Stock Price Prediction with Denoising Autoencoder and Transformers

Zhiyang Chen *

Department of Electrical Engineering and Computer Science, University of California, Berkeley, Berkeley CA 94720, USA

* Corresponding Author Email: yang.chen@berkeley.edu

Abstract. Predicting stock market price movement has been a challenging problem for time series forecasting due to its inherent volatility. The introduction of machine learning techniques, such as the use of a recurrent neural network (RNN), have since dominated the field for stock trend forecasting and stock price prediction. RNNs have inherent model limitations that are solved with the introduction of the transformer model, which has since been used in many sequential classification and generative tasks. This study demonstrates the viability of a transformer-based model on the field of stock price prediction by comparing the predictions of a transformer-based model with some RNN-based baseline models on the hourly closing price of Apple (AAPL) stock prices. A dimension reducing simple autoencoder was also incorporated into the transformer model to increase performance. Although RNN-based models are shown to outperform the transformer model in accuracy, the transformer model leads in directional stock accuracy. This demonstrates that the transformer is applicable to compete with RNN based models within the field, along with the incorporation of an autoencoder to further integrate the transformer's power to this field.

Keywords: trend forecasting, stock price prediction, RNN-based models, transformer-based model, autoencoder.

1. Introduction

The impact of stock market movement is immense on economies. The task of predicting the volatility of the stock market holds significance by allowing more favorable financial decision making. Stock market prices are highly volatile and difficult to predict. One explanation is given by the efficient market hypothesis [1], stating stock prices reflect all important information related to the stock, and that information updates frequently. There are many various factors, such as political or economic events and even public sentiment, that comprise the information that influence a stock's price. This leads to a non-linear, noisy, and dynamic stock market that is erratic to predict. Stock market prediction as a problem full of rich contextual dependencies has been both challenging and resoundingly important in data science [2-4]. Before artificial intelligence was introduced, the field of stock prediction relied solely on statistical methods, which mainly relied on time-series forecasting. These approaches consisted of weighted moving average, regression analysis, autoregressive integrated moving average (ARIMA) [5], and generalized autoregressive conditional heteroskedasticity (GARCH) [6] models. However, there were limitations to these models, the most jagging of these limitations being that these models only accounted for simplistic patterns within time dependencies, which, when applied to financial forecasting, was inefficient due to the instability of financial time-series data. Due to advancements in the field of artificial intelligence, many algorithms were introduced that performed better than traditional statistical approaches. These algorithms deal precisely with modeling with complex data which are more suited for stock prediction than the traditional statistical methods.

Conventional machine learning (ML) algorithms such as k-nearest neighbors (KNN) [2], Bayesian analysis [7], support vector machines (SVM) [3] and decision trees [4] were introduced to the field of stock prediction. There was also heavy research on the usage of deep learning (DL) techniques, which is a branch of ML with models that have feature extraction built in to be learned by the network. Some of these include convolutional neural networks (CNN) [8], recurrent neural networks (RNN) [9], and specifically a variety of RNNs, the long short-term memory network (LSTM) [10] that is widely used for stock prediction. These DL techniques are generally built to process non-linearities in big data and perform superior to the statistical and traditional ML methods. These models have been used successfully in many fields such as computer vision, time series modeling, and natural language processing. The ability of deep learning networks to solve highly uncertain problems has led to its popularity within the field of stock prediction.

One particular model, the LSTM, has the advantage of preserving short term memory over long-time dependencies, so it is used in many sequential modeling tasks, most notably in natural language processing (NLP). As stock market forecasting is a time-series modeling task with long temporal dependencies, LSTMs perform well for the task of stock prediction. These deep learning models still have limitations. CNNs possess pooling layers and inherently dimensionally reduce, losing information, while RNNs struggle with the vanishing gradient problem over long sequences. The transformer [12] model was introduced recently without these limitations. The transformer itself is an encoder-decoder architecture which solely relies on the attention mechanism and performs similarly well in sequential modeling as RNNs. Transformer based models are currently state of the art models in the domain of NLP, with generative pretrained transformers being applied to many of the domain's challenges. The adoption of transformers for stock prediction has been relatively successful, as there are a growing number of models using attention-based models or transformers for stock prediction, as an alternative to the RNN based models. In the case of stock market prediction, transformers are powerful at capturing short-term recurrent trends and learning these feature trends within its model. Although the transformer has a weakness in capturing exceedingly long sequential dependencies, the strong feature learning feature of short windows allow the transformer to succeed.

Since transformers tend to capture information within small windows of sequential data, and the stock market is highly volatile within small windows, it can be favorable to denoise the data first with an autoencoder [11]. Studies have been done on the incorporation of an autoencoder with RNN based models and have shown reasonable success in modeling stock market prices. There are also models that solely rely on stacking denoising autoencoders which also obtain reasonable performance. This paper uses a similar scheme of processing using an autoencoder with a transformer model for stock prediction. The aim is to demonstrate the performance of transformer-based models on stock prediction for feature learning on financial temporal data. In addition, it will answer the question of whether a transformer-based model is capable of showing similar relevant results as state-of-the-art recurrent neural network-based models. Moreover, this study will integrate an autoencoder with transformer architecture, showing the benefits of data denoising techniques and lower dimensional representation on stock prediction and its ability to increase accuracy for transformer-based methods. The rest of the paper is as follows. Section 2 comprising of the data and method, in which the paper presents the experimental datasets as well as built models, Section 3 of the results of training and prediction using evaluation metrics, as well as comparison to other baseline models, and Section 4 the discussion of limitations and the conclusion.

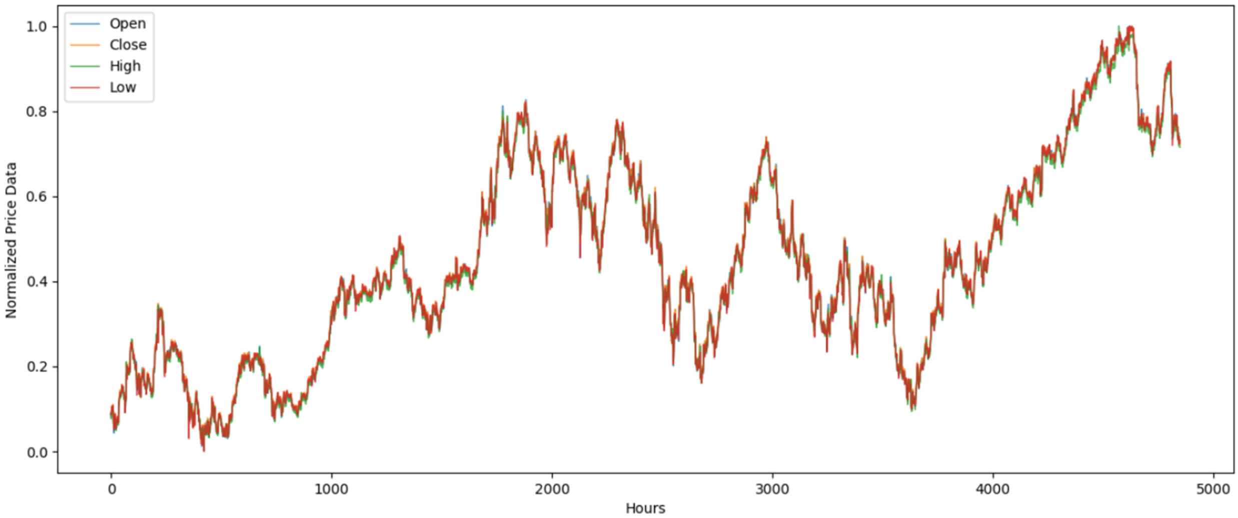


Figure 1. AAPL stock data used from the dataset (Photo/Picture credit: Original).

2. Data and Method

This paper uses the yahoo finance API for its experimental data. The daily stock market prices for Apple Inc. (AAPL) were taken from the API over six values, Open, High, Low, Close, Adjusted Close, and Volume. Open, High, Low, Close, and Volume (OHLCV) values are selected to quantify stock market prices. The first four of these are stock price values, while volume is a great indicator in stock market movements, and serves as an additional factor in predicting price movements within the stock market. To account for the transformer's ability to learn features of short sequences, as well as to provide sufficient training data, hourly values were used from a period of two years, from 2020-12-08 to 2023-09-13. The visualization is shown in Fig. 1.

The data was preprocessed using min-max normalization with the given formula:

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{1}$$

This was done for all 5 features and scaled them in the range of 0 to 1. Since stock prices are based on the valuation of the company, this normalization is necessary to keep training data from different stocks equally relevant to the model. The time series data was then split into windows of total size 72 with the first 64 values used for predicting the following 8 hours of stock prices. The architecture of the model is a simple preprocessing autoencoder used to denoise temporal data subsequently feeding the denoised data into a deep transformer network (seen from Fig. 2). The autoencoder model is self-supervised since the goal of the autoencoder is to reconstruct the input within its low dimensional representation. The autoencoder model is a traditional encoder-decoder model fitted over the OHLC to process the temporal data via denoising. The goal of using an autoencoder is to find a lower-dimensional representation of temporal OHLC and volume data. First, this study expands the dimensionality of OHLC to account for the various volatile factors within the stock market. Then this paper performs significant feature extraction and lower-dimensional representation with the autoencoder, and finally re-reduce the significant features into OHLC temporal data. A simple dense autoencoder was used for dimensionality reduction of raw stock data. The encoder and decoder are composed of a fully connected dense layer each.

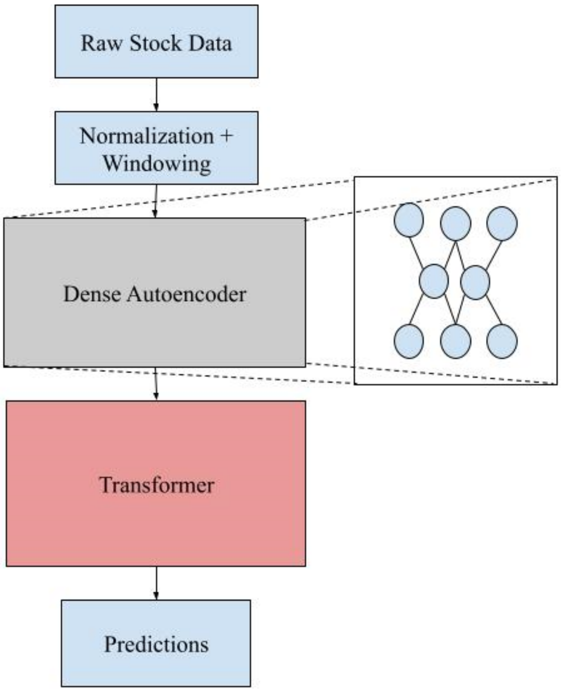


Figure 2. Model architecture (Photo/Picture credit: Original).

The transformer model is based on the original paper that introduced the model in 2017 [13]. Similar to the original, the transformer model used in this case is an encoder-decoder model using multi-headed attention based on scaled-dot product attention. The input to the transformer is the preprocessed OHLC data from the autoencoder. Volume data can be processed via normalization into labels similar to sentiment analysis and carried on as a significant feature in our temporal analysis. Using deeper layers allows for higher level characterizations of the time series data to be learned by the transformer to obtain more abstract and efficient recognition of patterns of the time series. Using a similar scheme to the original transformer paper, the model is an encoder-decoder transformer model with a 2 encoder and 2 decoder block stacks. More blocks stacking likely leads to overfitting, however this was not tested. The hyperparameters that were tested and performed well relative to the size are given in the table below. Dropout was also adopted since having a complex and large model led to overfitting (seen from Table 1).

Table 1. Model parameters

Encoder Layers	Decoder Layers	Attention Heads	FFN Hidden Size	FFN Filter Size	FFN Dropout
4	4	8	32	32	0.15

The two baseline models chosen to compare the transformer-based model to are the bidirectional LSTM based model and the bidirectional gated recurrent unit (GRU) based model, both with a simple 4 block stack. Since the windows predict 8 hours of closing data simultaneously, for any singular hour, the predictions from the 8 data windows that predict the price of that hour were averaged for the result. There are two primary metrics used in this paper. The first, also the loss function fitted onto our model, is the mean squared error (MSE). MSE is a standard choice for the loss metric in regression or estimation tasks, such as forecasting. Larger MSE correlates to predictions having larger deviation from truth labels, as it is computed by taking an average of the distance between the model's predicted label prices and the actual price. The mean absolute error (MAE) and the root mean squared error (RMSE) were also included as additional evaluators since both are derivatives of a summation over the error between predicted and truth stock prices. The formulae for the MSE, MAE, and RMSE are the following:

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(x_i - y_i)^2 \quad (2)$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|x_i - y_i| \quad (3)$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(x_i - y_i)}{N}} \quad (4)$$

The second metric is more specific to financial prediction tasks, this being the accuracy of the direction of predicted price movement. This metric measures the model's likelihood of making a correct prediction of price movement relative to that of the previous trading hour. This metric was also evaluated for closing price predictions for entire trading days. The metric was evaluated as follows:

$$\frac{1}{N}\sum_{i=1}^{d} f(x_i, y_i), where\ f(x_i, y_i) = \begin{cases} 1 & if\ (x_i - x_{i+1})(y_i - y_{i+1}) > 0 \\ -1 & if\ (x_i - x_{i+1})(y_i - y_{i+1}) > 0 \\ 0 & otherwise \end{cases} \quad (5)$$

3. Results and Discussion

The results of 15 epochs of training are given in Table 2. The model was trained on a single NVIDIA V100 without distributed training. The optimizer used was Adam with learning rate = 0.003. In the loss metrics the LSTM and GRU baseline models significantly outperformed the transformer in estimation, with the MSE of a simple 2 block LSTM model achieving an MSE of 0.0021 compared to the 0.0068 of the transformer. Similar indications of the lower MAE and RMSE metrics show that the RNN models are strong in short-period financial time-series predictions and perform more accurately than the transformer. The transformer model, however, did achieve a better metric on predicting the movement direction of the stock trend compared to the LSTM and RNN prediction data (seen from Fig. 3).

Table 2. Results of 15 epochs

Metric	Transformer	AE-Transformer	LSTM	AE-LSTM	GRU
MSE	0.0068	0.0046	0.0023	0.0021	0.0024
MAE	0.0690	0.0477	0.0370	0.0362	0.0364
RMSE	0.2625	0.2049	0.1279	0.1083	0.1344
Direction	0.1277	0.0957	0.0851	0.0957	0.1063

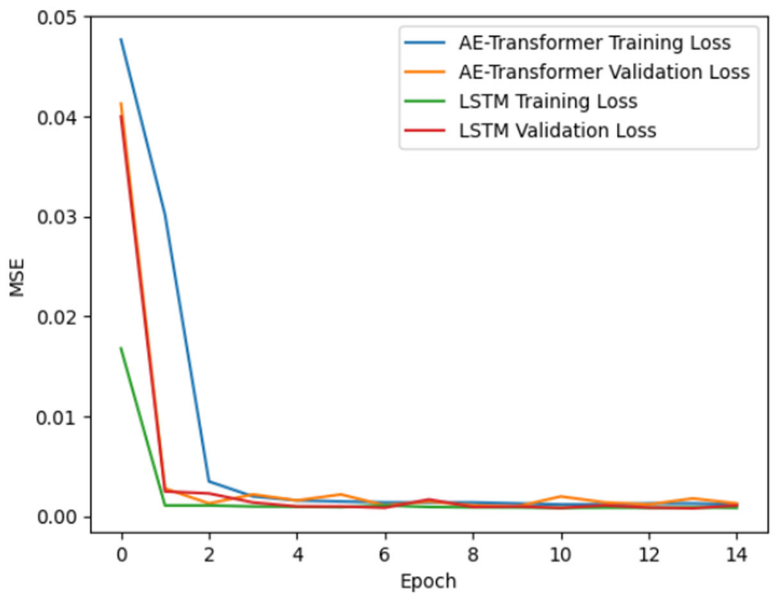
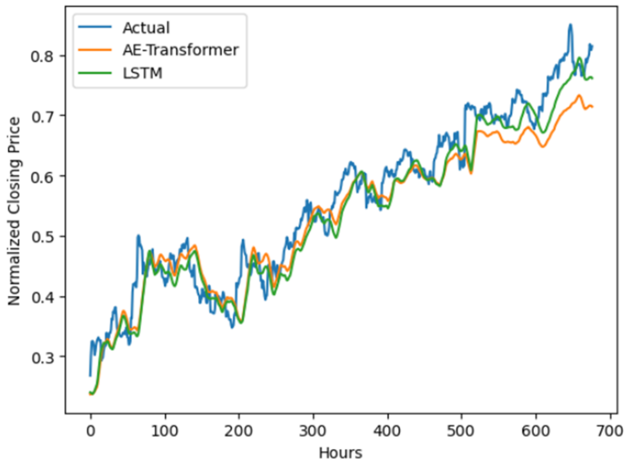
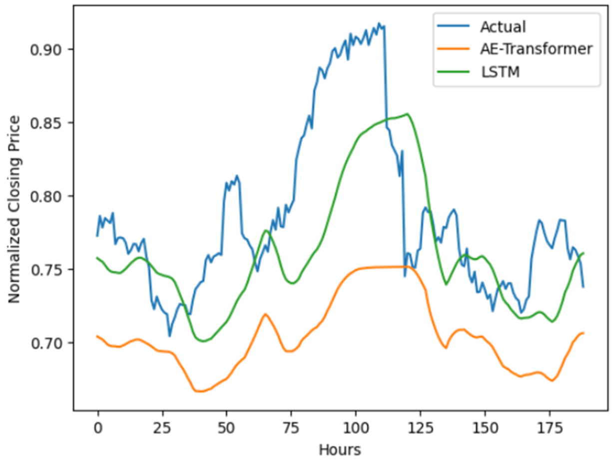


Figure 3. Training and validation losses (Photo/Picture credit: Original).

The training and validation losses over the epochs are given on the right. The models seem to quickly converge after the first few epochs and eventually have more validation loss than training loss. This is a signal of overfitting for both models. The second note is that the MSE is low magnitude due to the normalization of stock prices, leading to the square terms being much smaller than if the data were unnormalized. The quick convergence likely showcases that the learning rate that was used was high, leading us to obtain an incorrect representation early in training. The addition of the autoencoder significantly improved the performance of the transformer model, while less so for the LSTM. It is likely that the current hyperparameters do not represent the best hypothetical accuracy as a fully optimized model. This is due to a choice to use 4 encoder and 4 decoder block stacks in the model architecture. While using the 8 head multi-attention led to good results, using 4 block stacks was too complex and led to loss of interpretability, as well as overfitting. As in the prediction results presented in Fig. 4, it is evident that the validation data was predicted much more accurately compared to the test data.



(a)



(b)

Figure 4. (a) Validation Set Predictions for Closing Price. (b) Test Set Predictions for Closing Price (Photo/Picture credit: Original).

In particular, both the AE-transformer and LSTM models on the test data set are slow to adjust to rapid shifts in stock prices, as they are unable to predict sudden spikes in normalized prices. This is due to the models being unable to account for data that drive sudden spikes in stock prices, namely consumer sentiment, as well as other factors that contribute to the volatility of the prices. The results themselves, nonetheless, demonstrate the transformer's ability to accurately forecast stock series to the same degree of magnitude as the preferred state of the art LSTM model, and does even better at capturing movement signals for stock prices than that of the LSTM. The transformer is more accurately able to catch signals of short period movement trends. The use of an autoencoder is also shown to improve performance for transformer-based models. A major drawback not previously discussed is the long and more arduous training times of large models such as the transformer. However, given enough data and computing power, the transformer is much more parallelizable [14] than traditional RNN models. Although this study was unable to capitalize on this unique advantage of the transformer, it is shown that it is possible for transformer-based models to compete with RNN based models.

4. limitations and prospects

There are various limitations to this paper, the most significant being the lack of access to relevant computational power for large scale deep learning based models within the writing's time restrictions. Having access to more computational power would enable adding additional complexity to the model, adding robustness. There are also possibilities of examining longer intervals of stock prices as data points, such as daily or weekly stock price data points as opposed to the hourly data points accessed now. The original concern of using larger stock intervals as data points is twofold, firstly the potential lack of data for a large-scale complex transformer model to train accurate predictions. Secondly, utilizing longer stock price intervals and still maintaining the same number of training data points results in using stock prices over a decade or two, for which the older data is less relevant to current data due to the overbearing effect of inflation and globalization, which is difficult to account for. As an alternative, however, it is nonetheless possible to add complexity to the model by lengthening the predictive window itself, over the hourly data of the previous month rather than that of 2 trading weeks.

The autoencoder model itself could also have added complexity of a full-scale variational deep learning autoencoder, however an overly complex autoencoder could lead to more overfitting, as well as loss of financial significance of the model obtained. Performing complex feature extraction on financial temporal data could lead to representations that lose interpretability. Nonetheless, a deeper autoencoder model could facilitate the entire prediction sequence of the transformer being done in latent space, rather than simply using low dimensional reconstructed data. This might expedite training times significantly for large transformer models.

Another point to explore is the impact of various more complex attention mechanisms tuned for financial time-series prediction. One such model is the Autoformer [14], which uses the auto-correlation mechanic to replace self-attention to capture the inherent periodicity of time series data. Stock price data does not conform to the same type of periodicity as other popular time series forecast tasks; however, a similar line of thinking can be used to adapt a mechanism which capitalizes more on index metrics.

Lastly, additional financial data from other categories is likely to critically improve the model. For example, sentiment analysis was not incorporated in this work, as it is primarily focused on time-series forecasting of financial data. Financial news provides consumers incentive to drive the market in certain directions, so incorporating sentiment analysis classified via deep learning models as an additional metric, similar to the current role of volume, may increase the accuracy of the transformer model significantly. Other financial data, such as aggregate industry trends, may also aid as additional factors and estimators [15].

5. Conclusion

To sum up, the applications of transformer-based models in sequence modeling is growing, regardless if for classification or regression-based tasks. The model and mechanisms are flexible and can easily be fit to a task like stock prediction. Although stock prediction is a challenging task, especially due to the stock market's inherent volatility, it is demonstrated that transformers can be a viable alternative to RNN based models specified for this task. The advantage RNN based models have will lessen with the incorporation of larger data sets where the parallelizability of transformers will shine. This gap is also lessened significantly by feature extraction techniques such as the usage of lower dimensional representation for the data via an autoencoder. It is likely that further developments for transformer-based models, such as in areas like natural language processing or computer vision will permeate to improve their accuracy for time-series forecasting tasks such as stock prediction.

References

[1] B. G. Malkiel, Journal of economic perspectives 17 (1), 59 - 82 (2003).

[2] A. Khalid, International Journal of Business, Humanities and Technology 3 (3), 32 - 44 (2013).

[3] H. Yang, L. Chanand I. King, "Support vector machine regression for volatile stock market prediction," in Lecture Notes in Computer Science 2412, Intelligent Data Engineering and Automated Learning Conference (2002).

[4] T. S. Chang, Expert Systems with Applications 38(12), 14846 - 14851 (2011).

[5] G. P. Zhang, Neurocomputing 50, 159 – 175 (2003).

[6] H. Park, Econ 930 (Department of Economics, Kansas State University, KS, 1999) pp. 1 - 241.

[7] J. L. Ticknor, Expert Systems with Applications 40 (14), 5501 - 5506 (2013).

[8] E. Hoseinzade and S. Haratizadeh, Expert Systems with Applications 129, 273 - 285 (2019).

[9] A. M. Rather, A. Agarwal and V. N. Sastry, Expert Systems with Applications 42 (6), 3234 - 3241 (2015).

[10] D. M. Q. Nelson, A. C. M. Pereira and R. A. de Oliveira, "Stock market's price movement prediction with LSTM neural networks," in 2017 International Joint Conference on Neural Networks (IJCNN) (Anchorage, AK, USA, 2017), pp. 1419 - 1426.

[11] S. Lv, Y. Hou and H. Zhou, arXiv: 1912. 00712 (2019).

[12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser and I. Polosukhin, arXiv: 1706. 03762 (2017)

[13] G. Ding and L. Qin, International Journal of Machine Learning and Cybernetics 11 (6), 1307 - 1317 (2020).

[14] Y. Qin, D. Song, H. Chen, W. Cheng, G. Jiang and G. W. Cottrell, arXiv: 1704. 02971 (2017).

[15] H. Wu, J. Xu, J. Wang and M. Long, arXiv: 2106. 13008 (2021).