Model Analysis of Stock Price Trend Predictions based on Financial News

Yunhan Zhang and **Mingjia Zhu** and **Liuyang Zheng**
[yuz047, mzhu, lizheng]@ucsd.edu

Abstract

Financial news is an important source for people to learn information in financial field, such as the variation in stock market. News could also be a key factor to forecast change in the stock market. In this report, we introduce different methods we tried to predict the change of stock price by using financial news, including Bag-of-Words, AutoPhrase, LSTM, and BERT. Our experiments demonstrate that BERT outperforms other models.

1 Introduction

Nowadays, we are seeing a great amount of data generated everywhere. The Data will bring us both challenges and opportunities. As AutoPhrase (Shang et al., 2017) is developed, an automated and domain-independent phrase mining method, we see its potential to be used on extracting quality phrases from numerous daily financial news and editorials, to improve the current data prediction model using bag-of-words and other conventions as its entry. This experiment could be very promising since it can be a quick classifier of long text reports. Analysts then may use the result as a supportive reason for future investments. Thus, adapting this method could improve the efficiency with the same accuracy if used properly.

In this project, we will give a report of model analysis among conventional methods and AutoPhrase on predicting the change of stock price (increase or decrease). We will also include the comparison between the performances of AutoPhrase and deep learning methods on predicting the change of the stock price.

2 Dataset

2.1 Financial news dataset

We mainly use reports and news from Yahoo Finance as the input of our models. We found articles that only mention a single company to prevent the noise. For our input, there are two main types of articles: analyst report and press release news. We searched for the articles under "press releases" and "research reports" categories on the web page of a company and used beautiful soup to scrape the articles. We plan to collect news about 30 companies and 30-50 articles within the past one year for each company to test our models.

2.2 Stock price dataset

Our stock price dataset is from a kaggle project that updates all stock prices regularly and has been verified. The raw dataset covers 1657 major nasdaq stock prices from their beginning to present. We are selecting one at a time, and compute its 5-day-average, 10-day-average and so on from the time a particular piece of news is released. We started with Apple Inc. to test our methodology and to try automations. (Mooney, 2020)

2.3 Positive and negative words in financial news dataset

We utilized the "Financial positive and negative terms list" created by Bill McDonald. This dataset contains common positive and negative words in financial news. The dataset contains 354 positive words and 2350 negative words. It helps us to determine the attitudes of financial news articles.

2.4 FinancialPhraseBank dataset

This dataset contains the sentiments for financial news headlines from the perspective of a retail investor. And it contains two columns, "Sentiment" and "News Headline". The sentiment can be negative, neutral or positive. (Malo et al., 2013)

3 Models

In this section, we introduce each model, including the methods, algorithm, and details about them.

3.1 Baselines

We used the Bag-of-Words method to generate the baseline model. After removing stopwords, removing punctuation, and stemming the text, we performed Bag-of-Words to get the top n words with the highest frequencies. Then we compare the extracted words with positive and negative words in the financial news dataset. If the number of positive words is larger than that of negative words in a document, we give the output as "True". Below is how the model works:

```
#create a dictionary (dict) that
    contains each word (key) in the
    document with the number of times it
     appears in the text (value)
positive_word_count = 0
negative_word_count = 0
for each high frequency word do
  if word in positive_word_list
    positive_word_count += dict[word]
  else if word in negative_word_list
    negative_word_count += dict[word]

if positive_word_count >
    negative_word_count
  Output: True
else
  Output: False
```

3.2 AutoPhrase Model

After building the baseline model, we tried to explore more methods to predict the change of stock price. Instead of using native methods like Bag-of-Words to extract the words in articles, we considered to extract high-quality phrases. Therefore, we utilized AutoPhrase to perform phrase mining tasks. AutoPhrase is a framework that extracts quality phrases from text (Shang et al., 2017). After running AutoPhrase, we will get the top high-quality phrases in an article with their scores. We will then use the positive and negative word lists to determine the attitude of the words. Finally, we use the scores and their attitude to predict the stock price change. Below is how we get the score for each extracted phrase:

```
for each high quality phrase do
    Split phrase into words
    coefficients = list()
    for each word do
        if word in positive_word_list
            Append 1 to coefficients
        else if word in
    negative_word_list
            Append -1 to coefficients
    if sum of coefficients = 0
        coefficient = 0
    else if sum of coefficients >= 1
        coefficient = 1
    else
        coefficient = -1
    score of phrase = original score *
    coefficient
```

By following the method above, we get an adjusted score for each phrase. We then train an SVM classifier to make the prediction.

3.3 Doc2vec and LSTM Model

Given that the board of the company will always try to make a development plan, no matter short term or long term, we know that there might be some connections between each financial report in the analysis of these financial decisions. Thus, we tried to use the Doc2vec model to create numerical representation of documents. Compared to the Bag-of-Words model, it was designed to address the problem of word order and syntax that BoW has, and after the Doc2vec model is trained, we can also fit in a new sentence and predict its paragraph vector, and that will be very useful in our stock price movement prediction because that means every time we have a new financial article, we can find its corresponding paragraph vector and used as the input for our prediction model. We are going to use LSTM for our prediction model. LSTM stands for Long Short Term Memory Network, and it's a special version of Recurrent Neural Network(RNN). As shown in the graph(Figure 1) below:

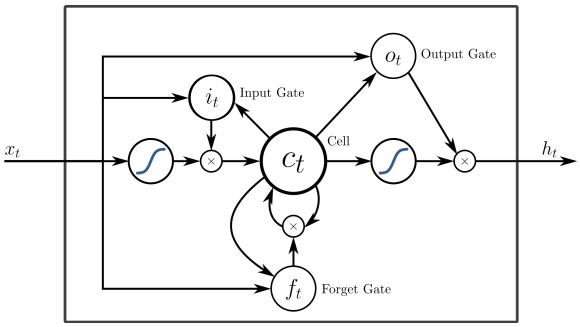

Figure 1: Work Flow of the LSTM Model

The LSTM architecture has a large range of modules for each time step update, and the output of each step update is controlled by a series of gates, which either add or remove the information from being updated to the cell state. With that being said, we will first obtain our vector embedding matrix from our Doc2vec model, and then we will fit in the embedding matrix as a weight to train our LSTM model, and from that we are able to predict the

movement of stock price from specific time span given the financial news documents. But the problem right now is that we do not have a large enough news article dataset to do proper neural network training, and the performance that we got from the current dataset that consists of a 30-ish number of financial reports is relatively trivial. However, our team has already found a new way to scrape the news articles and financial reports from the website and we should be able to test the model on a large scale of news sources. Even though we don't have the ideal result on our own dataset, we did test the model on a sentiment analysis dataset (Malo et al., 2013) that was found online consisting of only the financial news headlines, and a sentiment(neutral, positive, negative) assigned by a retail investor. And this dataset really shows us that Doc2vec and LSTM model has the potential to work with financial news and gives relatively good classification on binary labels.

3.4 BERT Model

While the main advantage of LSTM is that it keeps the long-term dependencies, BERT does more so we chose it to be the state of the art of our project. Upon our findings, we find that BERT is extremely helpful on our sentiment analysis model and good for our stock prediction. We could see from a typical BERT process illustrated by Devlin et al. The process(Figure 2) is in two parts. The pre-training process is using a model that can learn language well in an unsupervised way. By using the Masked Language Model, or MLM, BERT is able to read from left to right at the same time it reads right from left. This feature enables BERT to learn a sentence but not to learn a sequence of words. Besides, BERT is also implemented by NSP, or Next Sentence Prediction. This algorithm does a classification problem: if the some sentence B is following sentence A correctly, which guarantees an overall paragraph understanding by linking every sentence together. By bringing MLM, and NSP, we will have a very good understanding of the whole paragraph, knowing almost the true meaning of each word, sentence, and all. Lastly, since BERT is built on transformers, it can be accelerated by GPU executive units by doing parallel computing, which not only provides us learning performance, but also lower the computing requirements by improving its efficiency. BERT is also the first NLP technique that relies on self-attention mechanisms. This gives

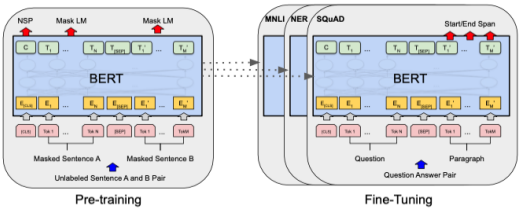

Figure 2: BERT Process Representation

BERT the ability to determine the change of meaning of the words. As Lutslevch suggests in the editorial, the word "is" often changes meaning when the paper goes. BERT, however, could often make a good association with the correct one. " That boy says the animal beside his mom is a cat. He is correct. It is an American Short-hair cat." In these sentences, the word "is" changes its meaning three times and BERT is designed for distinguishing these.(Devlin et al., 2019)

In the second phrase, BERT provides an extremely simple way to do the fine-tuning process. By adding one layer upon, one could already achieve a desirable output. Our approach is based on finBERT transformer from Hugging Face. After configuring the required packages and loaded the model, one should expect similar result and accuracy as writer's. The model we used is trained by Financial Phrase Bank by Malo et al.(Malo et al., 2013) This phrase bank is designed to train a financial sentiment analysis model, so we seeing it is a suitable model for our stock prediction pretrained model. As it is already tuned for analyzing sentiment for the financial texts, it could be mapped to our stock predictions situation. Since we are lacking resources of fetching enough news to pretrain this model particularly in our case, we have to choose accept this model and do classification upon this.

4 Results

4.1 Baselines

We experimented with different n and tried to predict the change of stock price of Apple Inc. after 5 days, 10 days, and 30 days of the news published date. We used 30 articles to test the model. The graph(Figure 3) below shows the change of accuracy. We could see that the baseline model performs the best for n=2000 and the stock price changes after 5 days, with an accuracy of 0.57.

For the prediction of the stock price change after 15 and 30 days, we did not get good results. In our other model, we will use more sophisticated models to achieve better accuracy.

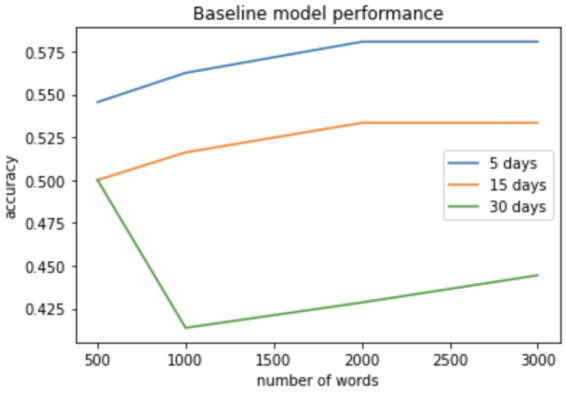

Figure 3: Performance of baseline model

4.2 AutoPhrase Model

We trained an SVM classifier with 100 news articles about Apple Inc. and predicted the stock price change by using 30 articles. The results we got for predicting the stock price change 5(Figure 4), 15(Figure 5), and 30 days(Figure 6) after the new publication date was 0.6, 0.58, 0.55, respectively. We have also created the confusion matrices for the results. In the graphs below, we could see that the model performs relatively well when the ground truth is 1. However, one problem is that the ground truth labels we used were imbalanced, as there're more 1s than 0s in the ground truth labels. We could increase the size of data if we have more time.

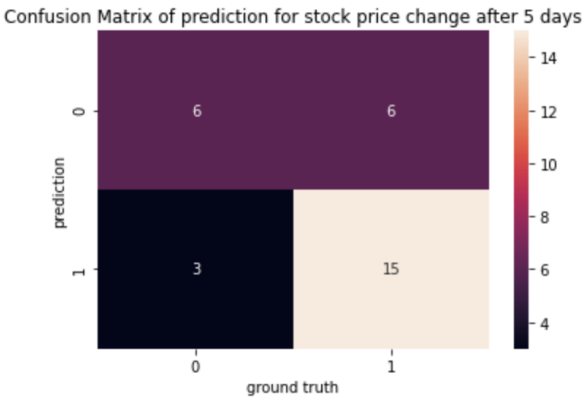

Figure 4: Confusion matrix of prediction by AutoPhrase model for stock price change after 5 days

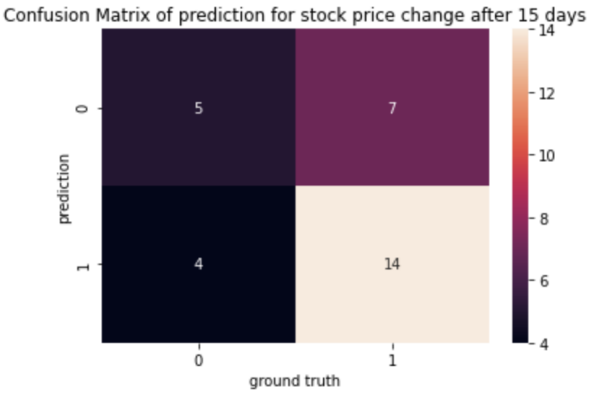

Figure 5: Confusion matrix of prediction by AutoPhrase model for stock price change after 15 days

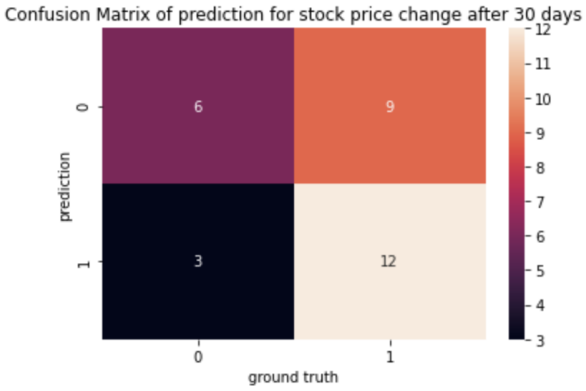

Figure 6: Confusion matrix of prediction by AutoPhrase model for stock price change after 30 days

4.3 Doc2vec and LSTM Model

We first tried to apply the Doc2vec and LSTM model on 30 financial news documents and stock price dataset, and then we predict the change of stock price of Apple Inc. after 5 days(Figure 7), 10 days(Figure 8), and 20 days(Figure 9) from the news published date. And the accuracy we got on the test set was around 0.6. We also obtained confusion matrices for these 3 categories respectively.

Although the accuracy for this dataset might not represent the true case due to the insufficient amount of training dataset we have, we still apply the model to a larger dataset we found online for sentiment analysis of financial news headlines. And note that it also has the binary labels where 0 stands for positive, 1 for neutral, and 2 for negative. And here below are the confusion matrix(Figure 10) for this classification process and accuracy(Figure 15) and loss(Figure 16) curves over the 50 epochs.

From these results, we can see that the confusion

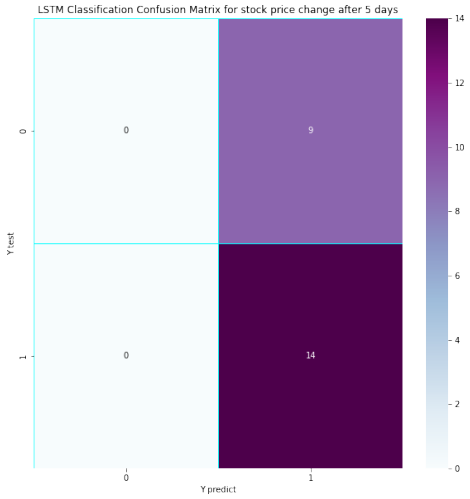

Figure 7: Confusion matrix of prediction by LSTM model for stock price change after 5 days

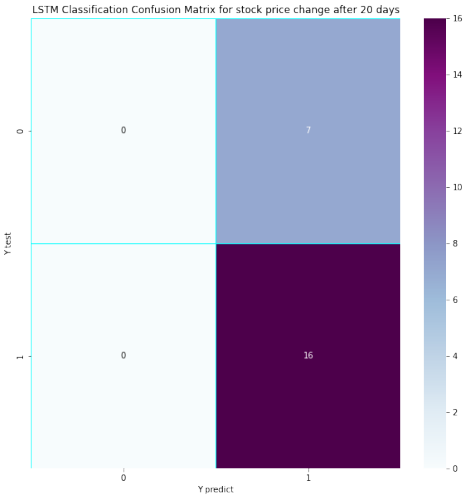

Figure 9: Confusion matrix of prediction by LSTM model for stock price change after 20 days

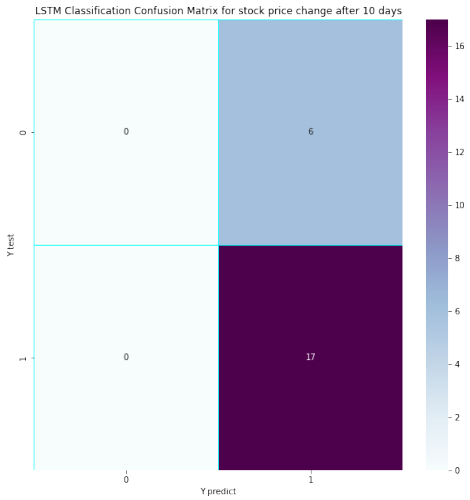

Figure 8: Confusion matrix of prediction by LSTM model for stock price change after 10 days

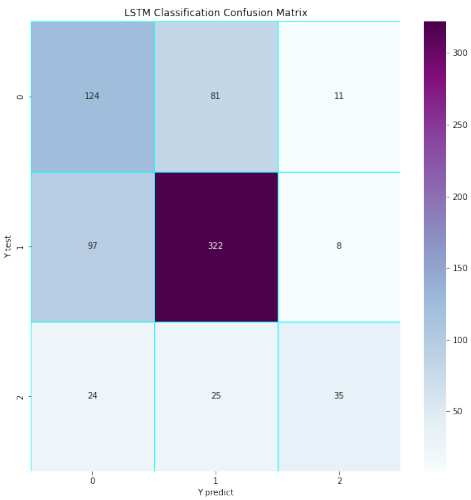

Figure 10: LSTM Classification Confusion Matrix

matrix is acting normally, and the model gives an accuracy of 0.66 for the test set and 0.67 for the validation set. Therefore, we confirmed the model's capability of predicting binary labels based on financial news dataset.

4.4 BERT Model

Built on finBERT model, we shall expect any text corpora has a output like Figure 11. Logit is the possibility that the model predicts of [positive, negative, neutral] for each sentence. And the prediction label is determined by logit where simply takes the largest possibility as the label. We also have a weighted sentiment score accordingly. We select the sentiment score and prediction label to create one extra machine learning layer to predict the stock price trend.(pro)

We can see from the confusion matrices(Figure 12, 13, 14) that the predictions are fairly accurate.

	sentence	logit	prediction	sentiment_score
0	Shares in the spin-off of South African e-comm...	[0.94092953, 0.020505333, 0.038565114]	positive	0.920424
1	Bob van Dijk, CEO of Naspers and Prosus Group ...	[0.03830802, 0.009095704, 0.9525963]	neutral	0.029212
2	REUTERS/Piroschka van de Wouw\nProsus comprise...	[0.08153216, 0.0048310524, 0.91363674]	neutral	0.076701
3	There is "way more demand than is even availab...	[0.86930275, 0.012840522, 0.11785665]	positive	0.856462
4	"It's going to be an interesting hour of trade...	[0.22605549, 0.021615589, 0.752329]	neutral	0.20444
5	Euronext had given an indicative price of 58.7...	[0.13808317, 0.012395258, 0.8495215]	neutral	0.125688
6	The shares jumped to 76 euros on opening and w...	[0.75047046, 0.018825809, 0.23070371]	positive	0.731645

Figure 11: Raw Output of finBERT.predict

Though the dataset is too small to be conclusive, we already have a great trend that the true predictions are much higher than false predictions no matter the labels distribution goes. This reveals BERT's potential to outperform any other existing models and may surprise us after a more thorough fine tuning with more datasets. Please note that BERT model are based on a smaller subset dataset than other models, since finBERT is not supporting GPU at our end which takes disproportionate time, and the running results are inconclusive.

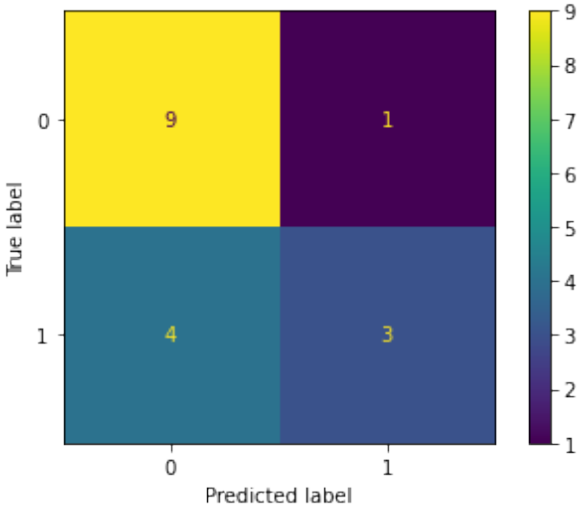

Figure 12: Confusion Matrix of BERT by week.predict

5 Discussion & Conclusion

In sum, we have worked on combination of conventional machine learning models, the AutoPhrase models. We have also worked on Deep Learning models like LSTMs, and BERT. In our preliminary analysis by far, we've seen that AutoPhrase has roughly the same performance as LSTMs. And BERT outperformed LSTMs, and all other models. However, we do see limitations that this BERT sentiment analysis model is not perfect for stock price prediction. There are still works to do to enhance its performance, a current 72% to as high as 85%

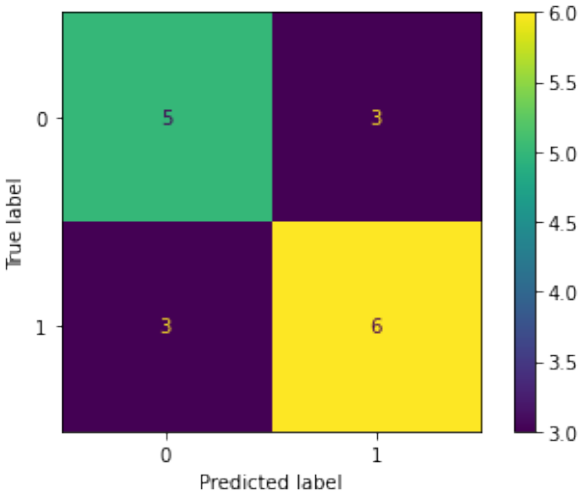

Figure 13: Confusion Matrix of BERT by month.predict

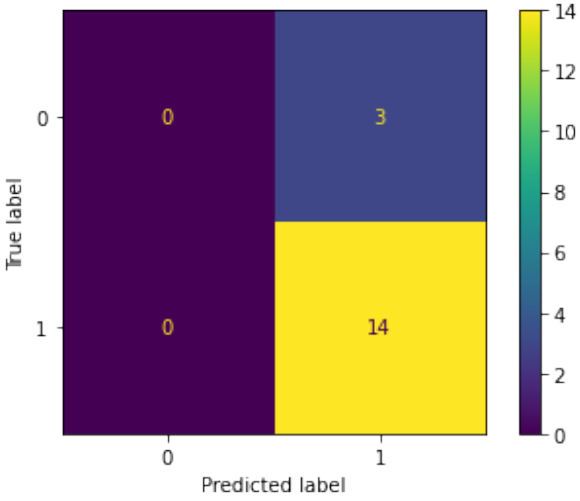

Figure 14: Confusion Matrix of BERT by quarter.predict

(Wells, 2020), which is an theoretical result after we are able to acquire diversified data. We may also take steps further to run simulations of this predictor and to feed it into RNN models, which we will continue working on.

The model comparison does not have a solid conclusion set in stone. Due to our team works on this project in the middle of transition of x86 to ARM, some packages are not applicable and costs extra time, which is the main reason that the models are tested on different test set. At this point, we are expecting deep learning model, especially BERT works better in this task. We will continue working on this project and expand it to a broader

selection of stocks.

6 Appendix

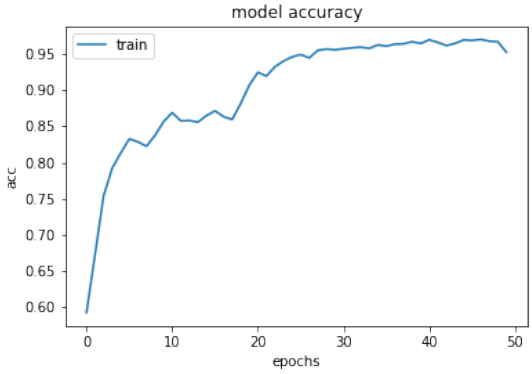

Figure 15: Model Accuracy over 50 Epochs

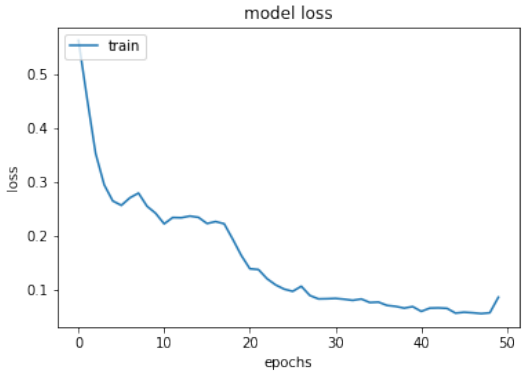

Figure 16: Model Loss over 50 Epochs

References

Prosusai/finbert$_h ugging face$.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Pekka Malo, Ankur Sinha, Pyry Takala, Pekka J. Korhonen, and Jyrki Wallenius. 2013. Good debt or bad debt: Detecting semantic orientations in economic texts. *CoRR*, abs/1307.5336.

Paul Mooney. 2020. Stock market data (nasdaq, nyse, sp500). https://www.kaggle.com/paultimothymooney/stock-market-data.

Jingbo Shang, Jialu Liu, Meng Jiang, Xiang Ren, Clare R. Voss, and Jiawei Han. 2017. Automated phrase mining from massive text corpora. *CoRR*, abs/1702.04457.

Jack Wells. 2020. Stock-market-prediction-nlp-bert. https://github.com/Jack-Wells/Stock-market-prediction-NLP-BERT.