Humanities & Social Sciences Communications

ARTICLE

https://doi.org/10.1057/s41599-025-04761-8 OPEN

Stock market trend prediction using deep neural network via chart analysis: a practical method or a myth?

Erfan Radfar[1]✉

In this study, we investigate the feasibility of using deep learning for stock market prediction and technical analysis. We explore the dynamics of the stock market and prominent classical methods and deep learning-based approaches that are used to forecast prices and market trends. Subsequently, we evaluate prior research applicability for stock markets and their efficacy in real-world applications. Our analysis reveals that the most prominent studies regarding LSTMs and DNNs predictors for stock market forecasting create a false positive. Therefore, these approaches are impractical for the real market if the temporal context of predictions is overlooked. In addition, we identify specific errors in these studies and explain how they may lead to suboptimal or misleading results. Furthermore, we examine alternative deep learning architectures that may be better suited for predicting dynamical systems including CNN, LSTM, Transformer, and their combinations on real data of 12 stocks in the Tehran Stock Exchange (TSE). We propose an optimal CNN-based method, which can better capture the dynamics of semi-random environments such as the stock market, providing a more sophisticated prediction. However, our finding indicates that even with this enhanced method, the predictive aspect of vanilla DNN algorithms is minimal for an environment as noisy and chaotic as the stock market, particularly when working with small data sets. Finally, we discuss why our algorithm can avoid false positives and provide a better solution for time-series and trend prediction.

[1]Sharif University of Technology, Tehran, Iran. ✉email: erfanradfar200117@gmail.com

With rapid growth in usage of neural network-based algorithms in machine learning, alongside the ongoing race for developing the best large language models such as GPT, Llama, and DeepSeek, a critical question arises: to what extent can these models infer humans' intentions, whether as an individual entity or a collective decision-making machine? Interestingly, there have been promising results utilizing deep learning for the prediction and simulation of disturbance-filled dynamical systems such as fluids in turbulence (Lusch et al. 2018; Rudy et al. 2019), navigation (Djenouri et al. 2022; Zhu and Zhang 2021), and intelligent control and interaction (Xu et al. 2022, Schulman et al. 2017). Moreover, it has been observed that deep learning is able to effectively extract the dynamics and the relationships between parameters from data that are either too complicated for humans or not sufficiently noiseless and non-chaotic for classical methods.

The stock market indices are determined based on their market impact and subsequent capitalization. Accurate prediction of the stock market is therefore a complex task due to changes in the international and national markets. Consequently, data analysts, chart analysts, and scholars have tried to utilize deep neural networks for price prediction in a manner that it can reach a considerable profit within the desired interval. Hiransha (2018) employed a 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. Lee et al. (2019) designed a reinforcement learning-based network called DQN, which demonstrated extraordinary performance with daily scenarios excluding transaction fees, tax, and stock demand. However, their approach does not provide explicit price predictions and cannot forecast upcoming days' trends. In another study, Zhang (Junhao Zhang 2022) used policy gradient for predicting stock market prices, incorporating day-to-day updates to refine the model.

Thakkar (Thakkar and Chaudhari 2021) investigated the effectiveness and predictive power of various structures ranging from CNN to DQN and depicted that DQN achieves the highest directional accuracy. Moreover, due to the dynamic behavior of the stock market, (Noel 2023) employed the Nonlinear Autoregressive Exogenous (NARX) algorithm to predict the price of the next day. Xiongwen (Pang et al. 2020) proposed an embedding layer where multiple historical data from multiple stocks were fed as inputs into an embedded LSTM. In another study, Gulmez (2023) integrated the Artificial Rabbit Algorithm with LSTM to optimize the deep network hyperparameters, ultimately predicting the next day's stock prices. Phuoc et al. (2024) tested an LSTM structure to predict stock price trends in the Vietnam stock market, achieving the best performance for day-to-day prediction compared with other intervals.

Most of the studies mentioned appear to outperform any traditional stockbroker's prediction with more than 90% accuracy for a range of several months. However, despite the reported success, these methods are not widely adopted and used extensively, replacing classical methods such as ARIMA (Anon. n.d.; Dhyani 2020). This is in spite of ARIMA's limitations, which only allow for short-horizon predictions and low-number regression parameters. This discrepancy stems from the network's structure, which is a common issue in the literature, often producing misleading results; surprisingly, such studies are published by prestigious journals. In this paper we demonstrate why day-to-day price prediction cannot be used adequately to train neural networks. Meanwhile, we evaluate such models and compare them to a proposed alternative, which is more realistic and aligns more closely with analytical methods used by human experts. Our model does not attempt to predict price directly; rather, it focuses on predicting upcoming trends in the market, which is a more practical and feasible objective. Considering the long-term pattern of each stock and the relative independence of each period, we use a 100-day historical period as the input for our model instead of relying on day-to-day input. The reason behind this decision is extensively and thoroughly explained in this paper. Furthermore, we leverage the advantage of convolutional neural networks (CNN) in identifying relative recurring patterns within historical data. Additionally, our model allows for adjustable sensitivity, enabling us to fine-tune the network's hyperparameters based on expected random disturbances of the chosen stock market.

The results presented in this paper are based on training and testing conducted on 12 stocks from the Tehran Stock Exchange (TSE). Detailed information about these stocks, including their respective dates, is provided in Appendix 1.

Fundamental analysis vs technical analysis

There are two primary approaches for the analysis of the stock market. The first, fundamental analysis, is concerned with evaluating a company's financial statements and broader economic indicators to determine the intrinsic value of a security. The result of such an analysis aims to provide the true worth of investment based on factors such as the company's financial health, the market demands, growth prospects, and prevailing economic conditions. Investors perform fundamental analysis to decide whether to invest in a company based on its current and projected value (Anon. n.d.; Graham 1949). This approach to the market often allows the analysts to look beyond investors' preferences and the firm's marketing, foreseeing the company's potential for long-term success. With fundamental analysis, it can then be gauged if the security's market price is overvalued or undervalued. In case of undervalued prices, investors can expect a rise in the price, whether it happens in the following days or even the upcoming years (Graham 1949). Thus, this approach favors long term investment. Moreover, this method accounts for long-term geopolitical effects on markets. Therefore, it is arguably regarded as the most sophisticated method of investment.

The second, technical analysis, is considered a more specific and narrow approach toward investment. In technical analysis, investors identify patterns in market behavior, including stock prices, uptrends, downtrends, buy and sell volume, inflation, and other fundamental indicators like price-to-earnings ratio (P/E). This perspective is rooted in the "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

(SEWELL 2011). As an example, the window of time that is available for investors to make decisions based on earnings announcements is too short, and prices quickly incorporate information from these announcements. Therefore, based on new prices, investors can have an approximate indication about the profitability of the corresponding stock. If only previous prices are used for future price trend prediction, it is called "Chart Analysis."

Basically, technical analysts believe that based on stock prices and the pattern extracted from them, they can access the same information that fundamental analysts derive from news, earnings reports, and annual revenue. However, technical analysts hold the advantage of swift reaction to the market, as their decision-making process can happen on a daily basis. They could gain short-term profit margins while avoiding short-term losses.

To the authors' knowledge, no extensive study has been conducted to determine whether technical methods yield better results than fundamental methods. Many of the rules employed by technical analysts come from the advantage of hindsight. As a result, in real-world scenarios, numerous large, seemingly random behaviors can emerge, zeroing out any gain investors have achieved through previous technical predictions.

A recurring question in this domain is whether there exists any reliable technical rule that works consistently over time, or a pattern in the prices that can guarantee long-term profit during predictable windows. While some rules of thumb, such as shoulder pattern, trend compatibility, and other repeating patterns are available, there has not been many significant and scientific evidence for them. Ultimately, it can be deduced that chart analysis—and more generally, technical analysis—relies on identifying price patterns to predict future price movements (Schwager 2002).

Pattern recognition using classical method

As mentioned before, in the case of chart analysis, pattern recognition is the backbone of any kind of approach. One of the most popular pattern recognition techniques 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 as logarithm or normalizing. Additionally, it should have a correlation with its own prior deviations from the average. An ARIMA model can be regarded as a dynamical filter that attempts to separate the signal from the noise after fitting a suitable curve, extrapolating the signal for the future variables. In this manner, most time series can be identified if a model large enough is utilized. Besides, some seasonal (long-term) factors can be added to the model to include specific time-windows' effects (Anon. n.d.).

A nonseasonal ARIMA (p, d, q) model is defined as follow: First, let y denote the dth difference of Y, defined as follow:

$$\begin{aligned} \text{If d} &= 0:\ y_t = Y_t \\ \text{If d} &= 1:\ y_t = Y_t - Y_{t-1} \\ \text{If d} &= 2:\ y_t = (Y_t - Y_{t-1}) - (Y_{t-1} - Y_{t-2}) \\ &= Y_t - 2Y_{t-1} + Y_{t-2} \end{aligned} \tag{1}$$

In terms of y, the general forecasting equation is:

$$\hat{y}_t = \mu + \phi_1 y_{t-1} + \dots + \phi_p y_{t-p} - \theta_1 e_{t-1} - \dots - \theta_q e_{t-q} \tag{2}$$

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 $\varepsilon t e_t$ are generally assumed to be independent, identically distributed variables sampled from a normal distribution with zero mean. Based on defined equations, $\phi_t$andθ_t are calculated using the iterative method mostly, the least square error. Moreover, other classical methods for time series, such as the Kalman filter, are also used for data denoising to achieve smoother data for training (Deepika and Bhat 2021).

The basis of moving averages and other denoising filters like the Kalman lies in estimating a dynamic equation for the system of time series. In ARIMA, it is determined what degree of equation is needed, and the complexity of the corresponding system is determined by choosing *d*. A larger *d* accounts for more subtle changes in prices. In this situation, random noises would have a great adverse effect on trend prediction and create ungeneralizable results. Besides, most ARIMA models take the previous 10 days or less as input data (Dhyani 2020); this short time would not be enough to capture the more complicated dynamics of the stock market, which can last for a period of at least one fiscal quarter (3 months). Consequently, there is a clear need for models capable of capturing long-range dependencies and dynamics in stock market behavior.

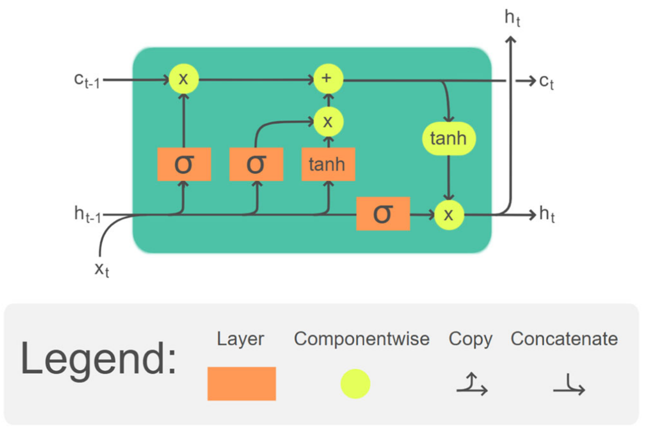

Fig. 1 LSTM cell structure (Yu et al. 2019).

LSTM for stock prediction (misleading flaw)

Neural networks are renowned for their capabilities in modeling highly non-linear systems, such as text-to-speech, text-to-picture, image-to-video, and image-to-dynamic states (Yarats 2021). Among the various architectures, one of the most promising and practical structures for identifying and predicting the states of dynamical systems is Long Short-Term Memory (LSTM). Each neuron (cell) in this system follows the algorithm in Fig. 1:

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 integrated with each other, can control the transmission and transformation of information within the cell, retaining or forgetting information over extended sequences.

1. Cell State (C_t): This is the memory of the cell, which carries most prominent information across time steps. Information which will be needed to form the context of upcoming inputs The cell state is updated through linear interactions, preserving the gradient flow and thus alleviating the vanishing gradient problem.
2. Input Gate (i_t): This gate performs as a controller, determining how new information from the current input and previous hidden state should be added to the cell state. It is generally calculated using a sigmoid activation function, determining the degree to which each component of the input vector should effect the cell state.
3. Forget Gate (f_t): This gate determines how much of the existing information in the cell state should be kept or forgotten. Like the input gate, it uses a sigmoid function to produce an output ranging from 0 to 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, integrating the new and the 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 profoundly effective for tasks involving sequential data, such as language modeling, time-series prediction, and speech recognition. By leveraging both 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 standard RNNs are susceptible to (Yu et al. 2019).

Despite the remarkable advantages of LSTM, its abuse will lead to misleading and suboptimal results. In many studies involving stock market forecasting using AI structure (Noel 2023) (Pang et al. 2020; Phuoc et al. 2024; Guangyu Ding 2020; Gülmez 2023), the price of each day or a fixed length period is provided as the input to the LSTM, with the output being the predicted price for following days. While such structure can seemingly achieve accuracy of up to 97% which is calculated using Eq. 3, these results often fail to account for critical real-world complexities.

$$Accuracy = \sum_{i=1}^{n}\left(1 - \frac{|\hat{y}_i.\gamma^i - y_i.\gamma^i|}{y_i.\gamma^i}\right) \tag{3}$$

In this equation γ is a discount factor (set to 0.99 in this study) to alleviate the effect of uncertainties in the prices over time. Here, y denotes the actual price, $\hat{y}$ represents the predicted price, and n is the number of samples. While, this rate of accuracy (up to 97%) for the long term would be considered superhuman if achieved by real trader, these results are misleading due to a critical flaw: the models are only predicting prices for the next day. In all of the stock markets used in these studies (Noel 2023; Pang et al. 2020; Phuoc et al. 2024; Guangyu Ding 2020; Gülmez 2023), a barrier is set for the 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%). As a result, if we pass today's price as tomorrow's prediction, we will end up with an accuracy of 95% to 98%. Therefore, such a high value as the accuracy baseline would indicate that these models—and, more exactly their loss function—are not properly calibrated. Consequently, they fail to find the patterns that could provide predictions better than using today's price as a constant estimate.

Interestingly, if these models' predictive diagrams are examined, it is realized that the forecasted values are the same as the real values but lag one day behind. To prove this hypothesis, the performance of a multilayer stacked LSTM model, similar to the one used in (Lusch et al. 2018) (Fig. 2) will be investigated. The reason for using stacked LSTM compared to multilayer LSTM is that the stacked LSTM has more connections, therefore deeper calculations can occur and more patterns will be identified between each sample of sequence. This model has shown great capability in text inference and language models which are one the most complicated data sequences (Onan and Toçoğlu 2021). However, as mentioned before, day-to-day prediction is not suitable for a practical analysis.

In Fig. 3, this LSTM model's result is demonstrated. The blue line shows the actual closing prices of stock number 2 during a 130 days period. The green line shows predicted stock if we update our LSTM every day by new data. This line is similar to the diagrams presented in (Noel 2023; Pang et al. 2020; Phuoc et al. 2024; Guangyu Ding 2020; Gülmez 2023). It may not be obvious, but by choosing a smaller time window for the stock, the lagging effect will be more apparent as demonstrated in Fig. 4.

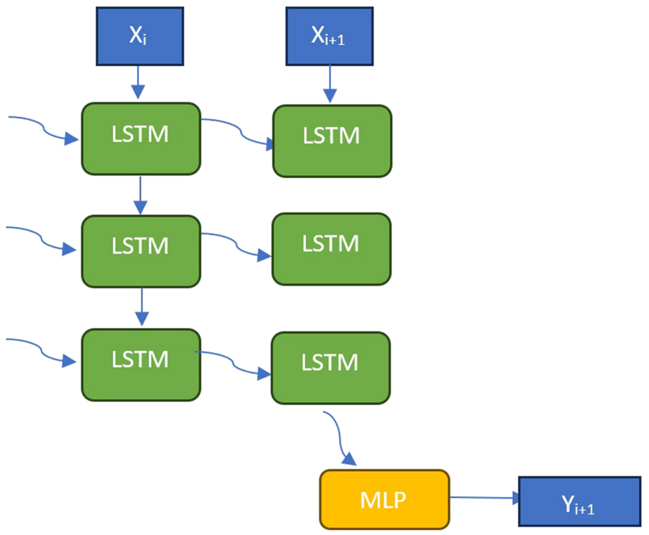

Fig. 2 Stacked LSTM structure.

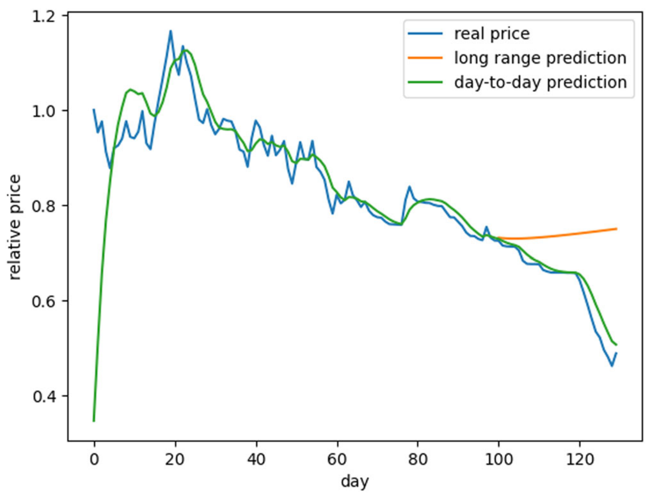

Fig. 3 Real and predicted sample from stock number 2 chart.

The lagging effect observed in LSTM models can be attributed to 3 main factors.

1. Recurrent neural networks (RNN) suffer from a forgetting phenomenon. Outputs of models are prone to memorize the latest sequence samples' information. Although LSTM networks mitigate this issue to some extent, they do not eliminate it entirely.
2. Neural models tend to converge to the most stable position in the state space. In the case of day-to-day information feeding, the most stable state often corresponds to predicting that tomorrow's closing price will be the same as today's. Such a model can provide an accuracy of around 95%, a skilled human predictor might enhance this by a few percent. Consequently, it would be challenging and ambiguous for researchers to determine if the model is trained enough or not based on this loss. Unfortunately, many studies conducted in this field are erroneously 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 the data movement is random,

information embedded in these data will not be sufficient to determine any dynamics for the system. Consequently, the data's mean would be the best estimation for upcoming sequences. Basically, in the most pessimistic way, stock market prices are some random walks. However, accurately estimating the probability distribution of these random walks requires information beyond stock charts. Critics of the random walk theory argue 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 (Smith 2023).

Methodology

Taking into account the limitations that have been observed in LSTM methods, we propose a structure that addresses the shortcomings of LSTM models. To mitigate the forgetting phenomenon, a transformer-based (Vaswani et al. 2017) model is used. Transformer utilizes a matrix that incorporate all previous data in a sequence, determining correlated values of the data. Therefore, it does not suffer from forgetting, however, all data segments should be provided as input, confining the model to a specific attention window (this study, the historical days of the stock). The cornerstone and the key advantage of the transformer despite its memory size (n^2) compared to LSTM (n.log(n)), is its ability for parallel computation. This capability has been a driving force behind the recent advancements in AI, with Chat GPT (Open-AI 2022) as an example. While transformers have demonstrated exceptional accuracy in large language models (LLMs), their potential can extend to other domains involving time series. To further enhance these model efficiency, feature-extractor models such as CNN-based models can be integrated.

The block diagram of multi head attention algorithm can be viewed in Fig. 5. However, as previously noted, stock market data is inherently much noisier and can be interpreted differently. Additionally, it can be disturbed by many factors outside the price charts. As a result, even the best possible predictions will inevitably include uncertainties that cannot be forecasted. To address this, we propose predicting an extrapolation for price series rather than the exact price. In this way, we can have control over precision demanded from the model. The extrapolation terms are designed to increase in value as days proceed, reflecting prices deviations and increasing uncertainties. Simultaneously, their influences are alleviated as time passes a certain point thus, some of the terms will vanish at the end of their prediction window. To achieve this, we use a combination of linear Dirac deltas approximation.

$$\begin{aligned} term_i &= C_i.\text{x}.e^{-(x-b_i)^2} \\ b_i &= \tfrac{i}{n_T} \\ x &= \tfrac{k}{N_p} \\ price\ of\ k^{th} day &= \sum_{i=0}^{4} term_i(k) \end{aligned} \tag{7}$$

In Eq. 7, C_i represents output of the network, n_T denotes number of extrapolation terms, N_p is the maximum number of extrapolation days (set to 30 days in this study), and a sample of these 30 days extrapolation can be viewed in Fig. 6.

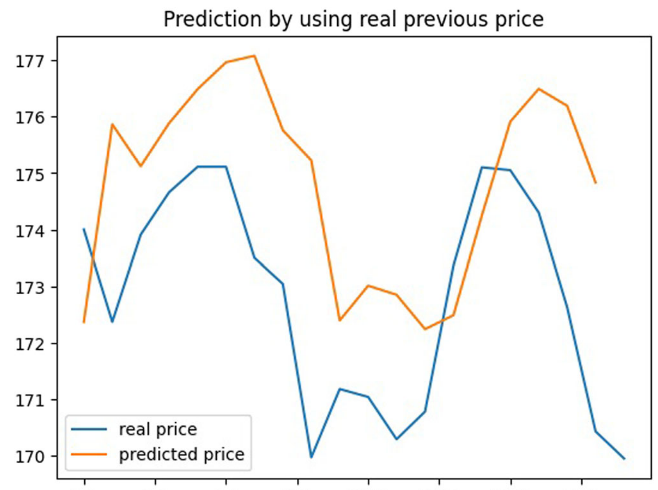

Fig. 4 Apple stock prediction using one layer LSTM (x axis indicates number of days.).

Models structure. These models take five parameters of each day as the input including closing price, number of shares traded, volume of trading, highest price and lowest price of the day. Additional parameters can be derived from these five inputs.

Transformer-based model. Inputs are first passed through some fully connected layer, to a double-layer residual multihead attention as demonstrated in Fig. 7. Residual networks (Kaiming He, 2016), incorporate feedforward to prevent neurons from experiencing exploding or vanishing gradients during the learning process. The fully connected layers in the residual block (dashed box) are recommended to have Leaky ReLU activation functions and dropouts with a probability of 80% which indicates how many of neurons are updated at each learning iteration. 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. Subsequently, a fully connected layer without any activation function outputs the coefficients of extrapolation terms (C_i). At last, using Eq. 7, the

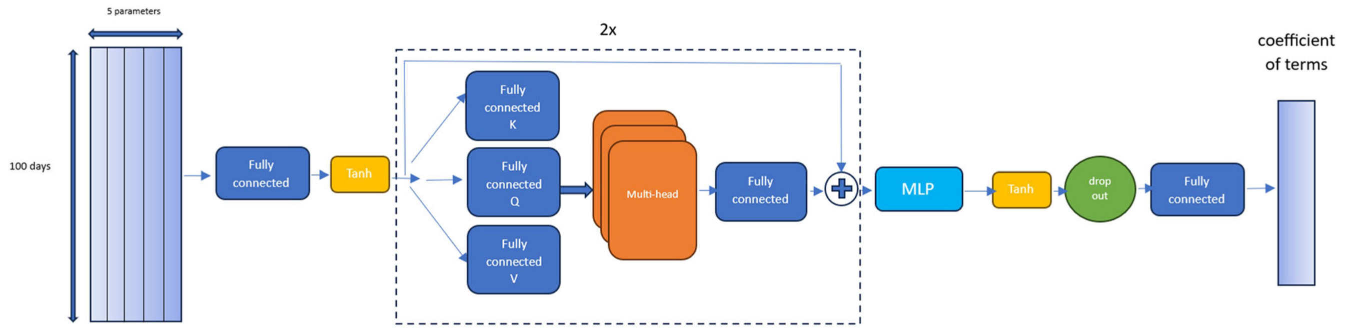

Fig. 5 Proposed transformer block diagram.

30-day predictions are calculated, and using Mean Square Error (MSE Loss), the network is optimized. The optimizer used in this study is Adam (Diederik and Kingma 2014) with an initial learning rate of 0.001 and chosen batch size of 64.

CNN-based model. The capability of CNN in pattern recognition has been demonstrated in numerous image classification models such as ResNet-50 and YOLO. Given that chart analysists rely on looking for pattern in historical data, CNN can serve as a useful tool for trend forecasting. The proposed CNN architecture is illustrated in Fig. 8. The same hyperparameters used in the transformer-based model, including optimizer settings and batch size, are also applied to the proposed CNN-based model.

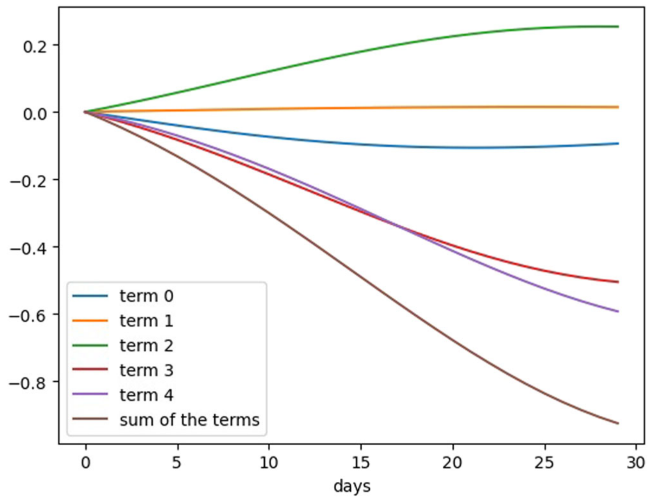

Fig. 6 Random extrapolation terms for output of the model.

Parameters tuning. For tuning parameters, including learning rate, dropout, and other hyperparameters included in Table 1, we ran each learning procedure till overfitting started (indicated by a decrease in training loss and an increase in test loss) or until the maximum training time of one hour on an RTX 3060 GPU was reached (this case was not observed in the training).

- **Dropout**: Tuned uniformly from 0 to 0.9. A value of 0.8 was selected.
- **Number of Layers**: Tuned uniformly from 2 to 8. Five layers were chosen.
- **Layer Size**: Tuned geometrically from 32 to 1024 ($32 \rightarrow 64 \rightarrow 128 \rightarrow \ldots \rightarrow 1024$). A size of 128 was selected.
- **Kernel Size**: Tuned uniformly from 2 to 7. A kernel size of 3 was chosen.
- **Learning Rate**: Tuned geometrically ($0.1 \rightarrow 0.01 \rightarrow 0.001 \rightarrow 0.0001 \rightarrow 0.00001$). A learning rate of 0.001 was selected.

It is important to note that the neural networks tend to overfit the stock market dataset due to the semi-random nature of these data. However, we observed that the chance of overfitting decreases as a larger number of stocks are used for training. Therefore, we recommend using significantly larger datasets (e.g., more than 1000 stock tickers) for each training instance to increase generalizability.

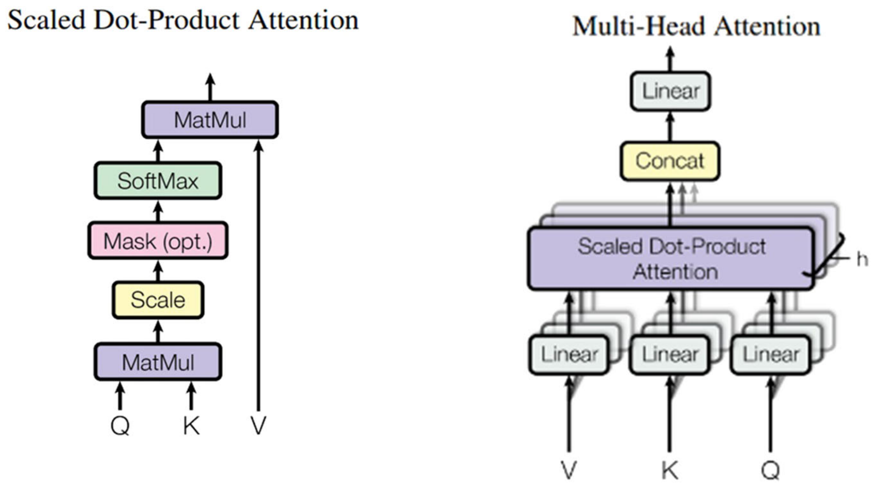

Fig. 7 Multi-head attention diagram.

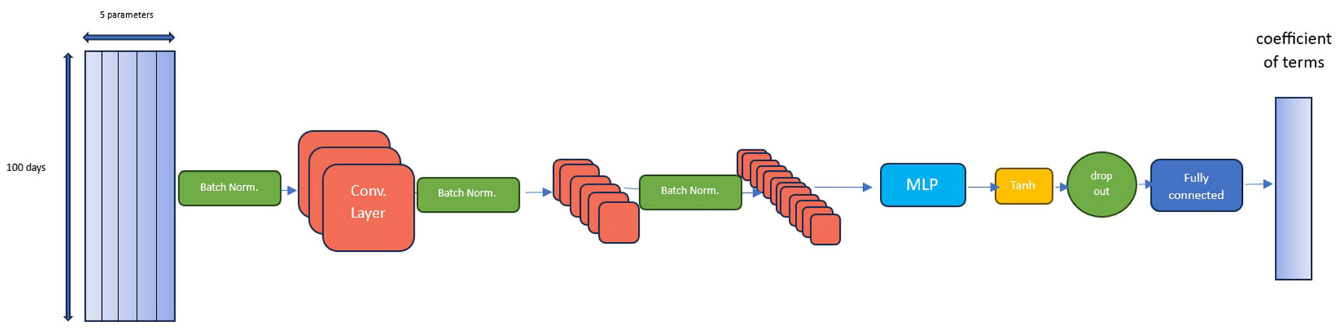

Fig. 8 Proposed CNN block diagram.

Dataset. The dataset used in this study was obtained from TSE via PyTSE library in Python. The models were trained using PyTorch, which provides parallel optimization through GPU acceleration. For training and evaluation, the dataset was split chronologically: the oldest 70% of the data was used for training, while the most recent 30% was reserved for testing and validation. Due to differences in the availability of historical data for each stock, the length of the time series differs across stocks, as detailed in Appendix 1. Importantly, the training and test datasets are completely separated, with no overlapping days or shared information. This way, it is ensured that the network does not interpolate or overfit the training data. In addition, this method guarantees that the model's performance is independent of different economic dynamics and seasonal factors.

Results and discussions

In this study, to evaluate our method, we compared its performance with a day-to-day LSTM model. Specifically, an optimized stacked LSTM model with the same number of parameters as our proposed method was trained on the Tehran Stock Exchange (TSE) data. Additionally, to account for limitations of day-to-day LSTM analysis, a vanilla MLP with the same output length and with a comparable number of parameters to the transformer-based model was also trained and tested on the stocks data. Their performances on evaluation data are demonstrated (For each model, training was initiated with five different random seeds, and an average of the top three results are demonstrated in Table 2) in Table 2. Besides, we need to determine whether chart data have any predictive information. Hence, we measure the accuracy of a model that outputs the 100th day price as a prediction for prices of days 101st to 130th; this model is called Const. Price (equivalent to a buy-and-hold strategy). Models' accuracies are calculated using Eq. 3.

Table 1 Parameters for proposed Models.

Learning rate	0.001
Optimizer	Adam
MLP (fully Connected)	(160*number of input days, 128, 256, 128, 64, number of output terms)
Activation function	Leaky ReLU (Tanh after MLP)
Dropout	0.8
CNN channel	(32, 64, 128, 128, 64, 32)
CNN Kernel	[3, 3]

As shown in Table 2, the day-to-day LSTM model prediction has a significant gap in results compared to constant output. This indicates that the LSTM model performs worse than the most naïve approach to predicting a continuous price. Consequently, models like the one depicted in Fig. 2, have no predictive power at all. In contrast, the models proposed in this paper marginally outperform the constant price model. The reasons we can suggest can be summarized into two main points:

- **Infeasibility of Chart Analysis Based on Previous Prices**: The stock market and chart exhibit local randomness, and there is no prominent information in historical data. While a company's stock price may partially reflect its intrinsic value, information embedded in prices is more retrospective than predictive. Many patterns identified by chart analysts have such low frequency that they can only provide random predictions.
- **Noise and Data Requirements**: The stock market is among the noisiest data sets available for AI training. To train a reliable network based on the stock market values, significantly larger data sets are required compared to other environments —probably encompassing more than 1000 stocks over a 10-year period—which demands great computing power and data extraction. Furthermore, if such extensive data is available, it is advisable to incorporate additional financial indicators, mainly involving fundamental analysis factors.

Despite these challenges, our proposed CNN-based model demonstrates better performance than the constant price method. This improvement is attributed to the generalizability of convolutional networks, which capture the average performance of each stock, enabling better prediction than constant price. As illustrated in Fig. 9, the output of the network is an almost specific linear curve that does not depend on the previous 100 days but reflects the total performance of stock during the interval chosen as the training dataset. Besides, Table 3 highlights the efficiency and the higher training speed for the proposed CNN-based model.

Our findings suggest that time-series prediction for the stock market environment is much more complicated than what previous works perceived and more dependent on numerous factors outside price, sales, volume, and highs and lows of the months before. While the dataset used in this work was not as extensive as the one used for training large language models, the framework proposed in this research can address the false positives created around LSTMs and Transformer efficacy for stock market prediction.

Table 2 Accuracy of models based on Eq. 3 (%).

	LSTM	MLP	Proposed CNN	Proposed Transformer	Const. price	ARIMA (2,1,2)
Stock 1	32.68	54.36	**54.41**	53.49	54.09	30.06
Stock 2	88.54	95.60	**96.01**	95.88	95.70	86.13
Stock 3	92.52	**94.24**	94.22	94.16	94.07	93.45
Stock 4	85.55	91.44	**92.15**	91.39	91.83	86.32
Stock 5	23.86	83.52	82.63	79.82	**84.06**	82.89
Stock 6	90.13	91.24	**91.88**	91.64	91.44	90.56
Stock 7	93.32	95.87	95.90	**96.00**	95.98	93.87
Stock 8	83.27	**93.06**	92.93	91.70	92.72	89.22
Stock 9	76.45	**87.33**	86.61	86.90	86.83	82.56
Stock 10	66.34	87.15	87.22	**87.23**	87.06	80.38
Stock 11	78.00	91.23	**93.03**	89.80	92.37	91.42
Stock 12	79.95	88.70	**89.76**	89.41	89.37	83.18
All Stocks	66.90	83.40	85.21	83.71	**85.25**	79.75

The bold values demonstrate the maximum performance within the presented methods.

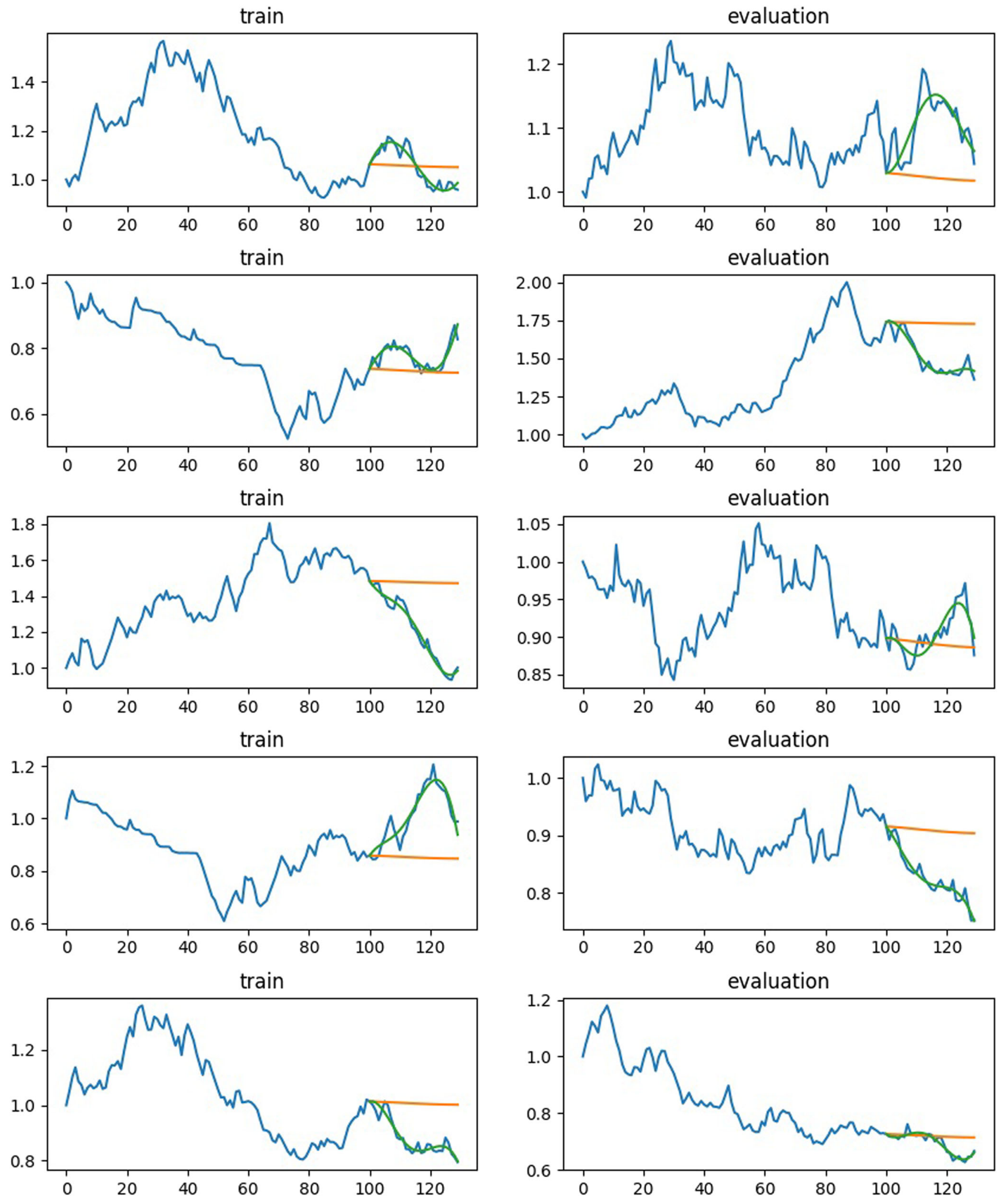

Fig. 9 Proposed CNN model results for stock number 2. Blue curve is the real data; green curve depicts desired extrapolation which model should converge to, and the orange curve shows model's prediction.

Conclusion

In this paper, we investigated the capability of medium-sized neural networks and their capability for learning the trends of the stock market and forecasting prices. We demonstrated why prior works utilizing LSTM are misleading and impractical for real-world trading environments. Meanwhile, we proposed two optimal methods based on transformer and CNN architectures which outperformed day-to-day LSTM models. However, these models learned to generate outputs that are largely independent of the previous 100 days, instead learning the average performance of each stock and marginally

Table 3 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

The bold values demonstrate the maximum performance within the presented methods.

outperforming the 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 the majority of firms' stock active in the market.

Our findings suggest that patterns claimed by chart analysts are insufficient to provide a reliable prediction and are more likely to happen randomly. Therefore, the most promising approach for stock price prediction involves integrating fundamental analysis tools, including financial and political news, annual reports, companies' product lifecycles, or their financial horizon. This kind of information can be encoded in a latent space.

Although time series and large language models operate under different dynamics (Tan et al. 2024), conventional methods of deep learning often underperform in complicated and noise-filled environments such as the stock market. These environments can be a great test set for evaluating the efficacy and the efficiency of time-series predictors. Financial markets are considered inherently chaotic and complex, posing challenges for both human experts and machine learning algorithms. The vastness, the deep correlation of financial networks, and the external disturbances add more complexity to these dynamics, making the stock market the ideal benchmark for AI models. However, the way we use these algorithms and how they produce false and true positives remains one of the most critical factors for researchers. This work enables future researchers to avoid these issues and conduct their work in a more meaningful and practical manner. Additionally, we demonstrated that a small number of the stock market tickers is insufficient for a neural network to achieve predictive. Therefore, datasets two to three orders of magnitude larger than those often used in this field are necessary for robust and capable models.

Data availability

Data used for training and testing for this paper are open sourced and available at Tehran Stock Exchange (https://tsetmc.com/). Data is provided within supplementary information files Also, the ticker data used specifically for this study is available at: https://github.com/Erfanmode/Stock-Market-Trend-Prediction-using-Neural-Network-via-Chart-Analysis/tree/main/tickers_data.

Received: 26 October 2024; Accepted: 13 March 2025;
Published online: 13 May 2025

References

Anon. n.d. https://people.duke.edu/~rnau/411arim.htm. [Online]
Anon. n.d. https://www.investopedia.com/terms/f/fundamentalanalysis. [Online]
Chevalier G (2018) LARNN: linear attention recurrent neural network. https://arxiv.org/abs/1808.05578
Deepika N, Bhat MN (2021) An efficient stock market prediction method based on kalman filter. J Inst Eng: Series B 102:629–644. https://doi.org/10.1007/s40031-021-00583-9
Dhyani B (2020) Stock market forecasting technique using arima model. Int J Recent Technol Eng
Djenouri Y, Hatleskog J, Hjelmervik J, Bjorne E, Utstumo T, Mobarhan M (2022) Deep learning based decomposition for visual navigation in industrial platforms. Appl Intell 52:8101–8117
Graham B, McGowan B (2003) The intelligent investor. Zweig J (ed.), Harper-Business Essentials New York, pp. 45–80
Guangyu D, Liangxi Q (2020) Study on the prediction of stock price based on the associated network model of LSTM. Int J Mach Learn Cybern 11:1307–1317. https://doi.org/10.1007/s13042-019-01041-1
Gülmez B (2023) Stock price prediction with optimized deep LSTM network with artificial rabbits optimization algorithm. Expert Syst Appl 227:120346
Hiransha M, Gopalakrishnan EA, Vijay KM, Soman KP (2018) NSE stock market prediction using deep-learning models. Procedia Comput Sci 132:1351–1362
He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778
Kingma DP, Ba J (2015) Adam: {A} Method for Stochastic Optimization. In: Bengio Y, LeCun Y (eds) 3rd International Conference on Learning Representations, {ICLR} 2015, San Diego, CA, USA. http://arxiv.org/abs/1412.6980
Lee J, Kim R, Koh Y, Kang J (2019) Global stock market prediction based on stock chart images using deep Q-network. IEEE Access 7:167260–167277
Lusch B, Kutz N, Brunton SL (2018) Deep learning for universal linear embeddings of nonlinear dynamics. Nat Commun 9: 4950. https://doi.org/10.1038/s41467-018-07210-0
Noel D (2023) Stock price prediction using dynamic neural networks. arXiv preprint arXiv:2306.12969
Onan A, Toçoğlu MA (2021) A term weighted neural language model and stacked bidirectional LSTM based framework for sarcasm identification. IEEE
Open-AI (2022) Introducing ChatGPT. [Online] Available at: https://openai.com/index/chatgpt/
Pang X, Chang V, Lin W (2020) An innovative neural network approach for stock market prediction. J Supercomput 76:2098–2118
Phuoc T, Anh PTK, Tam PH et al. (2024) Applying machine learning algorithms to predict the stock price trend in the stock market – The case of Vietnam Humanit Soc Sci Commun - Nature 11:393
Rudy SH, Kutz JN, Brunton SL (2019) Deep learning of dynamics and signal-noise decomposition with time-stepping constraints. J. Comput. Phys. 396:483–506
Schulman J, Wolski F, Dhariwal P, Radford A, Klimov O (2017) Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347
Schwager JD (2002) Getting Started in Technical Analysis. Wiley (ed.), John Wiley & Sons, Bahman, pp. 17–73
SEWELL M (2011) History of the efficient market hypothesis, Research Note RN/11/04. University College London, London
Smith T (2023) Random walk theory: Definition, how it's used, and example. Investopedia. https://www.investopedia.com/terms/r/randomwalktheory.asp, (Accessed on 23 Nov 2023)
Tan M, Merrill MA, Gottesman Z (2015) Deep residual learning for image recognition. Comput Vis Pattern Recognit
Tan M, Merrill MA, Gupta V, Althoff T, Hartvigsen T (2024) Are language models actually useful for time series forecasting? NeurIPS 37:60162–60191
Thakkar A, Chaudhari K (2021) A comprehensive survey on deep neural networks for stock market: the need, challenges, and future directions. Expert Syst Appl. 177:114800
Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN (2017). Attention is all you need. NIPS 2017
Xu J, Makoviychuk V, Narang Y et al. (2022) Accelerated policy learning with parallel differentiable simulation. arXiv preprint arXiv:2204.07137
Yarats D, Kostrikov I, Fergus R (2021) Image Augmentation Is All You Need: Regularizing Deep Reinforcement Learning from Pixels. In: International Conference on Learning Representations, https://openreview.net/forum?id=GY6-6sTvGaf
Yu Y, Si X, Hu C, Zhang J (2019) A review of recurrent neural networks: lstm cells and network architectures. Neural Comput. 31:1235–1270
Zhang J, Lei Y (2022) Deep reinforcement learning for stock prediction. Sci Program 2022:5812546
Zhu K, Zhang T (2021) Deep reinforcement learning based mobile robot navigation: a review. Tsinghua Sci Technol 26:674–691

Author Contributions

The data collection, theorical novel methods, and the coding of program associated with this paper, were all done by Erfan Radfar.

Competing interests

The author declares no competing interests.

Ethical approval

This study has exclusively used open-access data from companies actively traded on the Tehran Stock Exchange (TSE). As no individual participants were directly involved in the research, we confirm that ethical approval was not required, and no ethical guidelines or codes were violated during the course of this investigation.

Informed consent

This study has exclusively used open-access data from companies actively traded on the Tehran Stock Exchange (TSE). As no individual participants were directly involved in the research, we confirm that informed consent was not required.

Additional information

Supplementary information The online version contains supplementary material available at https://doi.org/10.1057/s41599-025-04761-8.

Correspondence and requests for materials should be addressed to Erfan Radfar.

Reprints and permission information is available at http://www.nature.com/reprints