Hugh Hoang 1833106

COSC 3337 *"Data Science I"* Fall 2022

Dr. Eick

Last Updated: Oct. 11, 11:15a

Task 3: Human activity detection using an LSTM network based on the sensor signals collected by a smartphone

Task3 Due: Sunday, October 30, 11:59p (no extensions will be given this time!)

Responsible TA: Navid

Total points: 50 + (10 Bonus)

Learning objectives:

- Getting familiar with RNN and LSTM.
- Learn how to tune model parameters and hyper parameters using k-fold cross validation.
- Learn how to use majority voting for labeling a test sample based on the outputs of different networks.

- Getting familiar with Bi-directional LSTM.
- (Bonus) Learn how to visualize time-series data using an LSTM encoder-decoder structure.

Classification of time series data is usually a difficult task. Effectively generating features from the raw data for a machine learning model often requires extensive domain understanding and the use of methodologies that are borrowed from signal processing. By automatically extracting features from the unprocessed data, deep learning techniques, like **recurrent neural networks** (RNNs), have recently obtained cutting-edge outcomes for time-series classification tasks. RNN uses sequential data or time series data. Their **memory**, which allows them to use data from earlier inputs to affect the current input and output, sets them apart from other neural networks. Recurrent neural networks' outputs depend on the previous items in the sequence, in contrast to standard deep neural networks' assumption that orders of inputs do not matter.

In this task, we are going to use a variation of RNNs known as Long short-term memory (LSTM) to classify six different human activities based on the data collected by a smartphone. The dataset can be downloaded from:

https://archive.ics.uci.edu/ml/datasets/human+activity+recognition+using+smartphones

The data were conducted on a group of 30 volunteers ranging in age from 19 to 48. Each participant used a Samsung Galaxy S II smartphone while engaging in six different activities (WALKING, WALKING UPSTAIRS, WALKING DOWNSTAIRS, SITTING, STANDING, and LAYING).

We recorded 3-axial linear acceleration and 3-axial angular velocity at a constant rate of 50Hz using its integrated accelerometer and gyroscope. The experiments were videotaped so that the data could be manually labeled. The resulting dataset was divided into two sets at random, with 30% of the volunteers chosen to create test data and 70% of the participants chosen to create training data. After applying noise filters as a pre-processing step, the accelerometer and gyroscope sensor data were sampled with fixed-width sliding windows of 2.56 seconds and 50% overlap (128 readings/window). The README.txt file contains more information about this dataset.

Download and extract the dataset. In this assignment, we use only **raw data** located in **"Inertial signal"** folder within "train" and "test" folders. The information about the corresponding subject for each row of the dataset can be found in subject_train.txt and subject_test.txt in train and test folders, respectively.

Write your code in **Python** or other language you prefer to answer the tasks listed below.

3.1. Briefly explain (in about one page) how an LSTM cell works. (How many gates does it have and what are the duties of these gates.) **Points: 5**

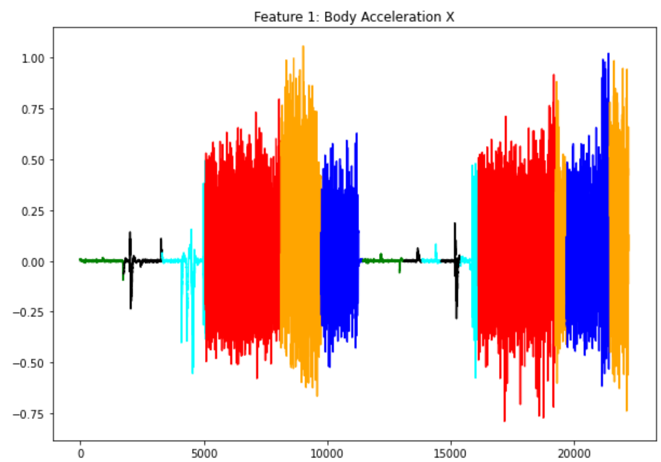
Long Short Term Memory networks are a special kind of RNN, capable of learning long-term dependencies. LSTMs have a chain of repeating modules of neural network, but the repeating module has a different structure than other RNNs as it has four network layers instead of one and interacts in a very special way. The key to LSTMs is the cell state which is usually shown by a horizontal line running through the top of a LSTM diagram. It functions somewhat like a conveyor belt and runs straight down the entire chain, with only some minor linear interactions allowing information to just easily flow along it unchanged. The LSTM can use this as it can remove or add information to the cell state, carefully regulated by structures called gates. They are composed out of a sigmoid neural net layer and a pointwise multiplication operation which outputs numbers between zero and one, describing how much of each component should be let through with a value of zero means "let nothing through," while a value of one means "let everything through!". An LSTM has three of these gates, to protect and control the cell state with in order the first gate's role having to decide what information we're going to throw away from the cell state, this is called the "forget gate layer". The next gate is responsible for deciding what new information we're going to store in the cell state called the "input gate layer"which decides what values we'll update with the tanh layer to create a vector of new candidate values. After this we update the old cell state by using the LSTM cell by multiplying the old state by what we chose to forget in the forget gate forgetting the things we decided to forget earlier. Then we add the information to the old candidate values to get the new one. Finally, we need to decide what we're going to output, which is the last gate, the output gate. This will output information based on our cell state, but will be a filtered version as it first runs a sigmoid layer which decides what parts of the cell state we're going to output then, puts it in the cell state through tanh to push the values to be between −1 and 1 and multiply it by the output of the sigmoid gate, so that we only output the parts we decided to. There are some slight variants to the LSTM, one popular LSTM variant, which includes adding "peephole connections." This means that we let the gate layers look at the cell state. Another variation is to use coupled forget and input gate, instead of separately deciding what to forget and what we should add new information to, we make those decisions together. We only forget when we're going to input something in its plac and only input new values to the state when we forget something older.

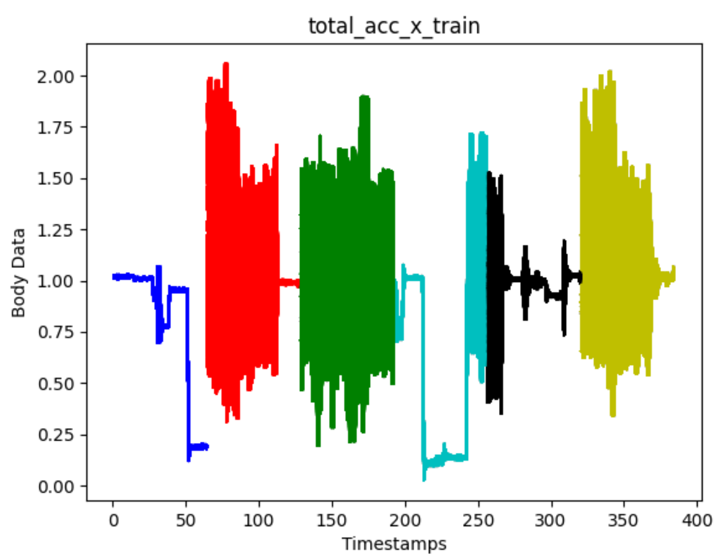
3.2. Explain (in about one paragraph) the advantage of LSTM networks over traditional RNNs.

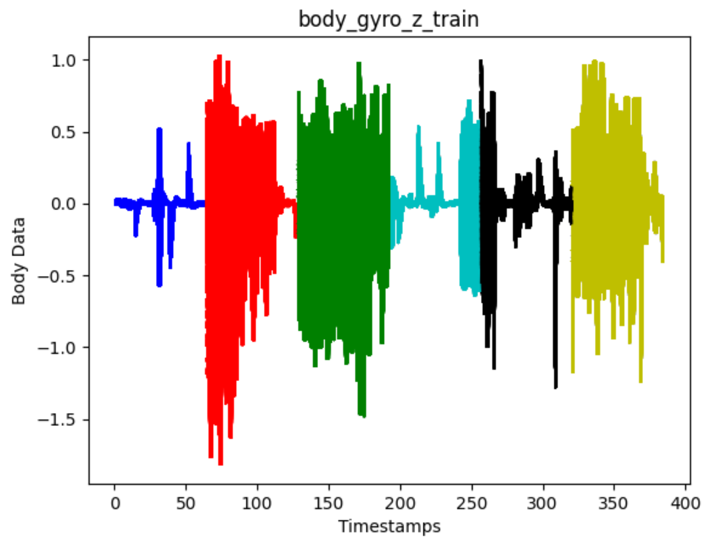
Traditional Recurrent Neural Networks are neural network algorithms that can memorize or remember the previous inputs in memory which is something that LSTMs can do as well. LSTM includes a 'memory cell' that can maintain information in memory for long periods of time which means it can handle the information in memory for a longer period of time than compared to RNN.The big difference or advantage LSTM have over traditional RNNs is that it is difficult to train RNNs for long-term memorization while LSTMs performs better in these kinds of datasets as it has more additional special units that can hold information longer.

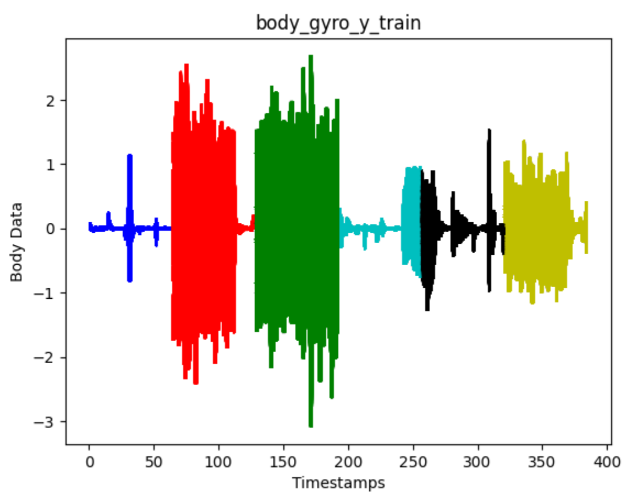
3.3. For subject 1 in training data, plot nine time series corresponding to nine different features. Note that there is a 50% overlap between each row of the data as mentioned above. The color of the line plot should change according to the corresponding activity. Use red, blue, orange, black, green and cyan for WALKING, WALKING UPSTAIRS, WALKING DOWNSTAIRS, SITTING,

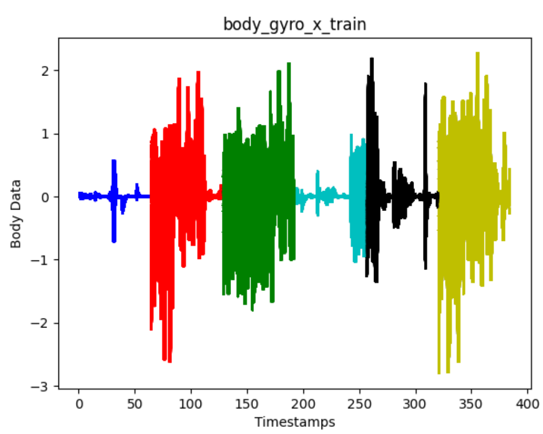
STANDING, and LAYING, respectively. Interpret the obtained plots for discriminating the different activities. **Points: 3**

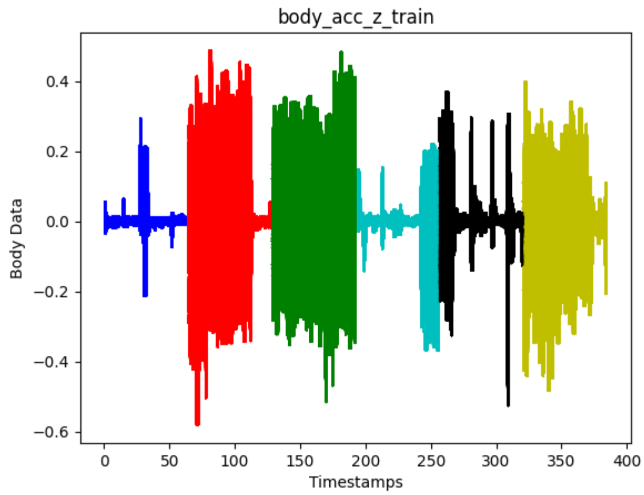


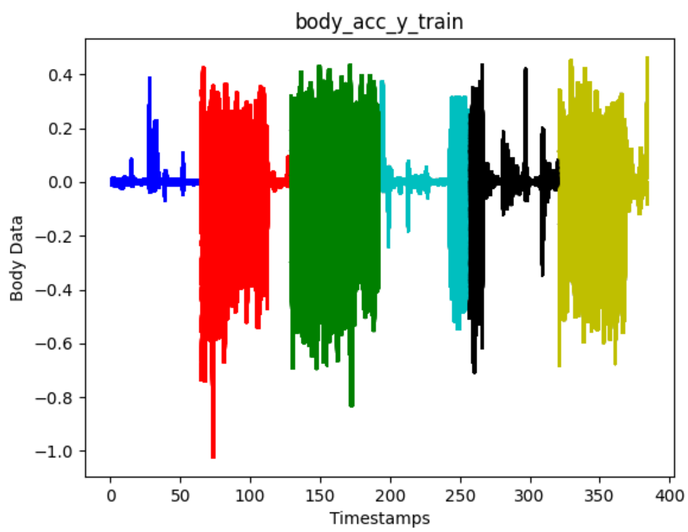


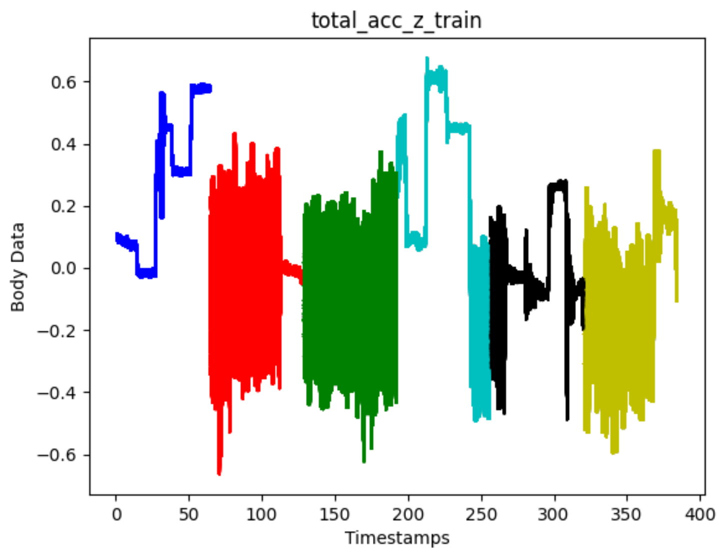

body_gyro_z_train
Body Data
Timestamps
1.0
0.5
0.0
−0.5
−1.0
−1.5
0
50
100
150
200
250
300
350
400

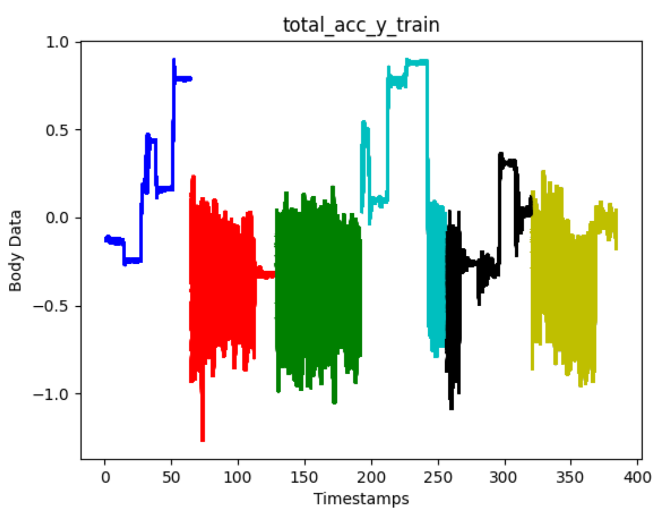

body_gyro_y_train
Body Data
Timestamps
2
1
0
−1
−2
−3
0
50
100
150
200
250
300
350
400


body_gyro_x_train
Body Data
Timestamps
2
1
0
−1
−2
−3
0
50
100
150
200
250
300
350
400


body_acc_z_train
Body Data
Timestamps
0.4
0.2
0.0
−0.2
−0.4
−0.6
0
50
100
150
200
250
300
350
400


body_acc_y_train
Body Data
Timestamps
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8
−1.0
0
50
100
150
200
250
300
350
400





3.4. Briefly explain k-fold cross validation. **Points: 2**

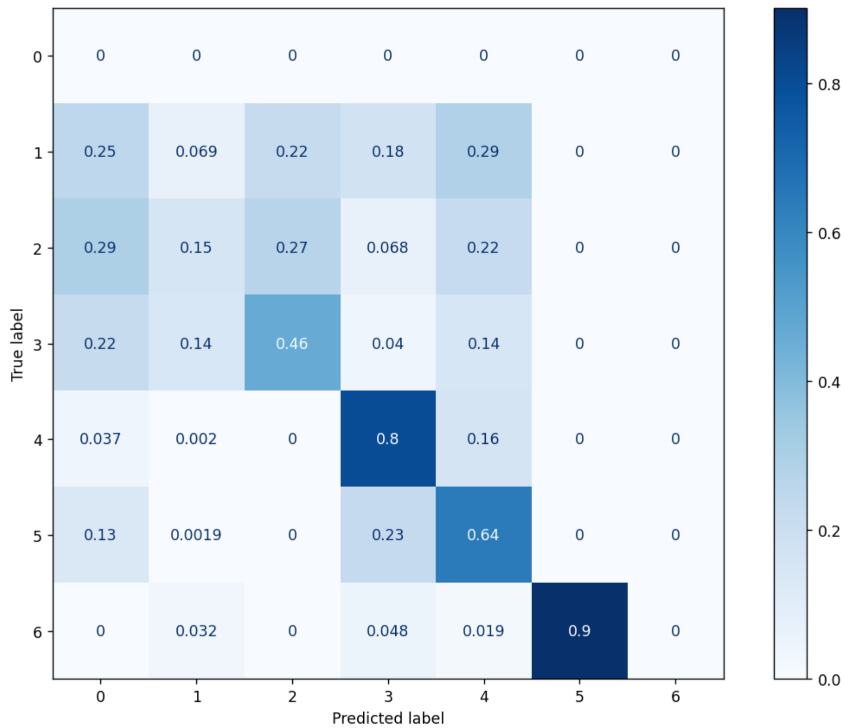
Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample. The procedure has a single parameter called k that refers to the number of groups that a given data sample is to be split into.Cross-validation is primarily used in applied machine learning to estimate the skill of a machine learning model on unseen data. That is, to use a limited sample in order to estimate how the model is expected to perform in general when used to make predictions on data not used during the training of the model.It is a popular method because it is simple to understand and because it generally results in a less biased or less optimistic estimate of the model skill than other methods, such as a simple train/test split.The general procedure is as follows:Shuffle the dataset randomly, Split the dataset into k groups, For each unique group:Take the group as a hold out or test data set,Take the remaining groups as a training data set,Fit a model on the training set and evaluate it on the test set,Retain the evaluation score and discard the model,Summarize the skill of the model using the sample of model evaluation scores

3.5. Design an RNN network to classify six different activities. Use the input with all nine features. Tune model parameters and hyper parameters and choose one that results the best accuracy based on the 5-fold cross validation. You must use grid search for tuning your parameters. You can consider three values for each parameter and find the best settings based on the highest accuracy obtained using grid search. Report all confusion matrices, accuracies and recalls for cross validation step. **Points: 5**

Grid search parameters

epoch:20 and hidden_layer:20 as my best acc: 72%

Confusion Matrix:



Accuracy:

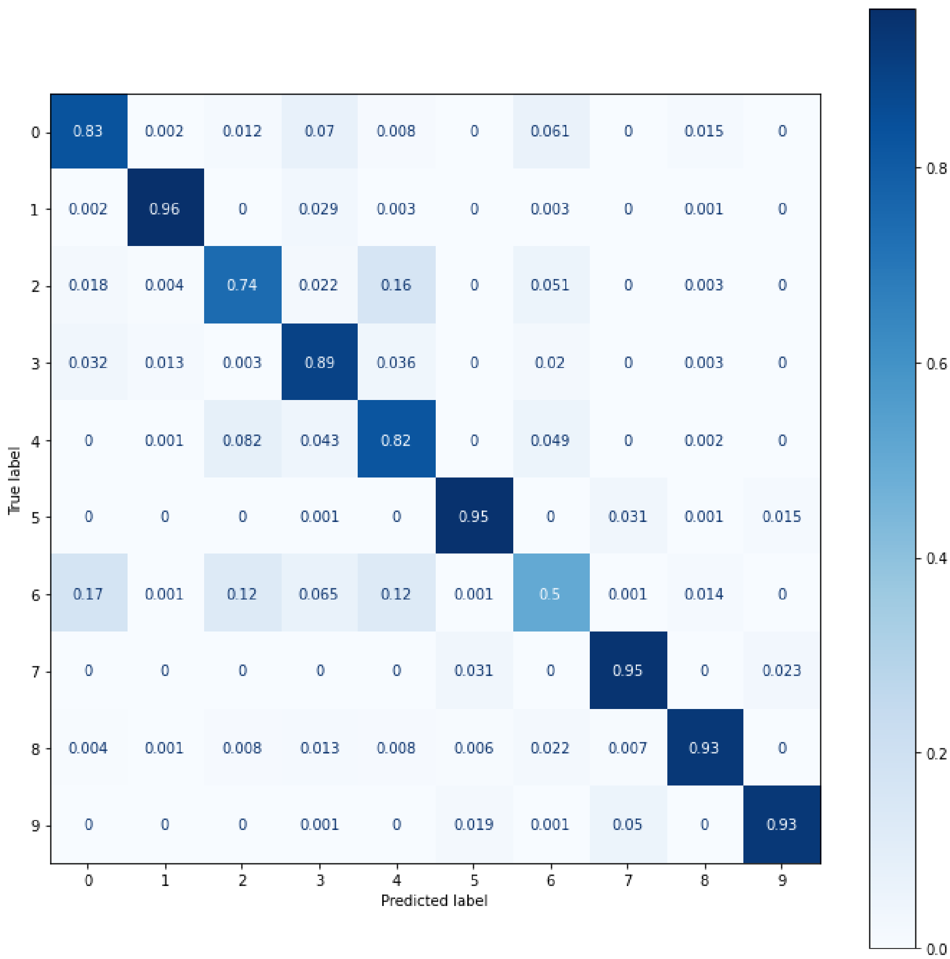
0.5748218297958374

Recall:

0.6104512810707092

3.6. Train your RNN network with selected settings found in previous task using all training samples (with all nine features) and report the confusion matrix, accuracy and recall. Also, report the confusion matrix, accuracy and recall for testing set. **Points: 3**

Training Confusion Matrix:



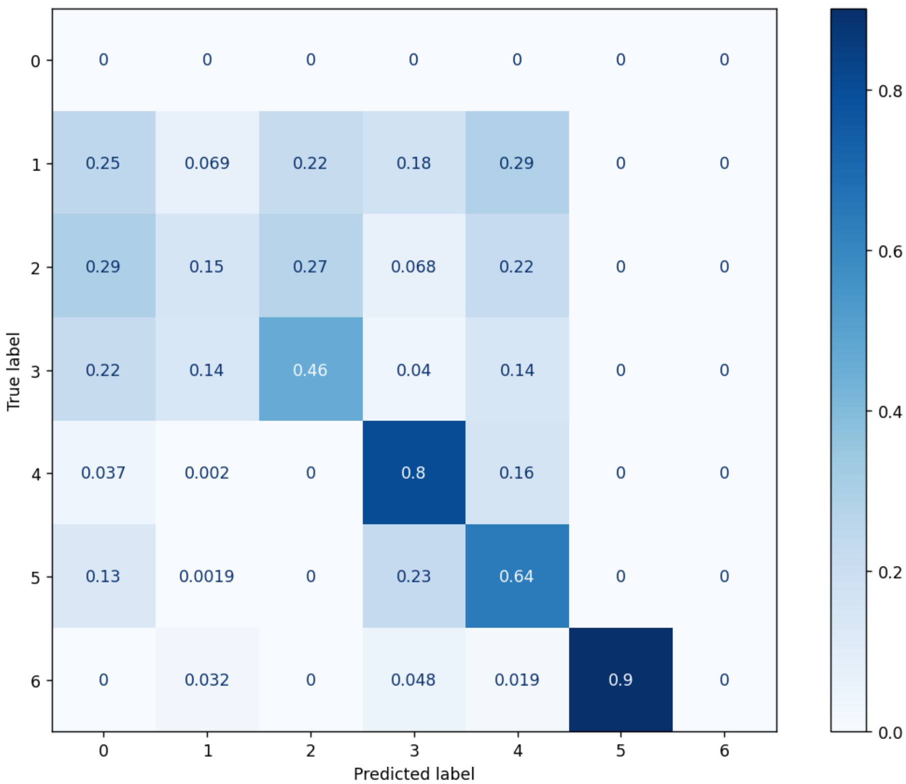
Training Accuracy:

0.6318218547956375

Training Recall:

0.6950040402752013

Testing Confusion Matrix:



Testing Accuracy:

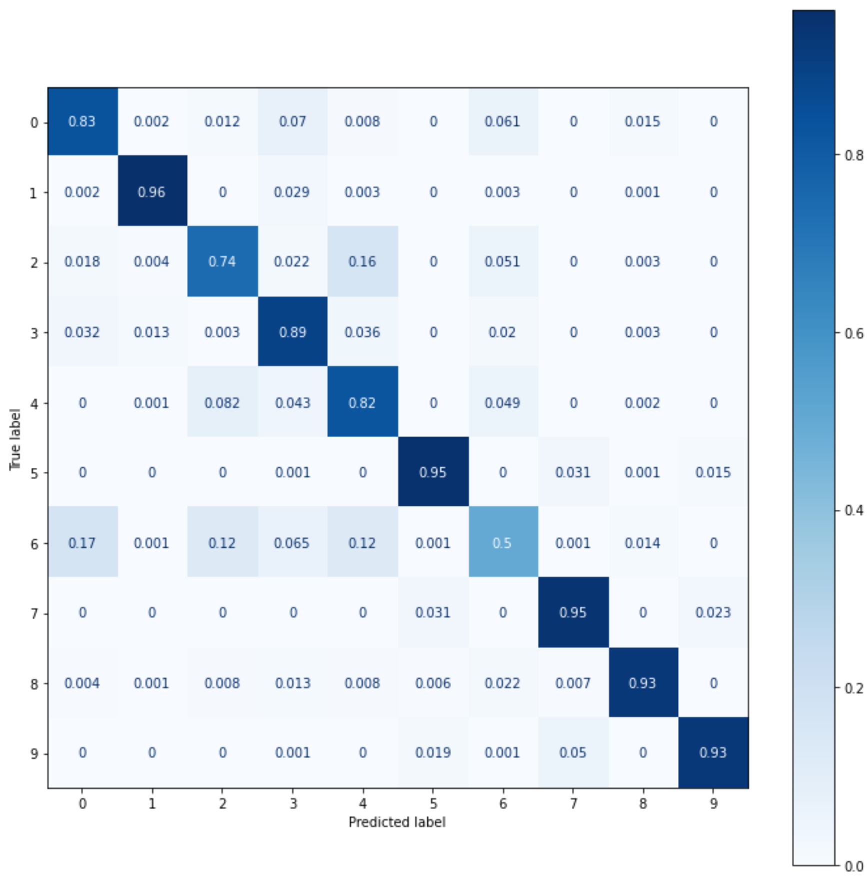
0.6556641889388691

Testing Recall:

0.6687774727176465

3.7. Replace RNN with LSTM structure and repeat 3.5 and 3.6 for designing and training an LSTM network. Compare obtained results. Discuss the reason(s). **Points: 10**

Training Confusion Matrix:



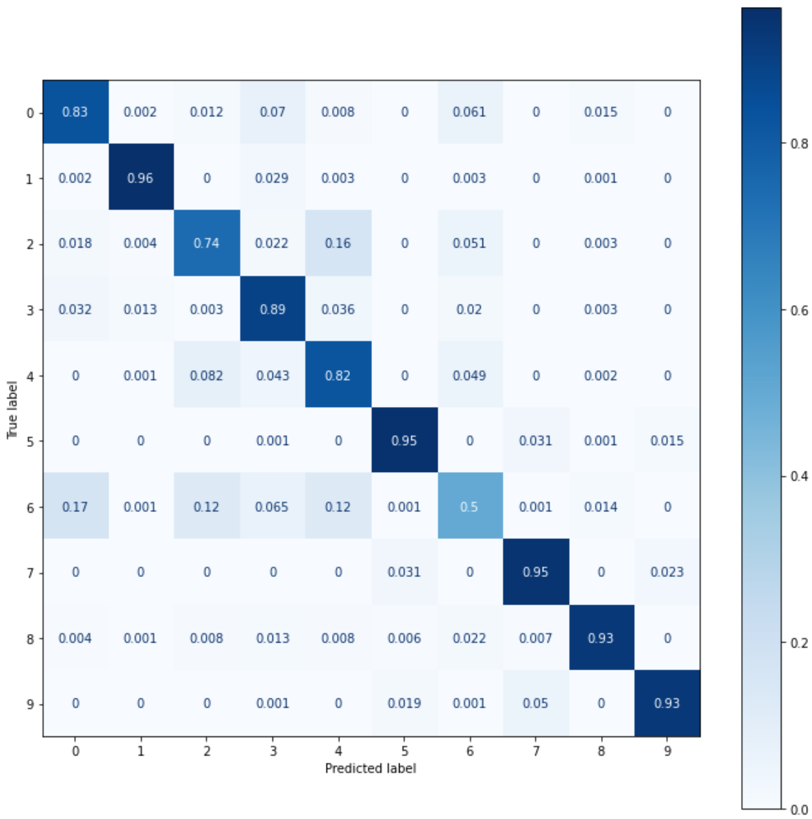
Training Accuracy:

0.6318218547956375

Training Recall:

0.6950040402752013

Testing Confusion Matrix:



Testing Accuracy:

0.6556641889388691

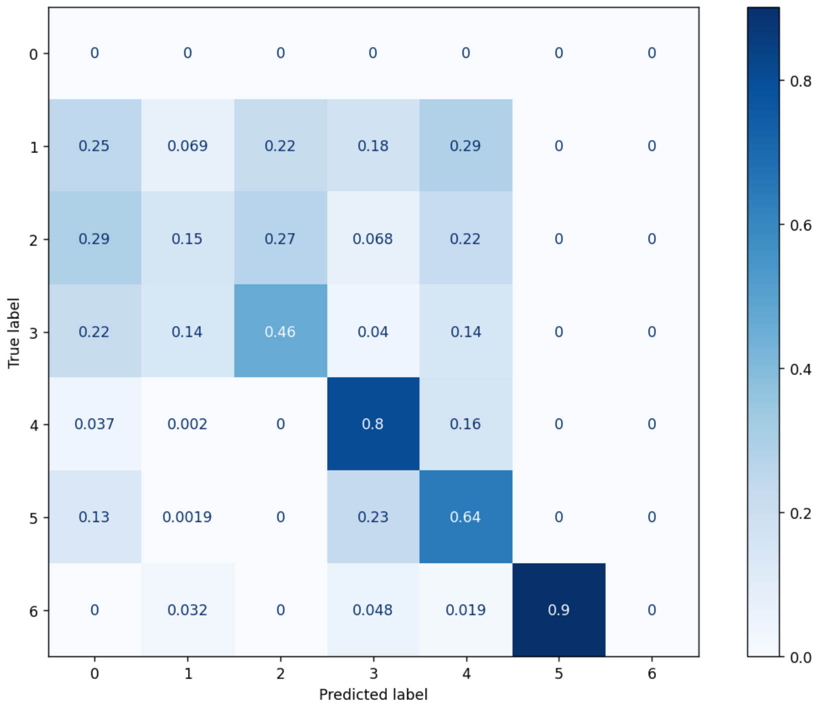
Testing Recall:

0.6687774727176465

We see from the results that the LSTM tends to always be more accurate and have a higher recall than the regular RNN structure. We can also tell from the confusion matrix that there are changes with what we chose as the most accurate feature. This is most probably due to differences between RNNs and LSTM which is that it is difficult to train RNNs for long-term memorization which is why LSTMs tend to perform better in these kinds of datasets as it has more additional special units that can hold information longer.

3.8. Now consider the data as nine separate univariate time-series. Design nine separate LSTM networks and feed each of them with the training samples with a single feature (nine features → nine separate LSTM networks). Use majority voting to label each sample in testing set. Report the confusion matrix, accuracy and recall for testing set. Discuss the reasons of the obtained result by comparing it with 3.7. **Points: 6**

Testing Confusion Matrix:



Testing Accuracy:

0.7524940317327326

Testing Recall:

0.7901187333193692

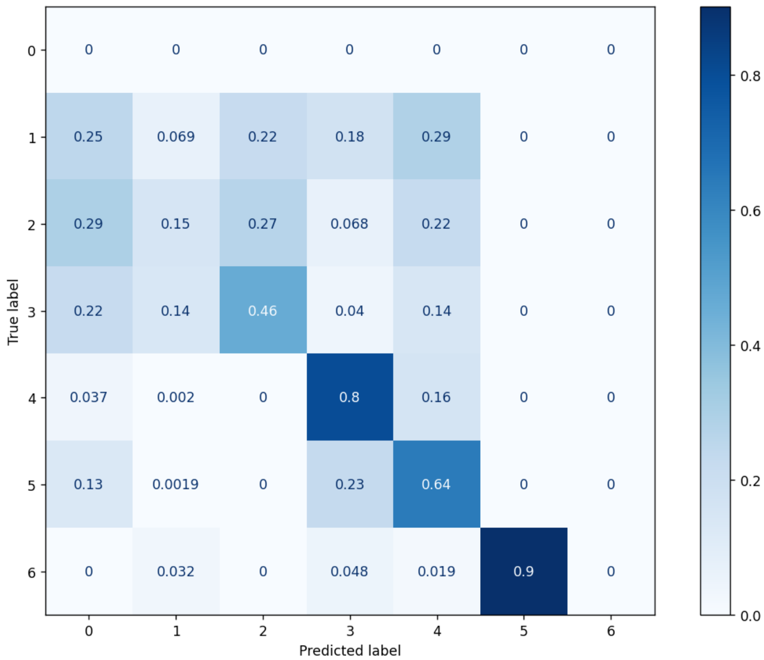
We see from the results that the our nine separate LSTM networks tends to always be more accurate and have a higher recall than the multivariate LSTM structure. We can also tell from the confusion matrix that there are changes with what we chose as the most accurate feature. This is probably because univariate analysis are more useful in understanding the distribution of values for one variable while the multivariate analysis allows us to understand the relationship between several variables.

3.9. Briefly explain what is a bi-directional LSTM? And why does it sometimes provide better performance over a simple LSTM network? **Points: 3**

Bidirectional long-short term memory is the process of making any neural network to have the sequence information in both directions backwards or forward. In bidirectional LSTM, our input flows in two directions, making it different from the regular LSTM which can only make input flow in one direction, either backwards or forward. However, in bi-directional, we can make the input flow in both directions to preserve the future and the past information which can be useful in case we need to do that.

3.10. Design and train a bi-directional LSTM. Report all confusion matrices, accuracies and recalls for 5-fold cross validation and testing set. Discuss the results. **Points: 6**

Testing Confusion Matrix:



Testing Accuracy:

0.8619477090756755

Testing Recall:

0.9395230028924863

We see from the results that the bi-directional LSTM tends to be more accurate and have a higher recall than the regular RNN structure. We can also tell from the confusion matrix that there are changes with what we chose as the most accurate feature. This is most probably due to differences between bi-directional LSTMs and standard LSTM which is that it we can make the input flow in both directions to preserve the future and the past information

2.11. Write a conclusion or summary (at most 30 sentences!) about what you learned in this task.

Through this exercise I began to learn alot about the manipulation of raw data to create something more complex like graphs, statistics, and RNN/LSTM models. I learned about things like the classification of time series data, effectively generating features from the raw data for a machine learning model, using deep learning techniques like recurrent neural networks for time-series classification tasks. I learned more about the coding aspect for RNNs and how they use sequential data or time series data to generate meaningful and useful data. This was probably the most difficult aspect as this was something that we really did not have much specific knowledge on. I also explored more about the variations of RNNs one of which was the Long short-term memory (LSTM) which we used to classify our features. There was also a modified LSTM which was the Bi-directional LSTM which enabled the LSTM to go either backwards or forwards. There was also more technical knowledge that we were a bit more familiar with from previous exercises like extracting data from files and manipulating datasets to fit our needs. Some more specific skills from the problems were things like how to tune model parameters and hyper parameters using k-fold cross validation. As well as learning how to use majority voting for labeling a test sample based on the outputs of different networks. As a whole I understood more in depth on what goes on with some of the techniques needed to extract necessary data from and how to use or even refine it.

Points: 5

2.12. (BONUS*) In this task, you will learn to use LSTM structure to visualize the time series in a 2-dimensional space. The network consists of one encoder and one decoder. The encoder converts the time series into a vector. The decoder takes this vector as its initialization to produce the input sequence. The network will be trained based on the difference between the original time series and the predicted time series. Design and train an LSTM encoder-decoder. Obtain the encoded time series produced by the trained encoder for testing samples. Plot a supervised scatter plot using the obtained 2-dimensional vectors and the testing labels. Interpret the findings! **Points: 10**

Remark: *Most of the points of the tasks will be given to summarizing and interpreting the obtained results. Therefore, by showing only graphs and tables without any discussions you will get a few points of that task! Your discussions and explanations in your report must be **at least 2 pages long**. This excludes any tables and figures (only text!) except for subtask 1. You can include graphical representations of LSTM cell to support your explanations. Also, you can ignore to report all confusion matrices for cross validation step in subtask 3.8. But if you report all, it is going to be considered as bonus points!*

***Attention!** Please **DO NOT** send me your codes for debugging before submission.*

How to submit:

*You must separate your codes from your report. The report must include your discussions, explanations and your plots and tables. Do not combine it with your codes! Please upload a **ZIP** file including your report in **PDF** format and your code files in Blackboard.*