Assignment 3: Recurrent Neural Networks for Stock Price Prediction, Deep Learning Fundamentals, 2024
Due: 11:59pm, 07/12/2024

Abstract

This assignment is to implement, describe, and test Recurrent Neural Networks (RNNs) to predict stock prices in the future, using Google Stock Price dataset on Kaggle. The submission will take the form of a conference paper.

1. Introduction

You need to explain the application, and review the related work here.

2. Task and dataset

You are required to build a RNN to predict Google Stock Price using the Kaggle dataset available from `https://www.kaggle.com/rahulsah06/gooogle-stock-price`. You can use vanilla RNN, Gated Recurrent Unit (GRU), or Long Short-Term Memory (LSTM) as taught in the lecture and demonstrated in the workshop.

The dataset provides the training set and testing set. But you should further split the original training set to training and validation sets. Validation set is for you to tune the hype-parameters. The dataset contains open, high, low, close, volume of the stock price for each day. The task is to train a model that is able to predict future stock price. A common way in stock market prediction is to use the past N days data (*e.g.* $N = 30$, including all open, high, low, close, volume) to predict the next M days data (*e.g.* $M = 1, 2, 3$, including all open, high, low, close, volume). There are other design choices, and it is up to you how to model this problem using a RNN. Whatever choice you make, you need to justify and evaluate your choice. Make it realistic. For example, people wish to predict the next day's price ($M = 1$), because they wish to predict to sell or buy. Your work will be mostly evaluated on performance, justification, analysis, insights, and usefulness (*i.e.* if it would be useful at all if someone wishes to use it).

3. Submission

The submission takes the form of a conference paper, and your code.

3.1. Report/paper submission

You need to write a paper such as might be submitted to a conference, and specifically a paper such as might be submitted to CVPR[1], which is one of the best conferences in Computer Vision. The paper must be in the CVPR format, and submitted as a pdf document. By far the easiest way to achieve that is to use LaTeX. LaTeXis a very powerful document formatting package, it's free, and it is the only way to generate well formatted documents that contain maths. It's also the easiest way to generate well formatted documents in general.

All the information about the CVPR paper format is available on their web site[1]. The paper must be all your own work, with no text copied from any other document. The paper you submit must be in the format specified for CVPR 2020, which is specified as part of the author instructions[1]. The easiest way to achieve this is to download the LaTeXtemplate and use that. You can use some other means if you really want to, but your paper needs to conform to the CVPR style specification. The only exception is that I don't mind if you use a4 paper rather than their preference for letter paper (it's a US conference).

Good paper exemplars? Go to the CVPR websites (not limited to 2020), and read the BEST PAPER AWARD papers and the oral papers in the past. You may not fully understand the the papers, but they should give you some idea what a look paper should look like.

3.2. Assessment

The purpose of the paper is to demonstrate that you understand the problem, and the solution. This means that your submission should have sections which broadly cover the following

- An introduction, which describes the problem, and the method/algorithm (5 points);

- A background section which describes competing approaches to the problem. Achieving this requires that you understand what the competing approaches do, how they do it, their advantages and shortcomings, and how they compare to the current approach. The methods you compare against here may well perform better than the method you are describing. The idea of this section is not that you show that yours is necessarily the best method available, but rather that you show that you understand enough about the literature in the area to be able to put it in context (10 points);

- A description of the method. This will typically require explaining some part of the algorithm in detail, and providing examples illustrating its effects and deficiencies. If you propose an improvement then you should describe how your method works, in enough detail that a reasonably skilled person would be able to implement it (30 points).

- Experimental Analysis. Describe the tests you have run, and your motivation for having run them. Report the results of the tests and the conclusions that you have drawn. Again, the goal is not to show that your method outperforms all comparators, but rather that you understand what the method aims to achieve, and can devise, execute, and report upon a set of tests which demonstrate whether it does so. If you have improved upon the base method then you have an opportunity here to show that your improvement is well motivated, and possibly even that it works (30 points);

- Code. In order to be able to test the method you will need to implement it. You can implement it in Python (preferred) or Matlab. You will also need to submit your code in `https://github.com`, which leaves the time stamp. **You can make your github code publicly available, OR it is important for you to give access to the teachers and tutors.** The tutor will not be marking the quality of your code, only checking that it shows enough evidence that you wrote it yourself, submitted in time, and no obvious error(s). Please make a separate section (Code section) in the report with the link to your github code (20 points).

- Conclusion. Demonstrate that you have learned something worthwhile from the process, possibly including ideas about what you might do to improve the method you are reporting on (5 points).

References

[1] CVPR. Ieee computer society conference on computer vision and pattern recognition. See `http://cvpr2020.thecvf.com/submission/main-conference/author-guidelines`.