Using news titles and financial features to predict intraday movements of the DJIA

Arjun Arora
Stanford University
aarora52@stanford.edu

Avoy Datta
Stanford University
avoy.datta@stanford.edu

Victoria Ding
Stanford University
vding1@stanford.edu

Abstract

In this work we use a deep learning model to predict the intraday directional movements of the Dow Jones Industrial Average (DJIA) stock market index using a combination of general news titles and technical data related to the index. Previous research in this domain has shown a combination of RNN and CNN-based architectures to be effective at capturing semantic information from financial news titles and model the temporal dependencies of a separate index, the S & P 500, on each day's news. When it comes to linguistic input, most existing methods draw exclusively from financial news articles. We have attempted to generalize this framework to general news titles, as well as experiment with attention mechanisms in modeling temporal variations in the index.

1 Introduction

1.1 Problem Overview

The DJIA stock index is a price-weighted average of stock prices from 30 of the largest publicly traded companies in the United States. The DJIA is one of the most closely watched market benchmarks. Traditionally, technical analysis has been used to make near-term forecasts on this index, based on the premise that future values of a financial time series are conditioned on its past values. However, recent work **[1]** has shown daily news titles impact stock price movements, while RCNN networks have been shown to extract information from both news titles and technical indicators **[2]** to make predictions that beat predictors that do not use news titles. Our goal is to predict the intraday movement of the DJIA (increase/stay the same or decrease), given news titles and financial technical indicators. We use the Recurrent Convolutional Neural Network (RCNN) model proposed in **[2]** as our baseline for comparisons, migrating the model from S and P 500 predictions to our current task of DJIA predictions and using general news headlines instead of financial news headlines.

1.2 Technical indicators

Besides news titles, our model uses as its input a set of seven technical indicators extracted from the financial time series n days before the day of prediction. These technical indicators, proposed in **[4]**, are functions of the following index values over a window of n days:

- Closing price at previous day
- Lowest and highest prices at previous day
- Moving average over past n days
- Highest high and lowest low over past n days

Preprint. Work in progress.

2 Related Work

Our project was motivated by the paper "On the importance of text analysis for stock price prediction." by Lee et al **[1]**. This paper was one of the first to show a significant improvement in predictive power of a stock (relative to an index) based on textual information. Even more important is that they show the variation in predictive power of text over time from a relevant event. The non-linguistic features remain relatively flat while varying the time interval, but the linguistic features give more predictive power in the very short term, compared to a longer period. This indicates that the effect of linguistic features diminishes quickly with time. While Lee et al. did not implement any deep learning methods, their conclusion about the importance of text analysis in stock price prediction informed our own problem statement.

Velay and Daniel, in their paper "Using NLP on news headlines to predict index trends," explored a variety of techniques to predict intraday movements in the DJIA, including logistic regression, support vector machine, and LSTM **[3]**. They achieved the highest accuracy using logistic regression, and with their LSTM, they experienced severe overfitting and ultimately did not improve on randomness.

In the paper "Deep learning for stock market prediction from financial news articles," Vargas et al. proposed a Recurrent Convolutional Neural Network (RCNN) in their task of predicting the intraday directional movements of the Standard & Poor's 500 Index (S&P 500). The RCNN model aims to obtain advantages from both models: CNN and RNN. CNN has a superior ability to extract semantic information from texts in comparison with RNN, and RNN is better at capturing the contextual information and simulating complex temporal characteristics. This model uses as input a set of seven technical indicators extracted from the target series and financial news titles published the day before the prediction day. It applies a two-step process to represent each news in the data set: first, a word2vec model is used to generate a word representation; second, an average of all the word vectors of the same title is performed,addressing sparsity in word-based inputs.

The incorporation of the seven technical indicators was originally proposed by Zhai et al. in their paper "Combining News and Technical Indicators in Daily Stock Price Trends Prediction"[**4**]. These include the Stochastic %K, Stochastic %D, Momentum, Rate of Change, William's %R, A/D Oscillator, and Disparity 5, which are commonly used to describe assets. The RCNN model proposed by Vargas et al. uses only news from the day before the forecasting day and outperforms a set of models that uses news from the past day, week and month. This reinforces Lee's paper's idea that the impact of news on stock diminishes with time. Moreover, the papers confirms the positive influence of the use of a hybrid input (news and technical indicators), leading to the conclusion that both sources of information are relevant. We implemented the Vargas RCNN model as our baseline, which we will describe in deeper detail later in this paper.

3 Approach to DJIA movement prediction

3.1 Baseline RCNN architecture

We implemented the baseline ourselves due to the unavailability of source code.[1] We implemented an RCNN model that takes in a sequence of embeddings and technical indicators as input (also known as the "embedding layer" and "technical layer," respectively). The embedding layer is comprised of a sequence of 25 article headlines, each of which is represented by a sentence vector (the average of the word embeddings in the headline). 300-dimensional pre-trained GloVe embeddings were used to generate inputs from the news titles.[2] For each news title, the baseline **averaged** over the words in the title and passed the result as *Title Data*. The *Technical Indicators* were obtained from financial series data using formulae described in **[3]**. The technical indicators fed in are those taken over the *5* days preceding the day of prediction.

The following hyperparameters were used for the RCNN baseline:

[1] Close examination of the experimental details revealed several inconsistencies in the hyperparameters of the model in [2]. The actual baseline implemented was thus based on assumed corrections to these inconsistencies, with suggestions from the project mentor.

[2] https://nlp.stanford.edu/projects/glove/ (Wikipedia 2014 + Gigaword 5)

- $N = 1$, with the Conv layer's kernel size and pool size set to 4 and 2 respectively. All strides were 1. 1-D convolution was carried out along the dimension
- Dropout probability of 0.5 in the layer following the convolution layer.
- Hidden state dimensions of 128 and 50 for the upper and lower LSTMs respectively.

A dense layer takes in concatenated *hidden and cell states* from the two terminal LSTMs as inputs and maps to a two-class output, which is converted to a binary probability distribution using a *Softmax* layer (a combination of *LogSoftmax* and *NLL* were used in the actual implementation for numerical stability).

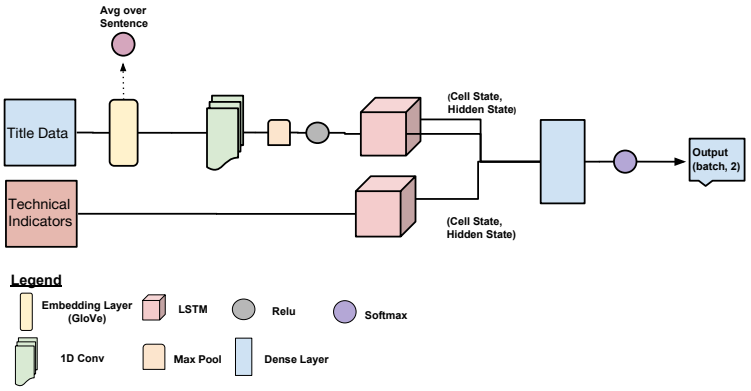
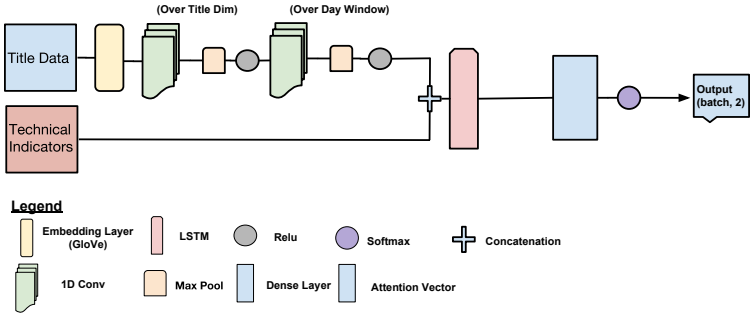


Figure 1: Baseline model using technical indicators and news headlines

3.2 RCNN with convolutional title embeddings

Regarding the news headline embeddings, the original baseline took the average of each word embedding to form an overall sentence embedding. This essentially assigns equal weight to each word and ignores the sequentiality of the words. To improve upon this shortcoming, we implemented another convolutional layer with max-pooling over the words in the title. This allowed for greater expressiveness of the context of the words in the titles.



hh

Figure 2: RCNN using 1D convolution over words in news titles

3.3 RCNN - conv titles with Multiplicative Attention

With this RCNN sequence model we sought to pass the last hidden states of the final LSTM through an attention vector to help our model focus on the days in the window that most helped in predicting the next day's movement. This attention vector was enforced to have a distributional prior (weights must sum to one and must be greater than zero). This also gives our model a semblance of interpretability as each weight corresponds to the the importance of a certain day in the window on the output of our model.

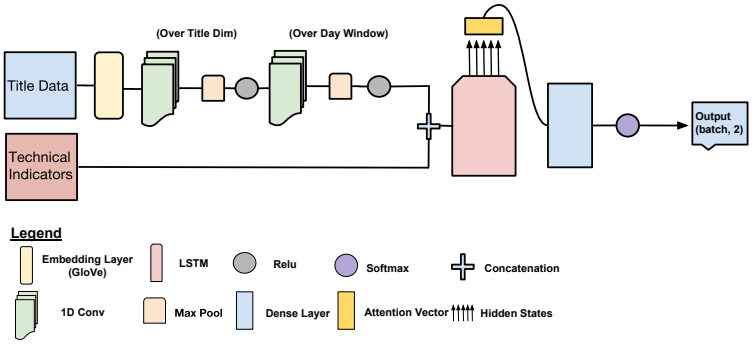


Figure 3: RCNN with attention

3.4 RCNN-LSTM with LSTM-based news embeddings

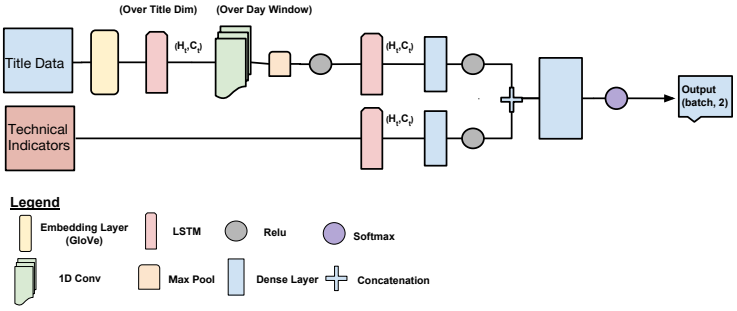


Figure 4: Sequential Title RCNN

The RCNN-LSTM is similar to the base RCNN but it does not ignore the sequentiality within each title in title data. Instead of relying on the first 1 dimensional convolution to produce these embeddings over the title dimension, we instead used an LSTM's output hidden states as our embeddings. Then we fed both Title and Tech Indicator data through a LSTM and Dense layer to allow the model to learn which elements of each data stream were important before we concatenated them and fed them through one last dense embedding.

3.5 Support Vector Machine

We implemented a SVM model with a **radial basis function** kernel because we wanted to compare its performance with our RCNN models. As input features, we used the same headline embeddings and technical indicators as we used with our RCNN models. However, realizing that financial patterns across long periods of time are difficult to extract, we decided to look at windows of 50 days at a time, splitting each window into train, validation, and test sets, and averaging the train and test accuracies over the entire dataset. Our goal with this model was to show how the expressivity of the machine learning model does not matter as much as the time window for financial time series. Given that the SVM has a smaller parameter space than the RCNN based models, one would expect poorer results; however, this is not the case, as we will discuss in the Analysis.

3.6 Logistic Regression

We implemented three variations of a logistic regression model for the purposes of comparing simpler models with our RCNNs. We tried using bag-of-words, n-grams[3], and GloVe embeddings. The GloVe

[3]Code for bag-of-words and n-gram models adapted from: https://www.kaggle.com/ndrewgele/omg-nlp-with-the-djia-and-reddit

embeddings were the same 300-dimensional ones used in the RCNN models as features extracted from the news headlines.

4 Experiments

4.1 Dataset

Our dataset was obtained from Kaggle at the following link:

`https://www.kaggle.com/aaron7sun/stocknews/home`

The dataset contains approximately 74,000 Reddit news article titles with their corresponding dates, as well as data pertaining to the DJIA per day (open, high, low, close, volume). For each day, the dataset also includes the label 1 if the DJIA close value rose or stayed the same and 0 if the value decreased. The dates for the entire dataset range from **August 8, 2008 to January 7, 2016**, excluding weekends, which gives us slightly under 2,000 dates to work with. Our project makes use of a combination of the top 25 article headlines per day and the **high**, **low**, and **close values** of the DJIA per day used to calculate a series of 7 technical indicators used as input. The technical indicators for the first 100 days are plotted in **Figure 5**. We predict a binary label of whether or not the value of the DJIA on a given day rose/stayed the same or if it decreased.

We split the data into training, validation, and test sets. Ordered chronologically, the first 1600 days were used for training (approx. 80% of the dataset), the next 200 (approx. 11%) for the validation set, and the last 180 (approx. 9%) for the test set.

Table 1: Class balance for the dataset

Dataset	Positive label (1)	Negative label (0)
Entire dataset	0.534	0.466
Training set	0.542	0.457
Validation set	0.482	0.517
Test set	0.530	0.469

4.2 Evaluation Methods

To train our model, we used the multiclass Cross-Entropy Loss objective, as specified in Vargas et al. The objective was defined as:

$$\mathcal{L} = \frac{1}{m} \sum_{i=1}^{m} y_i log(\hat{y_i}) + (1 - y_i) log(1 - \hat{y_i})$$

The model was trained with the **Adam** optimizer with a constant learning rate of 0.1 (although other values for this hyperparameter were experimented with) and a momentum of 0.9.

To evaluate performance, we used binary classification accuracy on the predicted labels, since this was the evaluation metric used in the majority of the baselines previously run on this dataset. We implemented this binary classification accuracy using Log Softmax and Negative Log Likelihood Loss for stability purposes.

We trained all RCNN models for 20 epochs, saving the model that did the best on the validation set.

4.3 Experimental Details

4.3.1 Random Sampling of Article Headlines

We started by experimenting with the baseline model and using all headlines for a given day, then proceeded to vary the number of daily titles used. We also experimented with including/excluding stop words from news headlines as well as taking a random sample of n number of news headlines

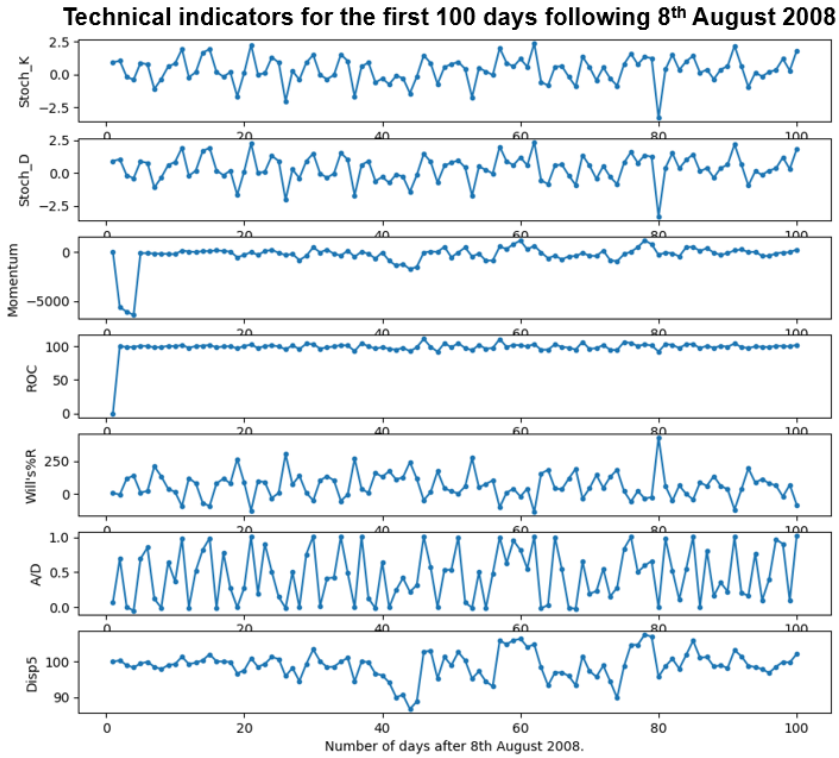


Figure 5: Technical indicators for the first 100 days in the dataset

from the top 25 headlines per day. Sampling n number of news headlines simulates a dropout layer for the input headlines and also serves to filter the data into a sparser form. Removing stop words from the news headlines also filters the input to preserve just the relevant information for our model. The RCNN parameters for our experiments remained the same as in our baseline.

4.3.2 Looking at specific training windows with SVMs

The purpose at looking at specfic training window sizes for SVM's was two fold. The first thing we wanted to test was to see how a learned model would behave across smaller time series. Secondly, we wanted to illustrate the inherent volatility of the market and how using long ago data to predict a current day trend would not be useful.

For these reasons we decided to implement our SVM with windowed training. Every n days (n = 25, 50 for our results) was treated as a separate "chunk" of the dataset. Within each chunk we fit the SVM to the first 80% of the data. Then we tested this fit on the next 20% of said chunk. In the next chunk we would discard the parameters learned in the previous chunk and start the process over. We then averaged our testing and training accuracy across chunks to produce our overall training and test accuracies reported in Table 1.

4.4 Results

See **Table 2**.

5 Analysis

5.1 General

As we have seen from the variety of different models tested upon this dataset and the variety of approaches, the task of predicting a financial time series is inherently a very difficult task. Each

Table 2: Results: Train and Test Accuracies for Different Models

Model	Train Accuracy	Test Accuracy
RCNN Baseline	0.535	0.526
RCNN (**18/25 titles randomly selected**)	0.501	**0.548**
RCNN-convolutional embeds (without attention)	0.514	0.516
RCNN-convolutional embeds **with attention**	0.492	**0.562**
RCNN-LSTM	0.500	0.526
RCNN-LSTM with only technical indicators	0.552	0.527
SVM, training window = 25 (*title window=5*)	0.673	0.500
SVM, training window = 50 (*title window=1*)	0.761	**0.579**
SVM, training window = 50 (*title window=5*)	0.626	**0.579**
Logistic Regression (BOW)	1.00	0.447
Logistic Regression (n-grams)	1.00	**0.564**
Logistic Regression (GloVe)	0.859	0.486

model has to learn how to generally predict how the DJIA changes for an entire year given training data from 2-6 years ago. Given that the stock market itself may not have a constant function for calculating movement, having a model learn to extrapolate based on the top /r/worldnews titles and some technical indicators from the past five days is inherently problematic. This memory issue in the stock market is best illustrated by our best performing models, the radial basis function SVM models with windowed training. As they moved across the dataset, they threw away any of the parameters associated with the previous 50 days (training window length) and instead learned entirely new parameters based on the current 40 days to predict the next 10. This seemingly trivial algorithm produced better results than RCNN models with 50x the training time and many more parameters. This is most likely caused by the RCNN's more odious task of predicting 2 years worth of DJIA movement from the previous 6 years' movements. Fortunately, compared to the standard results on the Reddit kaggle dataset (56 $\%$ test accuracy on a subset of samples), we beat the baseline with at least two models (SVMs). Our test accuracies still lag behind the ones reported in **[2]**. This indicates that the Dow Jones Industrial Average may not be as expressive/variant to news data as the S and P 500 is. Even with models that had more parameters with sequential embeddings (such as RCNN seq with attn), they didn't produce results much better than the original RCNN baseline model with randomized titles. This suggests that the volatility of the market cannot be accurately captured with the current techniques deployed.

Ironically the more "financially viable" method is to overfit the model based on small time series data and extrapolate to only a few values of movement after the training set.

5.2 Using attention to selectively look at news titles

We used a normalized Linear layer to allow our network to learn the optimal way to superpose hidden states from different days in the time window considered. Applying attention gave us our best deep learning model for classification, losing out only to the SVM. The attention started out evenly across the days, however, by the end of training the model focused the majority of its attention on the 2nd to last and previous day. This indicates that the market, according to our RCNN sequence model, takes 1-2 days to react

6 Conclusion

Our project hoped to explore the relationship between general news titles and stock prices, unlike most existing research that focuses primarily on financial news headlines. Overall, the RCNN models we explored performed close to random. In fact, the simpler SVM model outperformed all of the RCNN models. We were, however, able to improve upon the baseline by using another convolutional layer with max pooling to make sure the embeddings being fed into the network provide a better

Table 3: Normalized Attention distributions at different training periods

Epoch	Day *t-4*	Day *t-3*	Day *t-2*	Day *t-1*	*t*
1	0.15	0.22	0.315	0.17	0.145
20	0.0156	0.125	0.148	0.375	0.346
30	0.01	0.115	0.135	0.412	0.328

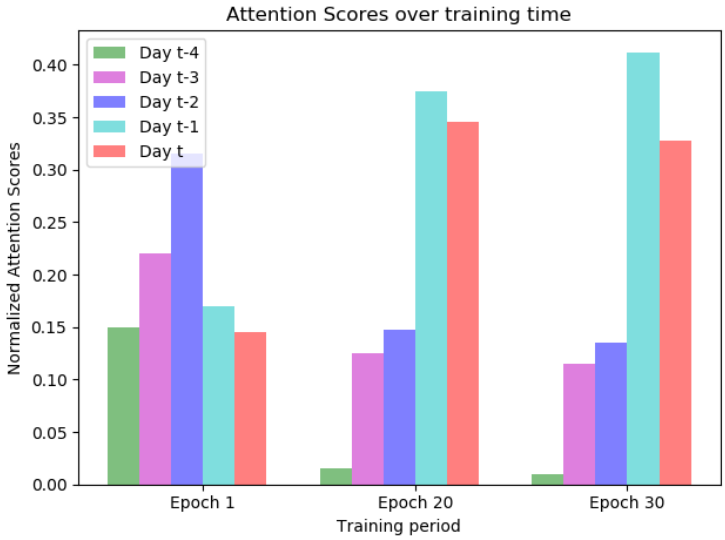


Figure 6: Normalized attention scores over training epochs

represention of the context expressed in each title, instead of simply averaging all the word vectors of each headline. This way, we can learn the embedding from context rather than relying on a pretrained model. We also implemented multiplicative attention, which improved our ultimate test accuracy.

Financial information in general is known to be extremely unpredictable, and the task of predicting stock movements remains an actively researched question by members of academia and industry alike. Future avenues of research could include experimenting with architectural adjustments, such as using transformers and transfer learning.

Acknowledgments

We would like to thank our project mentor Pratyaksh Sharma, Professor Chris Manning, and the rest of the CS 224N teaching team for their help and support throughout the quarter.

References

[1] Lee, H., Surdeanu, M., MacCartney, B. & Jurafsky, D. (2014). On the importance of text analysis for stock price prediction. Proceedings of the 9th Edition of the Language Resources and Evaluation Conference (LREC). 1170-1175.

[2] M. R. Vargas, B. S. L. P. de Lima & A. G. Evsukoff, "Deep learning for stock market prediction from financial news articles," 2017 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA), Annecy, 2017, pp. 60-65. doi: 10.1109/CIVEMSA.2017.7995302

[3] Velay, Marc, and Fabrice Daniel. “Using NLP on News Headlines to Predict Index Trends.” CoRR, 13 Aug. 2018. arxiv.org/abs/1806.09533.

[4] Zhai, Y.Z., Hsu, A.L., & Halgamuge, S.K. (2007). Combining News and Technical Indicators in Daily Stock Price Trends Prediction. *ISNN*.