Hybrid stock market prediction network

Tóth Zsombor, Holló-Szabó Ákos, Szakács Béla Benedek

Abstract

Predicting how the stock prices will change with time is one of the most important topics in the field of deep learning. Our experiment tries to use a hybrid solution to the problem, using both the past values of the stock's prices, but we tried to use a primitive NLP (natural language processing) method, using news headlines at the time as well. The concept was that our network can be robust enough to extract valuable data from a very diverse and seemingly random corpus of texts. We created a network that predicts the direction of the stock price for the next few days, not the exact value. We tested it on Amazon's stock price data.

Introduction

Stock market prediction receives prime interest from big corporations, for obvious reasons. This means various experiments have been done on the subject. Big companies are always on the lookout for efficient methods. This meant our experimental network was not only developed for our own amusement, but it has relevance, even considering the simplifications employed and the results.

We started the project mostly based on the data available. We went with a different experiment first for this project (a midi-based music-generator network, where we planned to use GAN), but the lack of coherent, easy-to-parse data was a huge barrier. So, we have chosen a different project, for which we acquired a lot of data relatively easily. This was the stock market data for big companies and news headlines collected with web-crawlers from big news sites. After we had the data, we started the research on the methods we can use.

Research

We have looked at articles on the subject. There are similar experiments done [1][2][3][4][5][6], some even used similar method to enrich the input data with natural language texts.

We also looked at other sources (https://www.datacamp.com/community/tutorials/lstm-python-stock-market) for technical information. We wanted to create a network robust enough to utilize the patterns in the article headlines.

We also examined the possible types of neural networks to use. We were particularly interested in those that were previously used for similar experiments, although we did consider other, less conventional methods as well. The main focus were LSTMs [7][8][9] (Long-Short Term Memory) and 1-dimensional convolutional networks [10].

Design

The initial plan was to create two separate networks for both the stock market data and the news headlines. We wanted to build a hybrid network that utilizes both sub-networks and merges them into one that can predict the changes, preferably with a certain accuracy. For the NLP part (the news headlines) we used LSTM, because of the numerous applications in this field. For the time-series prediction (the stock market prices) we used a 1-dimensional convolutional network. We then merge the output of both and feed it into a bidirectional LSTM.

Our goal was, at the beginning, accurate prediction concerning the results. After some research and the initial experiments, we realized that even though the network was able to predict movements, the accuracy was nowhere near what we hoped for, so we set a different goal: predicting the overall direction of the price value in the near future. We settled for this idea because an article [link] has provided some insight on the matter.

LSTM has been used widely for NLP tasks, mostly because of the assumption that natural languages follow patterns that can be very well processed by these networks. We could have used them for the stock data as well, but because training LSTMs is a very slow procedure, and the convolutional networks learn much faster, we went with that. However, we still decided for a bidirectional LSTM for the merge part of the network.

Development

As we have stated before, the project has started with a different idea, and after abandoning that, we came to the conclusion that only after acquiring sufficient data should we start developing the network. We decided on a schedule where we started with downloading, converting and trimming the data, and only after that did we start working on the network itself. The development of the network was more or less straightforward: we created an initial design and refined it over time.

Acquiring data

We started the project by looking up available data. We had the problem of two different datasets: we needed stock market data and news data from the same time period. We have chosen Kaggle for two reasons: first, we knew that data from their website was more or less fit for deep learning uses. Secondly, we knew that it had vast amounts of data about stock market, and about news headlines as well. The initial approach was "get as many as we can", and after downloading sufficient quantities of data from different datasets, we started processing it.

This was probably the most challenging part of the project. We downloaded several datasets from the following sources:

Stock price data:

- dgawlik/nyse*
- stexo92/gafa-stock-prices*
- borismarjanovic/price-volume-data-for-all-us-stocks-etfs*

News data:

- therohk/million-headlines*
- aaron7sun/stocknews
- snapcrack/all-the-news

- census/business-and-industry-reports
- census/total-business-inventories-and-sales-data

We used the datasets that are marked with a *.

After that we have looked into the data and converted them into a simple dictionary. We used unix timestamps for the time, because it was easy to convert into. We had to analyze the data in terms of time intervals. We found that the length of time periods in which the data was recorded varies greatly across the stock prices data. I addition, when we examined the news headlines data, we found out that some of the sources were almost unusable, since they only contained data from a very small time period (a few months).

In the end we have chosen to include one news headline dataset that contained a few years worth of data, and that has proven to be enough for a rough training, validation and test set. We have synchronized the datasets based on their timestamps, and then separated the sets.

We have chosen 8 big companies: Apple, Microsoft, Google, Facebook, Amazon, Intel, Nvidia and AMD. Their stock value data was inspected, and the common time period chosen. We also looked at the Pearson coefficients between the data of the various companies to see if using other companies' data in the network would enhance the results, but we have abandoned the idea. We created some visualizations of the data for manual checking.

We used normalization for the data, and slit it into train, test and validate sets. We have chosen the 70%-15%-15% distribution. We did it for both the news headlines and the stock market prices data. We used the stock data shifted by one day as the expected outcome.

Learning

After some experiments, we developed an architecture that looks like the following:

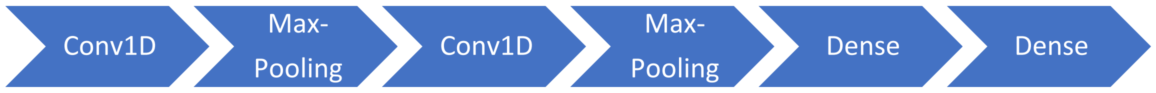


figure 1: The stock market prices branch

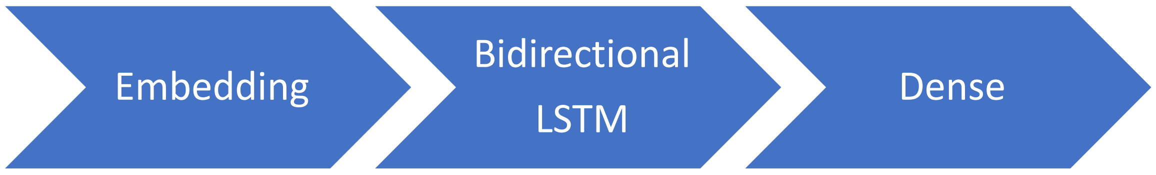


figure 2: The news headlines branch

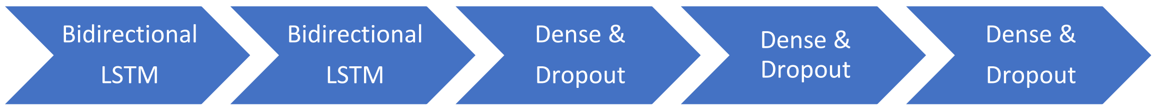


figure 3: The merged part

For the news headlines text we have only left in the top 300 words ordered by frequency, and examined the word count per day. We coded the words into integers and used that with an Embedding layer. We used ReLU activation, except for the last, output layer where we used sigmoid activation. We built in an early stopping mechanism that interrupted the learning if at the end of the epoch the value loss has not changed by more than 0.001 10 times in a row. We have saved the model after each epoch.

Evaluation

Our first attempts at training provided very bad, horrible results. We had to do a lot of tweaking in the structure of the network. We had discovered several bugs in the code until we reached acceptable results.

Train loss: ~0.04

Validation loss: ~0.04

Test Mean Absolute Error: 0.021

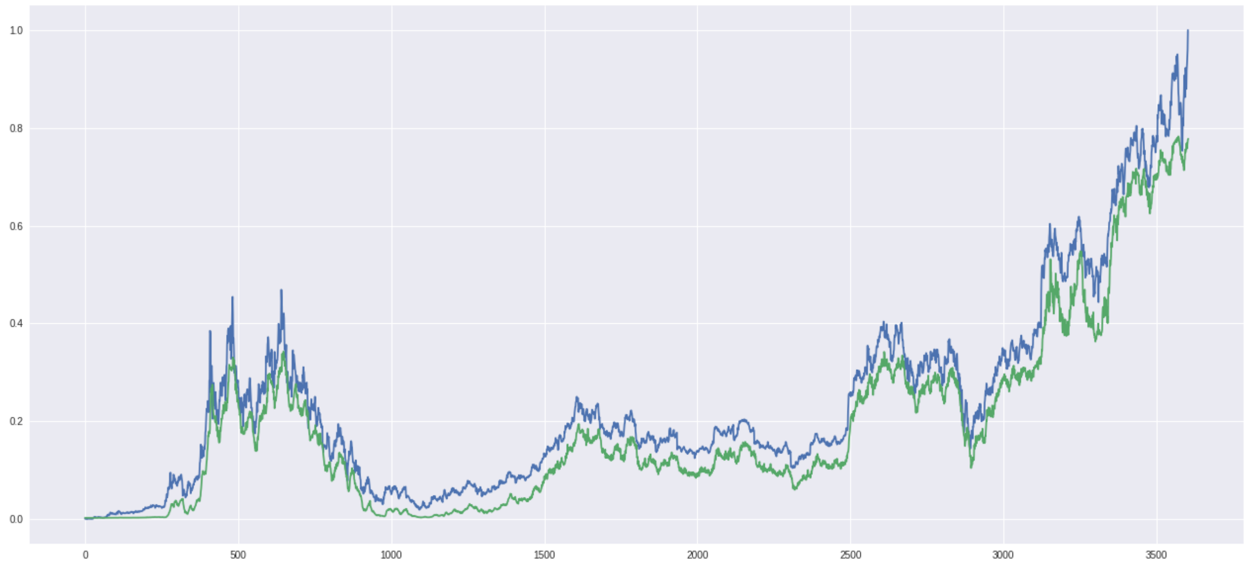


figure 4: The blue is the original training data, the green is the prediction

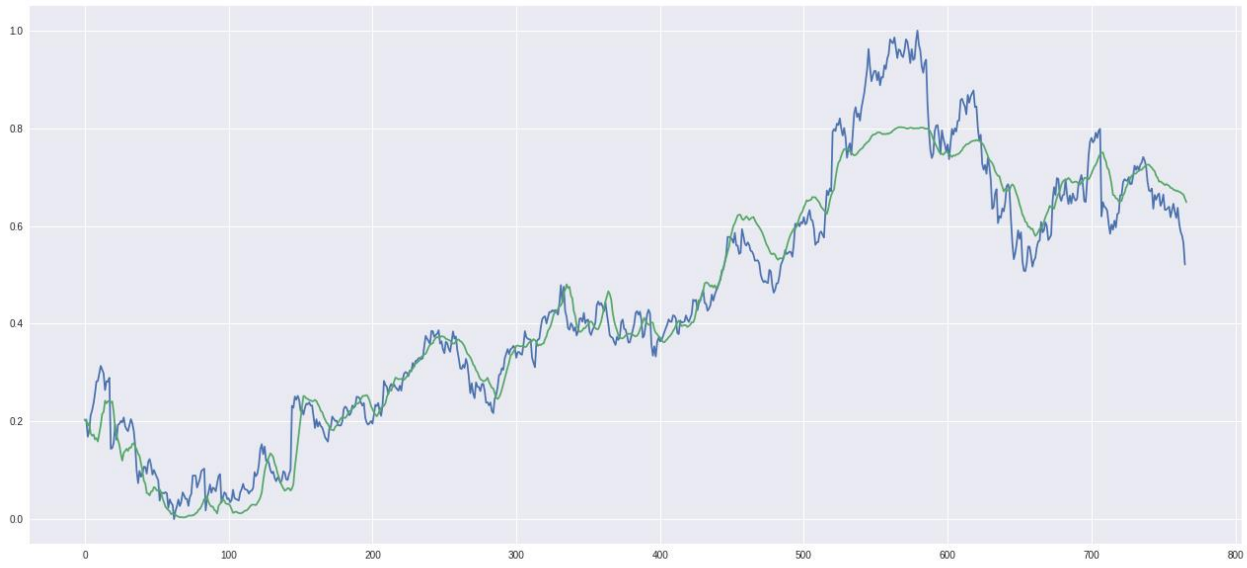


figure 5: The blue is the original validation data, the green is the prediction

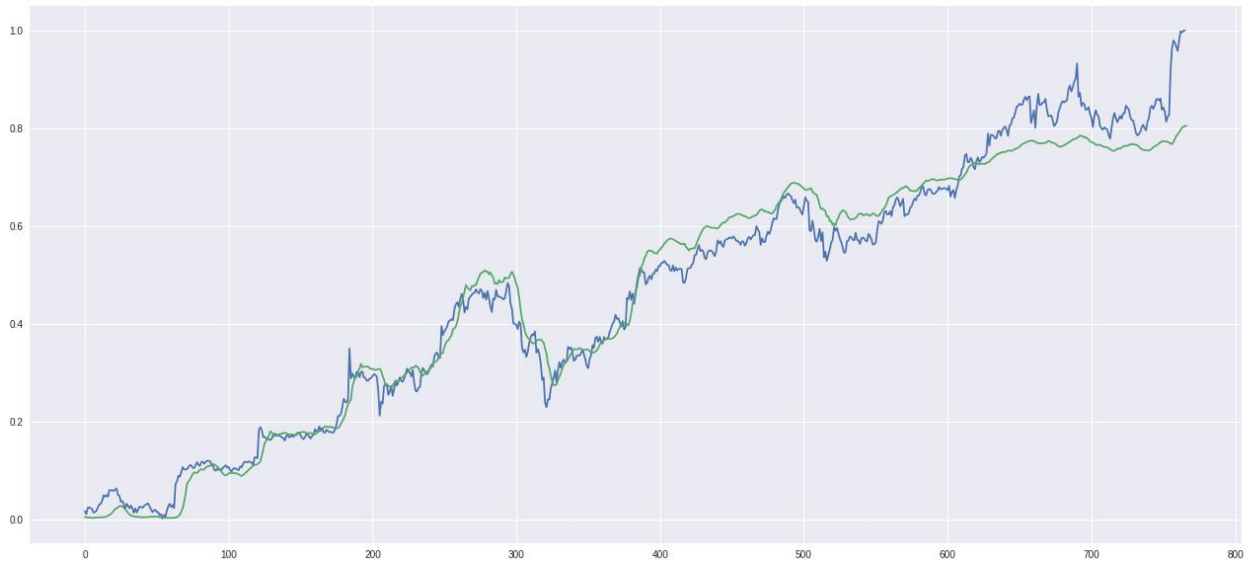
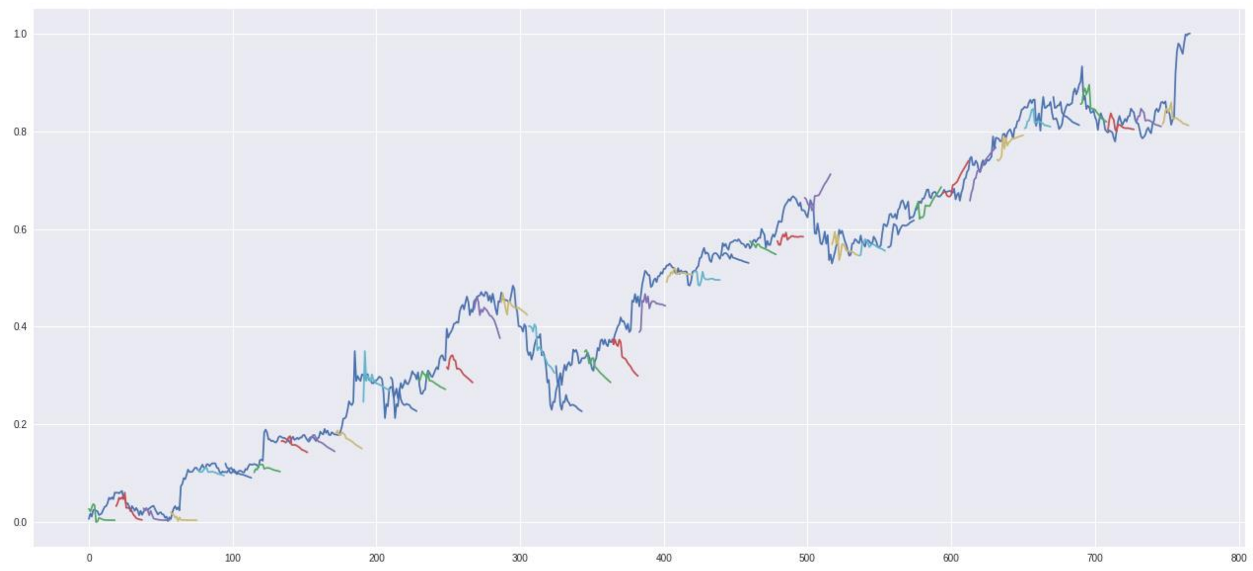


figure 6: The blue is the original testing data, the green is the prediction

Testing

Our goal was not exact prediction, but to forecast the overall direction for the next time period. For this, we tested it on the testing data, periodically making predictions. The results were the following:



The network can predict the direction, but the accuracy is not very great. We concluded that from the 40 test points the direction similar to the actual direction in 23 times.

Future development

The accuracy of the network needs significant improvement in the future. The NLP part needs a lot of refinement, as it is a fairly primitive one. The data we acquired was sufficient in quantity, but lacked quality, as it contained completely unrelated article headlines as well. This part definitely needs more development. LSTM is a very slow learner, we should look into using more convolutional networks instead in future versions.

References

[1] Xiong, Ruoxuan, Eric P. Nichols, and Yuan Shen. "Deep learning stock volatility with google domestic trends." *arXiv preprint arXiv:1512.04916* (2015).

[2] Akita, Ryo, et al. "Deep learning for stock prediction using numerical and textual information." *Computer and Information Science (ICIS), 2016 IEEE/ACIS 15th International Conference on*. IEEE, 2016.

[3] Ding, Xiao, et al. "Deep learning for event-driven stock prediction." *Ijcai*. 2015.

[4] V. Lavrenko, M. Schmill, D. Lawrie, P. Ogilvie, D. Jensen, and J. Allan, “Mining of concurrent text and time series,” in Proceedings of the KDD-2000 Workshop on Text Mining, 2000, pp. 37–44.

[5] P. Schumaker and H. Chen, “Textual analysis of stock market prediction using breaking financial news: The AZFin text system,” ACM Transactions on Information Systems (TOIS-09), vol. 27, no. 2, pp. 12:1–12:19, 2009

[6] Sundermeyer, Martin / Schlüter, Ralf / Ney, Hermann (2012): "LSTM neural networks for language modeling", In *INTERSPEECH-2012*, 194-197.

[7] F.A. Gers, J. Schmidhuber, F. Cummins “Learning to forget: continual prediction with LSTM” 9th International Conference on Artificial Neural Networks: ICANN '99, 1999 p. 850 – 855

[8] Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." *Neural computation* 9.8 (1997): 1735-1780.

[9] Gers, Felix A., and E. Schmidhuber. "LSTM recurrent networks learn simple context-free and context-sensitive languages." *IEEE Transactions on Neural Networks* 12.6 (2001): 1333-1340.

[10] LeCun, Yann, and Yoshua Bengio. "Convolutional networks for images, speech, and time series." *The handbook of brain theory and neural networks* 3361.10 (1995): 1995.