Eric Escareno
Christian Galleisky
Irvin Budwal
Joseph Lopez

CS478 Final Report: Stocks Deep Learning

Abstract

Our primary focus for this task is to utilize machine learning to predict graphs based on results from older stock market data. Designing the model to create accurate results and meaningful data is essential to this project. The data needs to be analyzed in such a way that information and patterns can be extracted from the output. Our approach has been to utilize Long Short Term Memory (LSTM). Essentially, we are composing cells as an assembly of input gates, output gates, and forget gates. The idea behind this is for memory to be stored in the cell for future use. The gates feed information to each other and regulate passing values through their network (multilayer neural network). This is short term memory that will last for a long period of time (values and information are retained within the gates/cells).

Our team initially decided to try to model the emotional state of the Nasdaq stock market in order to understand when the market is emotionally overbought or oversold. What we hoped to do is to understand when we should expect some up or down movement in a particular security(equity/stock), so as to not be shocked when it does actually occur. A level of technical analysis was thought to be required for adequate implementation of the model. We were able to learn various methods of technical analysis but found the undertaking of building our own deep learning system with them difficult. We have never done any work in Machine Learning and thus, worked through trial and error to refine the scope and goals of our project. We believe we did our best to deliver a quality product/analysis by semester's end but given more time we could have fleshed out more aspects of our analysis.

Introduction

Market psychology paired with deep learning can give us greater insight into how and when we should make trades. This is interesting to our group as most of us have very little investing experience so if novices can figure out how to leverage deep learning effectively there are many more possibilities. Readings that our team will utilize to gain contextual understanding in order to tackle this project are: *Getting Started in Technical Analysis* by Jack Schwager, and *Technical Analysis* explained by Martin Pring. Both books give detail on technical analysis which can be

used to make better predictions in stock price. A large portion of the data and formatting we will use can be found at:
https://www.kaggle.com/datasets/jacksoncrow/stock-market-dataset

Related Works
https://www.analyticsvidhya.com/blog/2018/10/predicting-stock-price-machine-learningnd-deep-learning-techniques-python/

https://medium.datadriveninvestor.com/predicting-the-stock-market-with-python-bba3cf4c56ef

https://deepnote.com/@deepnote/Stock-Analysis-in-Python-4cf17a26-d9bd-420a-a356-74d668ece385

https://www.kaggle.com/code/artemburenok/stock-analysis-monte-carlo-build-portfolio/notebook

There were many deep learning stock prediction projects for us to work in reference to and as most covered a quick overview of what was possible we felt we could improve upon existing work to build our own models. All of these resources were useful for us and gave crucial context which is why our goals and ideas changed many times before we settled on one. By far the biggest contribution to our project and our understanding of how predicting stock prices was even possible was this project:

https://www.kaggle.com/code/thebrownviking20/intro-to-recurrent-neural-networks-lstm-gru/notebook

Although relatively simple with only a few examples of using neural networks to predict stock prices, the project was well presented and easy for us novices to understand. We built and changed this project, incorporating our own dataset, different models, and applying our in class knowledge to get a unique approach to predicting these prices.

Data

We attempted to get current data by either web scraping on stock websites or trying a script that collects the data but ran into errors such as with the following link:

https://www.kaggle.com/jacksoncrow/download-nasdaq-historical-data/notebook

We were unable to find a means to quickly collect new data and decided to use existing data to build upon. Our initial research and brainstorming had us choose the Monte Carlo simulation method as our primary method to predicting stock prices, however this simulation proved unnecessarily cumbersome compared to the deep learning strategies we utilized. There are existing implementations of deep learning stock predictions which we built off of and tweaked. Our original expectation was to add different forms of data to include as context, but we had difficulty matching context to our specific historical dataset. Adding context to the model could prove useful if we were to expand upon this idea. We evaluated our results by checking whether

our model makes accurate predictions. Our biggest metric for accuracy was the root mean squared error and trying to minimize this value across various models. This allowed us to test our model under certain stressors, so that we can determine prediction accuracies. Essentially, by dividing data from our chosen subset into two segments, one segment is then used to train our model, while we use the other to compare it to. (Comparing independent vs dependent variable). Our results are representable as plots and we are able to compare the accuracy of our predictions across sample sizes to tweak our models. Ultimately, the goal is that our market analysis model will help streamline investment and trading for newcomers and people unfamiliar with that environment, for users to maintain their focus on spreads and trends.

We will evaluate our results by checking whether our model makes accurate predictions. We will need to learn technical analysis, which market psychology is an aspect of, to make our own predictions to compare against. Our results should be representable as plots and we should be able to compare the accuracy of our predictions across sample sizes to tweak our activation function. The ultimate goal would be to make the model as accurate as possible.

Essentially, we have made some changes to our project and its direction that we believe will be beneficial to our timeline. We have focused on reducing the approach of “emotional state” of the stock market as a means to replicate neural networking. On top of that, deciding to change our approach from exclusively following current data, to older backlogs containing lengthier existing data so we can better make analysis.

Deciding between both datasets was a difficult choice. Historical daily prices and volumes of all U.S. stocks and ETFs from (Kaggle) included a data set that only went up to 2017. It was very in depth and expansive, however the data included was static, brute forced to maintain numbers up to the year 2017. Historical daily prices of Nasdaq-traded stocks and ETFs also from (Kaggle) was the data set that we decided on. This is mostly because the dataset is extremely granular, the last dynamic update was in 2020, the contributors to the project included through and extensive documentation, and included was a data collection script that essentially allows us to update the stock data to more current “quarterly” fixtures.

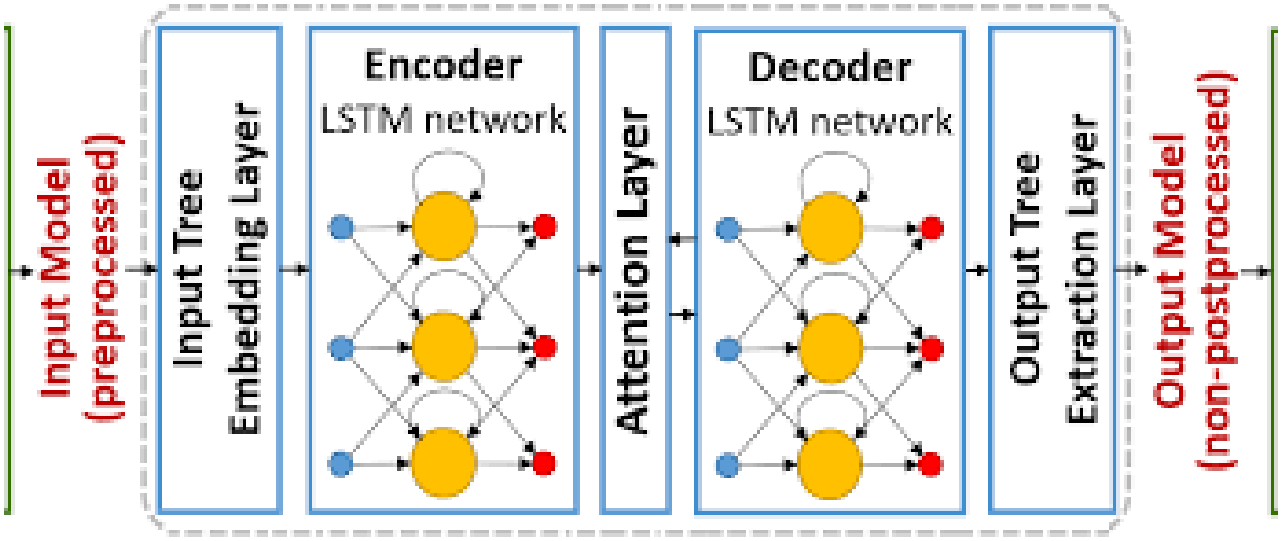
The basis for our model will essentially include close training and prediction according to moving averages. (This data can be cross referenced from data analyzed in the 1980’s to 1990’s)

The main force of our focus will be on developing a machine that will utilize LSTM. Long short-term memory will allow us to predict stock trends from specific sequences of variable lengths. From related works that have attempted this, we are confident in generating an algorithm that can successfully and effectively predict the movement of the stock market. This model would then be adjusted to individual stocks and stock exchanges, depending on the user’s desire.

The primary idea is that it can be applied to any stock and be an effective tool in future predictions. (Using it to predict apple stock, microsoft, etc.)

A latest iteration that we have seen utilized an LTSM algorithm to closely predict the trends from the IBM stock, with very minimal deviations. It's not 100% accurate, but the shape prediction is very close. We believe this will be an extremely useful tool for users that are in need of assistance for making smart buying choices, to help with involvement in the stock market.

Methods



We modified one of the researched related works to better improve and visualize the working prediction model. The original author's code only uses LSTM and GRU models along with a single graph to show accuracy. This approach seemed fairly minimal and we believe that it can be elevated using the tools we have learned about this semester. We utilized our knowledge of BiLSTM, Stacked BiLSTM, histograms, and other models/visualizations to better assess how well our model is doing. On top of that we refined our parameters and attempted to obtain faster runtimes. The input data is modified to fit the models. We found that the original parameter of 32 for batch size and 50 for epochs fits the model well into reasonable accuracy limits. We used a few different optimizers, not just one. Also the numpy reshape and MinMaxScaler function and API were invoked.

The input learning data was done on a 3X1 matrix with chunks of 60 data points as the major parameter. We did not use Validation data either, as opposed to the vast majority of our homework assignments. Rather, we used the predict command to obtain our results and accuracy.

Aside from our predicted/actual stock graphs and rsme values, we ran into difficulty getting the visualization methods taught in lecture to work with our dataset dimensions. This is likely our biggest failing as we can see general accuracy and the graph comparisons but are unable to analyze the learning curve, false positive/negative values, or overtraining of the model.

Experiments

In order to obtain the desired results, we tweaked our model's structure and hyperparameters to reduce the RSME value as much as possible. We attempted to implement ROC-AUC graphs to better understand our model's performance. Overall, we made sure the model predicted well in a short window of time. If the model is observed from a further distance we may misinterpret how accurate our model realistically is. A good example of this is with our original csv's dates ranging from 1962 to 2020. As the graph of the IBM stock is mostly flat before 1995 when using the full range of dates to train our model it created incredibly flat prediction lines which were poor compared to the actual graphs. By setting 1995 as the beginning of our data collection we can make better predictions across any stock.

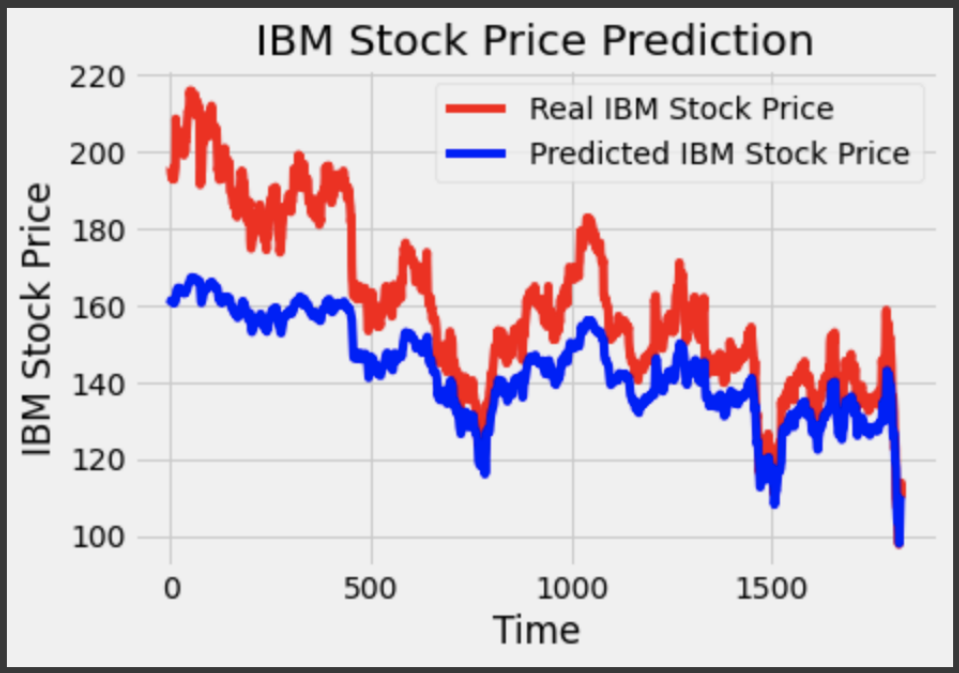
The basis of our training dataset is bound in values ranging from before the year 2003. Detecting these patterns was imperative to our research and to make our model successful. We realized that if we were able to feed trends and shapes that had appeared in earlier decades, the algorithm would have a solid indicator of where its values should go. We implemented a 70/30 split train, and the results were promising.

For further experimentation, we implemented an epoch of 6 and a batch size of 128. (This ultimately was to set up parameters for the BiLSTM architecture.

```python
# Training
epochs = 6
batch_size = 128

# LSTM layer architecture
n_lstm = 256
drop_lstm = 0.2

# The BiLSTM architecture
regressor_bilstm = Sequential()
# First BiLSTM layer with Dropout regularisation
regressor_bilstm.add(Bidirectional(LSTM(units=50, return_sequences=True, input_shape=(X_train.shape[1],1))))
regressor_bilstm.add(Dropout(0.2))
# Second BiLSTM layer
regressor_bilstm.add(LSTM(units=50, return_sequences=True))
regressor_bilstm.add(Dropout(0.2))
# Third BiLSTM layer
regressor_bilstm.add(LSTM(units=50, return_sequences=True))
regressor_bilstm.add(Dropout(0.2))
# Fourth BiLSTM layer
regressor_bilstm.add(LSTM(units=50))
regressor_bilstm.add(Dropout(0.2))
# The output layer
regressor_bilstm.add(Dense(1, activation='sigmoid'))

# Compiling the RNN
regressor_bilstm.compile(optimizer='rmsprop',loss='mean_squared_error')
# Fitting to the training set
regressor_bilstm.fit(X_train,y_train,epochs=50,batch_size=32)
```

Which also produced favorable results. Results that we felt confident in.

```python
# Visualizing the results for BiLSTM
plot_predictions(test_set,Bilstm_predicted_stock_price)
```



Conclusion

Based on our teams methods the GRU model was far and away the best performer in terms of accuracy. With the lowest rsme value and best predicted to actual graph similarity, it's clear

which model was making the most accurate predictions. The regular LSTM came in as close second with a very accurate model as well. By far the worst performing prediction model was our Stacked BiLSTM which include an RNN layer. This created a flat prediction curve and the worst rsme value of all the models. Ultimately, we were able to use our in class knowledge to test and tinker with our models but given more time we could have really pushed for better results and analysis. With our overaggressive ambitions we can see easy ways to improve and change this project further. Getting deeper into technical analysis and finding a method to use technical analysis to train a deep learning model could prove to be an interesting alternative to our approach. Collecting emotional data/context through polls or news could create a viable method to modeling the emotional state of certain stocks. For an easy way to expand upon our work we can include context such as:

Classical Economic indicators; capacity utilization, industrial production, inventories growth/contraction, unemployment rate, participation rate, savings rate, income growth rate, disposable income growth/contraction, etc…

Classical Business metrics: Price to earnings, price to earnings and growth, gross margin, operating income, Earnings before taxes interest and deductions (EBITDA), profit margin, Earnings per share, long term debt, assets, liabilities, assets minus liabilities, etc…

Classical stock chart ideas; shapes of stock price graphs and shapes of various stock chart metrics like RSI and Stochastic Oscillator, etc…

Overall we're proud of the progress we made together as a team to learn about the stock market, deep learning, and the limits of our ideas. Thank you for giving us an opportunity to grow.

Our Group plan for final deliverables:

- Abstract: Irvin
- Introduction: Chris, Eric, Joe, Irvin
- Related works research: Eric
- Data and processing: Chris, Eric, Joe
- Methods/approach: Eric, Irvin
- Experiments: Chris
- Conclusion: Joe