequal lengths. Usually, for each of the action trial window, the sequence data is normalized. These can be using statistical methods such as standard deviation, mean etc. But as discussed in earlier chapters, these methods require domain expertise and the aim is to train the model directly on raw data. Some of the other normalization techniques are truncating and padding. In truncating, the sequence is truncated so that all the sequences are of equal length. In padding, zeroes are added at the end of the

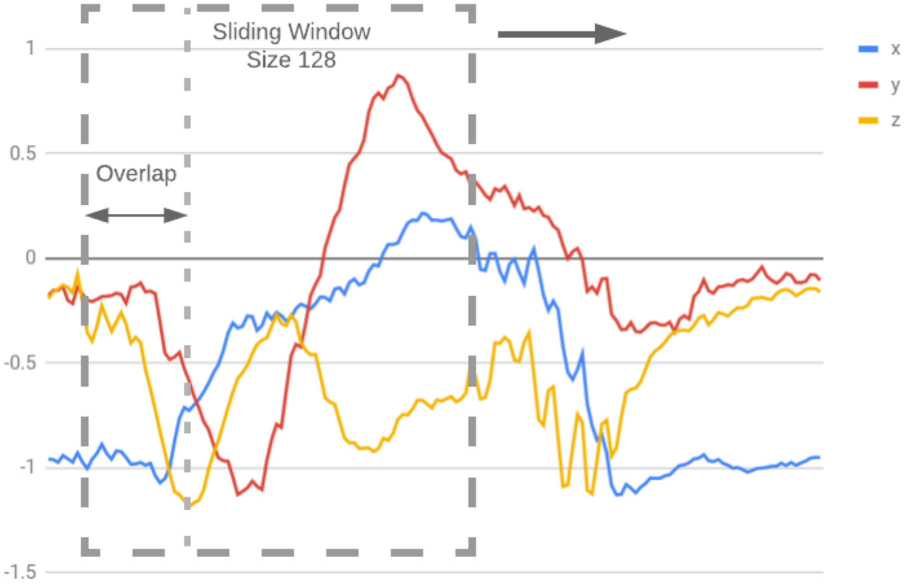


Figure 11. Sliding window of 128 timesteps

sequence so that all the sequences are of equal length. These techniques often lead to loss of temporal information. Hence, in this project the method of sliding window is adopted. The time series data is divided into several blocks. The sliding window moves to the next block which gets added to the sequence as shown in Figure 9. This maintains a fixed length of the sequence without the loss of temporal information.

If the time series data is given as,

$$(x_0,\ x_1,\ x_2,\ \ldots,\ x_{n\text{-}1},\ x_n,\ x_{n+1},\ \ldots) \qquad (10)$$

When the window size is fixed at k, the data interval becomes,

$$(x_{i-k},\ x_{i-k+1},\ \ldots,\ x_i,\ x_{i+1}) \qquad (11)$$

In this project, the shape of the input vector is N x W x 6, where W is the window size which is kept as 128, N is the total no. of windows calculated for the entire sample space and 6 is the x, y, z axes readings of accelerations and angular velocities from accelerometer and gyroscope respectively. Since, the sampling rate of the inertial sensor data in the dataset is 50 Hz, the time interval between two successive data points is 0.02s.

Activation

The activation used in this BLSTM model is the sigmoid activation function which is a logistic function. The model has the sigmoid activation function in the final output layer, which is the dense layer. Sigmoid activation function which is shown in Figure 10, is real-valued and differentiable which makes it capable of finding the gradients. It is the mathematical representation of a behavior of a biological neuron where the case of neuron firing or not, is indicated by its output. Based on the experimentation with different activation functions, sigmoid was the best one for this model. It is given by the equation,

$$f(x) = \frac{1}{\left(1+e^{-x}\right)} \qquad (12)$$

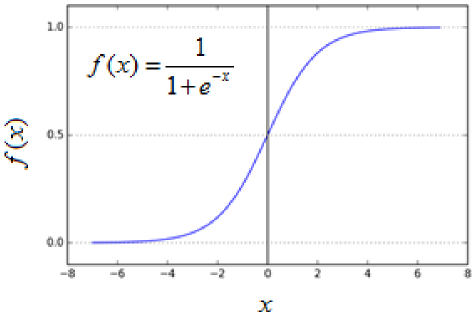


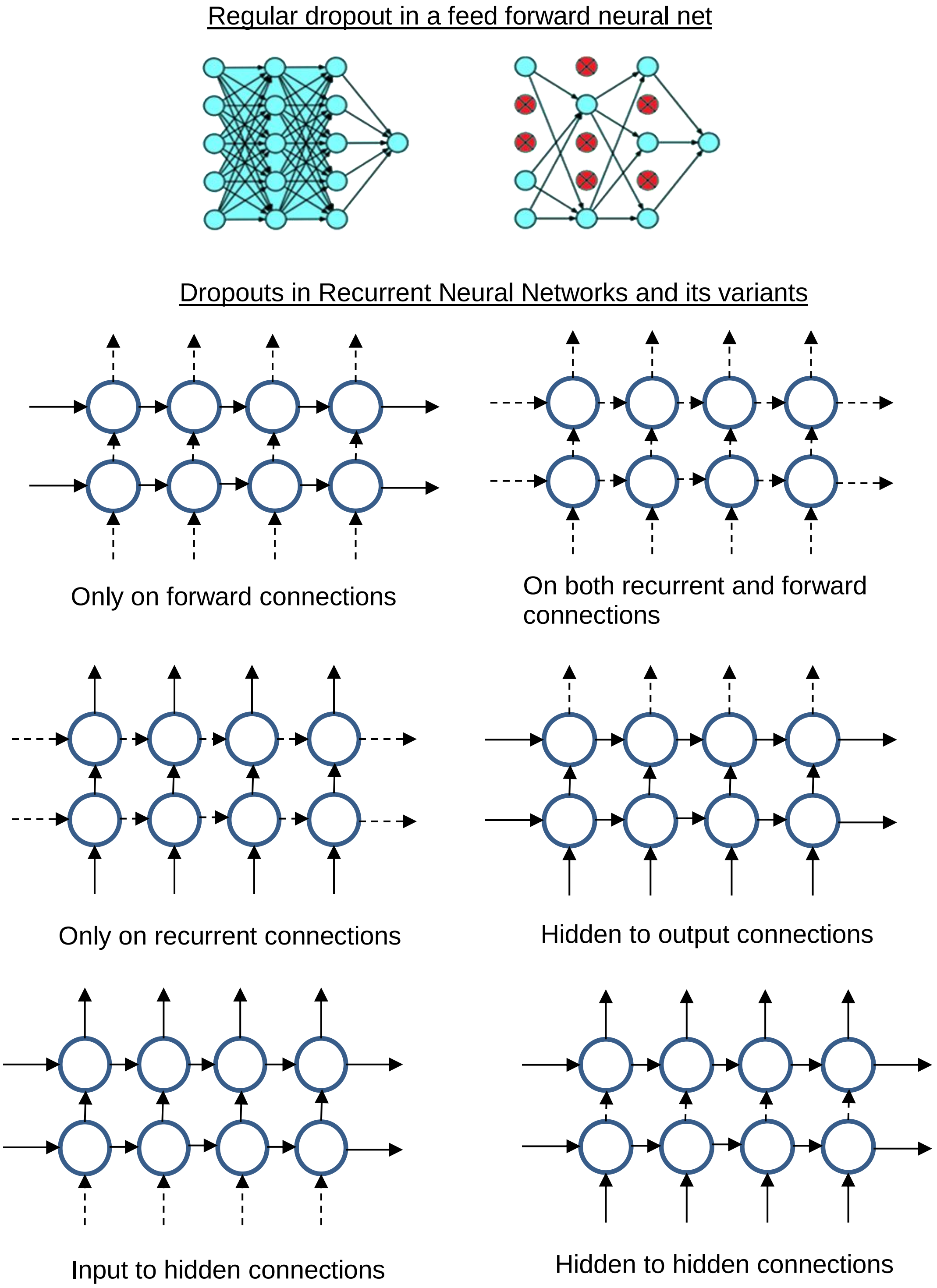
Figure 12. Sigmoid activation function

Regularization

The regularization used for this BLSTM model is dropout. Dropout is a regularization technique where randomly selected neurons are dropped out during training. It prevents the complex co-adaptations on training data [11]. Drop out reduces the chances of overfitting and has provided improvements on several difficult problems, such as in speech and image recognition [10,11].

Variations of dropout for LSTM have been suggested in the past. This includes, instead of applying dropout to the forward connections, the dropout is applied to the recurrent connections or a combination of both. Zaremba et al. [22] suggested using dropout in RNNs only in the non-recurrent connections. They experimented it for speech recognition, machine translation and language modeling. Gal et al. [23] proposed a recurrent dropout called variational dropout where the same dropout mask at each timestep is applied in the recurrent and forward connections. Moon et al. [24] proposed a recurrent dropout where the dropout at recurrent connections is applied at the cell states. Semenuita et al. [25] proposed a recurrent dropout where the dropout at recurrent connections is applied at hidden state update vectors. They further analyzed the sampling of dropout mask that is, once per sequence or once per time step.

In the BLSTM model of this project, the regular dropout or simply, dropout which is applied in forward connections is applied between the BLSTM layer and the output layer which is a dense layer, as the number of sequences becomes larger than the original sample sequence because of the sliding window. Further experimentation was done by combining this dropout of BLSTM hidden layer to output layer with a recurrent dropout in the BSTM layer. It is observed that the regular dropout, that is dropout in forward connections between the BLSTM layer and output was more effective than the combination of this dropout with the recurrent dropout. Therefore, the regular dropout between the BLSTM layer and output layer is used. This leads to significantly lower generalization error.



Regular dropout in a feed forward neural net

Dropouts in Recurrent Neural Networks and its variants

Only on forward connections

On both recurrent and forward connections

Only on recurrent connections

Hidden to output connections

Input to hidden connections

Hidden to hidden connections

Figure 13. Overview of regular and recurrent dropouts

Figure 13 above shows the regular dropout in a standard feed forward network and the use of regular dropout as well as recurrent dropout in recurrent neural networks and its variants. For simplicity instead of BLSTM, an RNN structure with an input layer, two layers of RNN and an output layer is shown. The solid lines depict no dropout in that connection whereas dotted lines depict dropout applied in that connection. The horizontal arrows are for recurrent connections and the vertical arrows are for forward connections. Input to hidden, hidden to output, hidden to hidden connections depicts the regular dropout applied between these layers, where hidden is the RNN layer.

Loss

Categorical Cross-Entropy

The loss function used in this model is the categorical cross-entropy loss. A loss function states the loss in predicting the outcome with the desired or true output. The objective in the training is to minimize the loss across the training iterations. The categorical cross-entropy loss is used when a probabilistic interpretation of the scores is desired. It measures the dissimilarity between the predicted label distribution and the true label distribution. It is given by,

$$-\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C}1_{y_i \in C_c}\log p_{model}\left[y_i \in C_c\right] \qquad (13)$$

where, the summation is over the observations denoted by i, and is N in number, and the categories c, which is C in number. The indicator function, $1_{y_i \in C_c}$ is for the observation i which belongs to the c category. The term $p_{model}\left[y_i \in C_c\right]$ is the probability predicted by the model for the observation i, to belong to the c category. The model outputs a vector of C probabilities, when there are more than two categories, each giving the probability that the input should be classified to the respective category.

Optimization

RMSProp

Per-parameter learning rates are maintained by RMSProp which are adapted on the basis of the average of recent magnitudes of the gradients for the weight. It is suitable where the weights change at a fast rate [16]. In this model, RMSProp works well for 15 activities. RMSProp divides the learning rate by an exponentially decaying average of squared gradients. RMSProp automatically decreases the size of the gradient steps towards minima when the steps are too large. The update equations for RMSProp given by the equations,

$$v_t = \gamma\ v_t + (1\text{-}\gamma)\ g_t^2 \qquad (14)$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{v_t + \varepsilon}}\ g_t \qquad (15)$$

where η is the initial learning rate and a good default is value 0.001, g_t is the gradient at time t, v_t is the exponentially decaying average of past squared gradients, ε is used to avoid ending up with a division by zero and γ which is the decay parameter and is generally set to 0.9.

Adam

Adam or Adaptive Moment Estimation optimization is based on adaptive estimates of lower-order moments [19] and works well for non-stationary objectives. In this case, as the activities were increased from 15 to 27, Adam works better than RMSProp. For a larger dataset, as seen in this project, Adam suits better than RMSProp. Adam keeps an exponentially decaying average of past gradients, m_t along with exponentially decaying average of past squared gradients v_t [16]. m_t and v_t are initialized as 0 vector because of which they are biased towards 0. The Adam update rule is given by the below equation 22, where $\bar{m}_t$ and $\bar{v}_t$ are bias-corrected first and second moment estimates [16] respectively.

First moment of gradients,

$$m_t = \beta_1 m_{t-1} + \left(1 - \beta_1\right) g_t \qquad (16)$$

Second moment of gradients,

$$v_t = \beta_2 v_{t-1} + \left(1 - \beta_2\right) g^2{}_t \qquad (17)$$

First moment bias correction,

$$\overline{m}_t = \frac{m_t}{1\text{-}\beta_1^{\,t}} \tag{18}$$

Second moment bias correction,

$$\overline{v}_t = \frac{v_t}{1\text{-}\beta_2^{\,t}} \tag{19}$$

Update rule,

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\overline{v}_t} + \varepsilon}\,\overline{m}_t \tag{20}$$

β_1, β_2 are the decay rates with values close to 1. $\beta_1$is usually kept around 0.9 while β_2 is kept at 0.99.

CHAPTER SIX

RESULTS AND OBSERVATIONS

Accuracy

The BLSTM model achieved an overall accuracy of 98.05% for 15 activities and 90.87% for 27 activities on the subject specific test dataset. Fig. 6 depicts the accuracy comparisons for the two set of activities on subject generic and subject specific test dataset with two different models, BLSTM and LSTM.

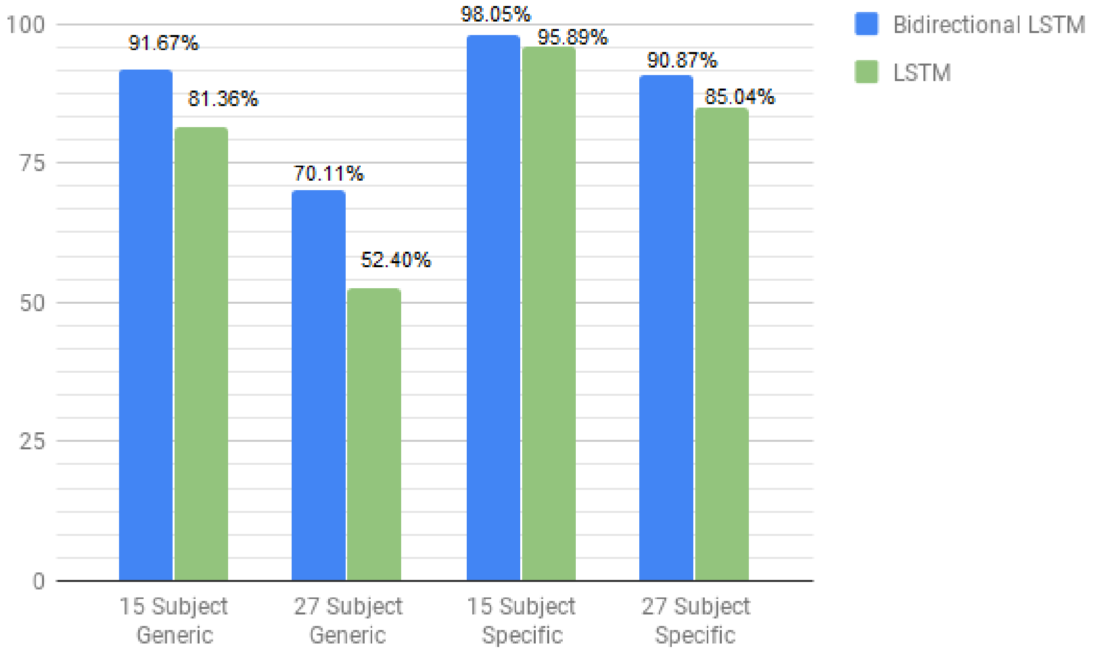


Figure 14. Accuracy of BLSTM vs LSTM

It is to be noted that as the sample size was increased from 15 to 27 activities, the performance of LSTM decreased from 81.36% to 52.40% for subject generic. It is observed that as the sample size increases from 15 to 27 activities, the BLSTM performs better than LSTM. The difference in accuracy between BLSTM and LSTM for 15 activities in case of subject specific is 2.16% whereas in case of 27 activities the difference increases to 5.83%.

Recall, Precision and F1 Score

Recall is the true positive rate and gives the measure of number of activities correctly identified as positive out of the total true positives, $\frac{\text{True Positive}}{\text{True Positive + False Negative}}$

Precision is the measure of number of items correctly identified as positive out of total items identified as positive, $\frac{\text{True Positive}}{\text{True Positive + False Positive}}$

F1 score is the measure of balance between recall and precision. It is the harmonic mean of recall and precision, $\frac{2 * \text{Recall} * \text{Precision}}{\text{Recall + Precision}}$.

Table 2. depicts these values in percentage for all the different set of activities in LSTM and BLSTM model in subject specific and subject generic test data. As can be observed, BLSTM in subject specific gives the best result of 97.74% and 90.34% for 15 and 27 activities respectively.

Table 2. Recall, precision and F1 score for different combinations of the model

	Mean Recall %	Mean Precision %	Mean F1 score%
15 Activities LSTM Subject Generic	82.54	82.83	78.98
15 Activities BLSTM Subject Generic	90.99	91.81	90.92
15 Activities LSTM Subject Specific	95.87	95.25	95.45
15 Activities BLSTM Subject Specific	97.90	97.67	**97.74**
27 Activities LSTM Subject Generic	53.84	55.73	54.77
27 Activities BLSTM Subject Generic	72.03	71.17	69.55
27 Activities LSTM Subject Specific	84.90	83.72	83.78
27 Activities BLSTM Subject Specific	90.65	90.58	**90.34**

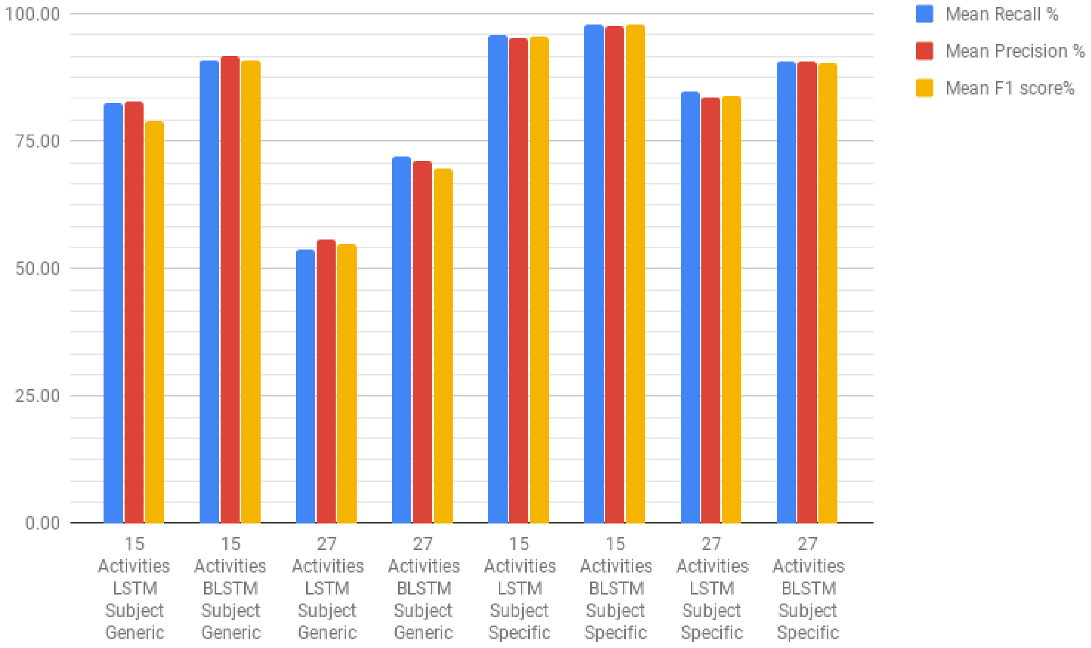
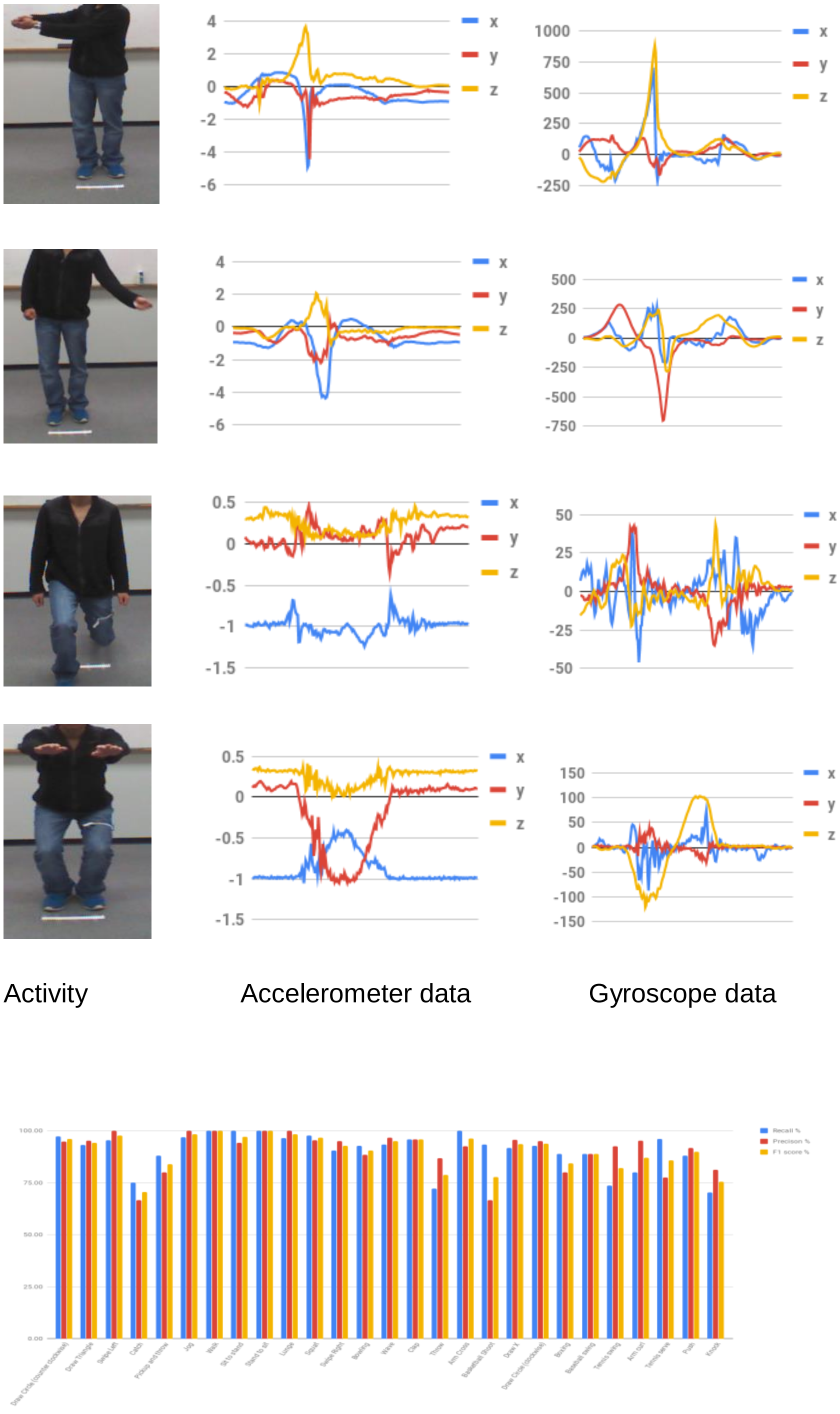


Figure 15. Recall, precision and F1 score for different combinations of the model



Activity

Accelerometer data

Gyroscope data

Results

Figure 16. Activity [5], input (data) and output (results)

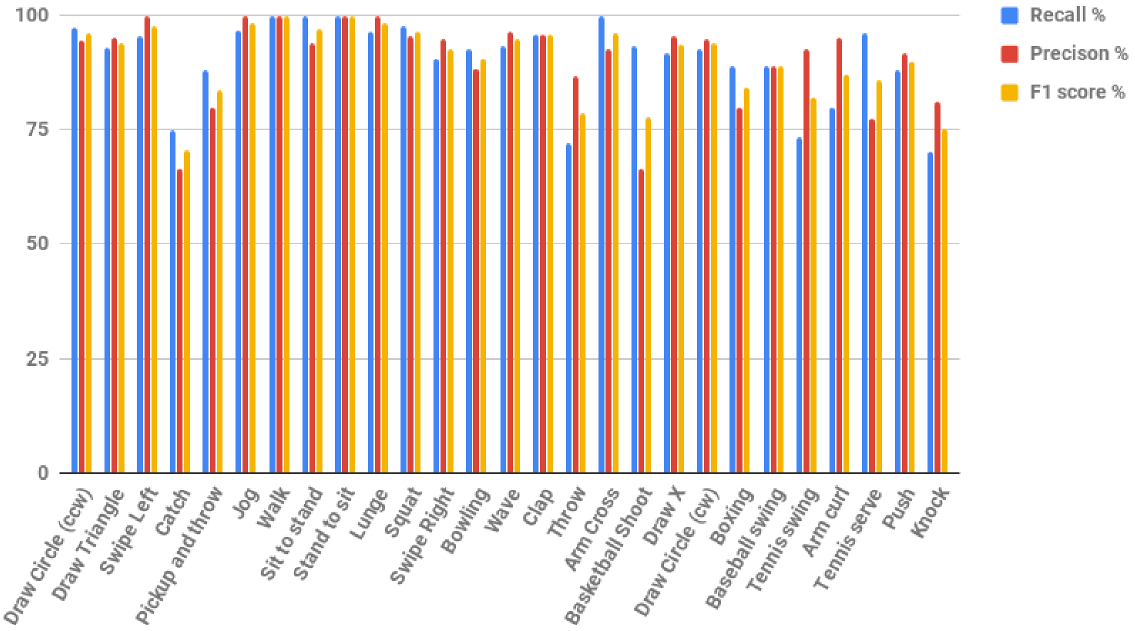


Figure 17. Accuracy metrics for BLSTM model on subject specific test dataset for 27 activities

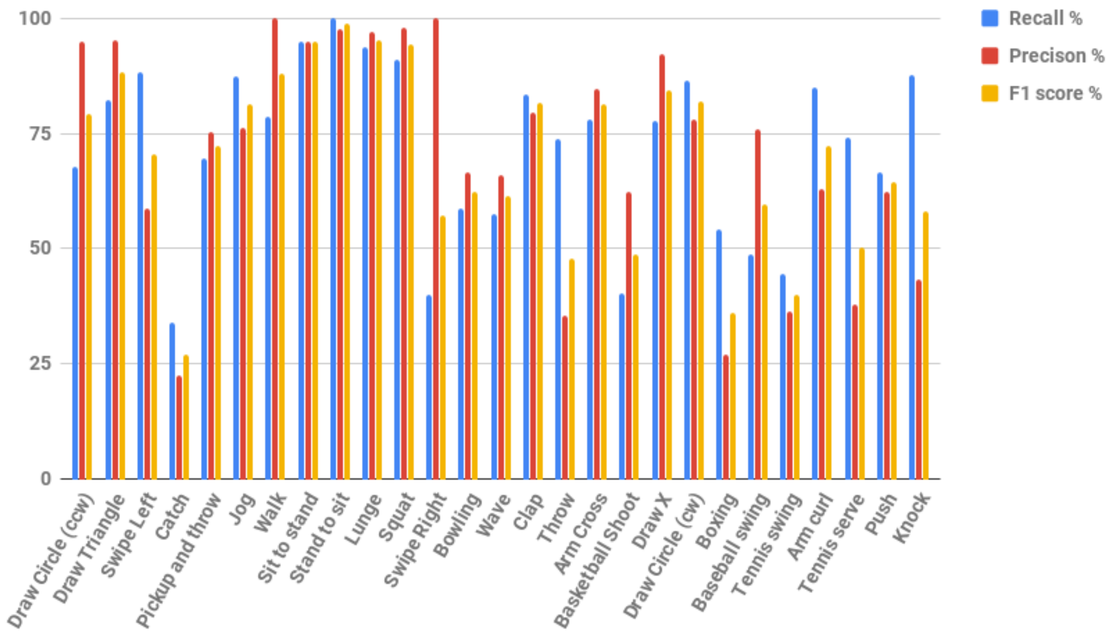


Figure 18. Accuracy metrics for BLSTM model on subject generic test dataset for 27 activities

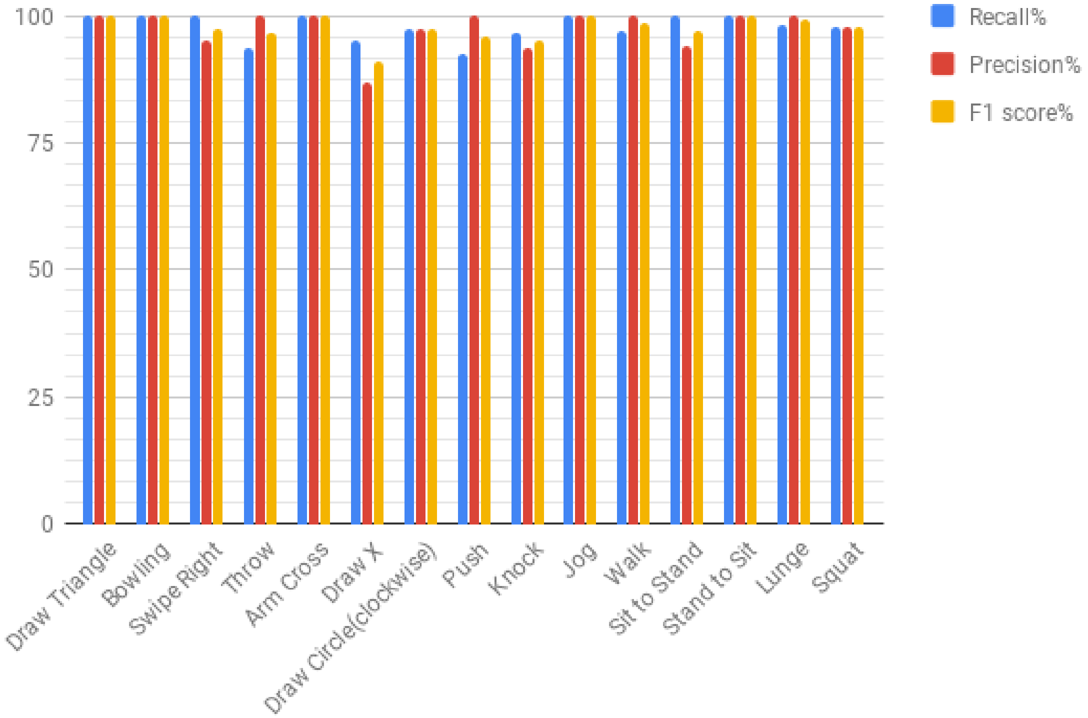


Figure 19. Accuracy metrics for BLSTM model on subject specific test dataset for 15 activities

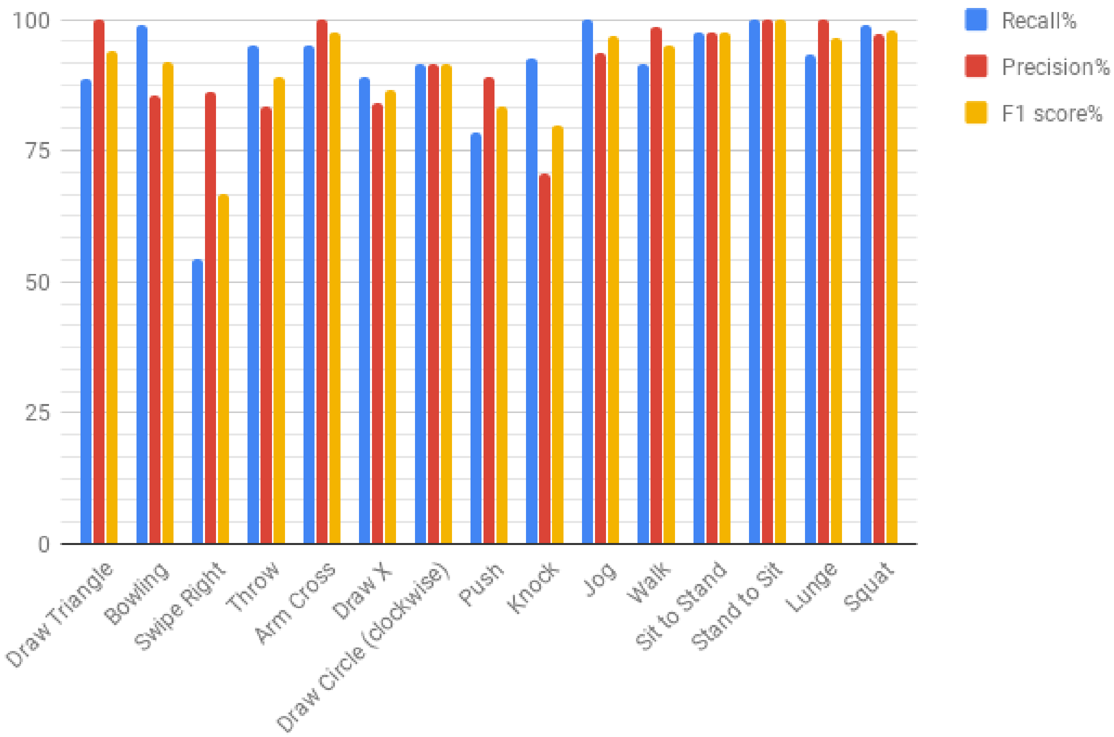
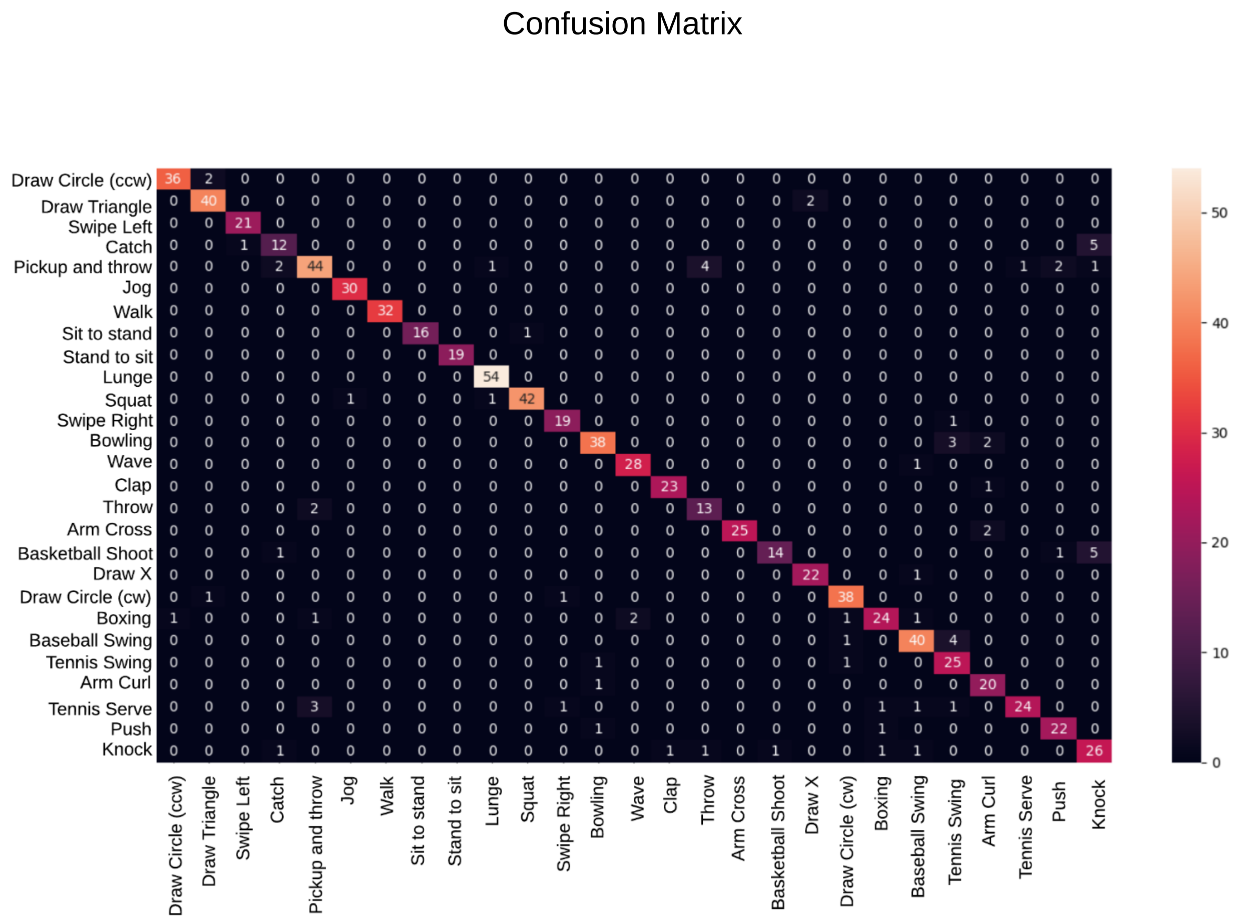
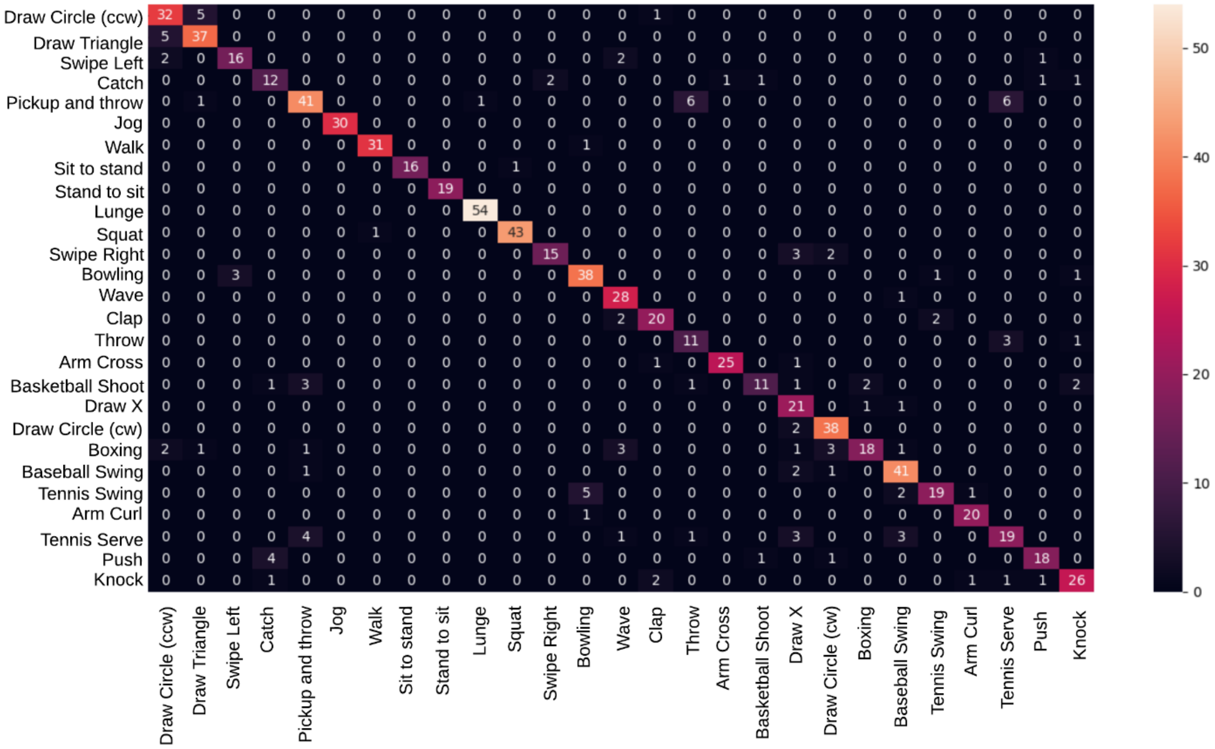


Figure 20. Accuracy metrics for BLSTM model on subject generic test dataset for 15 activities



	Draw Circle (ccw)	Draw Triangle	Swipe Left	Catch	Pickup and throw	Jog	Walk	Sit to stand	Stand to sit	Lunge	Squat	Swipe Right	Bowling	Wave	Clap	Throw	Arm Cross	Basketball Shoot	Draw X	Draw Circle (cw)	Boxing	Baseball Swing	Tennis Swing	Arm Curl	Tennis Serve	Push	Knock
Draw Circle (ccw)	36	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
Draw Triangle	0	40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0
Swipe Left	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
Catch	0	0	1	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5
Pickup and throw	0	0	0	2	44	0	0	0	0	1	0	0	0	0	0	4	0	0	0	0	0	0	0	0	1	2	1
Jog	0	0	0	0	0	30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Walk	0	0	0	0	0	0	32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Sit to stand	0	0	0	0	0	0	0	16	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Stand to sit	0	0	0	0	0	0	0	0	19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Lunge	0	0	0	0	0	0	0	0	0	54	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Squat	0	0	0	0	0	1	0	0	0	1	42	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Swipe Right	0	0	0	0	0	0	0	0	0	0	0	19	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
Bowling	0	0	0	0	0	0	0	0	0	0	0	0	38	0	0	0	0	0	0	0	0	0	3	2	0	0	0
Wave	0	0	0	0	0	0	0	0	0	0	0	0	0	28	0	0	0	0	0	0	0	1	0	0	0	0	0
Clap	0	0	0	0	0	0	0	0	0	0	0	0	0	0	23	0	0	0	0	0	0	0	0	1	0	0	0
Throw	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	13	0	0	0	0	0	0	0	0	0	0	0
Arm Cross	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	25	0	0	0	0	0	0	2	0	0	0
Basketball Shoot	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	14	0	0	0	0	0	0	0	1	5
Draw X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	22	0	0	1	0	0	0	0	0
Draw Circle (cw)	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	38	0	0	0	0	0	0	0
Boxing	1	0	0	0	1	0	0	0	0	0	0	0	0	2	0	0	0	0	0	1	24	1	0	0	0	0	0
Baseball Swing	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	40	4	0	0	0	0
Tennis Swing	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	25	0	0	0	0
Arm Curl	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	20	0	0	0
Tennis Serve	0	0	0	0	3	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	1	0	24	0	0
Push	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	22	0
Knock	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	1	0	0	1	1	0	0	0	0	26

Figure 21. Confusion matrix for 27 activities in BLSTM subject specific



	Draw Circle (ccw)	Draw Triangle	Swipe Left	Catch	Pickup and throw	Jog	Walk	Sit to stand	Stand to sit	Lunge	Squat	Swipe Right	Bowling	Wave	Clap	Throw	Arm Cross	Basketball Shoot	Draw X	Draw Circle (cw)	Boxing	Baseball Swing	Tennis Swing	Arm Curl	Tennis Serve	Push	Knock
Draw Circle (ccw)	32	5	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
Draw Triangle	5	37	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
Swipe Left	2	0	16	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	1	0
Catch	0	0	0	12	0	0	0	0	0	0	0	2	0	0	0	0	1	1	0	0	0	0	0	0	0	1	1
Pickup and throw	0	1	0	0	41	0	0	0	0	1	0	0	0	0	0	6	0	0	0	0	0	0	0	0	6	0	0
Jog	0	0	0	0	0	30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Walk	0	0	0	0	0	0	31	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Sit to stand	0	0	0	0	0	0	0	16	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Stand to sit	0	0	0	0	0	0	0	0	19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Lunge	0	0	0	0	0	0	0	0	0	54	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Squat	0	0	0	0	0	0	1	0	0	0	43	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Swipe Right	0	0	0	0	0	0	0	0	0	0	0	15	0	0	0	0	0	0	3	2	0	0	0	0	0	0	0
Bowling	0	0	3	0	0	0	0	0	0	0	0	0	38	0	0	0	0	0	0	0	0	0	1	0	0	0	1
Wave	0	0	0	0	0	0	0	0	0	0	0	0	0	28	0	0	0	0	0	0	0	1	0	0	0	0	0
Clap	0	0	0	0	0	0	0	0	0	0	0	0	0	2	20	0	0	0	0	0	0	0	2	0	0	0	0
Throw	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	11	0	0	0	0	0	0	0	0	3	0	1
Arm Cross	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	25	0	1	0	0	0	0	0	0	0	0
Basketball Shoot	0	0	0	1	3	0	0	0	0	0	0	0	0	0	0	1	0	11	1	0	2	0	0	0	0	0	2
Draw X	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	21	0	1	1	0	0	0	0	0
Draw Circle (cw)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	38	0	0	0	0	0	0	0
Boxing	2	1	0	0	1	0	0	0	0	0	0	0	0	3	0	0	0	0	1	3	18	1	0	0	0	0	0
Baseball Swing	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	2	1	0	41	0	0	0	0	0
Tennis Swing	0	0	0	0	0	0	0	0	0	0	0	0	5	0	0	0	0	0	0	0	0	2	19	1	0	0	0
Arm Curl	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	20	0	0	0
Tennis Serve	0	0	0	0	4	0	0	0	0	0	0	0	0	1	0	1	0	0	3	0	0	3	0	0	19	0	0
Push	0	0	0	4	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	18	0
Knock	0	0	0	1	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	0	1	1	1	26

Figure 22. Confusion matrix for 27 activities in LSTM subject specific

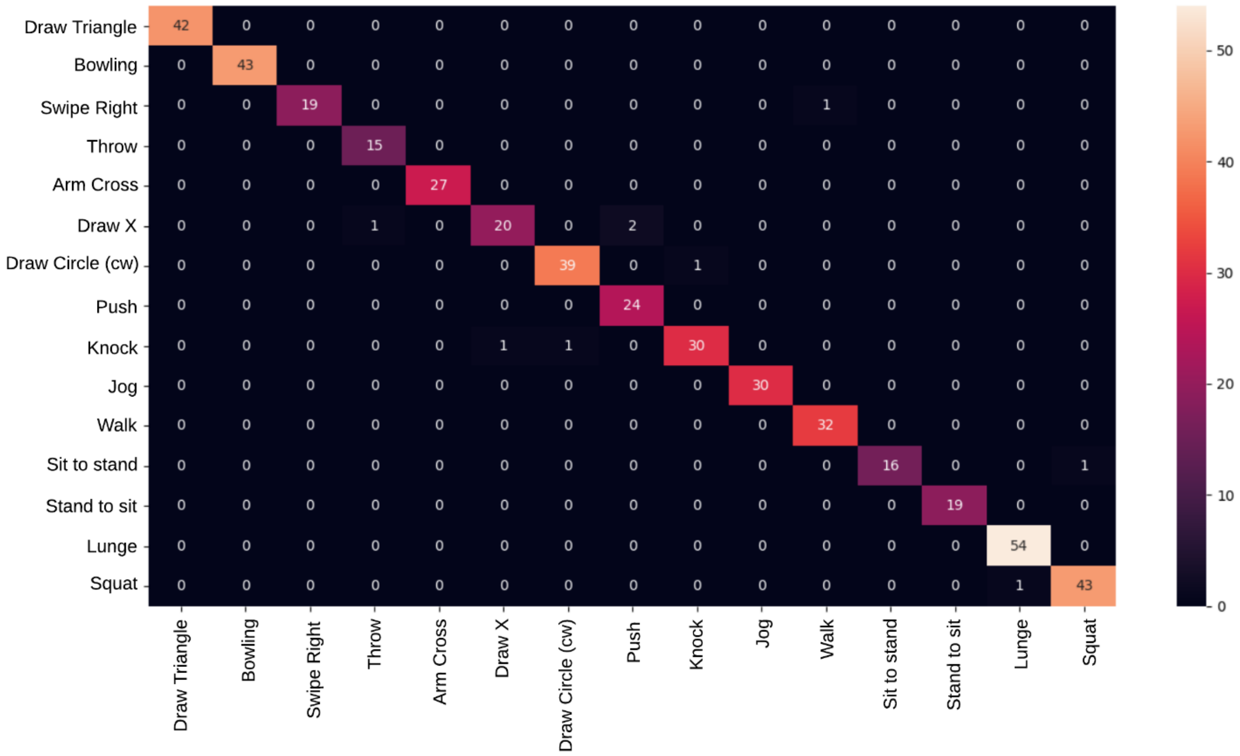


Figure 23. Confusion matrix for 15 activities in BLSTM subject specific

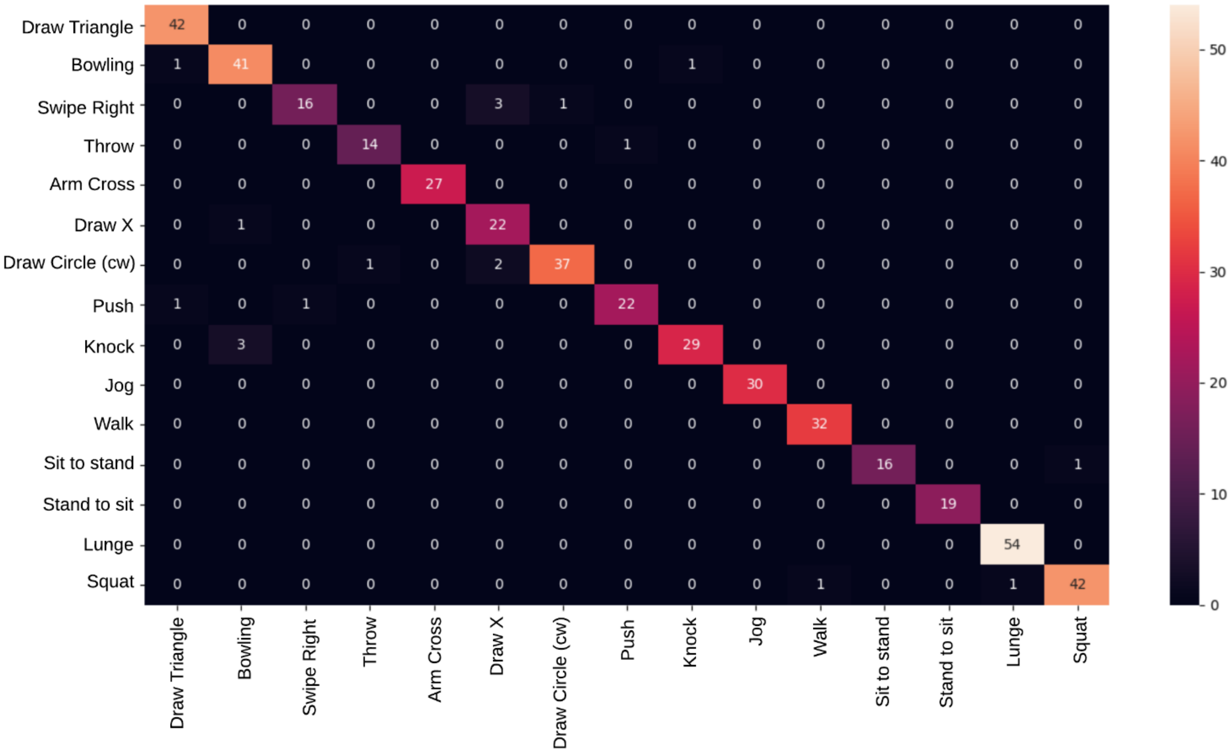


Figure 24. Confusion matrix for 15 activities in LSTM subject specific

CHAPTER SEVEN

CONCLUSIONS

Bidirectional LSTM is efficient to work directly on raw data from body worn inertial sensor. It yields results with good accuracy in a time series classification task. The BLSTM model of this project could achieve an accuracy of 98.05% and 90.87% for 15 and 27 activities, respectively. BLSTM is suitable for human activity recognition. On an average, majority of the 27 activities had a F1 mean score of 90%. This model which uses a large pool of activities is capable of distinguishing between closely related activities. This study observed that BLSTM yields results with better accuracy than LSTM for HAR on closely related activities.

REFERENCES

[1] Ordóñez, F.J.; Roggen, D. Deep Convolutional and LSTM Recurrent Neural Networks for Multimodal Wearable Activity Recognition. Sensors 2016, *16*, 115.

[2] Kellokumpu, Vili & Pietikäinen, Matti & Heikkilä, Janne. Human Activity Recognition Using Sequences of Postures. IAPR MVA 2005, 570-573.

[3] Z. Chen, Q. Zhu, Y. C. Soh and L. Zhang, "Robust Human Activity Recognition Using Smartphone Sensors via CT-PCA and Online SVM," in IEEE Transactions on Industrial Informatics, vol. 13, no. 6, pp. 3070-3080, Dec. 2017.

[4] Wang, Jindong & Chen, Yiqiang & Hao, Shuji & Peng, Xiaohui & Lisha, Hu. (2017). Deep Learning for Sensor-based Activity Recognition: A Survey. Pattern Recognition Letters. 10.1016/j.patrec.2018.02.010.

[5] C. Chen, R. Jafari, and N. Kehtarnavaz, (2015). UTD-MHAD: A Multimodal Dataset for Human Action Recognition Utilizing a Depth Camera and a Wearable Inertial Sensor, Proceedings of IEEE International Conference on Image Processing. 10.1109/ICIP.2015.7350781.

[6] L. Chen, J. Hoey, C. D. Nugent, D. J. Cook and Z. Yu, "Sensor-Based Activity Recognition," in IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), vol. 42, no. 6, pp. 790-808, Nov. 2012.

[7] Anguita D., Ghio A., Oneto L., Parra X., Reyes-Ortiz J.L. (2012) Human Activity Recognition on Smartphones Using a Multiclass Hardware-Friendly Support Vector Machine. Springer, Berlin, Heidelberg.

[8] Li, F., Shirahama, K., Nisar, M., Köping, L., & Grzegorzek, M. (2018). Comparison of Feature Learning Methods for Human Activity Recognition Using Wearable Sensors. Sensors, 18(3), 679. doi:10.3390/s18020679.

[9] Hochreiter, Sepp & Schmidhuber, Jürgen. (1997). Long Short-term Memory. Neural computation. 9. 1735-80. 10.1162/neco.1997.9.8.1735.

[10] Baldi, P., & Sadowski, P. (2014). The Dropout Learning Algorithm. Artificial intelligence, 210, 78-122.

[11] Hinton G, Srivastava N, Krizhevsky A, Sutskever I, Salakhutdinov RR. Improving neural networks by preventing co-adaptation of feature detectors. 2012

[12] Schuster, Mike & Paliwal, Kuldip K. Bidirectional recurrent neural networks in IEEE Transactions on Signal Processing, 1997.

[13] N. Y. Hammerla, S. Halloran, T. Ploetz, "Deep convolutional and recurrent models for human activity recognition using wearables", 2016, [online] Available: http://arxiv.org/abs/1604.08880.

[14] Li, Y., Shi, D., Ding, B., Liu, D., 2014. Unsupervised feature learning for human activity recognition using smartphone sensors, in: Mining Intelligence and Knowledge Exploration. Springer, pp. 99–107.

[15] Ravi, D., Wong, C., Lo, B., Yang, G.Z., 2016. Deep learning for human activity recognition: A resource efficient implementation on low-power devices. 2016 IEEE 13th International Conference on, IEEE. pp. 71–76.

[16] Sebastian Ruder (2016). An overview of gradient descent optimisation algorithms. arXiv preprint arXiv:1609.04747.

[17] Lane, N.D., Georgiev, P., Qendro, L., 2015. Deepear: robust smartphone audio sensing in unconstrained acoustic environments using deep learning, in: UbiComp, ACM. pp. 283–294.

[18] Gers, Felix A. and Schmidhuber, Jurgen and Cummins, Fred A.. Learning to Forget: Continual Prediction with LSTM Neural Computation,2000, Vol. 12, pp. 2451-2471.

[19] Kingma, Diederik & Ba, Jimmy. (2014). Adam: A Method for Stochastic Optimization. International Conference on Learning Representations.

[20] Kim, E., Helal, S., & Cook, D. (2010). Human Activity Recognition and Pattern Discovery. IEEE pervasive computing, 9(1), 48.

[21] C. Chen, R. Jafari and N. Kehtarnavaz, "A Real-Time Human Action Recognition System Using Depth and Inertial Sensor Fusion," in *IEEE Sensors Journal*, vol.16, no.3, pp.773-781, Feb.1,2016. doi: 10.1109/JSEN.2015.2487358

[22] Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals, Recurrent neural network regularization. arXiv preprint arXiv:1409.2329, 2014.

[23] Gal, Y., & Ghahramani, Z. (2016). A Theoretically Grounded Application of Dropout in Recurrent Neural Networks. *NIPS*.

[24] Moon, T., Choi, H., Lee, H., & Song, I. (2015). RNNDROP: A novel dropout for RNNS in ASR. *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 65-70.

[25] Semeniuta, S., Severyn, A., & Barth, E. (2016). Recurrent Dropout without Memory Loss. *COLING*.