Identifying Market Sentiments for Financial Market Signals

Yuejiao Qiu, MS Data Science & Analytics, 2023
Ruoting Shen, MS Data Science & Analytics, 2023
Yuening Wang, MS Data Science & Analytics, 2023
Xiana Zhang, MS Data Science & Analytics, 2023

1 Introduction

Market sentiment has been a key driving factor for stock prices, since it reveals how investors view the market, and stock prices will change so as to reflect such opinion on how market will evolve. Major financial institutions have been inferring market sentiment through technical analysis, which is a trading discipline that analyzes statistical trends gathered from trading activity, such as price movement and volume; but few directly predict market sentiment by modeling because of the complexity of financial corpus and computational efficiency.

Nowadays, with the ease of information transmission, social media platforms, such as Twitter and Reddit, contribute to the most financial market sentiment, as it allows people to share and express their ideas freely. Twitter has always been a platform where individuals are allowed to share information, such as news, politics, education, or even financial guidance. There were multiple occasions, influential fellows were able to manipulate the financial market by their tweets.

Identifying market sentiment using text data on social media platforms have gained increasing popularity, especially after a price surge in GME (GameStop) stock price in January 2021. Before the price rise, investors on Reddit found out that many institutional investors were betting against GameStop by shorting its stocks, which finally resulted in a large number of Reddit users purchasing GME stocks massively, eventually leading to a fourteen-fold price increase. This has suggested that market sentiment is of crucial importance to stock price fluctuations.

Therefore, it is natural for us to deploy NLP techniques to analysis the sentiment from financial text data from social media platforms. Existing literature has been conducted in this domain, but mostly focusing on the formal texts, such as news articles and reports. However, posts and comments on social media platforms are full of internet slangs, jargons, and abbreviations, which makes it difficult to use transfer learning to analyze the discussions online that are relatively informal. In this project, we therefore aim to perform sentiment classifications on the financial tweets, trying out different methodologies and architectures, such as Bag-of-Words (BOW) model and fine-tuned BERT model. Through the comparison of different models and their performances, we will be able to tackle the important facts about these models as well as the task of sentiment analysis. Moreover, a neural network is built on top of the sentiments to predict stock price movement, which also helps to evaluate the sentiment model in a practical context.

2 Data

2.1 Dataset for Sentiment Analysis

In this project, the primary dataset we focus on is Financial Sentiment Analysis Dataset, which contains 5,322 finance-related tweets and their corresponding sentiment (positive, neural, or negative). Table 1 below provides a snapshot of the Financial Sentiment Analysis Dataset.

Sentence	Sentiment
For the last quarter of 2010, Componenta 's net sales doubled to EUR131m from EUR76m for the same p...	positive
The Stockmann department store will have a total floor space of over 8,000 square metres	neutral
$ESI on lows, down $1.50 to $2.50 BK a real possibility	negative

Table-1: Financial Sentiment Analysis Dataset Snapshot

The sentiment distribution is shown in Table 2 below.

Sentiment	Percentage
Positive	31.7%
Neutral	53.6%
Negative	14.7%

Table-2: Sentiment distribution of Financial Sentiment Analysis Dataset

Another dataset we will use to pre-fin-tune the BERT model is Sentiment140 Dataset, which is a significantly larger dataset containing 1.6 million general tweets and their implied sentiments.

2.2 Dataset for Price Movement Prediction

The text data we adopt for price movement prediction is the reddit submissions regarding GME (Gamestop) from Jan. 1, 2021 to Aug. 31, 2022, which covers 94,298 submissions. Their implied sentiments will be predicted using the trained sentiment analysis model. As for the stock price data, GME historic daily price is accessible from Yahoo Finance.

3 Models

3.1 Bag-of-Words Model

The very first model we consider for sentiment analysis is bag-of-words model, which is a very straightforward method for many NLP tasks, where text data are vectorized into a multiset of words, without considering grammar and context. After vectorization, we feed the data into the model designed below, which contains 4 dense layers, 3 dropout layers, and 772,565 trainable parameters.

After training for 15 epochs, the highest validation accuracy achieved is 67.84% and f1 score is 0.681.

```
Layer (type)                Output Shape              Param #
=================================================================
input_1 (InputLayer)        [(None, 6000)]            0

dense (Dense)               (None, 128)               768128

dropout (Dropout)           (None, 128)               0

dense_1 (Dense)             (None, 32)                4128

dropout_1 (Dropout)         (None, 32)                0

dense_2 (Dense)             (None, 8)                 264

dropout_2 (Dropout)         (None, 8)                 0

dense_3 (Dense)             (None, 5)                 45

=================================================================
Total params: 772,565
Trainable params: 772,565
Non-trainable params: 0
```

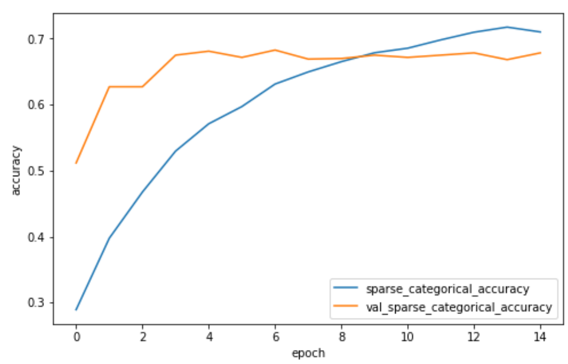

Fig-1: Bag-of-words model

3.2 BERT Models

In the following few sections, we opt to train the conventional BERT model with different options and test on the Financial Sentiment Analysis Dataset.

3.2.1 BERT without fine tuning

The following model design encapsulates the basic BERT model without fine tuning. In this stage, all the parameters in the BERT layer are freezed, which results in only 246,917 trainable parameters in the classifier, since the dense layers are trainable instead. The optimal model is trained for 100 epochs with mini-batch size of 64 and learning rate of 5e -5, where the best accuracy score is 64.59% and f1 score is 0.643.

Layer (type)	Output Shape	Param #	Connected to
input_ids (InputLayer)	[(None, 100)]	0	[]
token_type_ids (InputLayer)	[(None, 100)]	0	[]
attention_mask (InputLayer)	[(None, 100)]	0	[]
bert (TFBertMainLayer)	TFBaseModelOutputWithPoolingAndCrossAttentions(last_hidden_state=(None, 100, 768), pooler_output=(None, 768), past_key_values=None, hidden_states=None, attentions=None, cross_attentions=None)	108310272	['input_ids[0][0]', 'token_type_ids[0][0]', 'attention_mask[0][0]']
dense_4 (Dense)	(None, 256)	196864	['bert[1][1]']
dense_5 (Dense)	(None, 128)	32896	['dense_4[0][0]']
dense_6 (Dense)	(None, 128)	16512	['dense_5[0][0]']
dense_7 (Dense)	(None, 5)	645	['dense_6[0][0]']

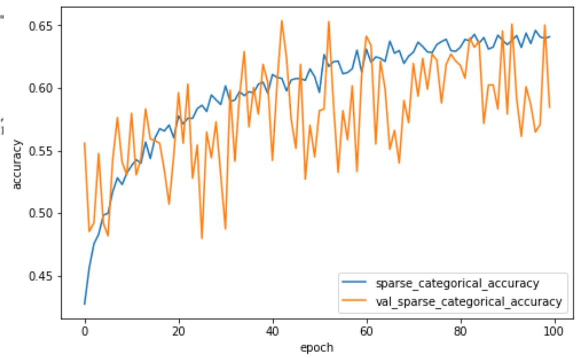

Fig-2: BERT without fine tuning

3.2.2 BERT with fine tuning

An improve we made from the previous model is to add a fine tuning process for basic BERT. Compared with the previous model, two major distinctions are that the parameters in the BERT layer are all unfreezed and dropout layers are added after dense layers. The former increases the number of trainable parameters to 108,557,189, which allows the parameters to be updated in each training epoch; the latter can prevent the model from overfitting. The results shows that fine tuning remains effective, since the validation accuracy increases to 75.5% and f1 score is 0.771 with 4 epochs training of mini-batch size of 64 and learning rate of 5e -5.

Layer (type)	Output Shape	Param #	Connected to
input_ids (InputLayer)	[(None, 100)]	0	[]
token_type_ids (InputLayer)	[(None, 100)]	0	[]
attention_mask (InputLayer)	[(None, 100)]	0	[]
tf_bert_model_1 (TFBertModel)	TFBaseModelOutputWithPoolingAndCrossAttentions(last_hidden_state=(None, 100, 768), pooler_output=(None, 768), past_key_values=None, hidden_states=None, attentions=None, cross_attentions=None)	108310272	['input_ids[0][0]', 'token_type_ids[0][0]', 'attention_mask[0][0]']
dense_4 (Dense)	(None, 256)	196864	['tf_bert_model_1[0][1]']
dropout_77 (Dropout)	(None, 256)	0	['dense_4[0][0]']
dense_5 (Dense)	(None, 128)	32896	['dropout_77[0][0]']
dropout_78 (Dropout)	(None, 128)	0	['dense_5[0][0]']
dense_6 (Dense)	(None, 128)	16512	['dropout_78[0][0]']
dropout_79 (Dropout)	(None, 128)	0	['dense_6[0][0]']
dense_7 (Dense)	(None, 5)	645	['dropout_79[0][0]']

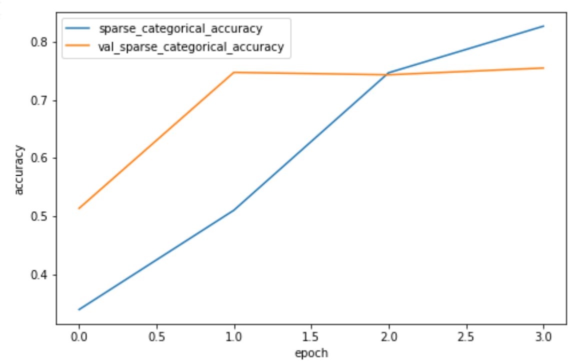

Fig-3: BERT with fine tuning

3.3 BERT Transfer Learning

All the previous models suffer from the limited availability of Financial Sentiment Analysis Dataset, since it only contains 5,322 finance-related tweets, which seems slightly insufficient for model training. Therefore, we decide to first pre-train the BERT model on a significantly larger dataset Sentiment140 Dataset with 120,000 general tweets, which are characterized by its abundance of internet slangs. Then we record the parameters obtained and transfer them onto the Financial Sentiment Analysis Dataset.

3.3.1 Pre-tuned BERT without fine tuning

Instead of the basic BERT model, we load the pre-tuned BERT layer on Sentiment140 Dataset and incorporate the parameters in our model. This time we freeze the parameters when training on the Financial Sentiment Analysis Dataset. This results in only 246,917 trainable parameters. The classifier is identical as the basic BERT without fine tuning, which contains 4 dense layers and is trained for 100 epochs with mini-batch size of 64 and learning rate of 5e -5. The validation accuracy is 69.06% and f1 score is 0.664, which both increase compared with the basic BERT model.

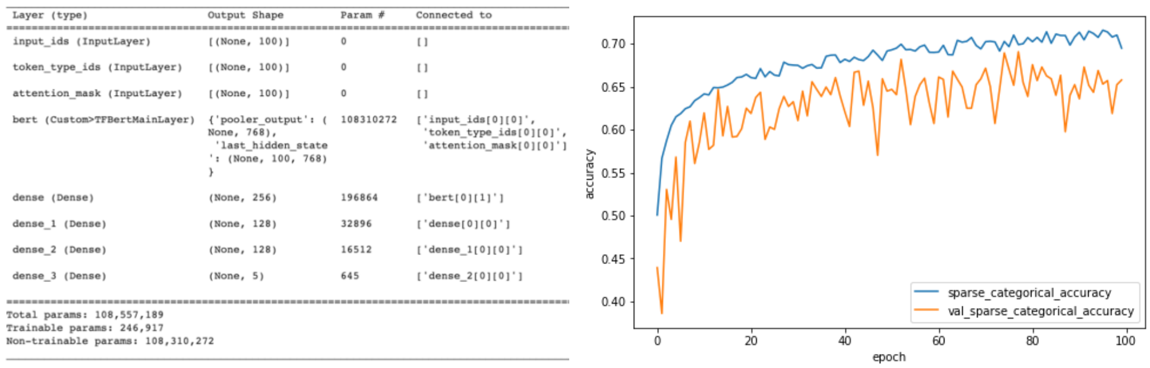

Fig-4: Pre-tuned BERT without fine tuning

3.3.2 Pre-tuned BERT with fine tuningError! Bookmark not defined.

We now unfreeze the parameters in the Sentiment140 pre-tuned BERT layers in the training process on the Financial Sentiment Analysis Dataset, which leads to 108,557,189 trainable parameters. As for the classifier, it is the same as the previous fine-tuned model, which is trained for 4 epochs with mini-batch size of 64 and learning rate of 5e -5, and with 4 dense layers and 3 dropout layers which prevents the model from overfitting. Similarly, fine tuning is effective, as validation accuracy increases to 76.52% and f1 score increases to 0.775.

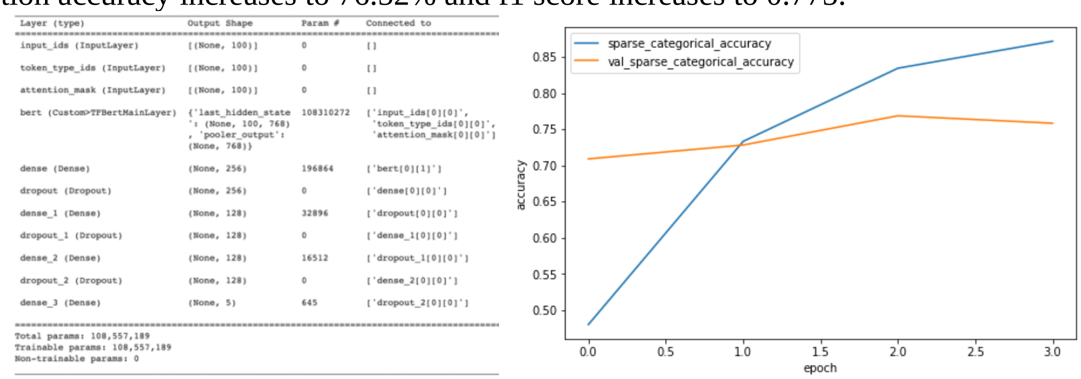

Fig-5: Pre-tuned BERT with fine tuning**Error! Bookmark not defined.**

4 Evaluation

4.1 Model Comparison

Finally, we put together all the models, configurations, and performances in the table below.

Section	Dataset	Training Size	Model	Fine Tune	Hidden Dense	Dropout	#Parameters	#Epochs	Best Accuracy	F1 Score
3.1	FinTweets	7512	BOW	N/A	(128, 32, 8)	0.75	772,565	15	67.84%	0.681
3.2.1	FinTweets	7041	BERT	N	(256,128,128)	N/A	246,917	100	64.59%	0.643
3.2.2	FinTweets	7041	BERT	Y	(256,128,128)	0.5	108,557,189	4	75.50%	0.771
Prep	Sentiment140	120,000	BERT	Y	None	N/A	108,313,348	1	Pre-tuned BERT	
3.3.1	FinTweets	7041	Pre-tuned BERT	N	(256,128,128)	N/A	246,917	100	69.06%	0.664
3.3.2	FinTweets	7041	Pre-tuned BERT	Y	(256,128,128)	0.5	108,557,189	4	76.52%	0.775

Table-3: Model comparison

Generally speaking, the best model on sentiment analysis is the pre-tuned BERT with fine tuning in 3.3.2, which yields the highest accuracy and f1 score. The other fine tuning model on basic BERT ranked the second. It is also worth noticeable that in terms of f1 score, both BERT and pre-tuned BERT model does not outperform the simple bag-of-words model.

From the comparison between 3.2.1 and 3.2.2, or between 3.3.1 and 3.3.2, it is clear that models with unfreezed parameters and fine tuned in the training process yield a much higher accuracy score than the models with freezed parameters.

Moreover, based on the model comparison between 3.2.1 and 3.3.1, or between 3.2.2 and 3.3.2, we discover that pre training BERT on Sentiment140 also contributes to better model performance, especially for the not fine-tuned models that accuracy increases from 64.59% to 69.06%, while for relatively large dataset, the effect of pre-tuning BERT is mitigated.

4.2 Correlation Analysis

First of all, we use the pre-tuned BERT with fine tuning model to get the predictions of sentiments for the Reddit submissions. The results we get are as follows:

Sentiment	Count
Positive	61,656
Neutral	18,765
Negative	13,877

Table-4: Sentiment predictions on Reddit data

Then we aggregate the Reddit sentiment by date and count the number of each sentiment in a given day, as well as their corresponding percentage.

We calculate the correlation of various sentiment indicators (count and percentage) with both prices and smoothed prices using a 3-day-span average.

	Open	High	Low	Close	Return
neg_count	0.190421	0.295703	0.034448	0.135550	-0.088591
pos_count	0.208103	0.294619	0.077112	0.167200	-0.049275
%neg	-0.072633	-0.063947	-0.111795	-0.103508	-0.081021
%pos	0.132717	0.102020	0.183161	0.154084	0.012134

Table-5: Correlation of sentiment with stock price

	Open	High	Low	Close	Return
neg_count	0.112454	0.183123	0.002548	0.069588	-0.071774
pos_count	0.143459	0.206967	0.044253	0.110450	-0.038988
%neg	-0.053496	-0.039359	-0.094069	-0.077686	-0.067056
%pos	0.166088	0.137713	0.203534	0.178016	0.005419

Table-6: Correlation of sentiment with smoothed stock price

Unfortunately, none of the correlations appears to be highly statistically significant, in terms of value. This is reasonable because complexity of stock market where there are many different factors that could influence stock prices, as well as the unpredictability of stock prices. Generally, positive sentiment seems to be more related to GME stock price than negative sentiment. More specifically, positive sentiment count is more associated with GME stock price than percentage count. However, this result might be affected by the data, due to the short squeeze of GME stock in late January 2021, where there was a blast of discussions of GME on Reddit, which actually resulted in an increase of both positive and negative submissions.

Delayed time effect is also assessed through time lagged cross correlation. With the increase of time lag, the cross correlation displays a roughly monotonic decreasing trend. Only negative count reaches the highest correlation at lag=1, but as mentioned before, it can be potentially affected by the short squeeze of GME. Given the fact that the correlation value is not significantly large either, there is not enough evidence for a delayed time effect for market sentiments on stock prices. This also proves that the US stock market is efficient to some extent, that stock prices could quickly adjust to the various information outside.

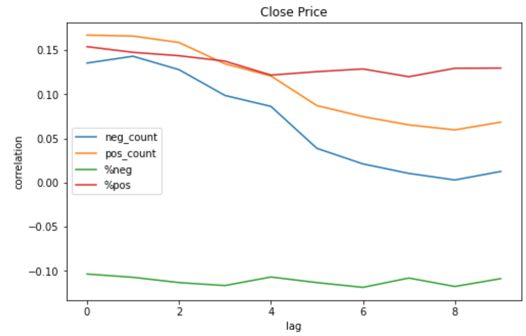

Fig-6: Time lagged cross correlation

4.3 Stock Price Movement Prediction

In this section, instead of predicting the exact price of the stock, our target is to predict whether the price will go up, namely the return will be positive or not.

For sentiments, we use positive count, which seems to be the most significant among all sentiment indicators. Together with sentiments, there are also 6 technical indicators fed into the neural network, which consists of MOM (momentum), relative strength index (RSI), normalized average true range (NATR), commodity channel index (CCI), on-balance volume (OBV), and average directional index (ADX). They are acquired by the TA-Lib python package based on the price data (open, high, low, close) of GME. And they are normalized before fed into the neural network.

The architecture of Long Short-Term Memory networks, as shown below is adopted in this project in order to predict the stock price movement of GME stock price, as LSTM can utilize its ability to memorize information further back in time in a sequence of data. Before fitting data into the model, the price data are transformed into blocks of data, with a time step of 14 days, which correspond to a week.

Layer (type)	Output Shape	Param #
lstm_30 (LSTM)	(None, 14, 8)	512
dropout_30 (Dropout)	(None, 14, 8)	0
batch_normalization_30 (BatchNormalization)	(None, 14, 8)	32
lstm_31 (LSTM)	(None, 14, 16)	1600
dropout_31 (Dropout)	(None, 14, 16)	0
batch_normalization_31 (BatchNormalization)	(None, 14, 16)	64
lstm_32 (LSTM)	(None, 8)	800
dropout_32 (Dropout)	(None, 8)	0
batch_normalization_32 (BatchNormalization)	(None, 8)	32
dense_10 (Dense)	(None, 1)	9

Fig-7: LSTM model architechture

With 10 epochs of training, the best accuracy achieved is 69.83% for training and 55.74% for testing. Given the limited size of Reddit data, which only covers more than one and a half years, the results are still promising.

5 Possible Further Study

After reviewing the most state-of-the-art research on BERT fine tuning and financial sentiment analysis, we have found several aspects that we can improve. First, it is feasible to set a decaying learning rate for lower layers, which intuitively would be more capable of containing more general information. Moreover, instead of the conventional BERT model, FinBERT is also a very good alternative, which is a BERT model further pre-trained on financial sentiment corpora, such as financial news, reports, etc. If given the opportunity, we are willing to extend this project to the above-mentioned directions.

Reference

Dogu Araci. Finbert: Financial sentiment analysis with pre-trained language models. CoRR, abs/1908.10063, 2019.

Adam Bear. What happened to gme stock? 2 2021.

Paolo Ripamonti. Twitter sentiment analysis.

Qiang et al. Sentiment Analysis for Meme Stock Price Prediction.