Forecasting Old School RuneScape’s Market using an LSTM RNN

Matthew Manoly, Steven Proctor, Harrison Ratcliffe, Zachary Taylor

December 10th, 2018

Abstract

In this paper, a Long Short-Term Memory Recurrent neural network is used to forecast the prices items in Old School RuneScape’s in-game market. A dataset consisting of item prices and trends over a month-long period was constructed and used to test the network. The LSTM RNN looked at sequences of four days and attempted to predict the following fifth day. A custom accuracy equation was developed and used in place of Keras’ own accuracy equation as it was reporting inconsistent accuracies during testing. After being tested on the dataset, the LSTM RNN achieved an average accuracy of 18%.

Introduction

Forecasting financial markets using a neural network has been pursued for both industry and academic purposes for many years now. Starting as early as the 1990's, researchers have attempted to accurately predict markets using neural networks. Many people who invest in the stock markets do so through stock brokers who act as a middle man and know how the market works. These stock brokers keep track of recent trends to try and get money back through frequent stock trades. In doing so, these stock brokers earn a commission based upon how much they make. Being able to predict markets could increase their commission size, or if used by the general public, could avoid having to pay commission at all.

Old School RuneScape is a Massively Multiplayer Online Role-Playing Game (MMORPG) developed by Jagex Inc. Like many other MMORPGs, RuneScape features an in-game marketplace, known as the Grand Exchange, where players can trade, sell, and buy items using in-game currency. The price of each item follows a supply-demand model comparable to real world markets with some influence from Jagex Inc. Just like stock brokers, players will often track market trends in order to determine the best time to buy or sell an item for the most profit. Thus, accurately predicting how the market will behave can greatly benefit players to increase wealth. This project aims to predict RuneScape's in-game market by using data on the trends of each item over a fixed period. To do this, it implements a Long Short-Term Memory(LSTM) Recurrent neural network (RNN).

Recurrent neural networks are a type of artificial neural network where connections between nodes form a directed sequence, performing the same task for every element of a sequence with the output being depended on the previous computations. RNNs achieve this by using an internal memory system, making them ideal for processing sequential data such as time series or speech recognition. LSTM is an architecture for RNNs that enables the neural net to remember values over long time-intervals and handles long term dependency issues that may arise when using RNNs [6]. Thus, LSTM RNNs are particularly useful for examining sequential data that may contain gaps or irregularities. This can be seen in similar projects which also attempt to accurately predict financial markets.

Current State of Similar Projects

Many projects and research today are still focused on accurately predicting changes in a financial market using artificial neural networks. These projects focused on forecasting changes in real world markets, such as housing markets or the stock market, rather than the markets found in video games. Despite there being little to no research on predicting in-game markets for video games, the underlying framework for predicting real world financial markets can be applied to in-game markets as well. Thus, this project can look at similar financial market predictions when determining which neural network to use and identifying any potential issues.

In recent years, there have been many projects trying to improve and evaluate how well financial markets can be predicted. Several different artificial neural network architectures have been used for market forecasting with convolutional neural networks (CNN) and LSTM RNN's as the two most common architectures. In 2015, a study suggested that an LSTM RNN worked best for processing data for financial markets due to its usefulness for processing sequential data. This study was focused on predicting China's stock market using records from Shanghai and Shenzhen using data from 1990 to 2010 [1]. They were only able to achieve an accuracy of 27.2% but recognized that the accuracy could be improved by refining the data and focusing on a certain stock, rather than processing all the stock market data at once [1]. A similar study conducted in 2016 using an LSTM RNN achieved a peak prediction accuracy of 59% and an average prediction accuracy of 54% when attempting to predict stock market changes for 6 companies across different industries [2]. It then applied the predictions in a stock trading simulator to evaluate the feasibility of the model, finding that the neural networks predictions resulted in a net profit across all 6 companies [2]. This study did not compare how other RNN based models, such as Continuous-Time RNN, or any CNNs [2]. Wei Bao, Jun Yue, and Yulei Rao then improved upon the use of LSTM RNNs by including stacked autoencoders (SAE) and wavelet transforms (WT), allowing the neural net to learn more irregular features [7]. Their model outperformed 3 other architecture models, each tested on the same 6 stocks [7].

In addition to LSTM RNNs, CNN's have grown in popularity for forecasting financial markets. When applied to market forecasting, CNNs were found to perform reasonably well, achieving reasonable accuracies when predicting both market trends and volatility [8]. Persio and Honchar [3] compared how different artificial neural network architectures perform when forecasting stock markets. Their paper evaluates an RNN, CNN, and a multi-layer perceptron (MLP) all trained and tested on the same data using Keras, with slight tuning based on the architecture. They found the CNN to be the most accurate for predicting the data, however it only slightly outperformed the RNN and MLP [3]. Up until this point, the current state of the art neural network model for forecasting financial markets was a convolutional neural network with a single event embedding layer (EB-CNN), proposed by Ding et al [9]. They determined that a CNN based prediction model worked best because CNNs can quantitatively analyze the influence of events over long sequences of data and can extract the most representative feature for the prediction model [9]. The event embedding layer in their network helped to reduce feature sparsity present in the dataset and was found to work better than simple event layers or a word embedding layer [9]. They were able to achieve a prediction accuracy of 65.08% when tested on stock market data. In 2018, Xu et al. [5] attempted creating a new architecture for forecasting stock market trends that could outperform the EB-CNN model proposed by Ding et al. [9]. They used an artificial neural network with an LSTM layer, convolutional layer, and an entity embedding layer (EB-CNN-LSTM), achieving an accuracy of 66%, narrowly beating the EB-CNN model [5]. They compared their model with other CNN based architectures proposed by previous projects and surpassed them as well when trained and tested on the same dataset [5]. Their EB-CNN-LSTM seems to currently be the best model of neural network for achieving the best accuracy when predicting financial markets.

Predicting financial markets is a challenging task. Issues in the dataset such as gaps, irregularities, or noise can heavily affect accuracy, as well as the choice of neural network architecture. As seen in the Chinese Stock Market study [1], when attempting to follow the trends of many stocks at once, there can be a large loss of accuracy. Additionally, since results are so heavily dependent on data, it is difficult to compare architectures across multiple studies unless tested on the same dataset, as was done in Persio and Honchar [3]. It seems that currently the best accuracy is achieved by both narrowing the focus of the dataset and combining different types of neural net architectures.

Implementation

The dataset for this project was planned to initially contain information on each item available in RuneScape's market. This was accomplished by gathering data from an online item database API service which contains information on each item's current price and trend as well as today, 30 days ago, and 90 days ago. Using a web scraper built in Python, information for each item was extracted in JSON format. The web scraper saved each item's current, today, and 30-day data for this dataset, scraping it each day over the period of a month. Using the today and 30-day trends, a dataset of almost a full month of sequential data was constructed. The data was stored as text files, with each text file containing information about each item for a specific day. This method of acquiring the dataset was chosen because an already completed dataset for RuneScape's market could not be found. After gathering as much data as possible, each day was read into a Python notebook file, parsed to select useful data, and stored in a multi-dimensional matrix to be fed into the neural network.

An LSTM RNN was initially proposed for this project, as they excel in predicting, processing and classifying data in a time-series. An LSTM RNN architecture was chosen over a CNN and other architectures from previous studies due to prior experience using an LSTM RNN and deciding that it tends to be the most reliable architecture for financial time-series forecasting. Since the data is a time-sensitive group where the trend over a few days determines the next day, an LSTM RNN seemed like the obvious choice.

The network looks at four days and tries to determine the fifth from the data it is presented. Because of this, a custom accuracy equation was created to determine the correct accuracy to test against the networks predictions. The equation $1/(1+|x - y|)$ -- where x is the actual price for the day and y is the predicted price for the day -- gives an accuracy that moves towards zero as the prices get further from each other. The converse is also true, as the equation returns a number closer to one as the predicted price moves closer to the actual price. After averaging every item, the accuracy value will show the total accuracy for all entries. When the neural network was being constructed, two LSTM layers were found to produce the best accuracy. Adding any additional LSTM layers resulted in the accuracy growing much slower and significantly increased runtime. Shuffling the data also did not seem to impact accuracy, as it remained fairly consistent. The network starts rather slowly, with most tests starting at around a single percent of accuracy but quickly reaching up to 24%.

Results

The network produced varying results from many tests of different layers and parameters. With a single LSTM layer, batch size of one and sequence size of two, the network reached an average accuracy of 10% with a peak at 15%. The LSTM RNN returned an average accuracy of 11% with a peak accuracy of 19% when using a single layer with a batch size of one, a sequence size of four and twenty-two days of data. Adding an additional LSTM layer resulted in an average accuracy of 13%, a peak accuracy of 24.3%, and an average of 18% over the final 50 epochs. The goal was to match the 27% accuracy from the LSTM for stock market predictions article by Kai Chen, Yi Zhou and Fangyan Dai [1] since their model is similar to the one in this paper. Both models are tested on a large and diverse set of data, so they seemed to be a likely candidate to match. With an accuracy of 24%, the neural net can predict what the price will be on the following day for about 24% of items. Several other parameter configurations were tested, including increased batch size, increased and decreased learning rate, and added dropout. All of these resulted in a lower accuracy, sometimes significantly.

The results from this model, when compared to the accuracies and results from previous studies, are somewhat lackluster. This may be due to a number of factors, including the choice of neural network, the parameters and layers of the network, and the dataset itself. An LSTM RNN seemed like the most logical choice; however, a higher accuracy may have been achieved if a

CNN was used, as seen in Persio and Honchar [3], Ding et al. [9], and Xu et al. [5]. Additionally, different parameters and layers could have been tested to try and increase accuracy. More LSTM layers could have been added, given that more time was available to test the neural network. Lastly, the dataset may have been too small or simply contained too many irregularities. Despite these potential problems, the networks results show that RuneScape's markets can be somewhat accurately forecasted but requires a better model for more accurate predictions.

Successes and Failures

Several problems were encountered as the project progressed. When first attempting to gather the data the web scraper crashed for certain items and if server requests were made too quickly then a bad server response was returned. This was mitigated by removing the crashing items from the dataset and slowing down the polling rate to not trip Jagex's anti-DDOS protection. Additionally, a day of scraping was missed due to errors within the web scraper causing a gap in the data; however, this likely did not heavily impact accuracy due to the LSTM layer in our neural network. It is also important to note that the data was not scraped at the same time every day, which may have introduced slight irregularities. Another possible issue with the data is interference from Jagex Inc. Jagex releases updates every Thursday that can, and do, alter item prices. Furthermore, these updates can increase or decrease the demand of an item, affecting the price. This causes irregular rises and drops which may have an impact on prediction accuracy. Another issue encountered was with the 30 days ago data. For some reason, whenever that data was added to be trained on, the network struggled to gain even a single percent of accuracy over 100 epochs. This occurred both when the 30 days ago data was trained on by itself, and when it was combined with the current data to form a larger dataset. Looking at the data, it is not clear why this happened, and after several attempts to fix it, it was decided to not include it. Finally, Keras' accuracy equation for LSTM RNNs with a sequence size higher than one returns very round numbers, a fraction of the days it's currently looked at. This is not the best approach for reporting the networks accuracy as it would require getting the entire day correct to gauge itself as accurate. This issue was solved by creating a unique accuracy metric, as discussed above.

Despite encountering a multitude of problems, several choices for this project worked rather well. Scraping each items information from Jagex's Exchange API was fairly easy and gave a large amount of data to work with. While it does not contain lasting records for the items, it does give enough information to construct a sizable dataset and can continue to grow as long as it is consistently scraped. The custom accuracy model solved several problems during testing and helped gauge accuracy far better than Keras' implementation. This proved to be the biggest success as it helped quantify the effectiveness of the network.

Future Work

In the future, this neural network could be further refined to improve accuracy. This can be done by re-configuring the parameters or changing to a different architecture. Some architectures to consider would be a CNN or even a LSTM RNN with a convolutional layer, as seen in Xu et al. [5]. From there, different architecture could be tested against each other on the same dataset to determine which gives the best accuracy. The dataset could also be expanded to include more items or a longer sequence of days. Gaps and irregularities in the dataset could be reduced by consistently gathering data at the same time each day, however there is no way to account for interference from Jagex Inc. An additional way to improve accuracy might be focusing on a single item or only a few items at a time. Even though this might not actually improve the networks prediction accuracy overall, it could provide a better accuracy for the market trends of a single item and is worth testing. Another avenue of future work is implementing a function that returns the predictions from the neural network. This would allow for testing the predictions in RuneScape to see if they result in a net profit. If so, then the network could be used by players looking to track certain items. Lastly, the neural network proposed in this paper could also be tested on other video game markets, specifically other MMORPGs.

Conclusion

This paper proposed using a recurrent neural network with an LSTM layer to forecast RuneScape's in-game market. The dataset was manually constructed using Jagex's online item database service, gathering information on the price trends of items over the span of a month. The data was then parsed and fed into the neural network as a multi-dimensional matrix. A recurrent neural network with a single LSTM layer was tested on the dataset, achieving an average accuracy of 15% with a peak accuracy of 19%. By adding a second LSTM layer, the average accuracy was increased to 18%, with a peak accuracy of 24%. While this result was less than ideal, the model in this paper still somewhat answered the question "can RuneScape's in-game market be accurately forecasted?" Further work can be done to improve this model's overall accuracy. Once that has been achieved, the network could be used by players to accurately predict video game market trends.

References

[1] Kai Chen, Yi Zhou, and Fangyan Dai. *A LSTM-based method for stock returns prediction: A case study of China stock market,* 2015 IEEE International Conference on Big Data (2015), 2823-2824, DOI 10.1109/BigData.2015.7364089.

[2] Qiyuan Gao. *Stock Market Forecasting using Recurrent Neural Network,* University of Missouri-Columbia, 2016.

[3] Luca Di Persio and Oleksandr Honchar. *Artificial Neural Networks Architectures for Stock Price Prediction: Comparisons and Applications,* International Journal of Circuits, Systems, and Signal Processing **10** (2016), 403-413.

[4] Xiaochen Chen, Lai Wei, and Jiaxin Xu. *House Price Prediction Using LSTM,* The Hong Kong University of Science and Technology, 2017, arXiv:1709.08432.

[5] Bo Xu et al. *Stock Market Trend Prediction Using Recurrent Convolutional Neural Network.* NLPCC: CCF International Conference on Natural Language Processing and Chinese Computing (2018), 166-177.

[6] Jitendra Kumar, Rimsha Goomer, Ashutosh Kumar Singh. *Long Short Term Memory Recurrent Neural Network (LSTM-RNN) Based Workload Forecasting Model For Cloud Datacenters.* Procedia Computer Science **125** (2018), 676-682.

[7] Bao W, Yue J, and Rao Y. *A deep learning framework for financial time series using stacked autoencoders and long-short term memory,* PLOS ONE **12** (2017), 1-24. doi:10.1371/journal.pone.0180944

[8] Jonathan Doering, Michael Fairbank, Sheri Markose. *Convolutional Neural Networks Applied to High-Frequency Market Microstructure Forecasting.* 2017 9th Computer Science and Electronic Engineering (CEEC) (2017), 31-36. doi: 10.1109/CEEC.2017.8101595.

[9] Ding, X., Zhang, Y., Liu, T., et al. *Deep learning for event-driven stock prediction.* International Joint Conference on Artificial Intelligence (2015), 2327-2333.