CS230

Long Term Stock Prediction Based On Financial Statements

Shujia Liu
shujia@stanford.edu

Abstract

This paper proposes a model with LSTM and fully connected layers to predict long term stock trendings based on financial statements. Two data augmentation techniques are applied on structured data: 1) adding random noise to data fields; 2) erasing partial information from training examples. The performance of the proposed approach is demonstrated on real-world data of about 7,000 stocks listed on Nasdaq.

1 Introduction

Predicting stock trending is a popular topic with the rise of machine learning. Some build models based on feature engineering with key indicators, such as Price-to-Book Ratio, Price-to-Earnings Ratio, Debt-to-Equity Ratio, which requires a handful of human efforts. Some build models based on financial news, which could suffer from insufficient sources of news. A majority of the research focuses on short term prediction, trying to maximize the return on investment in a minimum amount of time. Short term prediction has a relatively higher risky level, also subjected to higher short-term investment tax penalties. There are way less research papers for long term prediction of stocks, it could be due to people are less interested in long term prediction, relative work is done but not published. This project focuses on building an end-to-end LSTM model for long term stock prediction based on historical financial statements (including balance sheet, income statement, cash flow statement).

2 Related work

Some related work includes using financial key indicators to predict stock trendings, which needs quite a bit of work in feature engineering [5]. Some develop a specific strategy to maximize investment return based on accounting ratios [4]. Some research work focused on training models with pure numerical information (historical stock price) for stock prediction [6], some focused on training with pure textual information (financial news) [7], some used both [1]. Training with numerical information could predict stock trendings well on historical data, due to the fact that there are corelations between stocks in the same industry in the same period of time, but may not generalize well to the stock trendings in the future due to market shifting over time. Training with textual information could suffer from insufficient scources of news: one example is that Gamestop traders could discuss their strategies in reddit today, but move to discord tomorrow, which is hard to keep track of relative news or discussions of Gamestop's stock. Most of the research focused on short term stock tranding prediction [2], which has a relatively high risky level: one example is the roller coaster stock price changes of Gamestop around the end of January, 2021. Some used convolutional neural networks to aggregate the impact of news on stock prices in a longer term [3], like one month or two.

CS230: Deep Learning, Spring 2021, Stanford University, CA.

3 Dataset and Features

The dataset contains about 40,000 items: 38,000 in training set, 1,000 in dev set, 1,000 in test set. Each item in the dataset contains 3 continuous data points for one stock, including features from balance sheet, income statement, cash flow statement. Dataset is crawled and parsed from `https://www.nasdaq.com/`. Examples of data could be founded in `https://github.com/sugia/tradeX/tree/main/data`.

Features used in models are:

1. Balance sheet features, including 30 data fields: cash and cash equivalents, short-term investments, net receivables, inventory, other current assets, total current assets, long-term investments, fixed assets, goodwill, intangible assets, other assets, deferred asset charges, total assets, accounts payable, short-term debt / current portion of long-term debt, other current liabilities, total current liabilities, long-term debt, other liabilities, deferred liability charges, misc. stocks, minority interest, total liabilities, common stocks, capital surplus, retained earnings, treasury stock, other equity, total equity, total liabilities and equity.
2. Income statement features, including 18 data fields: total revenue, cost of revenue, gross profit, research and development, sales general and admin., non-recurring items, other operating items, operating income, add'l income/expense items, earnings before interest and tax, interest expense, earnings before tax, income tax, minority interest, equity earnings/loss unconsolidated subsidiary, net income-cont. operations, net income, net income applicable to common shareholders.
3. Cash flow statement features, including 18 data fields: net income, depreciation, net income adjustments, accounts receivable, changes in inventories, other operating activities, liabilities, net cash flow-operating, capital expenditures, investments, other investing activities, net cash flows-investing, sale and purchase of stock, net borrowings, other financing activities, net cash flows-financing, effect of exchange rate, net cash flow.

4 Methods

Several steps are applied in data pre-processing:

1. To construct the input of a LSTM model, 3 continuous data points are connected to be 1 training item, with 30 features from balance sheet, 18 features from income statement, 18 features from cash flow statement, for a total of 66 features in each data point.
2. Training labels fall into five categories based on annual stock price percentage change one year after the last financial statement release date for an item. Those five categories are:
   (a) annual percentage change smaller than -50%;
   (b) annual percentage change greater or equals to -50% and smaller than 0%;
   (c) annual percentage change greater or equals to 0% and smaller than 50%;
   (d) annual percentage change greater or equals to 50% and smaller than 100%;
   (e) annual percentage change greater or equals to 100%;
3. Two data augmentation techniques are applied:
   (a) One is to generate more training data by adding a small percentage change (from -5% to 5%) to each field in one item.
   (b) The other method is to ease out fields from one of the three financial statements (balance sheet, income statement, cash flow statement) to generate new items.
4. In the training set, there are 4,135 items in the first category, 16,111 items in the second category, 16,312 items in the third category, 1,068 items in the fourth category, 374 items in the fifth category. To deal with the imbalance training data in each category, the final training set is constructed by random sampling 7,600 items from each category.

The model is constructed with 5 layers:

1. The first layer is a LSTM layer with 256 nodes, with tanh as the activation function;

2. The second layer is a Dropout layer, with 0.1 as dropout rate;
3. The third layer is a Fully Connected layer, with 256 nodes, with tanh as the activation function;
4. The fourth layer is another Dropout layer, with 0.1 as dropout rate;
5. The last layer is a Softmax layer with 5 nodes.

The model is trained with Adam optimizer, for the benefit of leveraging momentum and RMSprop. Categorical cross-entropy is used as the loss function.

$$\hat{y}_i = \frac{e^{a_i}}{\sum_j e^{a_j}} \tag{1}$$

$$loss = -\sum_i y_i log\hat{y}_i \tag{2}$$

5 Experiments

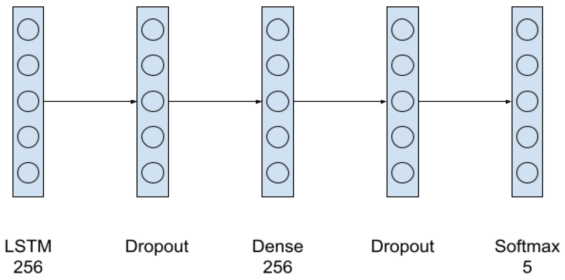

Figure 1: Illustration of the proposed model and its parameters.

Some hyper parameters that are chose in the model as shown in Figure 1:

1. The learning rate of the Adam optimizer is using a default value 0.001. This value is chosen because we would like the training procedure to be smooth without missing the local minimum.
2. The mini batch size is set to 32. This value is chosen to accelerate the convergence, while avoiding divergence in the training process.
3. The first layer is set to a LSTM layer because it could capture the hidden patterns in the sequence data. The number of nodes is set to 256 because we want to maintain the information captured in the sequence data without sacrificing too much of the training speed.
4. The second layer is set to a Dropout layer with dropout rate 0.1 because training with dropout nodes has a regularization effect, which makes the model generalize well to data it hasn't seen before. And a lower dropout rate is sufficient to keep most part of information to pass to the next layer.
5. The third layer is set to a Fully Connected layer because we want this layer to learn the complex function to extract information about stock performance. The number of nodes is set to 256 because we want to maintain the information from the previous layer without sacrificing too much of the training speed.
6. The fourth layer is set to another Dropout layer with dropout rate 0.1 because this could make the model generalize well to data it hasn't seen before.
7. The final layer is set to a softmax layer because the task is defined to be a multi-class one-hot vector prediction.

The primary metric used in this project is accuracy.

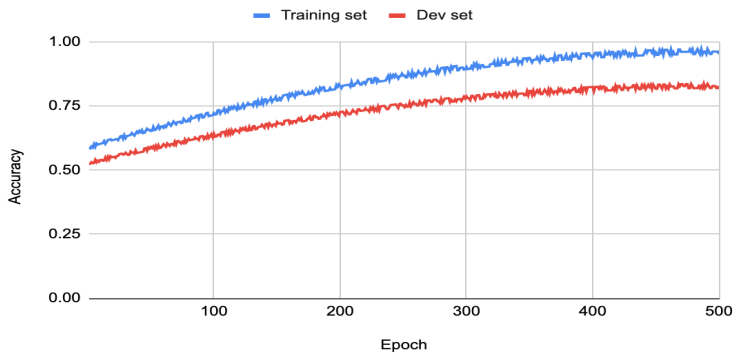

Figure 2: Accuracy vs epoch in training set and dev set.

According to Figure 2, the model didn't overfit on the training set, thanks to the two data augmentation techniques applied in training data (one is to add random noise into training data, the other is to erase out partial data fields from training data) and two dropout layers in the model.

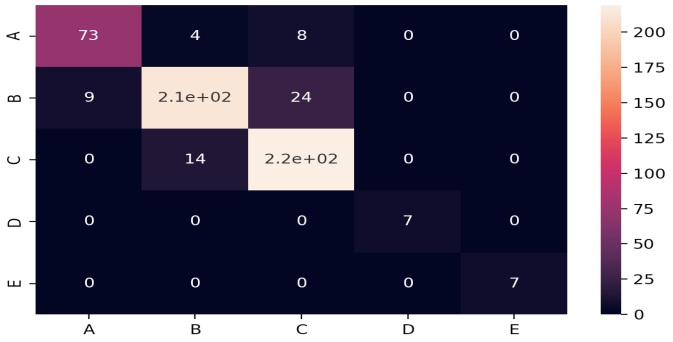

Figure 3: Illustration of the confusion matrix.

From Figure 3, we know that most of the cases that the model failed to predict is between category B, which includes stocks with annual price percentage change in range [-50%, 0%), and category C, which includes stocks with annual price percentage change in range [0%, 50%). This could be related to the fact that, a large amount of stocks (1,500 stocks out of a total number of 7,000) has a tiny price change very close to zero during one year, which is fine for the purpose of suggesting stocks for long term investment. For category D and E, including items with high investment returns, the model successfully predicts all items fall into these two categories, which indicates that the model is making good predictions on the items that we care the most.

Model	Structure	Training set accuracy	Dev set accuracy	Test set accuracy
A	LSTM256, Dropout, Dense256, Dropout, Softmax5	0.95	0.89	0.86
B	LSTM512, Dropout, Dense512, Dropout, Softmax5	0.97	0.72	0.71
C	LSTM128, Dropout, Dense128, Dropout, Softmax5	0.82	0.59	0.58
D	LSTM256, Dense256, Softmax5	0.97	0.63	0.61
E	LSTM256, Dropout, LSTM256, Dropout, Dense256, Dropout, Softmax5	0.91	0.70	0.66
F	LSTM256, Dropout, Dense256, Dropout, Dense256, Dropout, Softmax5	0.89	0.68	0.61

Table 1: Comparison of accuracy between models with different structures.

In Table 1, by comparing model A, model B, model C, we know that the number of nodes 256 seems to be a great fit for this training task. By comparing model A and model D, we know that those two dropout layers are doing a good job in helping the model to generalize to data that has never been seen before. By comparing model A and model E, we know that one LSTM layer is sufficient to capture patterns in sequential data. By comparing model A and model F, we know that one fully connected layer is sufficient to make good predictions and generalize to new data.

6 Conclusion and Future Work

Long term stock trading prediction is a non-trivial task with way less attention than it should be. This paper proposes a model with LSTM and fully connected layers to predict long term stock trendings based on financial statements. Experimental results show that two data augmentation techniques are quite powerful in training models on structured data: 1) adding random noise to data fields; 2) erasing partial information from training examples.

For future work, more historical data in financial statements could be critical in terms of improving the model's accuracy, like extending 3 data points in an item to 10 data points in an item for model training. Predicting stock performance in an even longer period of time, like 2 years, 5 years, or 10 years, could be another interesting topic to do research on.

7 Contributions

Shujia Liu contributes to all parts of this project, including data crawling, data parsing, data processing, model training, result analysis, paper writing.

This paper and the research behind it would not have been possible with the exceptional support of my project mentor, Avoy Datta.His enthusiasm, knowledge and exacting attention to detail have been an inspiration and kept my work on track from my first encounter with the project proposal to the final draft of this paper.

References

[1] Ryo Akita, Akira Yoshihara, Takashi Matsubara, and Kuniaki Uehara. Deep learning for stock prediction using numerical and textual information. In *2016 IEEE/ACIS 15th International Conference on Computer and Information Science (ICIS)*, pages 1–6. IEEE, 2016.

[2] Kai Chen, Yi Zhou, and Fangyan Dai. A lstm-based method for stock returns prediction: A case study of china stock market. In *2015 IEEE international conference on big data (big data)*, pages 2823–2824. IEEE, 2015.

[3] Xiao Ding, Yue Zhang, Ting Liu, and Junwen Duan. Deep learning for event-driven stock prediction. In *Twenty-fourth international joint conference on artificial intelligence*, 2015.

[4] Robert W Holthausen and David F Larcker. The prediction of stock returns using financial statement information. *Journal of accounting and economics*, 15(2-3):373–411, 1992.

[5] Jane A Ou and Stephen H Penman. Financial statement analysis and the prediction of stock returns. *Journal of accounting and economics*, 11(4):295–329, 1989.

[6] Eberhard Schöneburg. Stock price prediction using neural networks: A project report. *Neurocomputing*, 2(1):17–27, 1990.

[7] Robert P Schumaker and Hsinchun Chen. A quantitative stock prediction system based on financial news. *Information Processing & Management*, 45(5):571–583, 2009.