**COSC 3337 Task 3: Human Activity Detection using**

**LSTM Neural Networks**

Robert Duque

1582182



LSTM stands for Long Short-Term Memory which is a neural network consisting of multiple gates and states that can store long term memory to keep important information. In an RNN, we cannot store important data information through a long period time which can prove to be inefficient, LSTMs are used just like RNNs but can store important information that can be helpful for future states through a long period of time. There are 3 important gates that we need to discuss before jumping into implementation of an LSTM: Forget Gate, Input Gate, and Output Gate. Before I start explaining the gates, I need to make clear what a hidden state and a cell state is. **Cell state** is the long-term memory component that stores and loads information from previous neurons that can span back to the beginning of the neural network. **Hidden state** is the same in LSTMs as it is in RNNs, the hidden state is the information that is carried from the previous neuron and is overwritten at every neuron based on current information.

**Forget Gate:**

The forget gate is used to decide which information needs attention or can be ignored for the duration of the network. The forget gate essentially takes the sigmoid function of the previous hidden state plus the current input and multiplies the summation with the current cell gate that will determine if the current input is worth a position in the cell gate. If the number is 0, that means we forget the current input and if the number is 1, we apply it to the cell state that will hold the information for long term storage.

**Input Gate:**

The input gate is used to update the cell state in the current neuron. The way this is calculating is by adding the values of the previous hidden state and the current input and passing the summation through a sigmoid function that will produce a value between 0 and 1 which will tell the network if the current input is of importance or not (0 being not important and 1 being very important). The summation will also be passed through a tanh function that will produce a value between -1 and 1 that will help regulate the network. Both these values will be multiplied with each other, and the result will be added to the cell state.

**Output Gate:**

The output gate is responsible for producing a hidden state that will then be passed to the next neuron. The previous summation between the previous hidden state and the current input is passed through a sigmoid function and the current cell state (that the neuron calculated from the previous 2 gates) will be passed through a tanh function. Both these outputs will be multiplied, and the factor is the current hidden state that will then be passed through to the next neuron.



The advantages of an LSTM model over a RNN model are the conservation of important memory that has longer-term dependencies of later time stamps in our data. In RNN, each cell only considers the previous cell for its values and nothing else which can lead to important cells being forgotten in later cells that deem that specific cells data to be important to its hidden state output. LSTMs also help with vanishing/exploding gradients in the back propagation steps by introducing the input and forget gates which allow for better gradient flow control to prevent the gradient in the input layer from becoming too large or too small.

Text

Description automatically generated

Chart, histogram

Description automatically generatedChart, histogram

Description automatically generated

Chart, histogram

Description automatically generatedChart, histogram

Description automatically generated

Chart, histogram

Description automatically generated

From the above plots, we can tell that most of these features are immensely affected if the activity is either WALKING, WALKING UPSTAIRS, or WALKING DOWNSTAIRS. The way I am interpreting this data is that extraneous activity that requires heavier body movements will greatly affect any of the nine features that we have in the dataset. One thing I want to point out is that the STANDING attribute does not affect any of the features even slightly which is peculiar because SITTING and LAYING offer more of a variety on each time stamp and they are sedimentary activities just like standing.

****

K-fold cross validation is the process of distributing a testing set and a training set depending on the number of folds provided. Let’s say we have a dataset that is split 80% for the training set and 20% for the testing set. When we apply K-fold cross validation, we will take a different set of training/testing set depending on the folds value we are given while maintaining the proportion of the original split. K-folds is useful to thoroughly predict the accuracy of a model by using different training/testing sets of the data and computing an accuracy score for each Table

Description automatically generatedmodel fitting fold. To the right, we see a diagram of what K-fold cross validation does to a dataset with the same proportions as mentioned above. We can take the accuracy and each iteration and divide that number by the total number of iterations to get a better accuracy score for the model.

Table

Description automatically generatedText

Description automatically generated

I was able to obtain the test accuracies for each parameter combination and for each splitting of the data from K-folds cross validation, but I was not able to achieve this with confusion matrices or recall scoring. I was having trouble putting multiple scoring parameters for grid search so I will settle with just the test accuracies. Here I only had 3 parameters for units and 2 parameters for epochs because it was very slow to run on my computer.

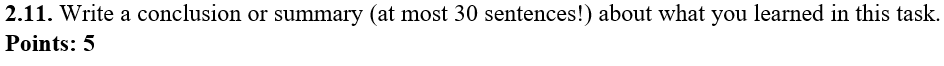


Table

Description automatically generatedI used the same parameters for LSTM as I did in the RNN, and I only reported the accuracies for each k-fold split. As we can see, the accuracy of every parameter combination and split has vastly increased from the previous RNN model. Although it did take a longer duration to fit the model onto the dataset, it has provided a more accurate model in predicting the exercise category.



In a bi-directional LSTM, the input flows in two directions which means that an input is fed into a backwards layer and a forward layer. This bi-directional choice gives the LSTM the ability to preserve both past and future information from any cells in the network given the current input cell. This can be very beneficial to predict certain information when the sequence of said information is important for predictions.



In this task, I have learned how to implement a neural network in python and learned how to visually interpret the data given to us. I have also been given tasks that were quite challenging to me but can be absolutely useful to me in the future such as knowing the differences between a Recurrent Neural Network, Long Short-Term Memory Neural Network, and a Bi-Directional LSTM Neural Network. These all focus on assigning specific data points to cells called neurons that can be used to help predict a certain value by fitting a model with data points that are to be evaluated as the first parameter and the actual values of the current dataset. This furthers my knowledge in a very popular topic that can prove very useful to me in the future if I wanted to create a model that is used for predicting values and/or scenarios. We also took a deeper look into K-Folds Cross Validation which is used to further validate a model by shifting the training and testing sets of the dataset we want to evaluate to see how well the model performs with different entries being the testing sets. I have also learned that although LSTM networks take a little longer to compile and evaluate, they give higher accuracy for testing data. The reason why it takes such a long time is that we are implementing more values in our data structure rather than forward feeding a hidden state. We are doing a multitude of computations that will tell the model what values need to be saved for later hidden states and which values can be discarded which can increase compile time. I unfortunately was not able to get Bi-Directional LSTM down in python but I fully understand the concept as being an LSTM with two cells for each input state. Each cell is used for either forward parsing or backward parsing through the neural network so that we can see if there is important information or values that pertain to the current input not only from the past neurons but also looking forward into future cells. This Bi-Directional parsing would surely increase compile time, but I feel it will significantly increase accuracy of any dataset that we fit the model to. From the following data, we can use Neural Networks that can predict efficient workout regimens for an individual based on their preference of exercising. We can predict what exercises are necessary to meet certain cardiovascular goals that will help the overall health of the individual and it can be done by using Neural Networks to help interpret data given from a trial. I have also learned that Neural Networks can work with different parameters that can be tuned to which set a parameters gives the best accuracy for our K-Fold split of our data. This is beneficial to me because I have a better understanding of how Neural Networks take in parameters that can either increase or decrease the efficiency of the algorithm all together.