Predict Parkinson Disease using wearable data

Aloi Alfredo, Riccio Francesco

December 31, 2021

Abstract

This is a project for the Deep Learning course held in the Master Degree in Computer Science of Università degli Studi della Calabria.

Contents

1 Introduction **3**

2 Task 1: next value prediction **4**
2.1 Data understanding and preparation 4
2.2 Modeling 4
2.3 Evaluation 8
2.4 Finding better parameters 9
2.5 All in one model 9

3 Task 2: anomaly detection **13**
3.1 Data understanding 13
3.2 Data preparation 13
3.3 Modeling 13
3.4 Evaluation 13
3.5 Conclusions 13

1 Introduction

Parkinson disease, the second most neurological disorder that causes significant disability, reduces the quality of life and has no cure. Approximately, 90% affected people with Parkinson have speech disorders. Deep learning has the potential to give valuable information after processing that can be discovered through deep analysis and efficient processing of data by decision makers. The main problem is the identification of OFF symptoms perceived by the study subject with reasonable accuracy from real-world data collected. OFF periods are times when Parkinson's disease medication, namely levodopa, is not working optimally. As a result, symptoms return. OFF periods can include both motor symptoms, such as tremor and rigidity, and non-motor symptoms, such as anxiety. The goal is to consolidate data collection by identifying the most important variables to solve the problem.

2 Task 1: next value prediction

We have 3 time series (X, Y, Z) recorded each 10 seconds. For the first sub-task we consider, for each time series, sequences of five minutes (`window_size = 30`) every one minute (`window_shift = 6`), predicting the next value in the series. The evaluation metric is the Mean Absolute Error, in particular we have to go below:

- 81.06 for X;
- 85.26 for Y;
- 79.94 for Z.

We then choose, for the second sub-task, the one time series with the worst value of the evaluation metric, looking for a better combination of window size/window shift.

2.1 Data understanding and preparation

We have two files, `train.csv` and `test.csv`, each one containing three columns, representing the three time series X, Y and Z. The `test.csv` file will be used for the evaluation, while the `train.csv` file, that contains 144910 rows, will be used for the training part.

Since the three time series to be used in the training steps are inside one single dataset, we start the preparation of our data by first splitting it into the three different time series. For each one of the generated time series:

1. we produce a set of sequences, basing ourselves on the aforementioned window size and window shift, with the corresponding labels, representing the element that follows each sequence in the series;
2. we normalize the values inside the produced sequences using a Standard scaler, which scales the data according to a Normal distribution;
3. we, additionally, split the produced sequences to obtain training and validation sets to use in the training step, using a ratio of 80:20.

2.2 Modeling

We choose to build a neural architecture starting with three LSTM layers followed by four Dense layers, with the last one made of just one neuron, which is our output. Between each layer of the architecture we choose to put

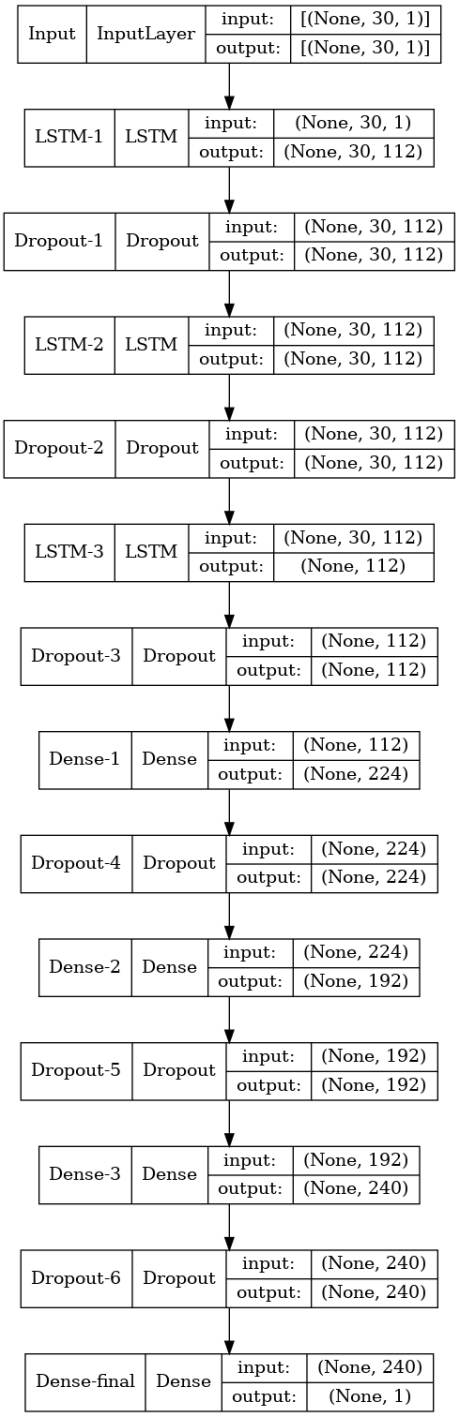

Figure 1: The model for Z time series

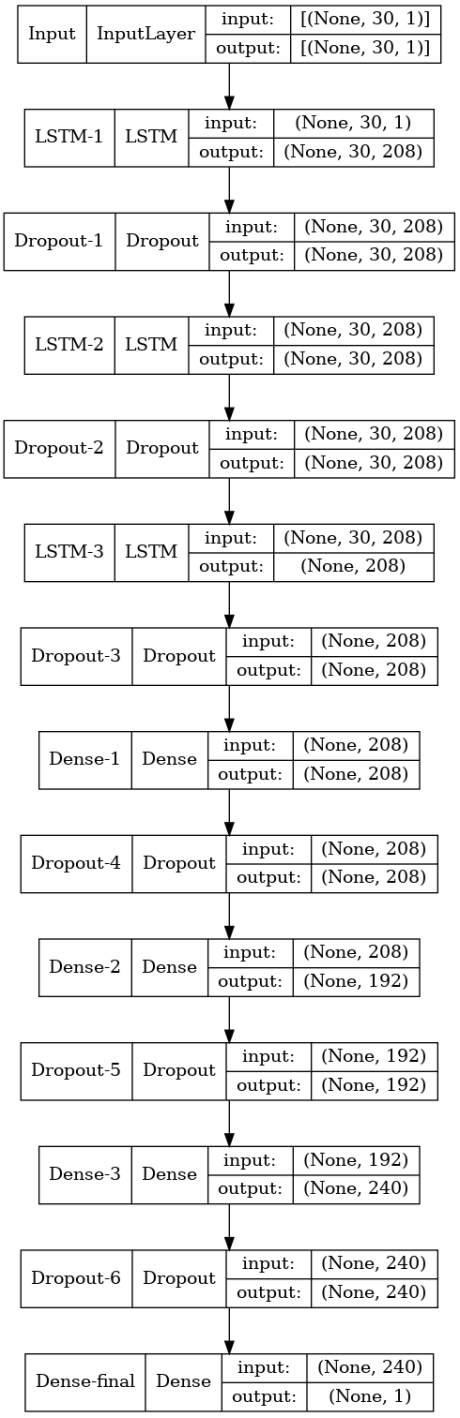

Figure 2: The model for Z time series

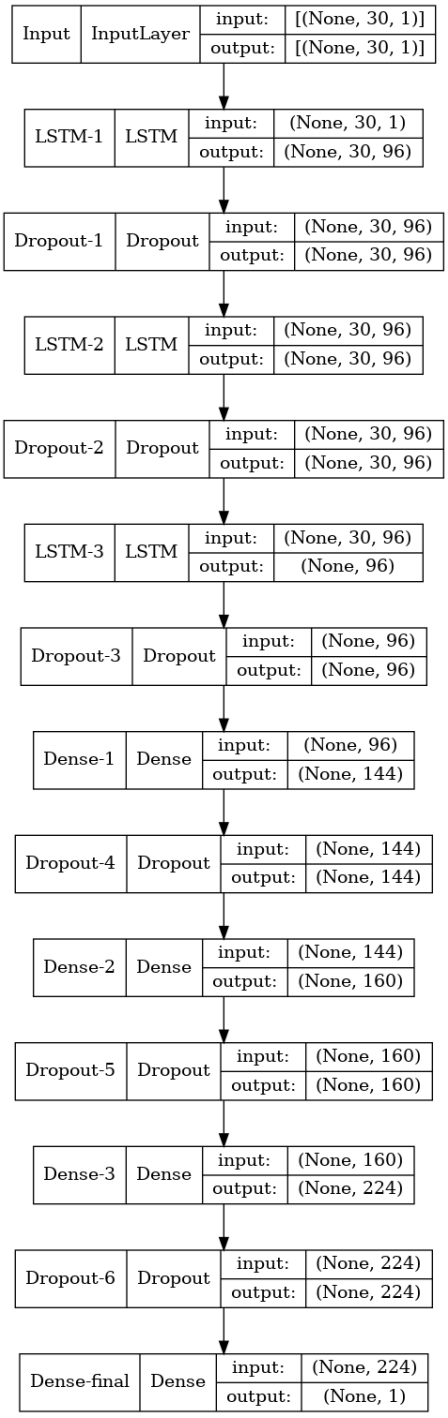

Figure 3: The model for Z time series

a dropout layer to mitigate the overfitting bias. We employ a Hyperband tuner to make the choice of the best model parameters for us. Next, we fit our models for 25 epochs using a batch size of 32, keeping track of the evaluation metric score of the validation data used during training, in order to save only the best behaving model; this is done using a Model Checkpoint as callback to the fitting process.

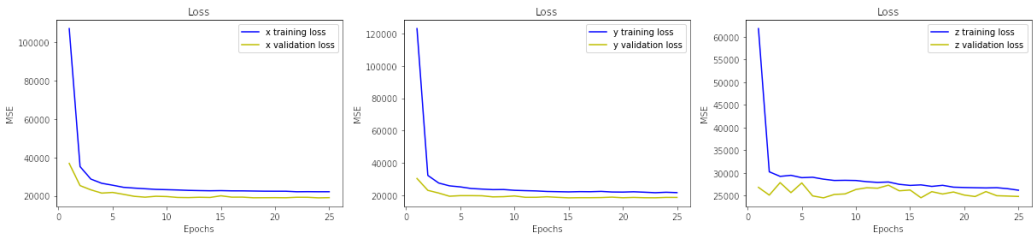

Figure 4: The losses during training

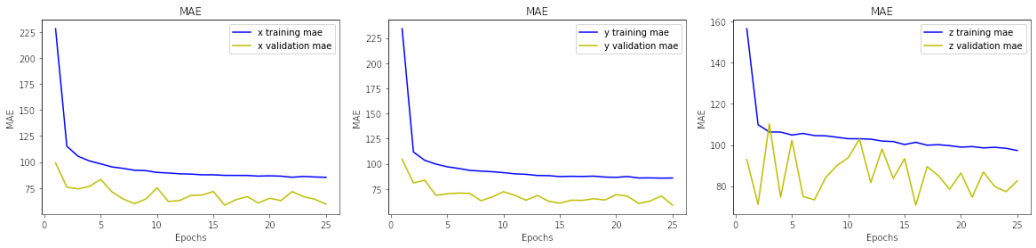

Figure 5: The MAE during training

2.3 Evaluation

Looking at the plots regarding the loss (Figure 4), we can observe how in the first epochs the models learn very quickly, suffering a sudden slowdown in the following epochs, but, anyway, the learning curve appears to be quite smooth for each time series. On the other hand, we can notice, in the plots regarding the Mean Absolute Error (Figure 5), that each time series has an irregular behaviour, especially the Z time series, with a high variability in the values of said metric during training.

To evaluate our models, we first have to apply the same preprocessing steps also to the test dataset, namely: splitting of the different time series, generation of sequences and normalization. Concerning the results obtained after testing (Table 1), these are quite good, all the evaluation metrics respect the set objective values. In particular, with regards to the X time series, we

	Target	Actual
X	81.06	**77.62**
Y	85.26	**84.31**
Z	79.94	**78.39**

Table 1: Testing results obtained

are far below the target threshold (77.62 compared to 81.06), while for the other two the difference is smaller (84.31 and 78.39 compared to, respectively, 85.26 and 79.94).

2.4 Finding better parameters

The objective of this subtask is to try to improve the results obtained previously, for the Y time series. Specifically, we want to find a better combination of window size and window shift. To make this possible, we have defined a search space, consisting of five values for the window size (18, 24, 30, 36 and 42) and four values for the window shift (1, 3, 6, 9). For each combination of window size/window shift we fit the model and test it, saving the respective evaluation. It can be noticed (Figure 6, Table 2) how models with a

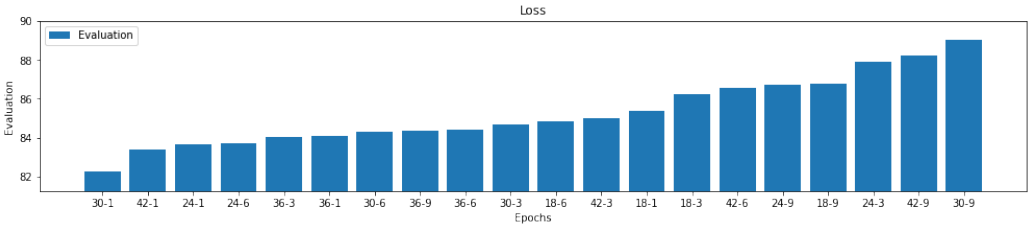

Figure 6: The evaluations for each combination

lower window shift tend to perform better, as opposed to those with a higher window shift, that do not perform in the same way.

2.5 All in one model

As an extra step we decide to build a model that takes as input the three time series at once, in order to see if we can obtain more accurate results. We use the same preprocessing approach used in the previous steps. Also here we use a Hyperband tuner to find the best parameters to use for our model, that is then fitted and tested.

Window size	**Window shift**	**Score**
18	1	**85.35**
18	3	**86.21**
18	6	**84.83**
18	9	**88.20**
24	1	**83.65**
24	3	**87.91**
24	6	**83.71**
24	9	**86.74**
30	1	**82.24**
30	3	**84.69**
30	6	**84.31**
30	9	**89.01**
36	1	**84.06**
36	3	**84.03**
36	6	**84.41**
36	9	**84.37**
42	1	**83.36**
42	3	**85.01**
42	6	**86.55**
42	9	**88.20**

Table 2: Testing results obtained

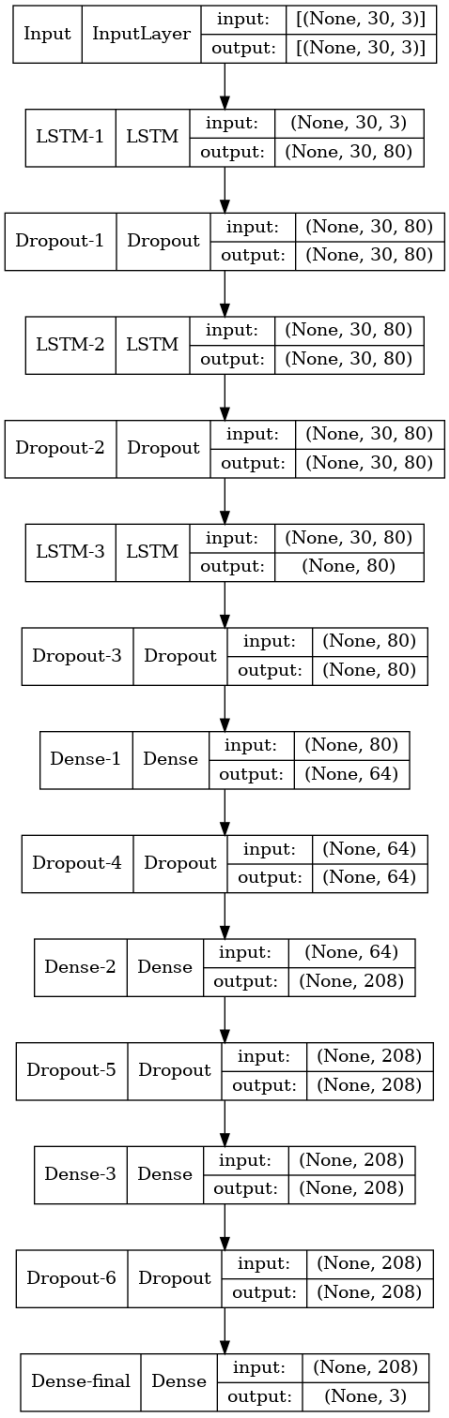

Figure 7: The all in one model

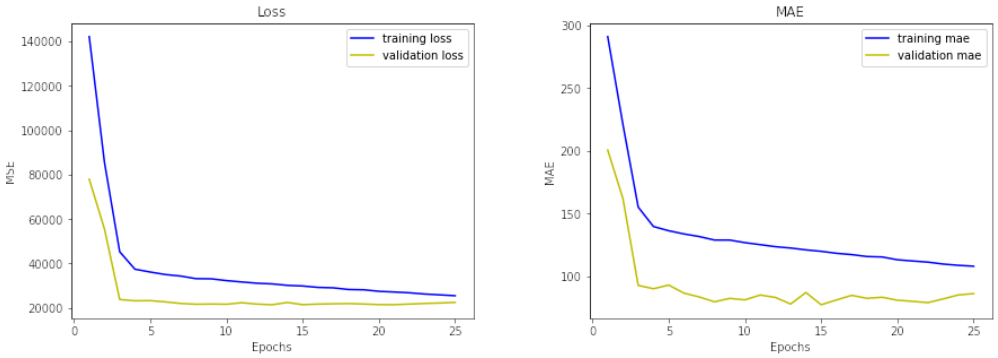

Figure 8: The all in one model loss and MAE

We are not surprised to see (Figure 8) that the plots are similar to the ones observed before. We can also notice that the model is now more easily keeping a not so high variation in evaluation metric during training, whereas before the values could vary wildly. In spite of this, the evaluation of the trained model on the test set has given a result of 93.86, that is higher than the worst result obtained by the same architecture using the three time series separately.

3 Task 2: anomaly detection

General description about the task

3.1 Data understanding

Description about the data...

3.2 Data preparation

All the steps: normalization etc.

3.3 Modeling

Model description

3.4 Evaluation

Talking about evaluation and results

3.5 Conclusions

conclusions about this task