Neural Networks and Deep Learning (2024)

Assignment (2):

Recurrent Neural Networks in the world of stocks

Shiraz University

Due date: 8th Khordad 23:59 [28 May 2024]

1 Introduction

Simple Recurrent Neural Networks (RNNs), such as Jordan and Elman networks, were among the earliest architectures developed for processing sequential data. While more advanced variants like LSTMs and GRUs have gained popularity in recent years, these simpler RNN models can still be effective tools for some applications.

In the domain of stock markets, simple RNNs can be applied to model and predict stock prices, trading volumes, or other financial indicators based on their historical values and patterns. The sequential nature of stock data, where each day's price is influenced by previous days' prices and market events, makes RNNs a suitable choice for capturing these temporal dependencies.

2 Problem Statement

For this assignment, you need to leverage simple RNNs to forecast the Tehran Stock Exchange Index by analyzing its historical price data.

2.1 Dataset: [DOWNLOAD LINK]

The provided dataset contains four years of historical data for the Tehran Stock Exchange Index, including the daily open, close, and final index values.

2.2 RNN for stock perdition

During the class lectures, you were introduced to different types of simple RNN architectures, such as the Elman network and the Jordan network. For this assignment, you need to select one of these architectures and implement it to predict the future index values of the Tehran Stock Exchange.

2.2.1 Construction of Sample Set

The stock price prediction problem is actually a time series problem, which can be expressed by the following formula:

$$x_n = f(x_{n-1}, x_{n-2}, \dots, x_{n-N})$$

This formula means that the closing price of the previous N trading day can be used to predict the closing price of the next trading day. The data of 966 closing prices should be divided into training samples and test samples; for the training samples, $x_1{:}x_N$ are selected to form the first sample, where $(x_1, x_2, \dots, x_{N-1})$ are the independent variable and x_N is the dependent variable, and $x_2{:}x_{N+1}$ are selected to form the second sample, where $(x_2, x_3, \dots, x_N)$ are the independent variable and x_{N+1} is the

dependent variable; and the rest can be carried out in the same manner; finally, a training set is formed as follows:

$$\begin{pmatrix} x_1 & x_2 & \cdots & x_{N-1} & x_N \\ x_2 & x_3 & \cdots & x_N & x_{N+1} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_i & x_{i+1} & \cdots & x_{i+(N-2)} & x_{i+(N-1)} \end{pmatrix}$$

In this set, each row is a sample, and the last column is the expected output. These samples are fed into RNN for training, and then the network model can be obtained.

If N is set to 7, it means that the closing price of the day is determined by the closing price of the previous six trading days.

For this part, you can directly use functions like *preparets* in MATLAB and its equivalents in Python.

3 Tasks

1. You must justify your choice between different RNNs you've learned during class lectures.
2. Use the last 150 days as your test set and the rest of data for training set.
3. You might need to normalize your data but it may bring some issues for this particular problem, think about these potential issues and offer a solution. **Hints:**
    a. Choose your normalization function wisely.
    b. Consider the implications of using the entire dataset, including the test set, for computing normalization parameters.
4. Explore the effects of using different values for parameter N.
5. Validate the effectiveness of your model by plotting its predictions alongside the true index values of Tehran Stock Exchange for both the training and test sets, **in one single plot, together**, ensuring your final results can be reliably applied to stock data analysis and prediction tasks.

    Here's an example of how your final results plot should look:

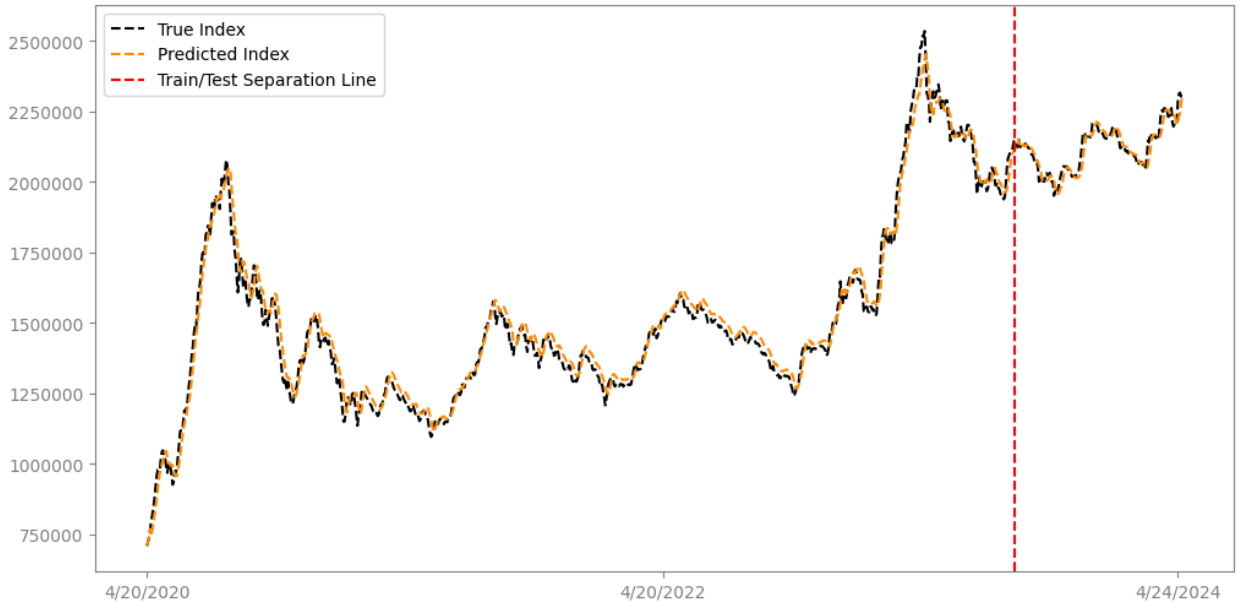

Figure 1: True index values of the Tehran Stock Exchange alongside predicted values generated by an RNN model.

Notes:

- Allowed programming languages: Python, MATLAB
- Any sign of cheating would result in a zero grade for this assignment.
- You should upload your submissions at:
  https://quera.org/course/add_to_course/course/16595/
  All of the files should be in a ZIP file named in this format:
  "FirstNameFamilyName-SudentNumber.zip"
  Ex: "AmirZamani-4023040.zip"
- Your reports should be in a PDF file including: key points of your implementation, explanation of your chosen approach, reports of your final results and answers of assignment questions (if given).