# Stock Market Trend Prediction using Deep Neural Network via Chart Analysis: A Practical Method or a Myth?

**Erfan Radfar**

Department of Mechanical Engineering, Sharif University of Technology, Tehran;
erfan.radfar@mech.sharif.edu

**Abstract**
In this paper we investigate the possibility of technical analysis for trade market via chart analysis using deep learning. We describe dynamics of stock market and prominent classical methods that are used to forecast prices and trends of market. Subsequently, we verify whether the previous works in this field are utilizable in real stock market and how much of prediction power we can obtain from the models proposed in the studies mentioned. Besides, we point out some errors in these studies and describe why they may not lead to favorable results or why those results can be misleading. Additionally, some other algorithm structures used in deep learning and which can be helpful for prediction of dynamical systems are introduced. Furthermore, we examine performance of these models such as CNN, LSTM, Transformer and their combinations on real data of 12 stocks in Tehran Stock Exchange (TSE). Finally, we propose the optimal method, which can better absorb the dynamics of semi-random environments like stock market and therefore help us to have more sophisticated prediction and grasp on chart analysis of chosen stock.

**Keywords:** trend prediction, deep learning, stock market, transformer, chart analysis.

## Introduction
With rapid growth in usage of neural network-based algorithms in machine learning in addition to the race for developing the best large language models such as GPT and Llama, there comes the question of how much inference these models can have about human's intention whether as an individual entity or a collective decision-making machine. Interestingly, there have been promising results using deep learning in prediction and simulation of disturbance-filled dynamical systems such as fluids in turbulence [1][2], navigation [3][4], intelligent control & interaction [5][6]. Furthermore, it is realized that deep learning is able to extract dynamics and relationships between parameters from data which are either too complicated for human, or not efficiently noiseless thus it can be exploited using classical methods.

The stock market indices are determined based on their market effect and subsequent capitalization. Accurate forecasting of the stock market is therefore a complex task due to changes in the international and national market. Consequently, data analysists, chart analysists and scholars have tried to utilize deep neural networks for price prediction in a manner that it can reach a considerable profit in short terms. In [7] used Multilayer Perceptron (MLP), Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM) and Convolutional Neural Network (CNN) for predicting the stock price of a company based on the historical prices available. Moreover, due to the dynamic behavior of stock market, [8] used NARX algorithm to determine price of the day after. In [9] an embedding layer is used and multiple stocks history are fed as the input of embedded LSTM.

All of these works mentioned, seems to follow stock trends better than any stock broker's prediction with more than %90 accuracy for a range of several months, yet we do not see such methods to be used extensively and be able to replace classical method such as ARIMA [10][11]. The reason comes from the fact that a wrong network's structure is common between scholars and the results are almost always misleading; yet they are published by prestigious journals like Nature [12]. In this paper we demonstrate why day-to-day price prediction cannot elaborately be used to train neural networks. Meanwhile, we test such models and compare them to a proposed more realistic and human analyst-like models. Our model does not predict price rather tries to forecast upcoming trend of the market which is more feasible. Considering the long-term pattern of each stock and relative independence of each period, we used a 100 days period as input of our model instead of day-to-day input. The reason behind this decision is extensively explained in this paper. Furthermore, we exploit advantage of convolutional neural network in recognizing relative pattern of in the historic data. In addition, we can have control on how much sensitivity is demanded from the model and tune the network parameters based on how much random disturbances are expected from stock market.

In this paper, results are trained and tested on 12 stocks in Tehran Stock Exchange (TSE) which are available in Appendix 1.

## Fundamental Analysis vs Technical Analysis
There are two major approaches for investing in stock market. Firstly, fundamental analysis is concerned with examining a company's financial statements and broader economic indicators to uncover a security's intrinsic value. The result of such an analysis should provide the investment's true worth based on a company's financial health, the market, growth prospect and economic conditions. Investors perform fundamental analysis to gauge whether or not to invest in a company based on its current and projected worth [13][14]. This approach to the market often allows brokers to see behind investor preference and firm's marketing to determine whether

the company has the potential for long-term success. With fundamental analysis, it can then be gauged if the security's market price is over- or undervalued. In case of undervalued prices, investor can expect rise in prices whether it happens in the following days or even the upcoming years [15]. Consequently, this approach is in favor of long-time investment. Besides, this method considers long term geopolitical effects that can occur within the range of markets. Therefore, it is arguably regarded as the most sophisticated method of investment

Secondly, technical analysis is viewed as a more specific approach toward investment. In technical analysis investors search for pattern in market behavior which involves daily stock price, uptrends, downtrends, buy and sell volume, inflation and other fundamental indicators like price to earnings ratio (P/E). This perspective is rooted in "Efficient-Market Hypothesis" which implies that asset prices reflect all information available about a stock. Moreover, it can be interpreted that investors cannot consistently beat the market and achieve profits based on a risk-adjusted basis since market prices should only react to new information [16]. As an example, the window of time which is available for investors to make decisions based on earnings announcement is too short and prices quickly incorporate information from these announcements. Therefore, based on new prices, investor can have an approximate indication about the profitability of corresponding stock. If only previous prices are used for future price trend prediction, it is called "Chart Analysis"

Basically, technical analysists believe that based on stocks prices and the pattern which can be extracted from them, they could have access to the information which fundamental analysists claim to extract from news, earning reports and annual revenue. However, technical analysists would have the advantage of swift reaction to the market because their decisions could be daily. They could gain short-time profit margins while avoid short-time losses. To the authors' knowledge, there have never been any extensive study on whether technical methods can have better results than fundamental methods. Most of rules derived by technical analysists come from the advantage of hindsight, thus for different realistic scenarios numerous large seemingly-random behaviors can be observed which zeros out any gain that investors have achieved through previous technical predictions.

The question that is always brought up is that if there exist any true technical rule that at least works most of the time and any pattern in the price which can assure long term profit and if its duration windows is predictable. There are some rules of thumb about this method but there has not been any significant and scientific proof for them and if they always work. For example, shoulder pattern, trend compatibility and other repeating patterns. Finally, it can be deduced that chart analysis and more generally technical analysis use price pattern to predict future prices [17].

**Pattern Recognition using Classical Method**

As mentioned before, in case of chart analysis, pattern recognition is the backbone of any kind of approach. One of the most popular pattern recognition technics is Auto-Regressive Integrated Moving Average (ARIMA). ARIMA models are, in theory, the most general class of models for forecasting a time series which can be manipulated to a stationary form by differencing, or perhaps in conjunction with other transformations such like logarithm or normalizing. A variable is stationary if its statistical properties are all constant over time. It does not have any trend and its mean has a constant amplitude. Additionally, it should have correlation with its own prior deviations from the average. An ARIMA model can be regarded as a dynamical filter that tries to separate the signal from the noise after fitting the suitable curve, the signal is extrapolated for the future variables. In this manner, any time series can be identified if a model large enough is designed. Besides, some seasonal (lang term) factors can be added to the model to include specific time windows effects [10].
A nonseasonal ARIMA (p, d, q) model is defined as follow:
First, let y denote the $d^{th}$ difference of Y, which means:

If d=0: $y_t = Y_t$
If d=1: $y_t = Y_t - Y_{t-1}$
If d=2: $y_t = (Y_t - Y_{t-1}) - (Y_{t-1} - Y_{t-2}) = Y_t - 2Y_{t-1} + Y_{t-2}$ (1)

In terms of y, the general forecasting equation is:

$$\hat{y}_t = \mu + \phi_1 y_{t-1} + \ldots + \phi_p y_{t-p} - \theta_1 e_{t-1} - \ldots - \theta_q e_{t-q} \quad (2)$$

Here the moving average parameters (θ's) are defined so that their signs are negative in the equation, following the convention introduced by Box and Jenkins. The error terms $e_t$ are generally assumed to be independent, identically distributed variables sampled from a normal distribution with zero mean. Based on defined equations, $\phi_t$ and $\theta_t$ are calculated using iterative method mostly, least square error. Moreover, other classical methods for time series such as Kalman filter is also used for denoising of data thus a smoother data is acquired for training [18].

Basis of moving average and other denoising filter like Kalman is to estimate a dynamic equation for system of time series. In ARIMA, it is determined that what degree of equation is needed and how much corresponding system would be complex by choosing 'd'. The larger 'd' is chosen, more subtle changes between prices are taken into account. In this situation, random noises would have great adverse effect on trend prediction. Besides, most ARIMA models takes previous 10 days or less as input data [11]; this short time would not be enough to capture more complicated dynamics of stock market which can last for period of at least one fiscal quarter (3 month). Therefore, the need for a long-range dynamical model can be felt.

**LSTM for Stock Prediction (Misleading Flaw)**

Neural networks have a reputation for identifying extremely non-linear systems; systems like text-to-speech, text-to-picture, image-to-video and image-to-dynamic states [19]. One of the most promising and practical structures for identification and forecasting of dynamical systems and time series is Long Short-Term Memory (LSTM). Each neuron (cell) in this system follows algorithm in figure 1:

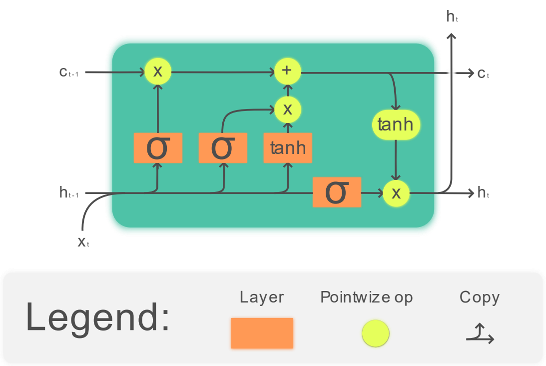


Figure 1. LSTM cell structure [20]

A Long Short-Term Memory (LSTM) cell is a specialized type of recurrent neural network (RNN) architecture, designed to address the limitations of traditional RNNs, particularly the challenge of learning long-term dependencies due to issues like vanishing and exploding gradients. The LSTM architecture features a unique cell structure that includes mechanisms to regulate the flow of information through the network.

**Structure and Components**

An LSTM cell is composed of several key components: the cell state, input gate, forget gate, output gate, and a set of activation functions. These components work together to control the transmission and transformation of information within the cell, enabling it to retain or forget information over extended sequences.

1. **Cell State ($C_t$):** This is the memory of the cell, which carries information across time steps. The cell state is maintained and updated through linear interactions, which helps preserve the gradient flow, thus alleviating the vanishing gradient problem.
2. **Input Gate ($i_t$ )**: This gate controls the extent to which new information from the current input and previous hidden state should be added to the cell state. It is typically calculated using a sigmoid activation function, determining the degree to which each component of the input vector should influence the cell state.
3. **Forget Gate ($f_t$)**: The forget gate determines how much of the existing information in the cell state should be retained or forgotten. Like the input gate, it uses a sigmoid function to produce an output between 0 and 1, where 0 represents complete forgetting and 1 represents complete retention.
4. **Output Gate ($o_t$)**: This gate decides the output of the LSTM cell for the current time step. The output gate is responsible for filtering the cell state through a non-linear transformation, usually a *tanh* activation function, to produce the new hidden state ($h_t$).
5. **Internal State Update**: The update to the cell state combines the input gate and the candidate cell state ($C_t$) which is an intermediate representation generated by applying a tanh activation to the input and previous hidden state—with the forget gate, facilitating the integration of new and existing information.

The architecture of LSTM cells allows them to effectively manage the balance between retaining information over long periods and updating with new information, making them highly effective for tasks involving sequential data, such as language modeling, time-series prediction, and speech recognition. By leveraging the forget and input gates, LSTM cells can selectively remember or forget information, making them robust against issues of long-term dependency and gradient degradation that afflict standard RNNs [21].

In spite fascinating advantages of LSTM, its abuse will lead to misleading results. In most of journals related to stock forecasting using AI structure [8][9][12][22][23], price of each day or a constant length period is given as the input to the LSTM and output is predicted price of following days. Such structure can seemingly achieve accuracy of up to %97 which is calculated with the equation 3. In this equation γ is a discount factor (in this study is set to 0.99) to alleviate uncertainties of prices as days go by.

$$\text{Accuracy} = \sum_{i=1}^{n}\left(1-\frac{\left|y_{\text{prediction}_i}\cdot\gamma^{i}-y_{\text{real}_i}\cdot\gamma^{i}\right|}{y_{\text{real}_i}\cdot\gamma^{i}}\right) \tag{3}$$

In equation 3, 'y' is the price and 'n' is number of samples. This rate of accuracy for a long term would be considered superhuman if it was done by a real trader but the misleading flaw of these kind of methods comes from the fact that corresponding models are only predicting tomorrow prices. In all of stock markets used in these studies [8][9][12][22][23], a barrier is set for daily price change of each stock which can be between 5% and 20%. Moreover, the variance of price change for each day is mostly under this barrier (between 2% and 5%). Consequently, if we pass today's price as tomorrow's prediction, we would end up with accuracy of 95% to 98%. Therefore, such high value as accuracy baseline would indicate that these models and more exactly their loss function, are not distributed correctly at all, thus they cannot find the patterns which could provide better prediction than the constant value of today.

Interestingly, if these models' predictive diagrams are examined, it is realized that forecasted values are the same as real value but lag one day behind. To prove this hypothesis, performance of a multilayer stacked LSTM model, similar to the one used in [1] is demonstrated here in figure 3. The reason for using stacked LSTM comparing to multilayer LSTM is that stacked LSTM have more connections and more complicated calculation can occur and be identified between each sample of sequence. This model has shown great capability in text inference and language models which is one the most complicated data sequences [24]. However, as mentioned before, day-to-day prediction is not suitable for a practical analysis.

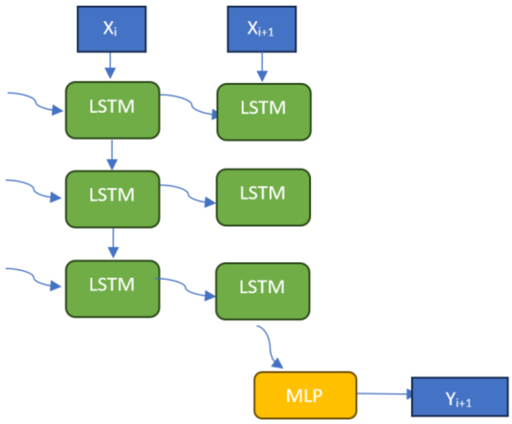


Figure 2. Stacked LSTM structure

In figure 3, this LSTM model's result is demonstrated. Blue line shows real closing price of stock number 2 during a 130 days period. Grean line shows predicted stock if we update our LSTM everyday by new data. This line is similar to the diagrams presented in [8][9][12][22][23]. It may not be obvious but if we choose smaller time window for any stock, the lagging effect will be more apparent as figure 4.

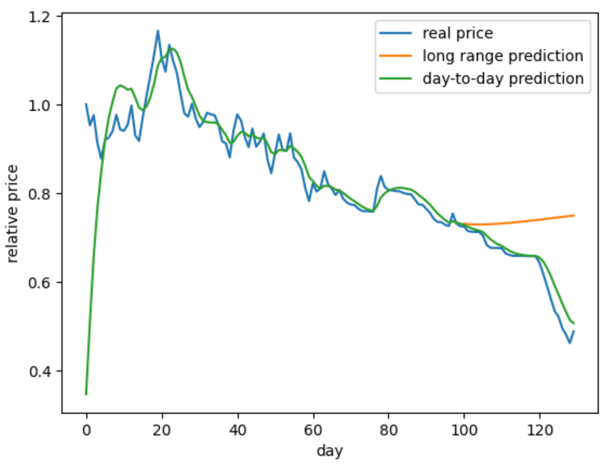


Figure 3. real and predicted sample from stock number 2 chart

The cause behind this lagging effect can be reduced to 3 main reasons.

1. Recurrent neural networks (RNN) suffer from a forgetting phenomenon. Outputs of models are more prone to remember latest sequence samples' information despite the fact that LSTM mitigates this effect to some degrees.

2. Neural models tend to converge to the most stable position in space of parameters. Therefore, if a model with structure of day-to-day information feeding is designed, the most probable position is the point where tomorrow's closing price is the same as today closing price. Such model can provide accuracy of around 95% while a real predicted can enhance this amount by few percents. Consequently, it is difficult and ambiguous to determine if the model is trained enough or not, let alone most studies conducted in this field are wrongly satisfied with such results.

3. As implied before, a well-designed neural network tends to converge to the most stable state. Hence, if there is no pattern in the data and data movement is random or information is not enough to determine system's degrees of freedom, then data's mean is the best estimation for upcoming sequence. Basically, in the most pessimistic way, stock market prices are some random walks. If these random walks' probability of distribution is demanded, the need for information out of stock charts are inevitable. There are some criticisms against theory of stock market random walk stating that, this theory oversimplifies the financial markets' complexity, ignoring the impact of market individuals' actions and their behaviors toward prices' movements, their side effects and their outcomes [25].

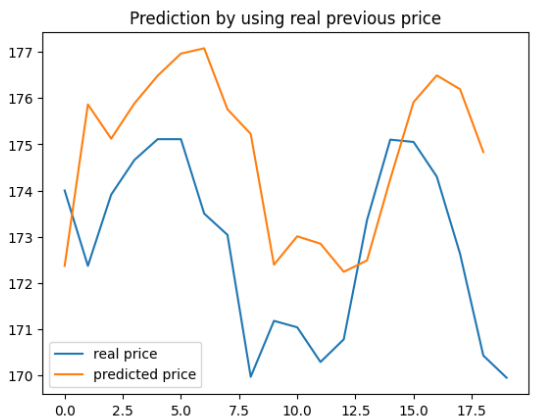


Figure 4. Apple stock prediction using one layer LSTM (x axis indicates number of days.)

**Methodology**

Taking into account all the flaws that have been observed in LSTM methods, we propose a structure which can resolve shortcomings of day-to-day LSTM. Firstly, to resolve forgetting phenomenon, a 'transformer' [26] based model is used. Transformer creates a matrix which include all previous data in sequence which can be used to determine correlated value of each data in sequence, comparing them to each other and therefore it does not suffer from forgetting. The cornerstone and main advantage of transformer despite its memory size ($n^2$) comparing to LSTM (n.log(n)) is its ability for parallel computation led to the boom of AI with Chat GPT [27]. However, transformer capability in accuracy has been proven for large language models (LLM) more broadly than any other problem. Although, we can show its efficiency for other problems involving time series.

*Multi-head Attention*

Multi-head attention is a crucial component of the transformer architecture, which is widely used in natural language processing (NLP) and other machine learning tasks. It enhances the model's ability to focus on different parts of the input sequence simultaneously, allowing it to capture various relationships and dependencies. Here's a detailed breakdown of multi-head attention:

*Key elements*

1. Attention Mechanism: The attention mechanism helps a model weigh the importance of different input elements when processing each element of the output. It computes a weighted sum of values, where the weights (attention scores) are determined by the similarity between a query and keys associated with the values.
2. Query, Key, and Value:
   - Query (Q): Represents the current token or word being processed.
   - Key (K): Represents all tokens or words in the sequence.
   - Value (V): Represents the values corresponding to the keys.

The attention mechanism computes a score for each key-query pair, indicating the relevance of each key to the query. The scores are then used to weight the corresponding values. In multi-head attention, the mechanism is applied multiple times in parallel, with different learned linear projections for the queries, keys, and values in each head. This allows the model to capture various types of relationships and interactions in the data. The process involves the following steps:

1. Linear Projections:
   For each head, the input data is linearly projected into different subspaces for queries, keys, and values. This means that separate sets of weights are learned for transforming the inputs into Q, K, and V for each head.

2. Scaled Dot-Product Attention:
   For each head, the attention scores are calculated using the dot product of queries and keys, scaled by the square root of the dimensionality of the keys. This scaling helps stabilize the gradients.
   The scores are then passed through a softmax function to obtain the attention weights, which are used to compute a weighted sum of the values.

3. Concatenation and Final Linear Layer:
   The outputs from each head are concatenated and passed through a final linear layer. This layer mixes the information from different heads and projects it back into the required output dimensionality.

*Advantages of Multihead Attention*

1. Multiple Perspectives:
   Different heads can learn to focus on different aspects of the input, such as various positional or semantic features. This is akin to having multiple "attention experts" in the model.
2. Improved Expressiveness:
   By combining information from multiple heads, the model can capture more complex patterns and dependencies in the data, which a single head might miss.
3. Parallel Computation:
   The multihead attention mechanism is well-suited for parallel computation, making it efficient to implement, especially with hardware accelerators.

*Mathematical Representation*

Let $Q$, $K$ and $V$ be the matrices representing the query, key, and value vectors, respectively. For each attention head, we have:

1. Linear projections:

$$Q_i = QW_i^Q, K_i = KW_i^K, V_i = VW_i^V \tag{4}$$

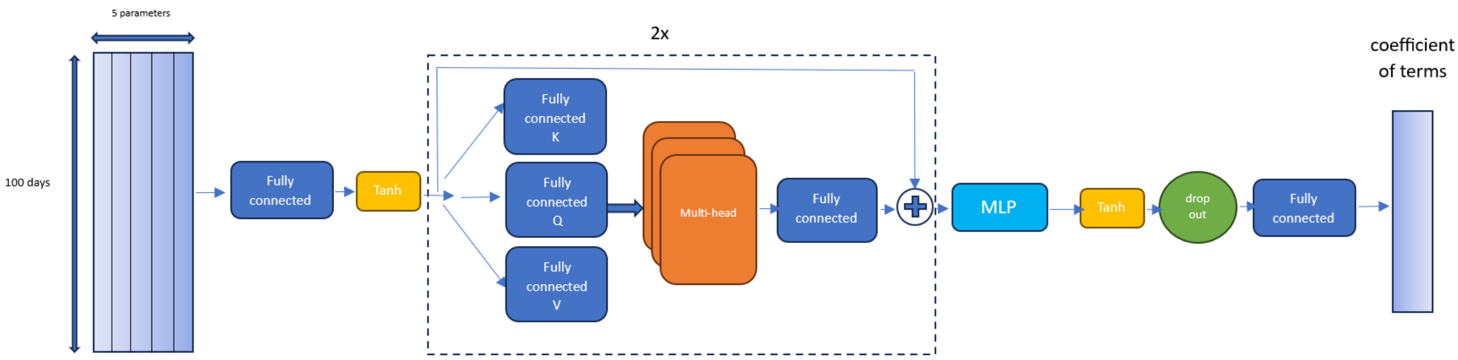


Figure 5. Proposed transformer block diagram

2. Scaled dot-product attention:

$$Attention(Q_i, K_i, V_i) \;=\; \text{softmax}\left(\frac{Q_i K_i^T}{\sqrt{d_k}}\right) V_i \qquad (5)$$

3. Concatenation and final projection:

$$\text{Multihead}(Q, K, V) \;=\; \text{Concat}(\text{head}_1, \dots, \text{head}_h) W^O \quad (6)$$

where $W_i^Q, W_i^K, W_i^V, W^O$ are learned projection matrices, $d_k$ is the dimensionality of the keys, and $h$ is the number of heads.

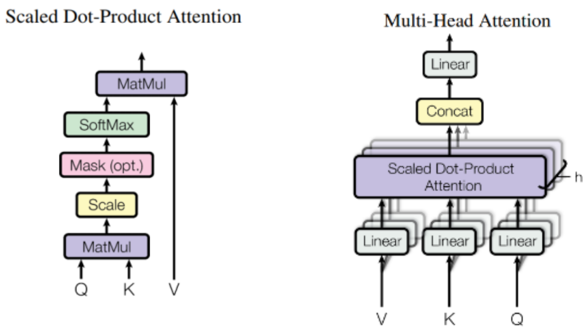


Figure 6. multi-head attention diagram

Block diagram of multi-head attention mathematical algorithm can be viewed in figure 6. Nevertheless, as stated before, stock market data is much noisier and can be interpreted in numerous ways. Additionally, it can be disturbed with many factors outside the price charts. Therefore, best prediction possible would have many uncertainties that cannot be forecasted. Consequently, we propose that an extrapolation for price series to be predicted rather than the price itself. In this way, we can have control on precision demanded from the network. The extrapolation terms should increase in value as days proceed doe to price deviation. At the same time, it's better to alleviate their effect as time passes a certain point thus, effects of some terms will vanish at the end of prediction period. Therefore, we use a linear dot of Dirac delta approximation.

$$\text{term}_i \;=\; C_i.\text{x}.\,e^{-(x-b_i)^2}$$

$$b_i \;=\; \frac{i}{number\ of\ extrapolation\ \ terms}$$

$$x \;=\; \frac{k}{total\ number\ of\ predicted\ days}$$

$$\text{Price of } k^{th} \text{ day} \;=\; \textstyle\sum_{i=0}^{4} term(k) \qquad (7)$$

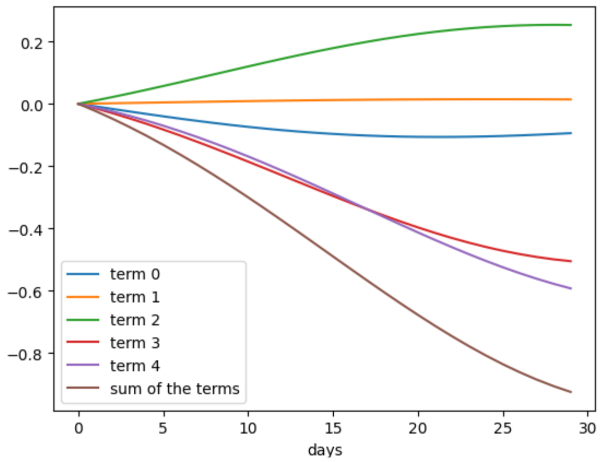


Figure 7. Random extrapolation terms for output of the model

In equation 7, $C_i$ is output of the network and a sample of 30 days extrapolation can be viewed in figure 7.

*Model Structure*

This model takes five parameters of each day as the input which include closing price, number of shares traded, volume of trading, highest and lowest price of the day. Other parameters can be derived from these five and after that, through some fully connected layer, data is passed to a double layer residual multihead attention as demonstrated in figure 5. Residual networks, use feedforward to prevent neurons from experiencing exploding or vanishing gradient during learning process. The fully connected layers in the residual block (dashed box) are recommended to have Leaky ReLU activation functions and dropouts with probability of 80%. Moreover, size of each layer is 320 and after residual block, a multi-layered perceptron (MLP) with four layers of size 128 and batch normalization between each layer, receive outputs and pass them through *Tanh* and a dropout layer. Finally, a fully connected layer without any activation function, passes out coefficients of extrapolation terms ($C_i$). At last, based on equation 7, 30 days prediction are calculated and using Mean Square Error (MSE Loss), network is optimized. Optimizer used in this study is Adam [28] with initial learning rate of 0.001 and batch size chosen is 64.

*Convolutional Network*
Additionally, capability of CNN in pattern recognition has been proven in numerous image classification models such as ResNet-50. Considering that chart analysists look for pattern in historic data, it can be a useful tool for trend forecast. Proposed CNN architecture can be viewed in figure 8.

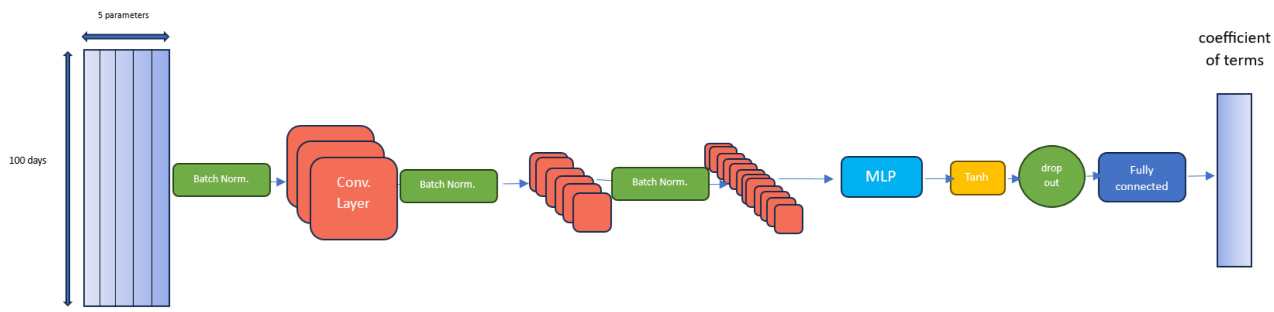


Figure 8. Proposed CNN block diagram

*Dataset*
Dataset chosen in this study are from TSE via PyTSE library in python and trained with PyTorch which enables parallel optimization using GPU. For training, the old 70% of data are separated and the recent 30% are used for test and validation. This way we can be sure that network does not interpolate and overfit to training data. Moreover, we can be sure that model works for any economical dynamics and it is independent of seasonal factors.

**Results and discussions**
In this study, to compare our method with day-to-day LSTM, an optimized model with same number of parameters of proposed method using stacked LSTM is also trained on TSE data. Additionally, considering the correction of day-to-day analysis, a CNN network with extrapolation output and a vanilla MLP with exact day prediction output, both with comparable number of parameters to transformer model, are also trained and their performance on test data is demonstrated[1] in table 1. Besides, we need to check whether chart data have any predictive information or not. Hence, we measure the accuracy of a model which only gives 100$^{th}$ day price as prediction of prices of days 101$^{st}$ till 130$^{th}$; this model is called Const. price. Models' accuracies are calculated using equation 3.

Table 1. Accuracy of models based on equation 3 (%)

| | LSTM | MLP | Proposed CNN | Proposed Transformer | Const. price |
|---|---|---|---|---|---|
| Stock 1 | 32.68 | 54.36 | **54.41** | 53.49 | 54.09 |
| Stock 2 | 88.54 | 95.60 | **96.01** | 95.88 | 95.70 |
| Stock 3 | 92.52 | **94.24** | 94.22 | 94.16 | 94.07 |
| Stock 4 | 85.55 | 91.44 | **92.15** | 91.39 | 91.83 |
| Stock 5 | 23.86 | 83.52 | 82.63 | 79.82 | **84.06** |
| Stock 6 | 90.13 | 91.24 | **91.88** | 91.64 | 91.44 |
| Stock 7 | 93.32 | 95.87 | 95.90 | **96.00** | 95.98 |
| Stock 8 | 83.27 | **93.06** | 92.93 | 91.70 | 92.72 |
| Stock 9 | 76.45 | **87.33** | 86.61 | 86.90 | 86.83 |
| Stock 10 | 66.34 | 87.15 | 87.22 | **87.23** | 87.06 |
| Stock 11 | 78.00 | 91.23 | **93.03** | 89.80 | 92.37 |
| Stock 12 | 79.95 | 88.70 | **89.76** | 89.41 | 89.37 |
| All Stocks | 66.90 | 83.40 | 85.21 | 83.71 | **85.25** |

[1] For each model, training was initiated with five different random seeds and average of top three results are demonstrated in table 1.

Table 2. Convergence time of models

| | LSTM | MLP | Proposed CNN | Proposed Transformer |
|---|---|---|---|---|
| Convergence time for training with all stocks | 40.3 s | 6.5 s | **4.6** s | 40.9 s |

As it can be seen, LSTM model of day-to-day prediction has large difference compared to constant output. This indicates that LSTM has a performance below the most naïve method of constant price. Therefore, models as like as one depicted in figure 3, have no predictive power at all. Meanwhile, models proposed in this paper, can barely outperform constant price model. The reasons we can suggest can be summarized into two main one:

- Stock market and chart analysis based on previous price is infeasible due to local randomness. Not all information available about future price of stocks can manifest itself in previous prices and other trading indictors. It is true that the value of a company is depicted in its stock price to some degree, but information available in prices are more hindsight than predictors factors. Most patterns introduced by chart analysists have such low frequency in being correct that probability of them happening as it is claimed, is basically random.
- Stock market is one of the noisiest datasets available for AI training. If we desire to train a network based on them, we need extremely larger data comparing to other environments; probably more than 1000 stock price for 10 years which demands great computing power. Additionally, in case having this much of information, it is recommended to include other financial indicators that belong to the fundamental analysis.

Nevertheless, our proposed CNN can have better performance than constant price method. This is due to the generalizability of convolutional network. This network found the average performance of each stock hence, it could give better prediction than constant price. It can be observed in figure 9 that output of network is an almost specific linear curve which does not depend on previous 100 days but depends on total performance of stock during the interval chosen as training dataset. Besides, based on table 2, we can observe efficiency and high training speed of proposed CNN.

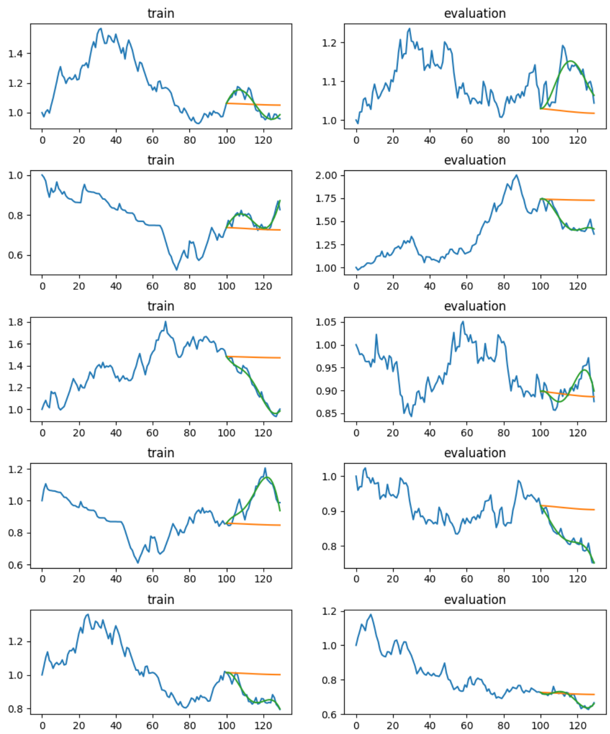


Figure 9. Proposed CNN model for stock number 2. [2]

**Conclusion**

In this paper, we investigated capability of medium-size parameter neural networks and their capability for learning trend of stock market and its ability to forecast prices. We demonstrated why previous works on this problem using LSTM have been misleading and impractical for trade environment. Meanwhile, we proposed two optimal methods based on transformer and CNN structure which could outperform day-to-day LSTM models. However, these models learned to pass outputs that are independent of previous 100 days yet learning average performance of each stock and barely outperforming constant price model. In brief, we deduced that historic prices of a stock and more generally chart data are not enough to have recognizable performance for trend prediction unless we involve majority of firms' stock active in the market.

Additionally, patterns claimed by chart analysists are not enough to have a meaningful prediction and are more likely to happen rarely. Therefore, the only way which can be exploited for stock price prediction is with the support of fundamental analysis tools whether it is financial and political news, annual reports, companies' lifecycle of product or their financial horizon.

[2] Blue curve is real data; green curve depicts desired extrapolation which model should converge to, and orange curve shows model's prediction.

**References**

[1] B Lusch, J Nathan Kutz, S L Brunton, 2018. "Deep learning for universal linear embeddings of nonlinear dynamics". *Nature communications.*

[2] S H Rudy, J Nathan Kutz, S L Brunton, 2019. "Deep learning of dynamics and signal-noise decomposition with time-stepping constraints". *Journal of Computational Physics*.

[3] Y Djenouri, J Hatleskog, J Hjelmervik, E Bjorne, M Mobarhan, 2022. "Deep learning based decomposition for visual navigation in industrial platforms". *Applied Intelligence*.

[4] K Zhu, T Zhang, 2021. "Deep reinforcement learning based mobile robot navigation: A review". *Tsinghua Science and Technology*

[5] Jie Xu, Viktor Makoviychuk, Yashraj Narang, Fabio Ramos, Wojciech Matusik, Animesh Garg, Miles Macklin, 2022. "Accelerated Policy Learning with Parallel Differentiable Simulation", *International Conference on Learning Representations*.

[6] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, Oleg Klimov, 2017, "Proximal Policy Optimization Algorithms". *cs-LG Arxiv.*

[7] Hiransha M, Gopalakrishnan E, Vijay Krishna Menonab, Soman K.P, 2018. "NSE Stock Market Prediction Using Deep-Learning Models", *ICCIDS 2018*

[8] Phd Thesis, David Noel, 2023, "Stock Price Prediction using Dynamic Neural Networks". *Arxiv.*

[9] Xiongwen Pang, Victor Chang, Weiwei Lin, 2020, "An innovative neural network approach for stock market prediction". *The Journal of Supercomputing*.

[10] https://people.duke.edu/~rnau/411arim.htm

[11] Bijesh Dhyani, Manish Kumar, Poonam Verma, Abhisheik Jain, 2020, "Stock Market Forecasting Technique using Arima Model", *International Journal of Recent Technology and Engineering (IJRTE)*

[12] Tran Phuoc, Pham Thi Kim Anh, Phan Huy Tam & Chien V. Nguyen, 2024, "Applying machine learning algorithms to predict the stock price trend in the stock market – The case of Vietnam". *Humanities and Social Sciences Communications - Nature*

[13] https://www.investopedia.com/terms/f/fundamentalanalysis

[14] Benjamin Graham, 1949, *"Intelligent Investor"*, Chap. 1

[15] Benjamin Graham, 1949, *"Intelligent Investor"*, Chap. 2,3

[16] SEWELL, Martin, 2011. *History of the efficient market hypothesis*. Research Note RN/11/04, University College London, London.

[17] Jack D. Schwager, 2002. *"Getting Started with in Technical Analysis"*, Chap. 1,2

[18] N. Deepika & M. Nirupama Bhat, 2021, "An Efficient Stock Market Prediction Method Based on Kalman Filter". *Journal of The Institution of Engineers (India): Series B*

[19] Denis Yarats, Ilya Kostrikov, Rob Fergus, 2021, "Image Augmentation Is All You Need: Regularizing Deep Reinforcement Learning from Pixels". *ICLR 2021 Conference*

[20] Guillaume Chevalier, 2018, "LARNN: Linear Attention Recurrent Neural Network", *Arxiv*

[21] Yong Yu, Xiaosheng Si, Changhua Hu, Jianxun Zhang, 2019, "A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures", *Neural Computation.*

[22] Guangyu Ding, Liangxi Qin, 2020, "Study on the prediction of stock price based on the associated network model of LSTM". *International Journal of Machine Learning and Cybernetics.*

[23] Burak Gülmez, 2023, "Stock price prediction with optimized deep LSTM network with artificial rabbits optimization algorithm". *Expert Systems With Applications 227*

[24] Aytug Onan; Mansur Alp Toçoğlu, 2021, "A Term Weighted Neural Language Model and Stacked Bidirectional LSTM Based Framework for Sarcasm Identification". *IEEE*

[25] Tim Smith, 2023, "Random Walk Theory: Definition, How It's Used, and Example". *Investopedia.*

[26] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, Illia Polosukhin, 2017, "Attention Is All You Need", *NIPS 2017*

[27] Open AI, 2022, "Introducing ChatGPT". https://openai.com/index/chatgpt/

[28] Diederik P. Kingma, Jimmy Ba, 2014, "Adam: A Method for Stochastic Optimization". *3rd International Conference for Learning*

**Appendix**

1- Name of stock used in TSE:

1) "آباد"
2) "آبادا"
3) "آپ"
4) "اپرداز"
5) "آساس"
6) "افرا"
7) "اهرم"
8) "انرژی"
9) "بترانس"
10) "بنیرو"
11) "پترول"
12) "وبملت"