Deep Learning - Assignment 1

Group 2

Instructor:
Siamak Mehrkanoon

Group Members:
Sheng Kuang (i6237193)
Yimin Yang (i6246099)

1. Brief answers for the questions

1.1 Which model do we use?

LSTM and **CNN** (LSTM demonstrates better result, see LSTM and CNN section)

1.2 How many previous steps do we feed into the model?

70 (see Figure 1)

1.3 What is the mean absolute error of predictions of each model:

10.449 (LSTM) and **50.347** (CNN)

2. Methods

2.1 Pre-processing

This step aims to normalize the original data and split them into training and validation datasets.

2.1.1 Normalization

The normalization approach we selected is ***MinMaxScaler***. The range of scaled data is from 0 to 1.

2.1.2 Data Transformation

In order to generate the data that can be used for training, we introduced ***WINDOW_SIZE*** to define the input length of the timestep. As a result, the data can be transformed into a NumPy array with size:

[1000 - *WINDOW_SIZE* + 1, *WINDOW_SIZE* + 1]

The first dimension represents the number of records, and the second dimension represents the length of consecutive sequences (WINDOW_SIZE for input, 1 for output).

2.1.3 Training & Validation Dataset:

80% of the data are used for the training set and the remaining 20% are used for the validation set. The datasets are listed below:

Train_X: [0.8 * (1000 - *WINDOW_SIZE* + 1), *WINDOW_SIZE*]
Train_Y: [0.8 * (1000 - *WINDOW_SIZE* + 1), 1]
Validation_X: [0.2 * (1000 - *WINDOW_SIZE* + 1), *WINDOW_SIZE*]
Validation_Y: [0.2 * (1000 - *WINDOW_SIZE* + 1), 1]

2.2 LSTM (High-priority for the assignment)

(Setting up using Pytorch, and this part has a **higher priority** in our assignment demonstration. CNN setting up and comparison are extensions for the assignment, we want to deal with the assignment using different approaches and compare the pros and cons.)

2.2.1 Motivation

LSTM is a type of recurrent neural network which is able to handle sequential data, especially in a scenario with variable input length. LSTM predicts the following step based on several previous steps.

2.2.2 Architecture

The LSTM model has 40 units, 10 neurons for each hidden state, no stack layers, no bidirectional propagation, and a fully connected layer on the top to output the value for the next value.

2.2.3 Hyperparameters

In order to fine-tune the LSTM model, we tested a series of hyperparameters and recorded the convergence of the model. Due to the limit of paper length, only the four most influential hyperparameters are listed below (i.e, Hidden size, Window size, Learning rate, Num layers):

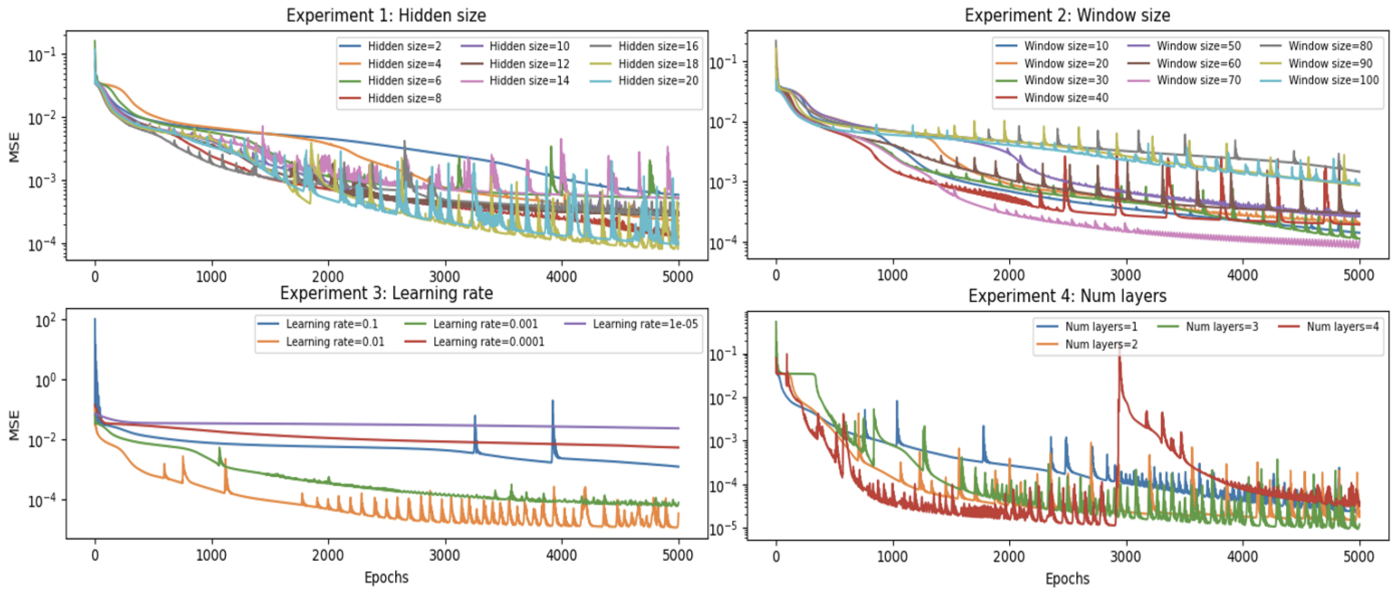

Figure 1. LSTM model convergence speed with different hyperparameters

Optimal hyperparameters of the model are listed below:

Name :	Value	Name :	Value
window_size :	70	learing_rate :	0.01
num_epochs :	5000	hidden_size :	8
optimzer :	adam	cost function :	MSE
batch first :	True	batch size :	fully propagate at once

2.2.4 Prediction

Figure 2 illustrates the 200-step predictions recursively generated by LSTM.

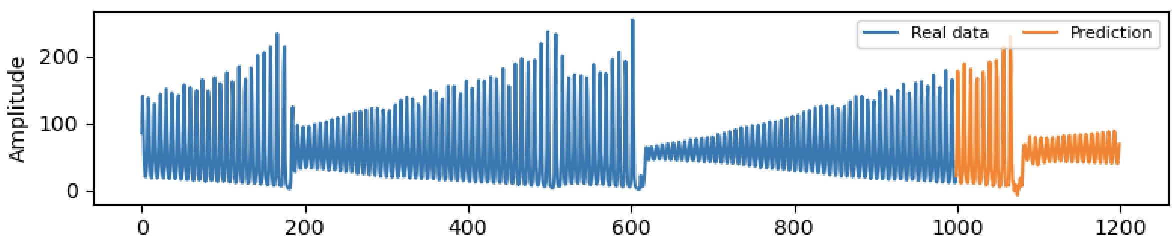

Figure 2. LSTM predictions for the next 200 steps

2.2.5 Mean Absolute Error

The mean absolute error (MAE) between the predictions and test data is around **10.449**. Figure 3 shows the comparison between predictions and test data.

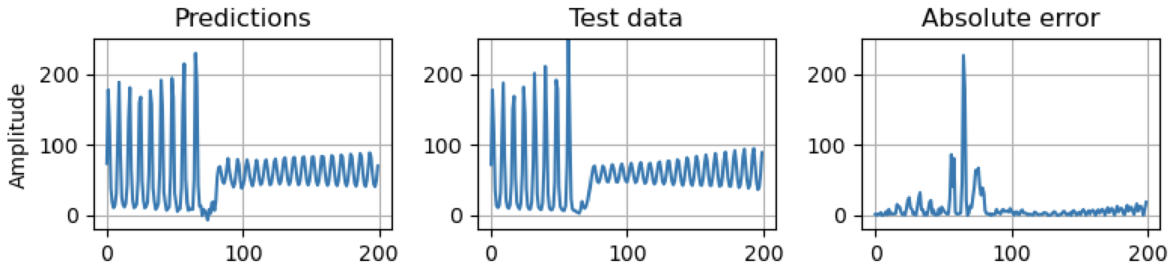

Figure 3. Comparison between LSTM predictions and test data

2.3 CNN (Sub-priority for the assignment)

2.3.1 Motivation

Besides LSTM, CNN is also commonly used to solve the problem of time series prediction. We compare CNN and LSTM as an **expansion** to the assignment.

2.3.2 Architecture

For CNN, we use Keras to build. The main architecture is a Convolutional layer, a max pooling layer, and a Flatten layer, finally followed by two Dense layers.

2.3.3 Hyperparameters

Name :	Value	Name :	Value
filters :	64	learing_rate :	0.01
kernel_size :	2	window_size :	40
activation_function :	'Relu'	cost function :	MSE
optimizer :	'adam'	batch size :	64

2.3.4 Prediction

We test the impact of parameters on the current model. Since some of the parameters have minimal impact, we only choose the two most important parameters (i.e., the window size and the batch size) with the greatest impact on the time series.

**(*In this part, in order to limit the content of the report, we will only show the results of prediction, we will not show the MSE vs Epochs figures here.)* **

2.3.4.1 window size

In this regard, we test the performance of the model with three values: 10, 40, 100. The experiments illustrate that the **best window size is around 40**; when it is close to 100, the prediction is kind of weird. The results are shown in figure 4.

2.3.4.2 batch size

In this regard, we test the performance of the model on three values: 32, 64, 128. After the experiments, we found that the **best batch size is around 64**, if the batch size is too small, the prediction will be very small. If the batch size is too big, the results will not be stable, The results show in figure 4.

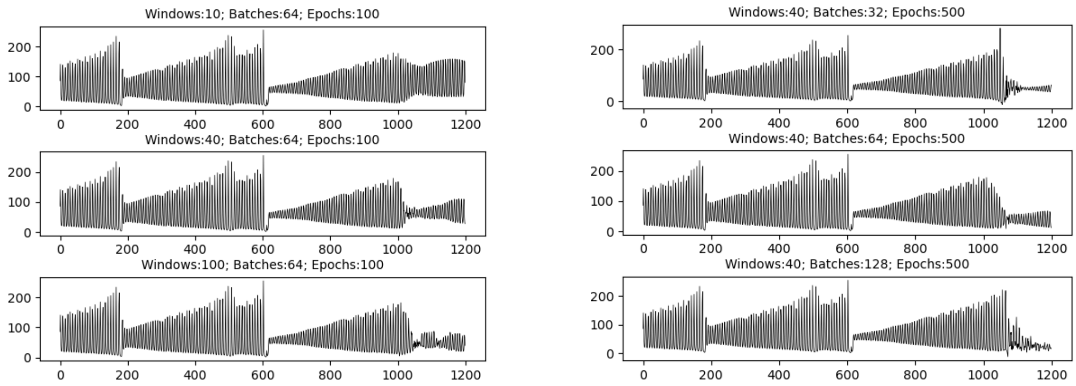

Figure 4. CNN predictions with different window sizes and different batch sizes

2.3.5 Mean Absolute Error

The mean absolute error (MAE) between the predictions and test data is around **50.347**. Figure 5 shows the comparison between predictions and test data.

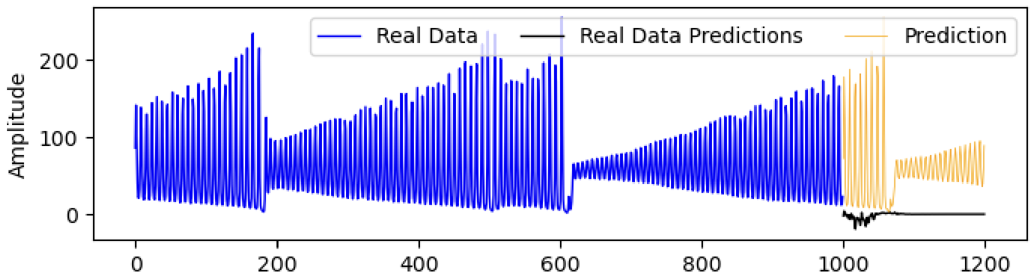

Figure 5. Comparison between CNN predictions and test data

2.4 Comparison

The aforementioned results show the LSTM predictions are better than CNN. The sequence is likely to be a kind of signal with a fixed time period. We can also observe the shape of predictions in Figure 3 and Figure 4. The absolute error in Figure 3 indicates that the LSTM model learns well in the aspect of frequency, time period, and amplitude. The maximum absolute error occurs near the junction of two consecutive time periods.

3. Conclusion

In this assignment, we completed the time series prediction by building the LSTM model and the CNN model and conducted many interesting experiments to explore the impact of different parameters and normalization. After this assignment, we obtain a deeper understanding of LSTM and CNN. Thanks for reading!