Stock Price Prediction Based on Artificial Neural Networks

Ran Li
University of Toronto St.George, Department of Statistics
(Dated: August 31, 2021)

Stock market consists an important portion of capital market, close relationship can be found between the volatility of stock market and the economic status. Wise investment in stock using the price oscillation can also help individual to earn money. Therefore, there is always high demand for stock price prediction. In this project, borrowing knowledge from the field of deep learning, we construct Convolutional Neural Network (CNN) and Long Short Term Memory (LSTM) models to predict close price for S&P 500, NASDAQ, Dow Jones Industry, RUSSELL and NYSE. The model evaluation shows significant improvement in prediction accuracy compared to the baseline model constructed by support vector regression (SVR).

Key Words: : Stock Market Prediction, Convolutional Neural Network, Long Short Term Memory

I. INTRODUCTION

Financial markets are considered as the heart of the world's economy in which billions of dollars are traded every day. Not only big companies can make profit from trading on the stock market, even a single person could benefit a lot from investing as long as we make wise decision to sell or buy stock based on judgement of stock price. Therefore, a good prediction of future behaviour of markets would be extremely important, helping the company and every individual to achieve economic goals.

However, since the behavior of stock market is highly nonlinear, various factors from macroeconomics to company regulations may affect the oscillation of market, predicting its behaviour is quite challenging. Luckily, with the advancement of computer science and technology, the mystery of stock market has been revealed to some extent. First of all, the rise of big data and cloud computing has made it easier for people to process and compute massive stock trading data. In some investment companies, quantitative analysis is gradually moving from manual to intelligent. Also, machine learning technology continues to advance, computers are becoming more and more "intelligent", and computers are likely to be the new key to solving stock forecasting problems [13].

Previously, many classical models are used in stock price prediction like seasonal autoregressive integrated moving average (SARIMA) model is utilized [6]. However, the prediction accuracy is limited due to the linearity of ARIMA and the chaos in stock market. Nowadays, machine learning techniques are the most popular methods to make such predictions like random forest [2], linear discriminant analysis, quadratic discriminant analysis, logistic regression. Artificial neural networks (ANN) and support vector machine(SVM)[6] are the most common algorithms among them. In our work, we will use SVM as our baseline model and compare it with our models built using ANN [14].

Deep learning, as a part of machine learning family, constructs multiple layers to progressively extract higher-level features from the raw input and is suitable for automatic features extraction and prediction, having made great contribution in computer vision, speech recognition and natural language procession [10]. Since influential factors for stock market is chaotic, utilizing deep learning to construct complex nonlinear relation seems helpful. To our knowledge, in this field, it is popular to construct back-propagation (BP) neural network [8], multiple layer perceptron (MLP) [14], convolutional neural network (CNN) and recurrent neural network (RNN) [14]. Provided the variability of influential factors, people also tried to incorporate feature selection tool into their models. For example, Zhong and Enke (2017) applied principle component analysis (PCA) in order to extract better features. A collection of different variables was used as input data while an ANN was utilized for prediction of S&P 500. The results showed an improvement of the prediction using the features generated by PCA. Such results motivate us to put more weights on feature selection. We designed the CNN layers in order to achieve this goal. Also, since stock price data is indeed time series data, we also construct long short term memory (LSTM) to reveal relation between the past and future. In this work, we find that our LSTM model and CNN model outperforms the baseline model SVM significantly in prediction.

The rest of this work will be summarized as followed: In section "Data", we will introduce our dataset. In section "Methodology", basic introduction of support vector machine, CNN and LSTM will be presented we will also show how we design our model based on this topic. Next, in result section, we are going to show how we pre-processed data and model testing results. Finally, we will discuss interpretation of results, model limitation and how improvement can be made in the last section.

II. DATA

Dataset that we used in this project come from Ehsan and Saman's work (Hoseinzade Haratizadeh, 2019) [1and it contains 5 stocks to put into analysis, which are S&P 500, NASDAQ Composite, Dow Jones Industrial Average, RUSSELL 2000, and NYSE Composite. Each dataset contains daily stock market information from 2009-12-31 to 2017-11-15 (1984 days in total), including daily close price, features from various categories of technical indicators, futures contracts, price of commodities, important indices of markets around the world, price of

major companies in the U.S. market, and treasury bill rates, 82 features in total. See Fig 6 in the Appendix section for detailed information. We will introduce data preprocessing in the "Results" section.

III. METHODOLOGY

A. Support Vector Machine

In machine learning, Support Vector machine (SVR) is known for its high prediction accuracy. Support Vector Regression gives us the flexibility to define how much error is acceptable in our model and will find an appropriate function to fit the data. When the relationship is linear, the regression plane is computed by minimizing the L^2-norm of the coefficient vector while constricting the absolute error less than or equal to a specified margin. Since it is potential to observe data points out of the range of error margin, we also introduce a slack variable ξ_i (ξ_i is below the margin and ξ_i^* is above the margin) for the deviation from the margin if the point is our of margin and add it on the objective function. It can be shown as formulas below:

$$\min\{\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi + \xi_i^*\}$$

such that

$$\begin{cases} y_i - \mathbf{w}x_i - b \leq \epsilon + \xi_i^* \\ \mathbf{w}x_i - b - y_i \leq \epsilon + \xi_i \\ \xi_i, \xi_i^* \geq 0 \end{cases}$$

where $\mathbf{w}$ are weights, ϵ is the specified margin, called the maximum error, C is hyperparameter that controls tolerance for number of points outside the predefined margin.

Indeed, such model can be generalized into nonlinear setting. The key idea is that we take nonlinear transformation of data into higher dimension with the help of kernel function and then apply linear SVR [2].

It is shown that support vector machine regression can be useful in time series prediction (Thissena and Brakel, 2003) [1]. Since it is less computational-costing, we first model the stock market using this model before constructing neural networks where we take $C = 1000$, $\epsilon = 0.1$, kernel function as "RBF", which is a widely chosen nonlinear kernel, and this model serves as the baseline.

B. Convolutional Neural Network

Convolutional neural network (CNN) is an important class of artificial neural network (ANN), widely applies to computer vision for image recognition and classification [9]. CNNs are improved version of fully-connected neural network, which means each neuron is connected with all the neurons in the next layer. However, such fully connected structure may meet two main difficulties in practice. The first one is overfitting, which occurs quite often that training accuracy is close to 1 but when the trained model is applied on test data, accuracy drops drastically. Also, when the input size is large(common situition when input is image), there is huge amount of parameters needed for estimation in the neural network, making computation time-costing. CNN models can be viewed as regularized version of such fully connected network which helps deal with the issue of overfitting and reduce computation by introducing concept of convolutional layer and pooling layer.

1. *Convolutional Layer*

   Convolutional layer is the main building block of CNN models. Input data is processed by a given size of learnable tensor called filter or kernel through convolution (a defined computation by taking dot product between numbers in filter with numbers in input data on corresponding position).By taking such convolution, we can effectively reduce amount of data and extract most useful feature. That is because through the optimization of numbers in the filter, we can find the optimal weight for each feature and ignore noise. Such optimization is usually realized through back-propagation by stochastic gradient descent(SGD) or batch gradient descent.

2. *Pooling Layer*

   Pooling layer is often added between convolutional layers or before flattening for fully connection. Different from convolution, pooling is a function for data clustering together to reduce dimension and avoid overfitting. Commonly used pooling functions are max pooling and average pooling.

In the first model, a CNN is built. We first apply convolution with a 82×1 filter. Such filter computes the linear combination of 82 features for each day. By optimizing filter numbers(weights), we can achieve the goal of selecting most useful features and reduce data. Such processing is the same idea when we use CNN to do image classification where convolution is applied to extract image information in space. Secondly, a 3×1 filter is applied for temporal information. Since stock data is indeed a time series data, information of past days have significant influence on the future. The first convolution reveals spacial information and the second one reveals temporal information. Next, we perform the 2×1 MaxPooling to avoid overfitting and repeat previous 3×1 2D convolution again for furture extraction.

Lastly, we flatten the reduced data, add the fully connected layer (dense layer) and apply "ReLu" as the activation function for final prediction. In Ehsan and Saman's paper, they build similar model but use "sigmoid" as final activation function to predict binary output, that is whether the close price in the next day will

be higher than the previous. See Figure 1 for the model visualization.

C. Long Short Term Memory Network

Long short term memory network (LSTM) is a kind of Recurrent Neural Network (RNN) supposed by Hochreiter and Schmidhuber (1997) [7]. Traditional RNN is able to take previous states into connection to catch temporal information and is useful in time series prediction. However the problem of gradient vanishing and explosion during back-propagation as well as its incapability in connecting information between large time gaps (long-term dependency problem) narrow its application. The invention of LSTM was developed to resolve these problems and become more widely used in time series prediction.

The key idea of LSTM is the concept of cell state. We use sigmoid function and previous hidden state (defined in classical RNN) to construct gates in order to control how much to forget or put into the cell state. Next, the update of cell state helps to update the hidden state and generates output. Fig 2 illustrates how each LSTM unit works [4]. It is verified that this structure solves the long-term dependency problem and also avoids gradient vanishing and explosion during back-propagation. LSTM have got great achievement in analyzing sequential data, contributing to the field of handwriting and speech recognition. The time-dependent characteristic of stock market motivates our utilization of LSTM. We will show how accurate of LSTM is making prediction.

Indeed, 2 models are built using LSTM. The first one is training data with all 82 features in the dataset while the second one is training data with close price in past days as the only feature. We intend to compare these two models and try to explain important features in predicting stock market. We add LSTM layer with 128 units followed by a LSTM layer with 64 units to track time reduce dimension. Next, two fully connected dense layers are added for final prediction. See table 2 for model summary.

IV. RESULTS

A. Experimental Setting and Data Preprocessing

We split our original dataset into two parts. Data points from 2009-12-31 to 2016-3-31 are used to generate traning data for the purpose of training each model and the close price of remaining data points from 2016-4-1 to 2017-11-15 make up the test dataset.

It is worth noting that different features may have significantly different ranges. If we directly plug in all the original values into the model, there will be more effect on some variables due to their larger absolute value which may lead to bias in our model. Therefore, we use the `MinmaxScaler` in sklearn package [16] to scale all the features as well as the response variables by using the formula below.

$$X_{scaled} = \frac{X - \min x}{\max X - \min X}$$

After such scaling, each feature should be within the same range $[0, 1]$ which helps us to resolve the problem.

For our baseline model using Support vector regression, all features including date are used to train the model.However, for the our CNN and LSTM models, we need to perform extra steps to create training data, that is before we would like to use close price in previous 60 days to predict future where the influence of past to future is taken into consideration. Therefore, for CNN and LSTM, each data point should contain all information in the past 60 days (including close price in past days). Training data for CNN and LSTM with 82 features have input size (1533, 60, 82) and for LSTM with 1 feature, we have input size (1553, 60, 1).

B. Results

We use `SVR` class from "scikit-learn" package on Python 3.9 with kernel chosen as nonlinear "rbf", $C = 1000$ and $\epsilon = 0.1$ to execute all the computation and `keras` from `Tensorflow` [17] is used to construct neural networks (CNN and LSTM) by adding corresponding layers as illustrated in the Methodology section. Note that we fit all the neural networks with `Adam` optimizer. This is chosen because efficiency in computation and little memory requirements [12].

Three images in Fig 3 show the convergence during model optimization as it can be seen that training loss is decreasing down to 0 as more epochs are run so does the validation loss. Indeed we can see that parameters converges really quickly. Results get stabilized only after a few epochs. However, we still run 50 epochs to ensure accuracy.

Figure 2 shows the predicted close price of S&P 500 in the next half year where green line represents true value. It can be told vaguely that LSTM prediction using only close prices in past days is the most accurate prediction. CNN with 82 feature is also close to the real curve. Figure 3 is the graph for prediction error of each model on S&P 500 prediction. Note that Figure 3 is created by calculating difference between values on each curve in Figure 2 and the green curve representing real values. Similar plots are created for other 4 stock market as well and will be found in the Appendix section.

Model Evaluation Measures

1. *Mean Squared Error*

$$MSE = \frac{1}{n}\sum_{i}^{n}(y_{true} - y_{pred})^2$$

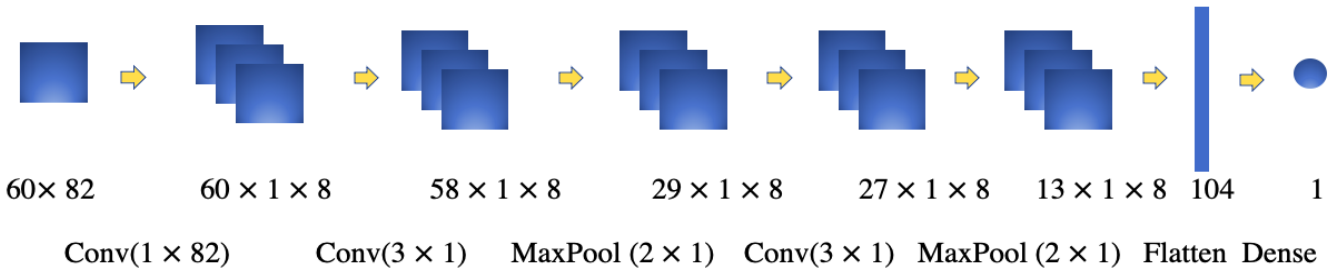
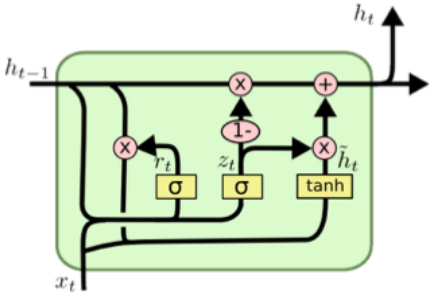
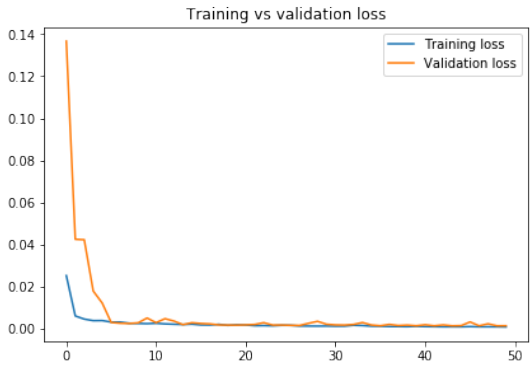


FIG. 1: CNN Model Visualization.

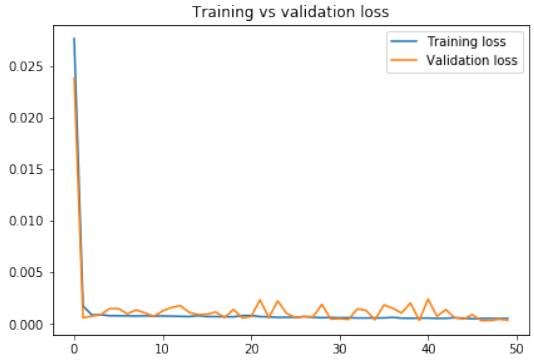


$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

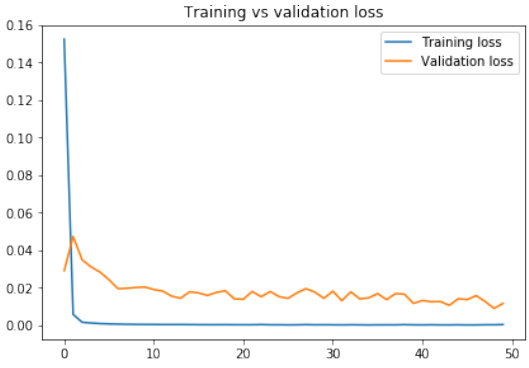
FIG. 2: LSTM Unit.



(a) CNN



(b) RNN(1)



(c) RNN(82)

FIG. 3: Convergence of ANNs

2. *Scaled Mean Squared Error*
MSE is computed as above except that y_{pred} are raw model prediction without inverse scaling transformation. This also shows how model loss in Figure 1 is computed.

3. *Percent of Correct Direction*
Percent of Correct Direction is the percent of correct prediction of whether the stock close price increases or decreases compared with the previous day.

4. *Mean Error*

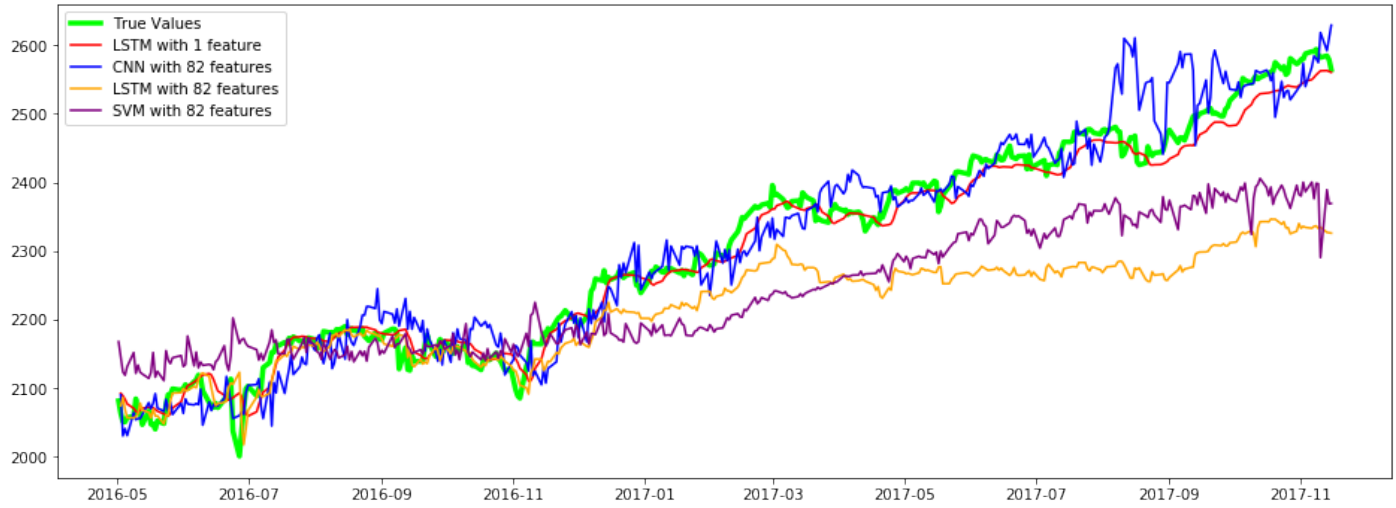


FIG. 4: Predicted close price of S& P 500 index using 4 models with comparison to the true values.

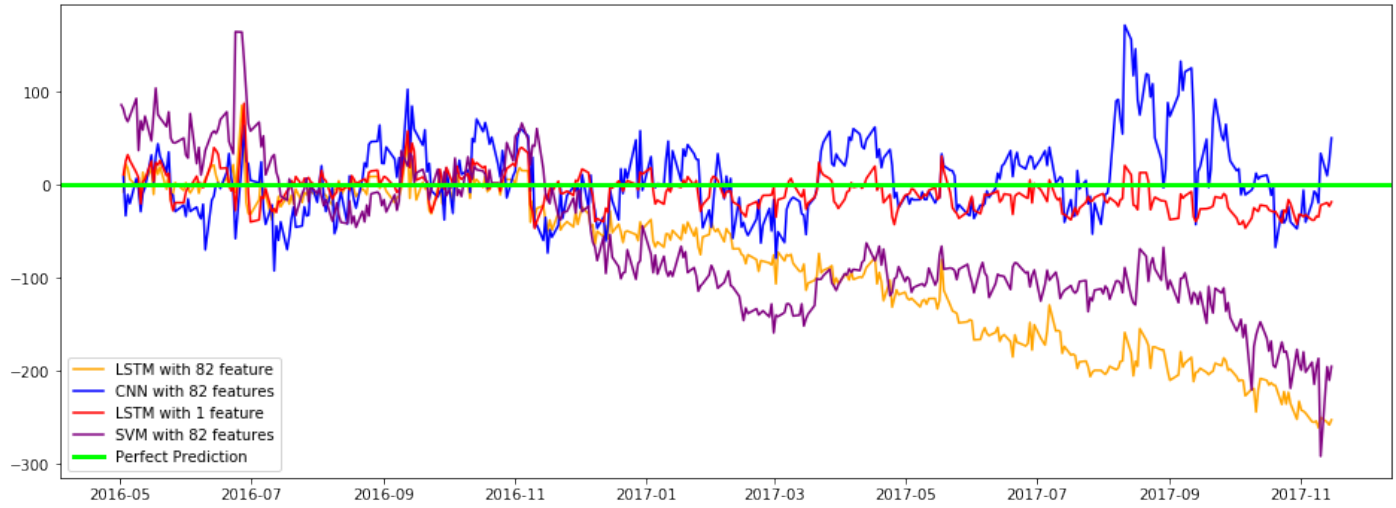


FIG. 5: Prediction error of each model.

$$Mean\ Error\ = \frac{1}{n}\sum_{i}^{n} y_{true} - y_{pred}$$

The sign of mean error can be used to detect whether the model tends to overestimate or underestimate the stock price.

Table 1 shows the summary of measures for each model on these 5 stock markets. Further interpretation will be presented in the next section.

V. DISCUSSION

In this work, we train 4 machine learning models (support vector regression, convolutional neural network, LSTM with 1 feature and LSTM with 82 features) to predict the close price of 5 different stocks from 2016 May to 2017 November.Although the stock market is highly non-linear and influenced by various factors, our prediction provides knowledgeable information regarding the current status of the stock price movement. Visualization for S&P 500 prediction along with plots in the Appendix for the other 4 stock markets provide a vague sensation for model accuracy. For S&P 500, the red curve is closest to the green curve, which represents 100% accuracy. But in order to perform deeper analysis, we need to start from Table 1.

Note that since different stocks have significantly different ranges, it makes no sense to compare MSE among different stocks directly. For example, S&P 500 is approximately within range 2000 to 3000, the maximum MSE is around 1000,000. While NYSE is within range 10,000 to 15,000 with maximum MSE around 25,000,000. Directly comparing MSE even for the same model is deceptive. We cannot evaluate prediction of same model on different stocks without scaling. Also, in order to evaluate models, we need to compare measures of models while holding market the same. MSE is the second moment of the error, and thus incorporates both the variance of the estimator (how widely spread the

		SVR	LSTM(82)	CNN	LSTM(1)
MSE	S&P500	9593	13097	4590	490
	NASDAQ	134900	448961	23139	8887
	DJI	1071849	1865929	366405	70825
	RUSSELL	9805	9704	3376	514
	NYSE	220113	223278	114641	9560
Scaled MSE	S&P500	0.027	0.01	0.0037	0.0004
	NASDAQ	0.028	0.046	0.002	0.0009
	DJI	0.025	0.025	0.005	0.0009
	RUSSELL	0.05	0.02	0.007	0.001
	NYSE	0.036	0.009	0.0049	0.0004
Correct Direction	S&P500	52%	44%	46%	50%
	NASDAQ	53%	46%	53%	55%
	DJI	53%	54%	54%	55%
	RUSSELL	57%	50%	56%	48 %
	NYSE	57%	46%	43%	51%
ME	S&P500	-62	-87	39	-2
	NASDAQ	-234	-525	-63	-48
	DJI	-396	-1043	450	-146
	RUSSELL	-72	-79	35	-11
	NYSE	-312	-333	234	23

TABLE I: Summary of Model Evaluation

estimates are from one data sample to another) and its bias (how far off the average estimated value is from the true value). The smaller MSE is, the better accuracy and stability we have for the prediction and it serves as the main measure to evaluate models. Note that for all the 5 stock markets, LSTM with past days close price as the only feature achieves the lowest scaled MSE and CNN with 82 features achieve the second lowest. Except for NASDAQ, LSTM with 82 features included obtain slightly lower scaled MSE than SVR for the other 4 markets. Both CNN and LSTM with 1 feature outperform our basline SVR models significantly. It is also worth noting that if we observe mean error of SVR or observe the SVR line in Figure 2, we find the SVR tends to underestimate price in all these 5 stock markets, similar phenomenon applies to LSTM with 82 features as well. But luckily, CNN and LSTM with 1 feature are pretty stable around line 0 in figrue 2, suggesting their better prediction.

Indeed, nice performance of CNN is not out of our surprise because feature selection is performed when we add the first convolutional layer with filter size (1×82) and historical information is also taken into consideration when we added the second Conv2d layer with size (3×1). Also, maxplooing layer avoids overfitting. However, it is surprising that CNN is worse than LSTM with only 1 feature. Since LSTM did better in memorizing long-term data, it seems that in prediction stock close price, we need this strong model to and reveal relationship between past and future. But the bad performance of LSTM with 82 features also shows that LSTM is not good at feature extraction as good as CNN, so maybe combining CNN and LSTM together will greatly improve the prediction. Specifically, we first add convolutional layer to extract most useful features for us and then apply LSTM to consider the past information. Indeed, such combination of CNN and RNN has been utilized in prediction on time series like air PM2.5 index, text classification and sentiment analysis [18].

Another place where we can improve our model is features that we use. In our dataset, 82 features include information like world stock markets, exchange rate of U.S. dollar and big U.S. companies but there are other factors that may affect stock price as well, for example news, composition of board of a company, policies, news or even public emotions. Behavioral economics tells us that emotions can profoundly affect individual behavior and decision-making. For example, in 2010, Bollen, Mao and Zeng performed sentimental analysis for the text content of daily Twitter to make inference on public mood and put it as feature to predict stock price [3].

In summary, although the high non-linearity of financial market makes it hard to predict, by incorporating techniques in deep learning, we can still make progress. But improvement for prediction accuracy is always required. Constructing CNN, RNN combined model or selecting more influential factors are two directions that we can move forward in this field.

VI. REFERENCES

[1] Hoseinzade, E., amp; Haratizadeh, S. (2019). CNNpred: CNN-based stock market prediction using a diverse set of variables. Expert Systems with Applications, 129, 273–285. https://doi.org/10.1016/j.eswa.2019.03.029

[2] Vijh, M., Chandola, D., Tikkiwal, V. A., amp; Kumar, A. (2020). Stock closing price prediction using machine learning techniques. Procedia Computer Science, 167, 599–606. https://doi.org/10.1016/j.procs.2020.03.326

[3] Bollen, J., Mao, H., amp; Zeng, X. (2011). Twitter mood predicts the stock market. Journal of Computational Science, 2(1), 1–8. https://doi.org/10.1016/j.jocs.2010.12.007

[4] https://colah.github.io/posts/2015-08-Understanding-LSTMs/

[5] Wei, D. (2019). Prediction of stock price based on LSTM neural network. 2019 International Conference on Artificial Intelligence and Advanced Manufacturing (AIAM). https://doi.org/10.1109/aiam48774.2019.00113

[6] Thissen, Brakel, amp; Weijer. (2003). Using Support Vector Machines for Time Series Prediction. https://doi.org/10.1016/S0169-7439(03)00111-4

[7] Sepp, H., amp; JÜrgen, S. (1997). LONG SHORT TERM MEMORY. Neural Computation, 9(8).

[8] Guresen, E., Kayakutlu, G., Daim, T. U. (2011). Using artificial neural network mod- els in stock market index prediction. Expert Systems with Applications,

38(8), 10389–10397.

[9] Habibi, Aghdam, Hamed (2017-05-30). Guide to convolutional neural networks : a practical application to traffic-sign detection and classification. Heravi, Elnaz Jahani. Cham, Switzerland. ISBN 9783319575490. OCLC 987790957.

[10] Hu, J.; Niu, H.; Carrasco, J.; Lennox, B.; Arvin, F. (2020). "Voronoi-Based Multi-Robot Autonomous Exploration in Unknown Environments via Deep Reinforcement Learning". IEEE Transactions on Vehicular Technology. 69 (12): 14413–14423. doi:10.1109/TVT.2020.3034800. S2CID 228989788. Archived from the original on 2020-11-16. Retrieved 2021-05-04.

[11] Due, G., Zhou, Y., amp; Han, R. (2018). Neural networks for stock price prediction.

[12] Diederik, K., amp; Ba, J. L. (2015). ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION.

[13] Jia, Z. (2017). A Stock Prediction Method based on News Features and the LSTM Model.

[14] Wang, J., Wang, J. (2015). Forecasting stock market indexes using principle com- ponent analysis and stochastic time effective neural networks. Neurocomputing, 156, 68–78.

[15] Di Persio, L., Honchar, O. (2016). Artificial neural networks architectures for stock price prediction: Comparisons and applications. International Journal of Circuits, Systems and Signal Processing, 10, 403–413.

[16] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

[17] Abadi, Martx27;in, Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., ... others. (2016). Tensorflow: A system for large-scale machine learning. In 12th *USENIX* Symposium on Operating Systems Design and Implementation (*OSDI* 16) (pp. 265–283).

[18] Huang, C.-J., amp; Kuo, P.-H. (2018). A deep cnn-lstm model for particulate matter (pm2.5) forecasting in smart cities. Sensors, 18(7), 2220. https://doi.org/10.3390/s18072220

VII. APPENDIX

Code generated results can be found in the following link:

https://github.com/ranli123/Stock-Price-Prediction-Based-on-ANN

In the next page, we will present feature descriptions in Fig 6, predicted values, predicted error of each model for four stock markets.

#	Variable	Description
1	Day	which day of the week
2	Close	Close price
3	Vol	Relative change of volume
4	MOM-1	Return of 2 days before
5	MOM-2	Return of 3 days before
6	MOM-3	Return of 4 days before
7	ROC-5	5 days Rate of Change
8	ROC-10	10 days Rate of Change
9	ROC-15	15 days Rate of Change
10	ROC-20	20 days Rate of Change
11	EMA-10	10 days Exponential Moving Average
12	EMA-20	20 days Exponential Moving Average
13	EMA-50	50 days Exponential Moving Average
14	EMA-200	200 days Exponential Moving Average
15	DTB4WK	4-Week Treasury Bill: Secondary Market Rate
16	DTB3	3-Month Treasury Bill: Secondary Market Rate
17	DTB6	6-Month Treasury Bill: Secondary Market Rate
18	DGS5	5-Year Treasury Constant Maturity Rate
19	DGS10	10-Year Treasury Constant Maturity Rate
20	DAAA	Moody's Seasoned Aaa Corporate Bond Yield
21	DBAA	Moody's Seasoned Baa Corporate Bond Yield
22	TE1	DGS10-DTB4WK
23	TE2	DGS10-DTB3
24	TE3	DGS10-DTB6
25	TE5	DTB3-DTB4WK
26	TE6	DTB6-DTB4WK
27	DE1	DBAA-BAAA
28	DE2	DBAA-DGS10
29	DE4	DBAA-DTB6
30	DE5	DBAA-DTB3
31	DE6	DBAA-DTB4WK
32	CTB3M	Change in the market yield on U.S. Treasury securities at 3-month constant maturity, quoted on investment basis
33	CTB6M	Change in the market yield on U.S. Treasury securities at 6-month constant maturity, quoted on investment basis
34	CTB1Y	Change in the market yield on U.S. Treasury securities at 1-year constant maturity, quoted on investment basis
35	Oil	Relative change of oil price (WTI), Oklahoma
36	Oil	Relative change of oil price (Brent)
37	Oil	Relative change of oil price (WTI)
38	Gold	Relative change of gold price (London market)
39	Gold-F	Relative change of gold price futures
40	XAU-USD	Relative change of gold spot U.S. dollar
41	XAG-USD	Relative change of silver spot U.S. dollar

(a) Feature Description 1

42	Gas	Relative change of gas price
43	Silver	Relative change of silver price
44	Copper	Relative change of copper future
45	IXIC	Return of NASDAQ Composite index
46	GSPC	Return of S&P 500 index
47	DJI	Return of Dow Jones Industrial Average
48	NYSE	Return of NY stock exchange index
49	RUSSELL	Return of RUSSELL 2000 index
50	HSI	Return of Hang Seng index
51	SSE	Return of Shang Hai Stock Exchange Composite index
52	FCHI	Return of CAC 40
53	FTSE	Return of FTSE 100
54	GDAXI	Return of DAX
55	USD-Y	Relative change in US dollar to Japanese yen exchange rate
56	USD-GBP	Relative change in US dollar to British pound exchange rate
57	USD-CAD	Relative change in US dollar to Canadian dollar exchange rate
58	USD-CNY	Relative change in US dollar to Chinese yuan exchange rate
59	USD-AUD	Relative change in US dollar to Australian dollar exchange rate
60	USD-NZD	Relative change in US dollar to New Zealand dollar exchange rate
61	USD-CHF	Relative change in US dollar to Swiss franc exchange rate
62	USD-EUR	Relative change in US dollar to Euro exchange rate
63	USDX	Relative change in US dollar index
64	XOM	Return of Exon Mobil Corporation
65	JPM	Return of JPMorgan Chase & Co.
66	AAPL	Return of Apple Inc.
67	MSFT	Return of Microsoft Corporation
68	GE	Return of General Electric Company
69	JNJ	Return of Johnson & Johnson
70	WFC	Return of Wells Fargo & Company
71	AMZN	Return of Amazon.com Inc.
72	FCHI-F	Return of CAC 40 Futures
73	FTSE-F	Return of FTSE 100 Futures
74	GDAXI-F	Return of DAX Futures
75	HSI-F	Return of Hang Seng index Futures
76	Nikkei-F	Return of Nikkei index Futures
77	KOSPI-F	Return of Korean stock exchange Futures
78	IXIC-F	Return of NASDAQ Composite index Futures
79	DJI-F	Return of Dow Jones Industrial Average Futures
80	S&P-F	Return of S&P 500 index Futures
81	RUSSELL-F	Return of RUSSELL Futures
82	USDX-F	Relative change in US dollar index Futures

(b) Feature Description 2

FIG. 6: Feature Description

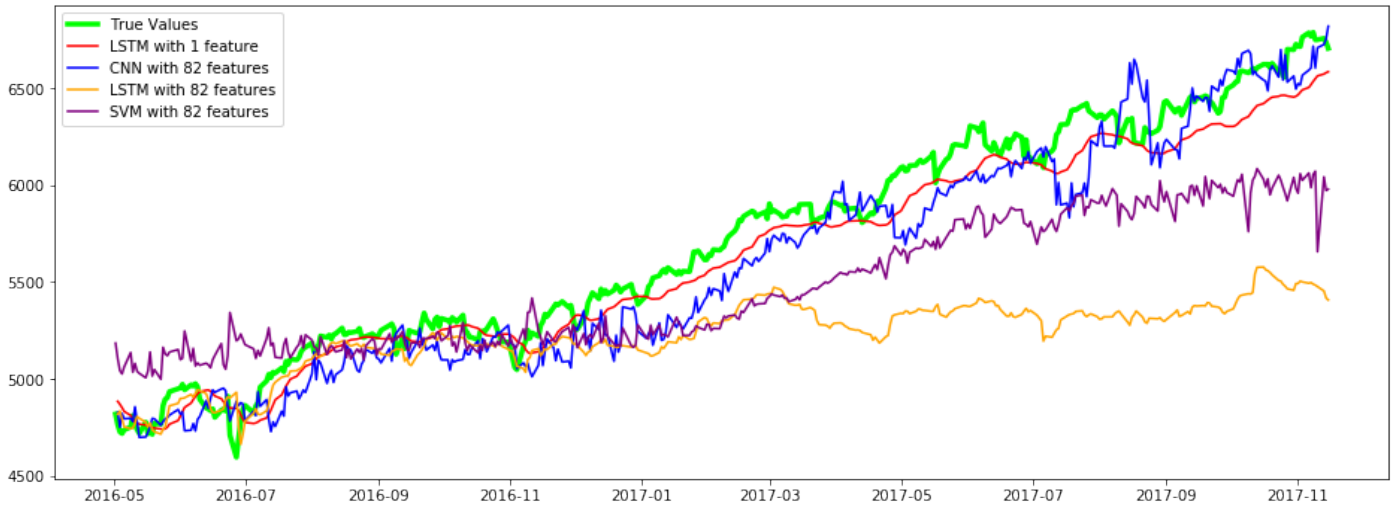
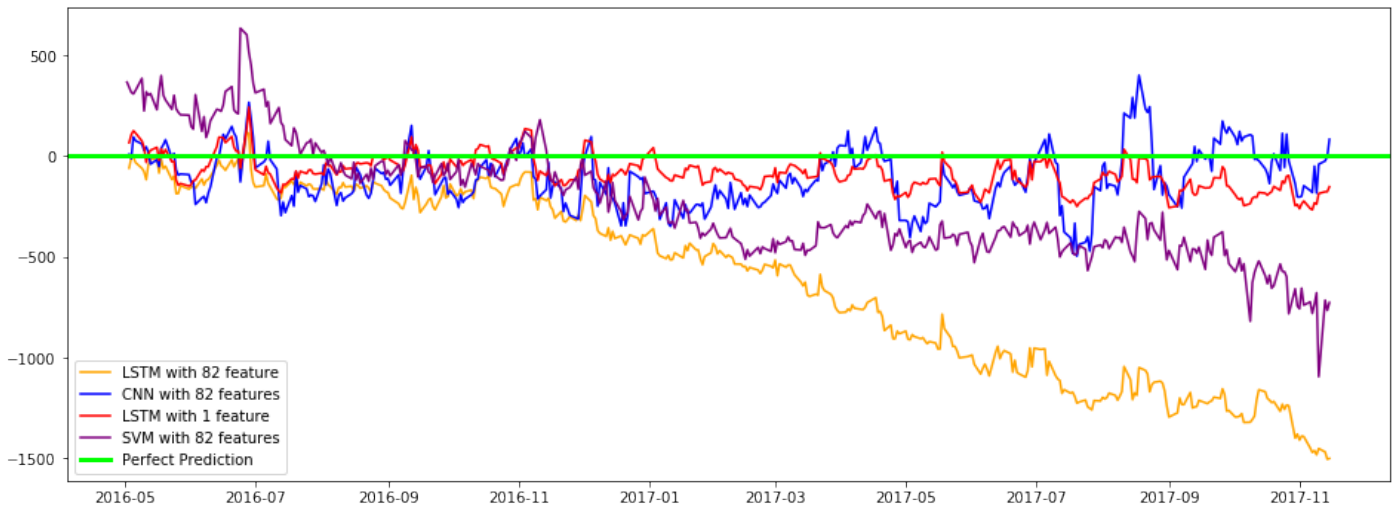


FIG. 7: Predicted close price of NASDAQ index using 4 models with comparison to the true values.

FIG. 8: Prediction error of each model for NASDAQ.



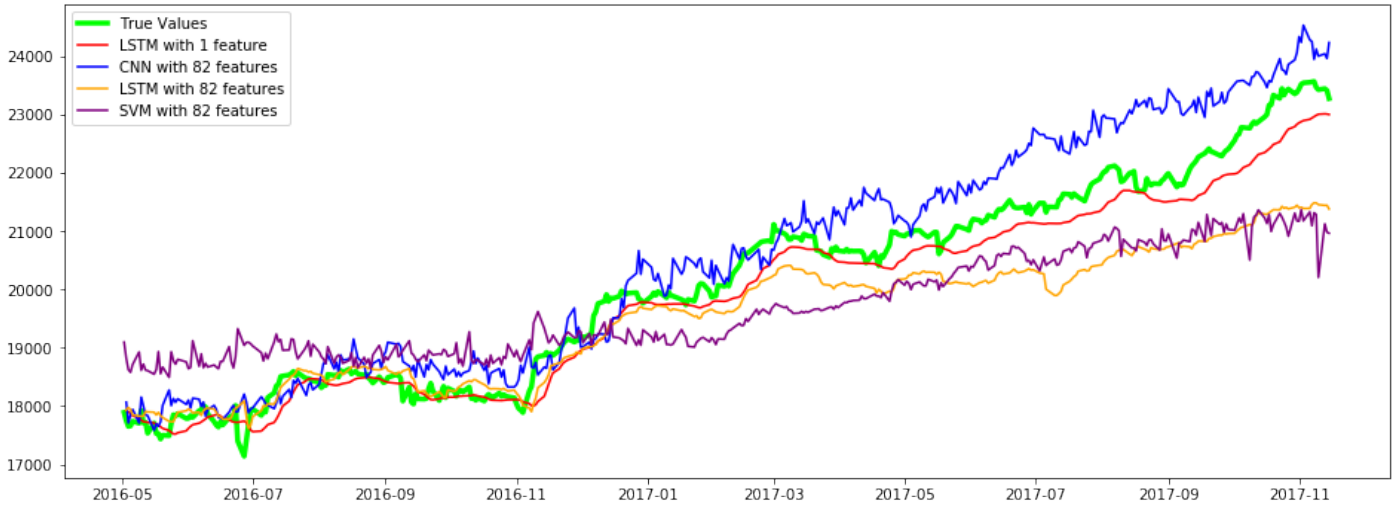


FIG. 9: Predicted close price of Dow Jones index using 4 models with comparison to the true values.

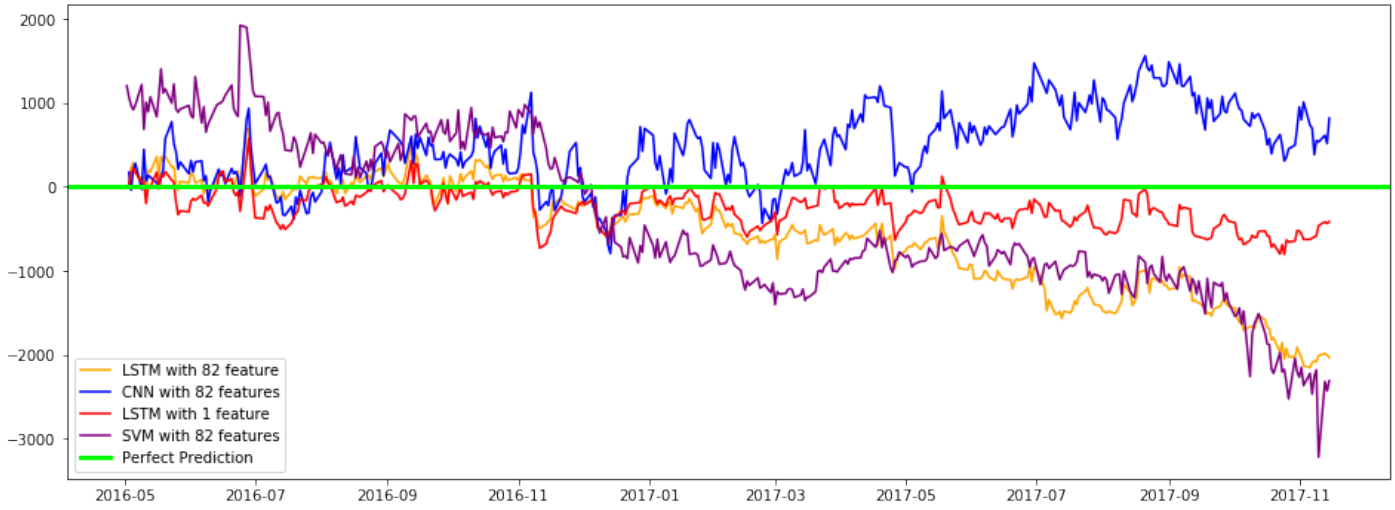


FIG. 10: Prediction error of each model for Dow Jones Industry.

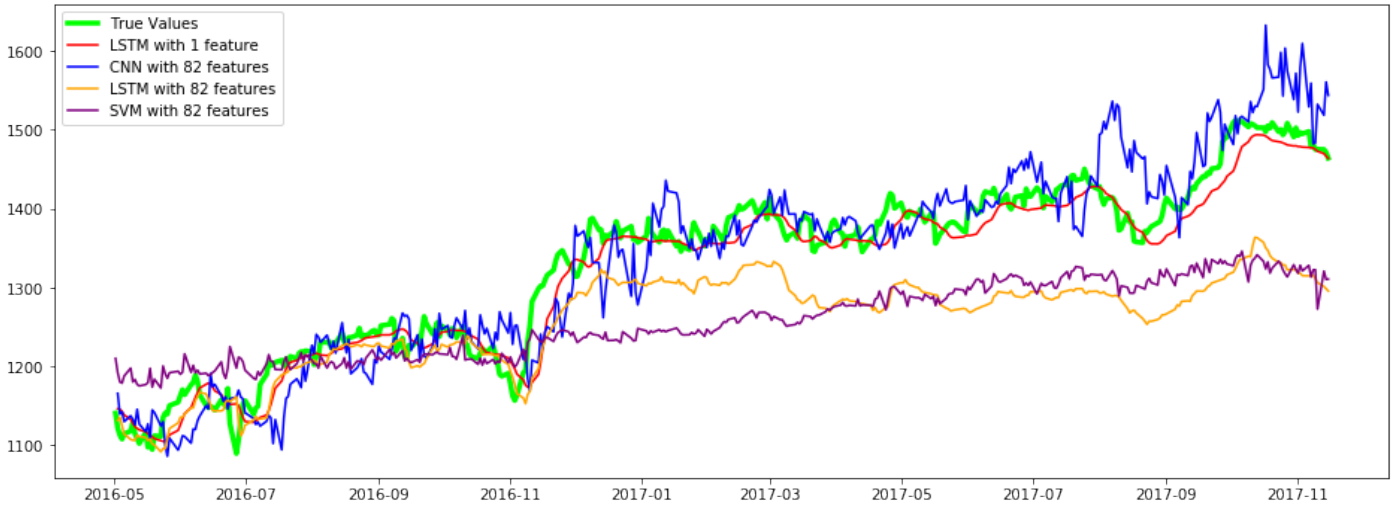


FIG. 11: Predicted close price of RUSSELL index using 4 models with comparison to the true values.

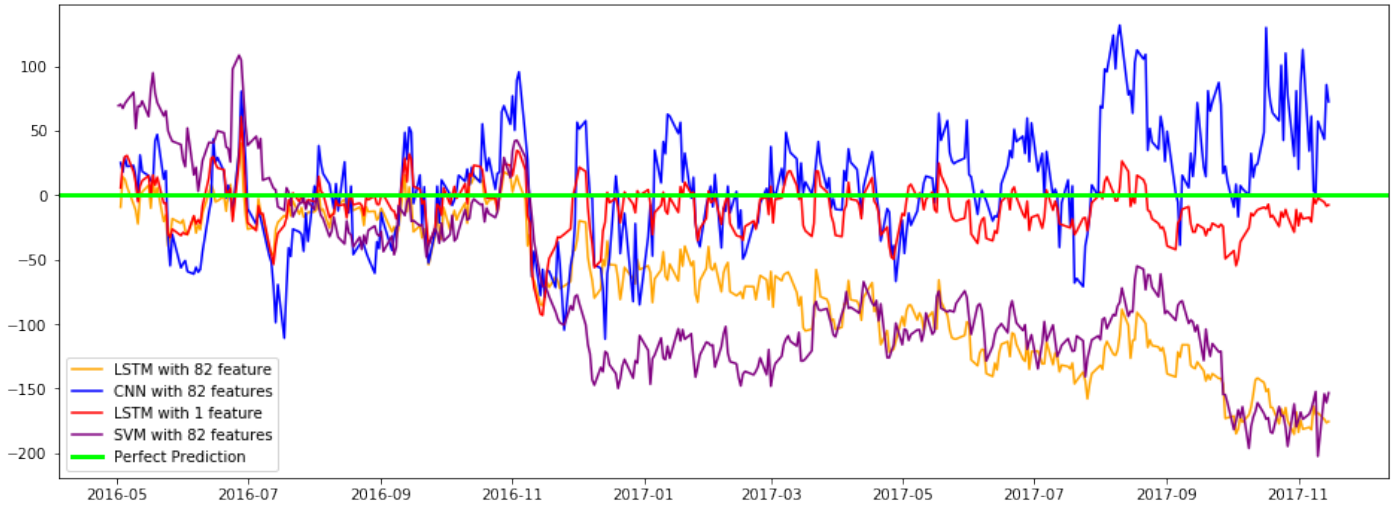


FIG. 12: Prediction error of each model for RUSSELL.

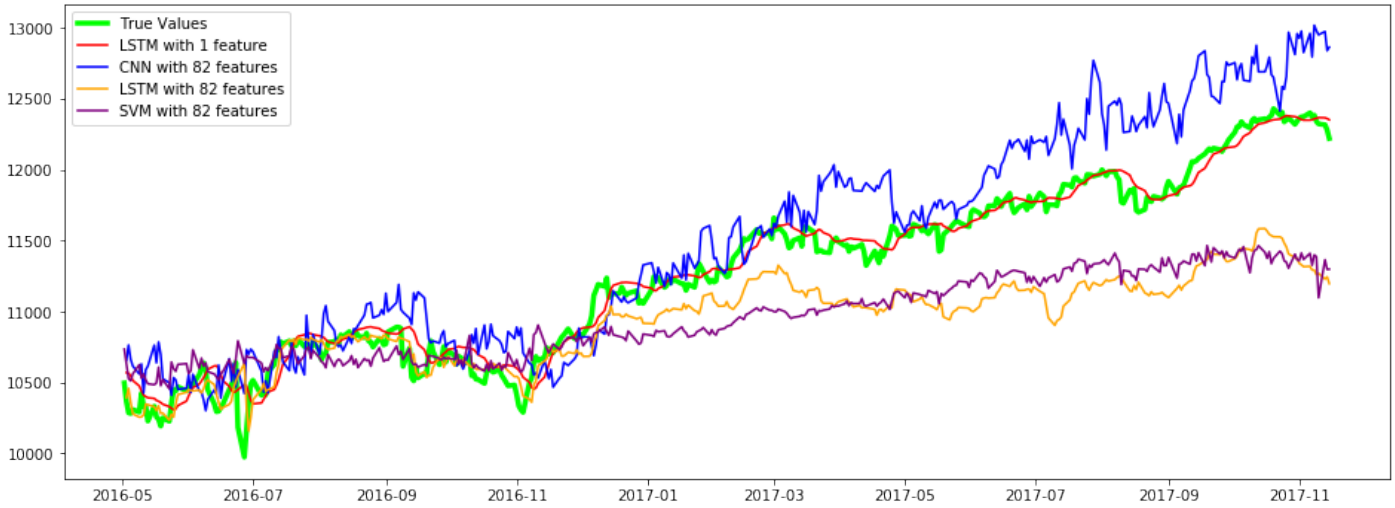


FIG. 13: Predicted close price of NYSE using 4 models with comparison to the true values.

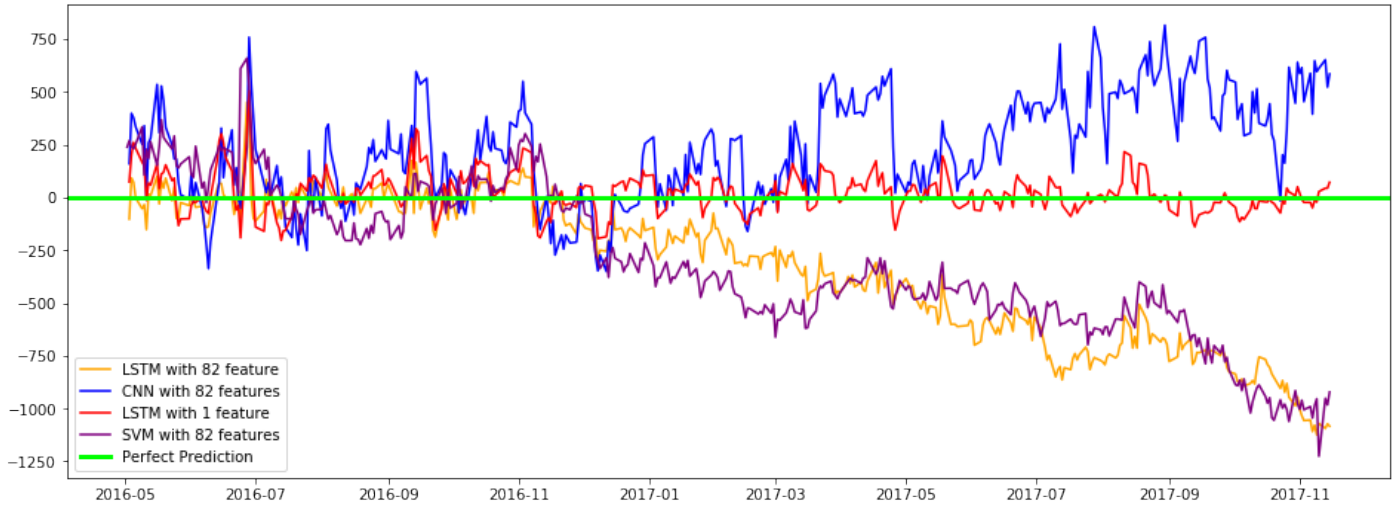


FIG. 14: Prediction error of each model for NYSE.