Evolving Systems
https://doi.org/10.1007/s12530-022-09481-x

ORIGINAL PAPER

An efficient real-time stock prediction exploiting incremental learning and deep learning

Tinku Singh[1] · **Riya Kalra**[1] · **Suryanshi Mishra**[2] · **Satakshi**[2] · **Manish Kumar**[1]

Received: 6 September 2022 / Accepted: 13 December 2022
© The Author(s), under exclusive licence to Springer-Verlag GmbH Germany, part of Springer Nature 2022

Abstract
Intraday trading is popular among traders due to its ability to leverage price fluctuations in a short timeframe. For traders, real-time price predictions for the next few minutes can be beneficial for making strategies. Real-time prediction is challenging due to the stock market's non-stationary, complex, noisy, chaotic, dynamic, volatile, and non-parametric nature. Machine learning models are considered effective for stock forecasting, yet, their hyperparameters need tuning with the latest market data to incorporate the market's complexities. Usually, models are trained and tested in batches, which smooths the correction process and speeds up the learning. When making intraday stock predictions, the models should forecast for each instance in contrast to the whole batch and learn simultaneously to ensure high accuracy. In this paper, we propose a strategy based on two different learning approaches: incremental learning and Offline–Online learning, to forecast the stock price using the real-time stream of the live market. In incremental learning, the model is updated continuously upon receiving the stock's next instance from the live-stream, while in Offline-Online learning, the model is retrained after each trading session to make sure it incorporates the latest data complexities. These methods were applied to univariate time-series (established from historical stock price) and multivariate time-series (considering historical stock price as well as technical indicators). Extensive experiments were performed on the eight most liquid stocks listed on the American NASDAQ and Indian NSE stock exchanges, respectively. The Offline–Online models outperformed incremental models in terms of low forecasting error.

Keywords Real-time forecasting · Incremental learning · Technical indicator · Intraday trading

1 Introduction

Stock prices are affected by several micro and macro factors, such as the global economy, healthcare situation, oil prices, interest rates, news articles, public sentiment, etc. It is a significant task for financial companies to forecast stock prices, and rational forecasts can mitigate market risks and produce substantial returns. Several papers and studies have been devoted to making the best predictions and models possible based on the presence of many factors. The complexity of stock price prediction has made it a challenging problem, which has resulted in several papers and studies trying to make the most accurate predictions and models possible, owing to the massive potential for profit associated with them. In high-frequency trading, there is a large volume of orders, proprietary trading, and a short retention period, according to the Securities and Exchange Commission (Menkveld 2013). According to Aldridge and Krawciw, the most typical deal in 2016 started at 10%–40% of trading volume and 10%–15% of exchange rate and assets (Aldridge

Riya Kalra, Suryanshi Mishra, Satakshi and Manish Kumar have contributed equally to this work.

✉ Tinku Singh
rsi2018006@iiita.ac.in

Riya Kalra
mit2021055@iiita.ac.in

Suryanshi Mishra
suryanshimishra2496@gmail.com

Satakshi
satakshi@shiats.edu.in

Manish Kumar
manish@iiita.ac.in

[1] Department of IT, Indian Institute of Information Technology Allahabad, Prayagraj, U.P., India

[2] Department of Mathematics and Statistics, SHUATS, Prayagraj, U.P., India

Published online: 21 December 2022

and Krawciw 2017). The frequency of performance tasks in the stock market has escalated to a fraction of a second due to the enormous expansion of the internet (Bagheri et al. 2014). High-frequency trading is currently a very popular form of trading whose sole aim is to maximize profits by buying and selling stocks in a short span. Data patterns may be more valuable than sentiments and news articles. Since most of the news surrounding stocks is not generated regularly, only those posted on Twitter and other outlets can be accessed. Because of the difficulty of analyzing so many factors simultaneously, this study focuses on high-frequency data, i.e., short-interval stock prices, to predict the current price.

Since stock market data is generated periodically, it is considered time-series data. Stock market data is a series of time-ordered data points associated with single or multiple time-dependent variables. It has local and global patterns produced by the movements of prices on a chart and is the basis of technical analysis. Time-series can be classified as univariate or multivariate. Univariate time-series models only have one dependent variable, whereas multivariate models consider multiple factors. Training a univariate time-series model simply relies on past price movements. While the current stock price is affected by many factors, such as the closing or opening price, univariate predictive models reduce this complexity to a single factor and ignore all other dimensions. Multivariate time-series forecasting models take into account a variety of factors, such as the relationship between closing and opening prices, various technical indicators, daily highs and lows, and moving averages. When dealing with stock market data, several time-series components like trends, periodic swings, seasonal patterns, and random volatility might contribute to improved stock price forecasting. The trends are the result of long-term effects and can increase or decrease the time-series value over time. Periodic swings occur over the length of a time-series and are aimed at capturing short to medium-term gains in stock prices. Irregular movements exhibit rapid changes in time-series that are difficult to repeat, such as COVID'19.

Forecasting stock market trends based on live-streaming data has been a challenge for financial analysts and researchers. A streaming data process differs from traditional processing tools, which store and process data in batches. Stock prediction is one of the most widely used applications that require the real-time processing of streaming data. However, making decisions is challenging due to the market's complexity and chaotic dynamics, as well as the numerous non-stationary, undecidable, and unpredictable factors involved. The need to estimate the domestic stock market in several countries makes accurate forecasting even more challenging because there are various cultures, traditions, and diverse sources that may impact investors' decision-making processes. Based on previous trends in financial time-series, professionals from diverse sectors have created numerous forecasting methodologies. To achieve promising performance, most of these methods require careful selection of the input variables, the establishment of a predictive model coupled with professional financial knowledge, and the use of various statistical methods. As a result, it is difficult for people outside of the financial industry to estimate stock values using such approaches. The fluctuating nature of data and the heterogeneity of data types makes forecasting more complex based on technical analysis. Our main objective is to devise a lightweight prediction for the number of companies with fair accuracy, useful enough for intraday trading.

The objective of this study is to predict the closing price of stocks for the coming 15 mins utilizing the current stock price extracted through streaming data along with technical indicators calculated through this data to improve accuracy. The idea is to train a model using high-frequency historical stock market data at short intervals and then apply it in real-time. It is difficult to spot a trend over a short period, such as 1 min or 5 mins because there is a lot of noise. With a longer time frame, such as 15 mins, it can be easier to identify patterns, support, and resistance. Therefore, in order to get more reliable outcomes, we will use the data stream of a 15-min interval for real-time forecasting. Two different approaches have been adopted for the study: incremental learning, where the model will update with every single collected current stock price from the data stream, and Offline–Online, where the model is retrained at the end of every trading session. Incremental linear regression has been utilized for incremental models, while the variants of LSTM and CNN have been adopted for forecasting through Offline–Online models. In the Offline–Online approach, Offline involves analyzing a batch of data and optimizing the model to make a prediction, whereas Online refers to taking samples from the streaming data and making the prediction. However, incremental learning targets building a learning model that adapts to new data without losing any existing knowledge. In the Offline–Online approach, the model is not fine-tuned on receiving every new instance from the stream, although it is tuned after each trading session since the stock market might be impacted by multiple factors throughout a session. The incremental learning model updates with each stream instance, so it does not require retraining with the entire dataset.

This paper makes the following contributions:

- Stock prices are transformed from a univariate to a multivariate time-series with technical indicators, which allows for better forecasting.
- The paper utilizes deep learning models in real-time forecasting, which has been achieved through the model training after the entire trading session, rather than after retrieving the next stock instance, while real-time lag val-

ues and technical indicators of the stocks are maintained using local variables.
- Offline-Online and incremental learning approaches are compared for real-time forecasting.
- It empirically demonstrates the performance of the proposed system on the eight most liquid stocks of the NASDAQ and NSE, respectively, for one year.

2 Literature review

A stock market forecast involves predicting the future movement of a stock's value on a financial exchange. The efficient forecasting of share prices offers investors great profit potential, and correctly predicting the price movement within a short span can result in substantial profits. Several methods have been proposed for forecasting the market and providing decision-making guidance. Stock prices, rather than being stochastic, can be viewed as discrete time-series that are based on well-defined numbers collected at regular intervals of time. To build the forecasting model, the time-series data must be stationary. The differencing approach can be applied to obtain stationary data from a non-stationary time-series. On the other hand, the trend information in the time-series will be ignored by the differencing technique. Different methods can be applied in this area, including statistical methods and machine learning models. Generally, statistical models assume that there is a linear correlation structure among the time-series values. However, the nature of the stock market time-series is non-linear, volatile, chaotic and highly noisy (Alves et al. 2018). The autoregressive method (AR), the moving average model (MA), the combination of both AR and MA, i.e., the autoregressive moving average model (ARMA), and the autoregressive integrated moving average (ARIMA) are all traditional statistical methods. The ARIMA model's popularity originates from its statistical features as well as the notable Box-Jenkins model-building methodology. However, ARIMA models are not able to capture nonlinear patterns, and resembling complex real-life problems with linear models is not always practical (Zhang 2003). The researchers proposed the Grader causality test, which elongates the analysis from a univariate to a multivariate time-series analysis. Using the vector autoregressive moving average (VARMA), a multivariate time-series forecasting model was developed, which can represent Vector Moving Average (VMA) and Vector Autoregressive (VAR) models flexibly (Liu et al. 2021). Using a generalized autoregressive conditional heteroscedastic (GARCH) model for conditional variances, Pellegrini et al. (2011) apply the ARIMA-GARCH model to the forecasting of a financial series. Since the ARIMA-GARCH models never converge to homoscedastic intervals, their prediction intervals may be inadequate.

Traditional time-series forecasting algorithms can capture linear correlations and yield good results for a small dataset. But these algorithms are not very effective when used for time-series that are large and complex, such as stock market time-series (Liu et al. 2021). As a result, researchers focused increasingly on machine learning and deep learning methods in this domain. Javed Awan et al. (2021) utilized machine learning algorithms and sentiment analysis for forecasting stock prices. As per the outcomes, linear regression, extended linear regression, and random forest produce more accurate outcomes than the decision tree. Several studies have used linear and non-linear support vector machines (SVMs) for the forecasting of financial time-series (Cao and Tay 2001; Kim 2003; Maguluri and Ragupathy 2020). However, overfitting is a problem with these models, and the algorithms are not good at predicting large datasets. As compared to other models, support vector regression has better accuracy, according to Behera et al. (2020). Tuarob et al. (2021) created an end-to-end framework containing three sub-models, i.e., Davis-C for data collection related to stocks in real-time, Davis-A for analysis, and Davis-V for visualization. Their framework demonstrates that a combination of machine learning algorithms outperforms a standalone machine learning algorithm by large margins. Vijh et al. (2020) developed two models: one that predicts the price trends for the next day using historical data, and another that predicts the price trends for the next month using historical data. They employed Logistic Regression, SVM, and Boosted Decision Tree to forecast the trend based on volume volatility, sentiment, and continuous up/down.

In recent years, deep learning methods have become increasingly popular for predicting stock market moves. From complex and inconsistent data, these approaches can extract significant characteristics and detect underlying nonlinearities without relying on human skill (Kumar et al. 2021). Several experts have used deep learning to improve stock forecasting and produce profits for shareholders. In financial time-series forecasting, deep learning methods like artificial neural networks (ANN), convolutional neural networks (CNN), long-short-term memory (LSTM), hybrid algorithms, and others lead to better outcomes than statistical and machine learning methods. Vijh et al. (2020) explored the ANN and Random Forest on multivariate time-series on five stocks to forecast the next day's closing price using features such as the previous day's open price, closing price, Moving Average, Highs, and Lows. Lu et al. (2020) proposed a hybrid CNN-LSTM stock forecasting method. The authors compared the suggested model's performance to that of MLP, CNN, RNN, LSTM, and CNN-RNN on the Shanghai Composite Index. According to the experimental findings, the CNN-LSTM came up with the most accurate stock price forecasting, with an MAE of 27.564 and an RMSE of 39.688. Wen et al. (2020)

utilized the PCA-LSTM, which used the PCA (Principal Component Analysis) technique to identify technical indicator features and decrease dimensionality, yielding more accurate forecasts. DJI. Ince and Trafalis (2008) focused on short-term forecasts and used the SVM model in stock price forecasts. Specifically, their main contribution consists of comparing MLPs with SVMs and finding conditions where SVM is more effective than MLP. Moreover, different trading strategies affect the results. They contribute primarily by comparing MLP and SVM and finding cases in which SVM works better than MLP. Moreover, different trading strategies also affect the results. Dan et al. (2014) demonstrated the forecasting capabilities of deterministic Echo State Networks (ESNs) in stock prediction applications. Their experiments with the S & P 500 dataset show that the deterministic ESNs have improved their efficiency by about 23% compared to the standard ESN while demonstrating a negligible gain in predicting accuracy. Li et al. (2022) presented an effective deep learning-based BiGRU-attention model for short-term voltage stability assessment. It extracts temporal relationships and performs well even with a limited training dataset.

Intraday traders work with minute-based or sometimes even second-based stock market data. As a result, it is very crucial to determine how to analyze useful information and identify whether the forecasting method can be effective in real-time on high-frequency stock market data. Shakva et al. (2018), utilized ANN to predict stock prices on the Nepal Stock Exchange. The authors tried to predict the percentage increase or decrease in stock prices every second minute. They used technical indicators along with the data from the past 30 min. Selvamuthu et al. (2019) proposed the use of Levenberg-Marquardt, Scaled Conjugate Gradient, and Bayesian Regularization algorithms for predicting stock prices on a common ANN architecture of 20 hidden layers. They used the high-frequency dataset of Reliance Private Ltd. from Thomson Reuter over one year with 15,000 data points per day and were able to obtain a MAPE of 99.9% using tick data and 98.9% over a 15-minute dataset. Zhou et al. (2018) present a generic framework for adversarial training to anticipate the high-frequency stock market using LSTM and CNN. To avoid complex financial theory research and challenging technical analysis, this model employs a publicly available index offered by trading software as an input, which makes it more suitable for the typical non-financial trader. Liu et al. (2021) suggest a general framework for automatically developing a high-frequency trading strategy using a PPO-based agent. The study compares the LSTM and MLP for price prediction based on bitcoin prices in real-time. The study demonstrates the effectiveness of a PPO-based LSTM agent over an MLP, which earns high returns even when the market is in a slump and the price fluctuates.

According to the literature survey, most of the papers only forecast using historical data and do not operate with real-time data. The majority of them utilize historical day-to-day closing prices rather than current stock prices and do not deal with short time intervals such as five minutes or fifteen minutes. Stock market data is highly volatile and produced in massive amounts, making it difficult to manage and much more difficult to forecast. A majority of the studies used univariate stock market forecasting models, which do not take advantage of the technical indicators and other influential features to improve their accuracy. To leverage the advantages of technical indicators, we have converted the univariate stock series to a multivariate series. Deep learning models are effective in stock forecasting but have limitations like complex model training and a long training time, which makes it challenging to train the model in real-time on the new stock instances. The motivation of this research is to use deep learning models in real-time to forecast high-frequency stock data and to leverage the advantages of technical indicators by converting the univariate stock series to a multivariate series. The system in this paper aims to fill the void left by existing models for high-frequency trading.

3 Dataset

The forecasting would be centered around high-volume stocks since intraday traders tend to be most interested in them because of buyers' and sellers' availability throughout the trading session. For this study, we selected financial time-series (stocks) from the Indian and U.S. stock markets. The Bombay Stock Exchange (BSE) and the National Stock Exchange (NSE) are India's two major stock exchanges. We selected NSE because the volumes traded there are far higher than those traded on BSE. NIFTY 50 is an index of the top 50 companies listed on the NSE; we considered the top eight Nifty-50 stocks (India NSE 2001). There are two major stock exchanges in the United States: NASDAQ and the New York Stock Exchange (NYSE). Due to the NASDAQ's volatility and the number of listed companies, we will be using NASDAQ-traded stocks instead of NYSE stocks. According to market capitalization, the NASDAQ-100 is an index of the top 100 publicly traded companies, so eight of the most traded stocks were selected for the study (Nasdaq 2022). The selected stock from both exchanges can be seen in Table 1.

The high-frequency historical data of 15-min time intervals and live feeds of NSE stocks have been extracted using web-scraping through Zerodha API [27]. AlphaVantage API provided both real-time as well as historical stock prices for NASDAQ stocks [28]. A snapshot of the live-streamed stock prices is presented in Fig. 1.

Table 1 Selected stocks for study

S. No.	NSE stocks	NASDAQ Stocks
1	RELIANCE INDUSTRIES (RELIANCE)	APPLE (AAPL)
2	HDFC BANK (HDFCBANK)	MICROSOFT (MSFT)
3	INFOSYS (INFY)	AMAZON (AMZN)
4	ICICI BANK (ICICIBANK)	GOOGLE (GOOGL)
5	HDFC (HDFC)	NVIDIA (NVDA)
6	TATA CONSULTANCY SERVICES (TCS)	TESLA (TSLA)
7	KOTAK MAHINDRA BANK (KOTAKBANK)	BERKSHIRE HATHWAY (BRK-B)
8	ITC LTD. (ITC)	FACEBOOK (FB)


```
"2022-02-28 14:40:00": {
    "1. open": "121.1050",
    "2. high": "121.3800",
    "3. low": "121.1050",
    "4. close": "121.2450",
    "5. volume": "38987"
},
```

```
"2022-02-28 14:35:00": {
    "1. open": "121.0998",
    "2. high": "121.2000",
    "3. low": "121.0388",
    "4. close": "121.1050",
    "5. volume": "27560"
},
```


Fig. 1 Snapshot of Live stream format of extracted data

4 Methodology

This study analyzes time-series forecasting models for efficient forecasting of stock prices utilizing high-frequency data (15-min intervals). The proposed approach is based on two different learning methods: incremental learning and Offline-Online learning. These methods are applied to univariate and multivariate time-series. The univariate time-series was created using stock prices, whereas the multivariate time-series was created using stock prices in conjunction with exponential moving averages (EMAs) and volume-weighted average prices (VWAPs). The incremental model is continuously updated as it receives new instances of the stock price from the live feed of the stock market. On the other hand, the Offline–Online learning model needs to be retrained after every trading session. A retraining of the model will enable it to adapt to the current market trends, volatility, and seasonality. Figure 2 displays a visual representation of the methodology.

A pre-processing step is required before the model can be adapted for forecasting. Preprocessing includes removing null values and duplicate instances, verifying the order of

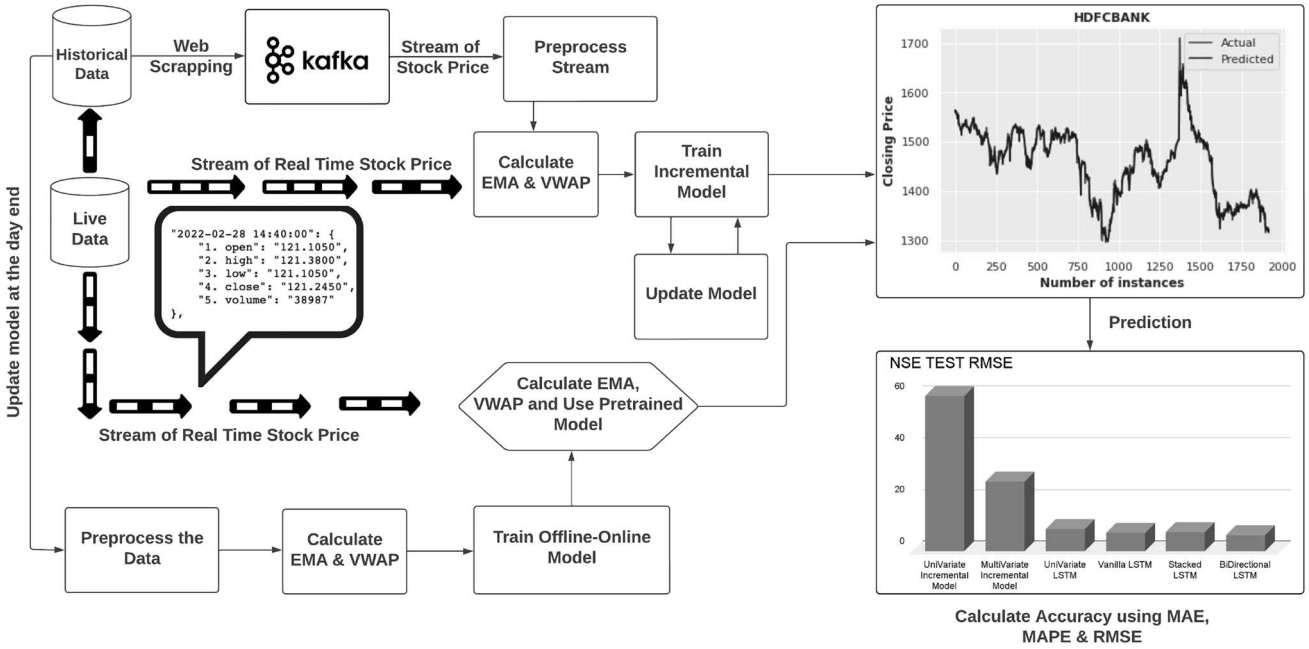

Fig. 2 Proposed model

instances, and finally converting the string date-time value containing UTC into a numerical timestamp. To perform the forecasting of the high-frequency stock market data, it is mathematically suitable to consider the time-series analysis with this condition $\{Y_t | t \in T\}$. A special type of examining stock prices (sequence of instances) collected over an interval of time is known as a "time-series analysis" (González et al. 2017). A time-series process $\{Y_t | t \in T\}$, is a stochastic process in which a set of random variables is ordered through time. T stands for index sets, which are distinct and separated evenly in time. Random variable Y_t is continuous. Let $i \in \mathbb{N}, T \subseteq \mathbb{R}$. A function $y : T \to \mathbb{R}^i, t \to y_t$ or, similarly, a set of indexed elements of $\mathbb{R}^i$,

$$\{y_t | y_t \in \mathbb{R}^i, t \in T\}$$

is called an observed time-series. It can also written as: $y_t(t \in T)$ or $(y_t)_{t \in T}$.

The variance function (fluctuation) of a time-series process (X_t) is defined as if $\forall t \in T$:

$$\sigma_t^2 = Var[Y_t], \sigma_t^2 = E[Y_t^2] - E[Y_t]^2, \forall t \in T$$

For historical stock data, if we assume that the mean and variance are constant then, $\mu_t = \mu$ and $\sigma_t^2 = \sigma_t$.

Therefore, the obvious estimate is:

$$\hat{\mu} = \frac{1}{n}\sum_{t=1}^{n} Y_t; \hat{\sigma}^2 = \frac{1}{n-1}\sum_{t=1}^{n}(Y_t - \mu)^2$$

A candlestick chart of the time-series for TCS over a 15-minute interval can be seen in Fig. 3. The stock prices are not generated at random but rather as a discrete-time-series created by collecting the numerical values at regular intervals. A candlestick shows the open, high, low, and close prices over an interval. Red candles represent the current closing price being higher than the previous candle's closing price, while green candles represent the lower price.

Fig. 3 Time-series of Stock Price

4.1 Technical indicators

In order to be effective in stock price prediction, traders utilize technical charts by analyzing price actions and technical indicators. There are numerous technical indicators that intraday traders use to determine when to buy or sell a particular stock, such as MACD and RSI. However, to capture current trends, $EMA(d)$ and VWAP are both suitable indicators. $EMA(d)$ is the average price of the stock in the previous d data points weighted exponentially (this way the prices of recent data points are given more weight). Since EMAs focus on recent price movements, they tend to respond more quickly to price changes. When trading intraday, it is considered reliable to use the value 5 to 20 for d in $EMA(d)$. It can be calculated as:

$$EMA_t = \alpha \cdot Y_t + (1 - \alpha) \cdot EMA_{(t-1)}$$

where EMA_t, Y_t denotes the EMA and closing price respectively at time t, and α is a smoothness coefficient between 0 and 1 that denotes the degree of weight reduction. For d previous observations α can be computed as:

$$\alpha = 2/(d + 1)$$

VWAP indicates the average price of the stock being traded in a day based on both price and volume. It is the ratio between the stock value and volume traded in a specific period. The indicator only works for one trading session and resets at the beginning of the next trading session. Suppose (Chen et al. 2013) that we have a large order v that must be executed during a specified time interval T. In that case, we must slice it into several smaller orders v_i and trade them over the time interval i from t_{i-1} to t_i ($t_i = i * L + t_0$, where L is the length of each time interval and t_0 is the start time for the trade).

$$VWAP_S = \frac{\sum_{i=1}^{n} p_i * v_i}{v}$$

$n = L/T$ denotes the trading periods, while $v = \sum_{i=1}^{n} v_i$ represents the time interval. To generate the multivariate time-series, the EMA(10) and VWAP are calculated using historical stock data, and the resultant series are combined with the stock price to get the final series.

4.2 Correlation/covariance analysis

The covariance and correlation functions define the level of dependency between the variables for any stock instances

(random variables) X_p and X_q. Auto-covariance function (ACVF) of the time-series $\{X_p, X_q | p, q \in T\}$ is defined as,

$$Cov[X_p, X_q] = E[(X_p - E[X_p])(X_q - E[X_q])]$$
$$\gamma_{p,q} = Cov[X_p, X_q] = E[X_p X_q] - E[X_p]E[X_q]$$

$\gamma_{p,q}$: Auto-covariance function of the given time-series.

The auto-correlation function (ACF) for the stochastic process is defined as:

$$Corr[X_p, X_q] = \frac{Cov[X_p, X_q]}{\sqrt{Var[X_p]Var[X_q]}}$$

For any two sets of stock instances $(r_1, r_2, ..., r_n)$ and $(s_1, s_2, ..., s_n)$, the sample of covariance and correlation functions are given as:

$$\hat{\gamma}_{r,s} = \frac{1}{n-1}\sum_{t=1}^{n}(r_t - \bar{r})(s_t - \bar{s}) \tag{1}$$

$$\hat{\rho}_{r,s} = \frac{\sum_{t=1}^{n}(r_t - \bar{r})(s_t - \bar{s})}{\sqrt{\sum_{t=1}^{n}(r_t - \bar{r})^2 \sum_{t=1}^{n}(s_t - \bar{s})^2}} \tag{2}$$

$\hat{\rho}_{r,s}$: Auto-correlation function of the stochastic process. However, for time-series data the ACVF and ACF measure the covariance/correlation between the single time-series $(r_1, r_2, ..., r_n)$ and itself at different lags.

Using Eq. 1 & 2 at lag 0, the ACVF $\hat{\gamma}_0$, is the covariance of $(r_1, r_2, ..., r_n)$ with $(r_1, r_2, ..., r_n)$ (or same series) and itself then,

$$\hat{\gamma}_0 = \frac{1}{n-1}\sum_{t=1}^{n}(r_t - \bar{r})(r_t - \bar{r})$$
$$\hat{\gamma}_0 = \frac{1}{n-1}\sum_{t=1}^{n}(r_t - \bar{r})^2$$

Similarly, the ACF $\hat{\rho}_0$, the correlation lies itself then,

$$\hat{\rho}_0 = \frac{\sum_{t=1}^{n}(r_t - \bar{r})(r_t - \bar{r})}{\sqrt{\sum_{t=1}^{n}(r_t - \bar{r})^2 \sum_{t=1}^{n}(r_t - \bar{r})^2}}$$
$$\hat{\rho}_0 = \frac{\sum_{t=1}^{n}(r_t - \bar{r})(r_t - \bar{r})}{\sum_{t=1}^{n}(r_t - \bar{r})\sum_{t=1}^{n}(r_t - \bar{r})} = 1$$

The auto-correlation function (ACF) & partial auto-correlation function (PACF) can be utilized to describe the order of stock price movements [31]. Let Y_t be the stationary time-series and Y_{t-h} with the lagged value of h. PACF estimates the degree of correlation between Y_t and Y_{t-h} but ignores the other time lags. We can predict x and y_3 with the help of y_1 and y_2 variables:

$$\frac{Cov(x, y_3 | y_1, y_2)}{\sqrt{Var(x|y_1, y_2)Var(y_3|y_1, y_2)}}$$

Here, y_1, y_2, and y_3 are the regression coefficients. In regression, x is a response variable, while the predictor variables are y_1, y_2, and y_3. A partial correlation exists between x and y_3, describing their association with y_1 and y_2 and indicating how dependent they are on one another. We define first-order with partial auto-correlation as being equal to first-order auto-correlation.

Based on Fig. 4 it can be observed that lag values of 15 min before at position 1 have a strong positive correlation with the current observations. In all three features, VWAP, Price, and EMA, the correlation is strong up to the lag value of 3 or up to 45 min, but beyond that, the correlation is not significant. Based on the analysis, a maximum of three lags are required for reliable forecasting. Furthermore, the lag values (3, 9, 27) were tested for reliability and consistency with the models in this study, and lag 3 was found to be reasonable in most scenarios.

4.3 Incremental approach

A stock price forecast is first derived through an incremental model. It uses incremental linear regression to predict the stock price for the next interval. Once the actual price for the next instance is captured through the data stream, it will estimate the prediction accuracy and update the model accordingly. The machine learning technique of incremental learning extends the existing model's knowledge by continuously using input data, i.e., by further training, it (Iscen et al. 2020). The ordered pair of (y_j, z_j) is denoted by the j^{th} pair of input and output observations. In the stock market, the correct output is considered to be $\mathcal{F}(y_j)$ if the system has provided data specified by a function $\mathcal{F}$. As a consequence of systematic noise or measurement error, the measured output z_j is consistent with $z_j = \mathcal{F}(y_j) + \epsilon_j$, where ϵ_j is inevitable, but hopefully, it is the minor term. If the function $\mathcal{F}$ has a m^{th} pair of observations, these ordered pairs are: $\{(y_1, z_1), (y_2, z_2), ..., (y_m, z_m)\}$. Even if we use $\mathcal{F}(y)$ to estimate z for an unobserved y, it will define a loss function $\mathcal{L}(z, \mathcal{F}(y))$ to evaluate the error which will occur. New observations that occur outside of our training set are classified as unobserved y. Here, the loss functions of the target function are $\mathcal{F}$.

Due to the linear regression, a linear function of the input vector is $\mathcal{F}(y) = W^T y$. Assume the loss function of the loss squared function is:

$$\mathcal{L}(z, W^T y) = (z - W^T y)^2$$

Therefore, the gradient of $\mathcal{L}$ with regard to a weight vector is defined as:

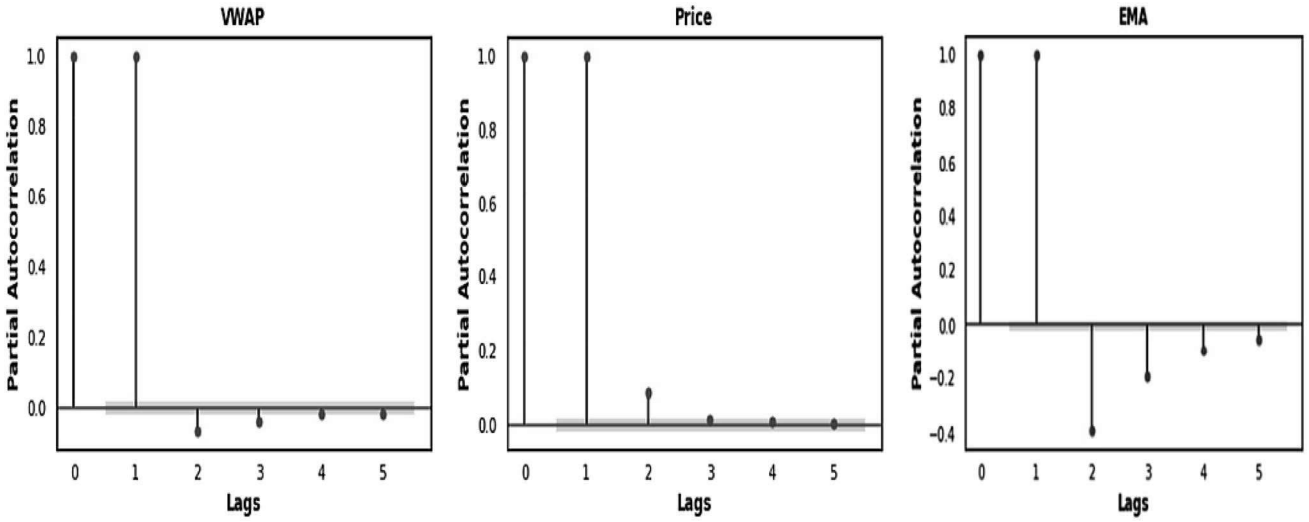

Fig. 4 Partial auto-correlation for VWAP, price and EMA

$$\nabla_W \mathcal{L} = -2(z - W^T y)y$$

Since the gradient represents the increased direction of the function, if we want the squared loss to decrease, we have to move the weight vector in the opposite direction of the gradient. We get the t^{th} observation y_t at time t, and we may estimate the outcome as follows:

$$\hat{z}_t = W_{t-1}^T y_t$$

Updated estimate of W is defined as:

$$W_t = W_{t-1} + \rho_t(z_t + \hat{z}_t)y_t$$

where $\rho_t > 0$ is known as step size. The step size is given as

$$\rho_t = \frac{\rho_o}{\sqrt{t}}$$

for some predefined constant ρ_o. The cumulative regret after t steps provides as a metric of effectiveness, which is defined as:

$$Regret = \sum_{t=1}^{T}(z_t - \hat{z}_t) - \sum_{t=1}^{T}(z_t - W_t^T y_t)^2 \tag{3}$$

Eq.(3), which is utilized in a financial decision-making system, where $W^T y_t$ is the optimum at step t, and the regret quantifies the total losses due to the non-optimal decisions.

4.4 Offline–online approach

For the Offline–online approach, financial time-series need to be converted into supervised learning problems to train a model. Since the model learns a function that maps a sequence of past observations as input to an output observation while it's being trained. That's why the dataset has to be prepared in the form of input samples. Each sample will take the current timestamp observation as the target value with n number of the previous instance as features where n is called lag observations. The model is trained after every trading session, and checkpoints are created for the trained model. The checkpoints help in storing the model's architecture, weights, and training configuration in a single file. Since the optimizer state of the model is recovered, it does not require retraining, and training can be resumed from the point at which it was stopped. A wide range of deep learning models for effective time-series forecasting have been utilized in this study, including LSTM and its variants; vanilla, stacked, and bi-directional LSTM, CNN, and CNN-LSTM.

LSTM is a type of artificial neural network (ANN) that excels at classification and regression tasks. LSTM (Graves et al. 2005) is a special kind of recurrent neural network (RNN) capable of handling long-term dependencies. The LSTM network is an advanced RNN, a sequential network, that allows information to persist. B-LSTM model is based on the bidirectional RNN model, which passes the information (Rathor and Agrawal 2021). It gives any neural network the ability to store the data backward or forward in both directions, at the same time. We can also have input flow in both directions, allowing us to save both previous and current data at any time step. These equations represent the forward (→) process as follows:

$$\overrightarrow{F_t} = \vec{\sigma}(\overrightarrow{W_f} * \overrightarrow{X_t} + \overrightarrow{V_f} * \overrightarrow{h_{t-1}} + \overrightarrow{Z_f})$$
$$\overrightarrow{I_t} = \vec{\sigma}(\overrightarrow{W_i} * \overrightarrow{X_t} + \overrightarrow{V_i} * \overrightarrow{h_{t-1}} + \overrightarrow{Z_i})$$
$$\overrightarrow{O_t} = \vec{\sigma}(\overrightarrow{W_o} * \overrightarrow{X_t} + \overrightarrow{V_o} * \overrightarrow{h_{t-1}} + \overrightarrow{Z_o})$$
$$\overrightarrow{C_t'} = tanh(\overrightarrow{W_c} * \overrightarrow{X_t} + \overrightarrow{V_c} * \overrightarrow{h_{t-1}} + \overrightarrow{Z_c})$$
$$\overrightarrow{C_t} = \overrightarrow{F_t} * C_{t-1} + \overrightarrow{I_t} * \overrightarrow{C_t'}$$
$$\overrightarrow{h_t} = \overrightarrow{O_t} * tanh(\overrightarrow{C_t})$$

In the backward (←) process, there are some equations as follows:

$$\overleftarrow{F_t} = \overleftarrow{\sigma}(\overleftarrow{W_f} * \overleftarrow{X_t} + \overleftarrow{V_f} * \overleftarrow{h_{t-1}} + \overleftarrow{Z_f})$$
$$\overleftarrow{I_t} = \overleftarrow{\sigma}(\overleftarrow{W_i} * \overleftarrow{X_t} + \overleftarrow{V_i} * \overleftarrow{h_{t-1}} + \overleftarrow{Z_i})$$
$$\overleftarrow{O_t} = \overleftarrow{\sigma}(\overleftarrow{W_o} * \overleftarrow{X_t} + \overleftarrow{V_o} * \overleftarrow{h_{t-1}} + \overleftarrow{Z_o})$$
$$\overleftarrow{C'_t} = tanh(\overleftarrow{W_c} * \overleftarrow{X_t} + \overleftarrow{V_c} * \overleftarrow{h_{t-1}} + \overleftarrow{Z_c})$$
$$\overleftarrow{C_t} = \overleftarrow{F_t} * C_{t-1} + \overleftarrow{I_t} * \overleftarrow{C'_t}$$
$$\overleftarrow{h_t} = \overleftarrow{O} * tanh(\overleftarrow{C_t})$$

Where, $\overleftarrow{F_t}, \overleftarrow{I_t}$, and $\overleftarrow{O_t}$ represent the backward forget input and output gate of the B-LSTM model. A weight matrix associates $\overleftarrow{W_f}, \overleftarrow{W_i}, \overleftarrow{W_o}$, and $\overleftarrow{W_c}$ with the inputs $\overleftarrow{X_t}$. Here $\overleftarrow{Z_f}, \overleftarrow{Z_i}, \overleftarrow{Z_o}$, and $\overleftarrow{Z_c}$ are the biased functions of the backward process model. The $\overleftarrow{\sigma}$, and *tanh* are the sigmoid and activation function of the model, respectively. $\overleftarrow{h_t}$ is the hidden state of the current timestamp, and $\overleftarrow{h_{t-1}}$ is the hidden state of the previous timestamp of the B-LSTM model. In the same way, the B-LSTM forward process works.

A vanilla LSTM (V-LSTM) (Wu et al. 2018) consists of an LSTM model with a single hidden layer and an output layer for the prediction. It can separate the effects of a performing variant change. In V-LSTMs there is a forget gate, allowing continuous learning. They also train using gradients rather than weight portions, as ESNs do.

For complex sequence classification challenges, stacked LSTM (Du et al. 2017) has become a reliable approach. An LSTM model with stacked layers can be called stacked LSTM (S-LSTM) architecture. When there is a long-term range between the data or a multivariate time dataset, connecting with several LSTM layers enhances the forecasting performance. Based on Fig. 5, X_t transmits the LSTM-1 layer with the hidden state h_{t-1} as the input vector and exists as h_t as the output vector, while h_t is the input vector for the LSTM-2 layer. The ultimate output, h_t^m, is generated when all of the LSTM-m layers have been stacked.

Figure 7 represents a hybrid CNN-LSTM deep learning model that is assessed to estimate stock prices, combining the benefits of both the CNN and LSTM models. The temporal dependencies are contained in the current input data and trained by the hybrid LSTM model. Figure 6 shows the architecture of the LSTM when the CNN input vector Y is input and the predicted data Z is output. The CNN system is integrated in such a way that it can handle multidimensional data. The information received in the input layer consists of various stock price sequences $\{T_1, T_2, ..., T_r\}$ which are mainly composed of the indicators dataset. Due to convolution and pooling layers, r convolution layers for each stock data have been used to produce r feature maps from the indicator dataset and to generate r feature vectors, which will also be referred to as an r channel. Every feature vector is fused in the matrix X_{T_r} as describes:

$$X_{T_r}^{dataset} = ReLU(dataset, T_r)$$

This convolution layer consists of a filter $W_c \in R^{g \times h}$, where g represents the dimension and h represents the step size in the feature vectors. As a consequence of the filter, the following feature vector is generated (Ren et al. 2015):

$$c = F(Conv(X_{T_r}^{dataset} * W_c) + B)$$

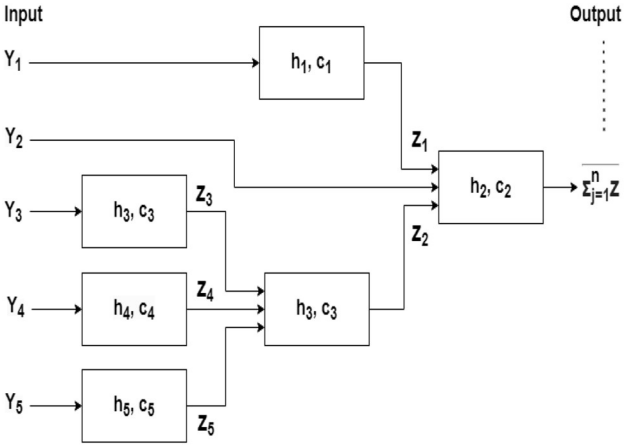

Fig. 6 Architecture of LSTM (input feature data Y from CNN and predicted output data Z from LSTM)

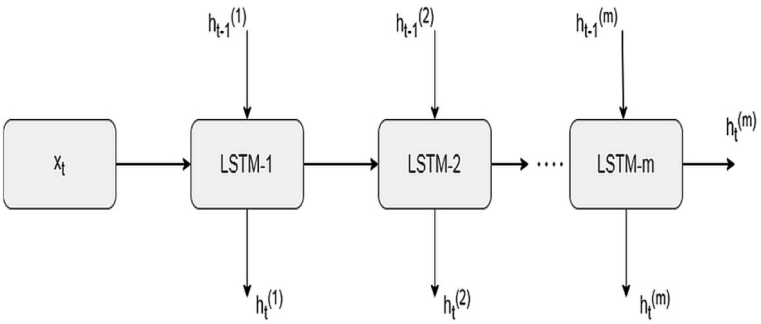

Fig. 5 Architecture of stacked LSTM

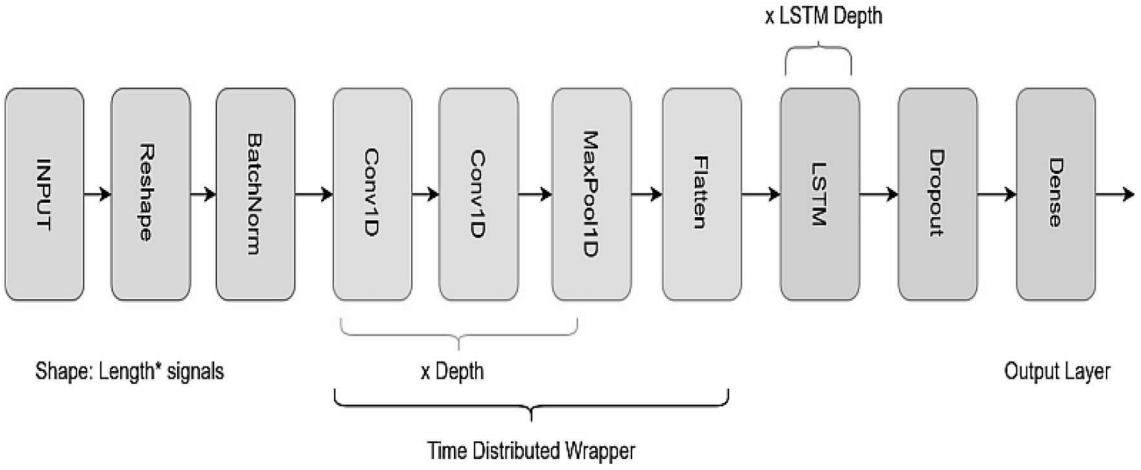

Fig. 7 Architecture of CNN-LSTM

Where, the bigoted vector is B, namely, the function's intercept, which will be used to achieve a linear classification. Based on the pool data, the most commonly used technique is to perform max operations on each filter result and get the output value as shown below:

$$X_{T_r} = [max(c)]$$

Here are the two reasons to clarify the max pooling operations: It removes the non-maximal values and speeds up the computations of the upper layer.

5 Performance evaluation

The accuracy of the forecasting models was assessed by measuring the mean absolute error (MAE), root mean square error (RMSE), and mean absolute percentage error (MAPE). The MAE is a model evaluation metric that is generally associated with regression models. Each prediction error represents the difference between the actual and predicted values of the model. It can be defined as:

$$MAE = \frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)$$

where y_j, $\hat{y}_j$ indicates the actual and predicted values, respectively. n represents the number of predictions. MAE provides equal weights to the errors, while RMSE is a quadratic measure since errors are squared before use, therefore it assigns higher weights to large errors. The formula for RMSE is as follows:

$$RMSE = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}$$

where y_j represents the j^{th} actual and $\hat{y}_j$ represents the j^{th} predicted values, respectively. While n shows the total number of predictions made. MAPE is similar to MAE but normalized by true observations. It shows how far the predictions of a model are from the respective actual values, on average.

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\frac{y_i - \hat{y}_i}{y_i}$$

6 Results and discussion

The experiments were carried out on real-time stock market data utilizing the Google Colaboratory, which uses Python 3.7 and offers a single GPU cluster with an NVIDIA K80 GPU, 12 GB of RAM, and a clock speed of 0.82 GHz. The proposed framework was being used to deploy numerous forecasting models on live data streams from the NSE and NASDAQ stock exchanges. For univariate and multivariate time-series of selected stock prices, the results were evaluated utilizing incremental learning and Offline-Online methods. Univariate time-series were derived from stock historical prices, whereas multivariate time-series include the EMA and VWAP along with the historical prices. In order to verify the effectiveness of the model, the models' forecasts were compared to the actual share prices of the eight most liquid stocks listed on the NSE and NASDAQ.

A variety of statistical performance measures, including RMSE, MAE, and MAPE, are being used to test the model's accuracy. MAE and RMSE are commonly used in financial analysis to measure the average gap between predicted and actual stock prices. The MAE is less biased for financial series with large values since it measures the average magnitude of those errors rather than taking into account the

direction of the errors. This could, however, not adequately reflect performance in case of large errors. RMSE is more informative when the overall impact is disproportionate to the increase in error. In contrast, MAE is more useful when the overall impact is proportional to the increase in error.

The incremental model learns from the data streams, where new data is constantly added. In this approach, initially, the model is trained with a small subset of data, and as a result, it shows a large deviation between actual and predicted stock prices. However, once the model is trained on a sufficient amount of data, the results get better. In the incremental learning process, there were 6404 instances of a one-year stock closing price recorded at 15-minute time intervals. EMA(10) and VWAP were calculated using the first ten instances, and initial model training follows these calculations. From the 11^{th} instance onward, training and testing are conducted simultaneously. Figures 8 and 9 illustrate the outcomes of the actual versus predicted prices

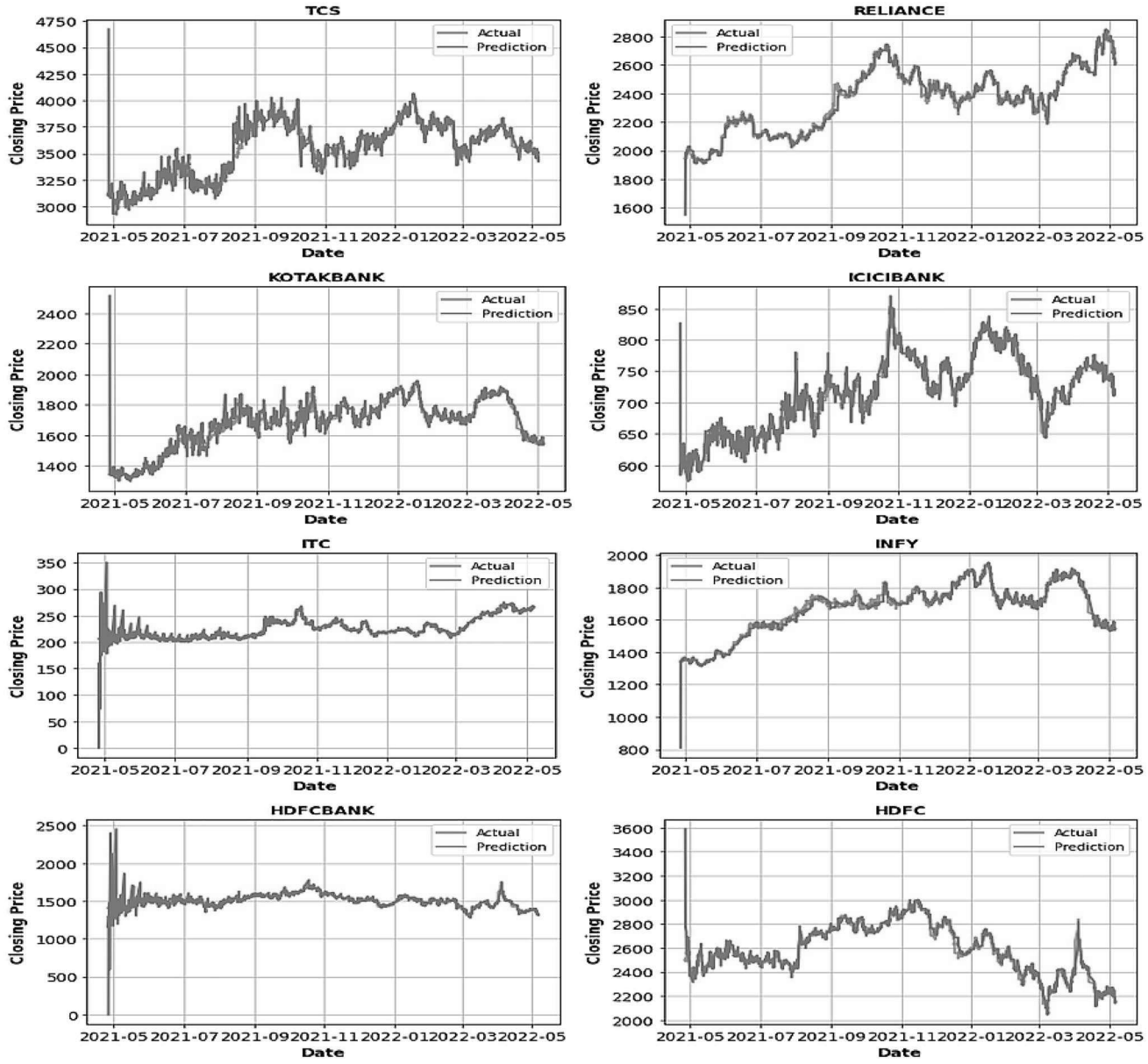

Fig. 8 NSE incremental model results

utilizing the incremental approach for the selected NSE and NASDAQ stocks, respectively.

Evaluations are done for both univariate and multivariate time-series. Univariate models are denoted by the abbreviation (U) along with the model name in this study, for example, univariate LSTM is denoted as LSTM(U). Table 2 compares the forecasting effectiveness of several machine learning models based on RMSE and MAE for the eight most liquid stocks listed on the Indian stock exchange NSE. Additionally, Table 3 illustrates the outcomes for the eight most liquid stocks listed on the American stock exchange NASDAQ. On stock price time-series, the multivariate incremental model INC outperforms its univariate counterpart INC(U), demonstrating the efficiency of technical indicators (EMA and VWAP) in estimating the future stock price.

Offline-Online models used in this study include the state-of-the-art deep learning models (LSTM and its variants,

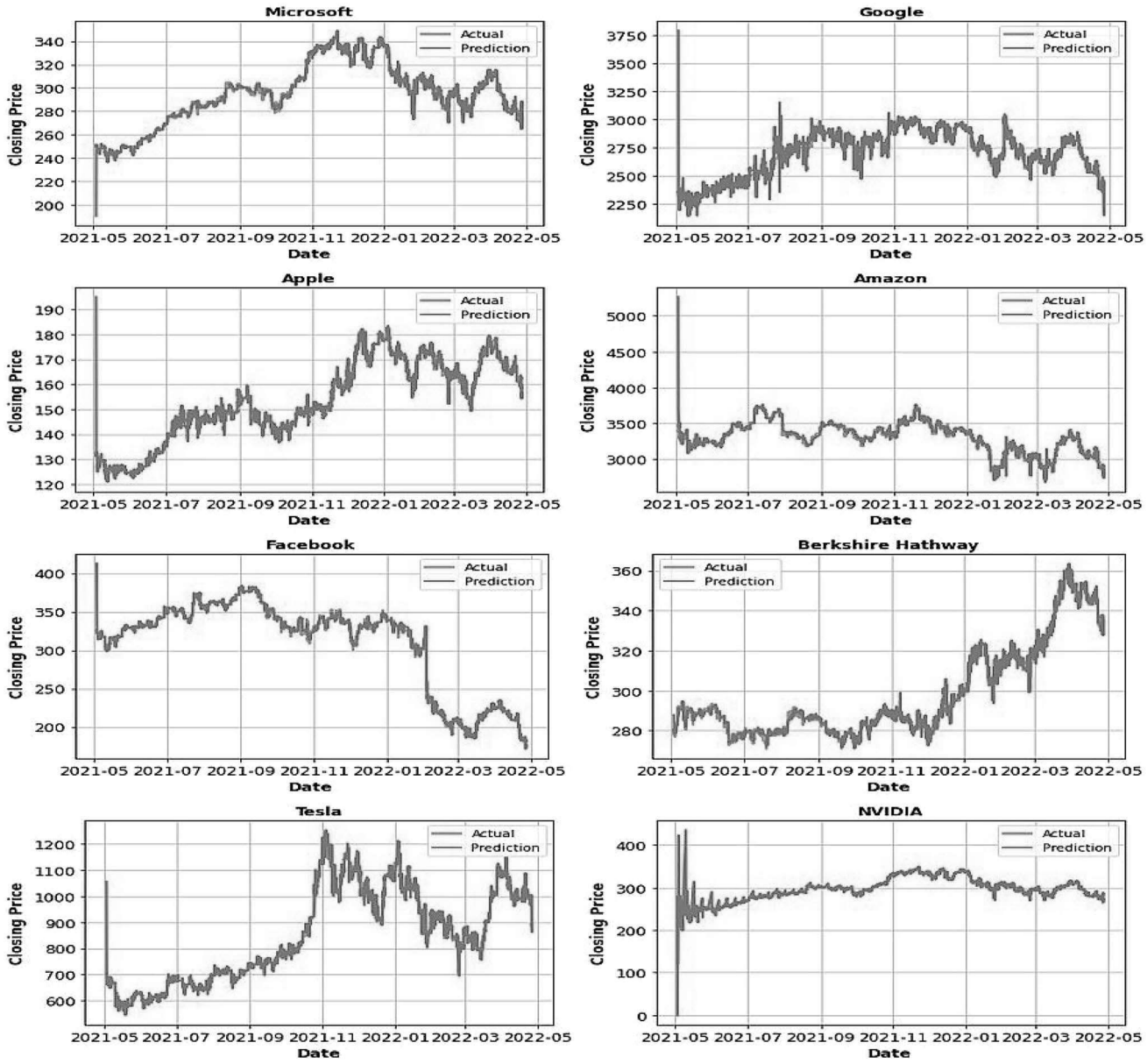

Fig. 9 NASDAQ incremental model results

Table 2 RMSE and MAE for the eight most liquid stocks of NSE

	Root mean square error (RMSE)							
Stock name	INC(U)	LSTM (U)	CNN-LSTM (U)	INC	S-LSTM	V-LSTM	B-LSTM	CNN
RELIANCE	83.080	54.870	21.634	16.850	11.410	12.150	7.700	10.939
HDFC	94.540	48.030	39.594	38.570	16.860	16.190	16.850	17.242
HDFCBANK	60.960	23.290	18.630	59.340	4.460	6.170	2.190	7.095
ICICIBANK	26.390	9.120	8.087	8.760	2.600	2.750	1.780	3.786
KOTAKBANK	38.320	41.310	26.294	36.200	10.210	8.920	7.450	9.691
ITC	8.510	2.020	1.663	1.920	0.270	0.430	0.350	0.778
INFY	67.040	13.340	16.494	18.110	5.330	6.180	4.390	8.028
TCS	112.450	53.460	37.141	54.720	15.430	12.430	11.350	12.948
	Mean absolute error (MAE)							
RELIANCE	19.793	40.196	16.070	9.445	8.260	8.950	5.582	7.830
HDFC	23.336	32.826	28.500	13.928	10.528	10.857	9.58	11.854
HDFCBANK	13.800	10.580	13.379	15.010	8.293	9.175	4.545	4.697
ICICIBANK	6.019	6.602	6.337	4.836	1.796	1.972	1.268	2.717
KOTAKBANK	13.111	31.043	19.978	19.652	7.541	6.545	4.985	7.308
ITC	1.923	1.369	1.235	1.998	0.184	0.233	0.265	0.540
INFY	33.087	9.915	12.321	10.407	8.093	8.498	6.545	5.471
TCS	28.909	42.925	27.178	27.103	11.347	9.295	8.073	9.042

CNN, and CNN-LSTM). The specifications of hyperparameters shared by all Offline-Online models (epochs: 50 with early stopping; activation function: relu; optimizer: adam with learning rate 0.003; loss function: MSE;) was finalized through grid search, while CNN and CNN-LSTM utilize the additional parameters (filters: 64; kernel-size:2; pool-size: 3 for max-pooling;). Fig. 10 shows the outcome of the grid search for hyperparameter optimization, and the green line highlights the consolidation of parameters that lead to the minimum MAE on the training dataset. On the training data, these hyperparameters were used for the final model training. The train test split was in the ratio of 70%:30%, since May 2021, 6404 instances of NSE stocks and 8892 instances of NASDAQ stocks have been extracted. After using 10 instances for calculating EMA(10), the remaining instances were split for training and testing purposes. It begins forecasting only after it has collected enough instances for training; thus, the forecasting is reliable from the start.

Figure 11 shows a comparison of RMSE for different models across different companies for both NSE and

Table 3 RMSE and MAE for the eight most liquid stocks of NASDAQ

	Root mean square error (RMSE)							
Stock name	INC (U)	LSTM (U)	CNN-LSTM (U)	INC	S-LSTM	V-LSTM	B-LSTM	CNN
AAPL	4.282	0.354	0.915	1.203	0.155	0.109	0.086	0.281
MSFT	7.039	0.395	2.205	1.493	0.256	0.381	0.169	0.763
AMZN	108.994	20.059	28.952	47.039	34.882	22.173	21.225	23.504
TSLA	19.306	4.746	8.599	7.970	5.941	6.861	3.742	3.860
GOOGL	90.697	13.381	23.371	43.205	20.201	23.570	12.794	15.526
NVDA	7.789	0.198	3.812	0.796	0.221	0.533	0.367	1.052
BRK-B	9.185	0.820	1.383	1.269	0.374	0.368	0.223	0.631
FB	9.394	3.190	2.385	2.380	2.061	2.240	0.860	1.348
	Mean Absolute Error (MAE)							
AAPL	1.808	0.872	0.720	0.563	0.474	0.720	0.406	0.324
MSFT	1.391	12.400	1.808	0.796	0.004	0.003	0.002	0.751
AMZN	25.707	0.872	19.375	19.385	0.750	0.720	0.642	10.391
TSLA	4.909	0.951	5.532	3.540	0.794	0.785	0.680	2.012
GOOGL	24.119	11.634	16.967	22.217	18.710	16.008	9.601	10.817
NVDA	3.094	0.671	2.631	0.796	0.670	0.554	0.574	1.022
BRK-B	1.808	1.492	1.018	0.890	0.670	1.232	0.574	0.844
FB	1.972	2.368	1.279	0.943	2.981	1.954	0.794	0.852

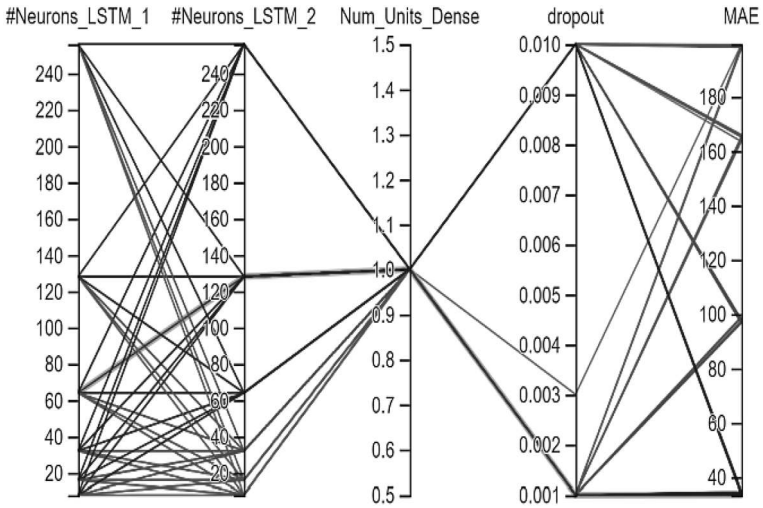

Fig. 10 The output from the tensorboard log file for hyper-parameter tunning through grid search

NASDAQ stocks using a line graph. Deep learning models outperform incremental models in terms of performance because they remember long patterns and can manage volatility and trends better. But, these models require training at the end of every trading session to be updated with the latest trends, seasonality, and sudden changes in the market. Forecasting outcomes using multivariate time-series was better, demonstrating that historical values alone cannot forecast better outcomes. Both LSTM and CNN produce good results, but B-LSTM outperforms others across all stocks on the NSE and NASDAQ in terms of low RMSE and MAE. Based on Fig. 11, it might seem that RELIANCE, HDFC, and TCS results are less accurate since their RMSEs for different models are on the higher side and so widely spread. However, model accuracy cannot be determined by RMSEs or MAEs since each company's stock price ranges

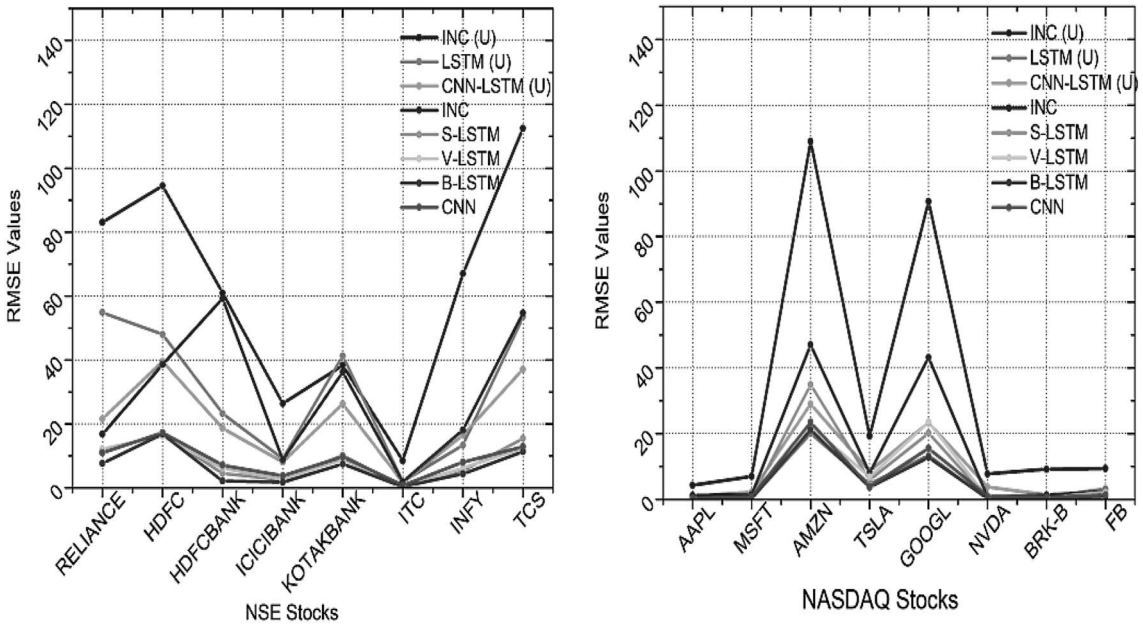

Fig. 11 Stocks RMSE for different models

may differ. For example, TCS's stock price lies in the range of 3000, while ITC has a stock price range of 200. As a result, TCS's RMSE values for all models will be greater than ITC's. This is because even a 1% difference in TCS pricing equals 30, whereas for ITC it's only 2. Since Apple, Microsoft, Berkshire Hathaway, and Facebook have low stock prices, their RMSE is relatively low. The MAPE of the models should be compared on the same financial series for comparisons of their accuracy.

MAPE is the most common measure to evaluate the model's forecasting accuracy. Since it utilizes the percentage error and is scale-independent, it can be used to compare the model's accuracy for the stocks in the different price ranges. Table 4 shows the MAPE results for the eight most liquid stocks listed on the NSE and NASDAQ, respectively. When considering univariate models, incremental linear regression is better than LSTM, while CNN-LSTM is the most effective. Figure 12 illustrates the graphical representation of Table 4 for a better interpretation of the results. LSTM on univariate time-series shows a high standard deviation on MAPE compared to other models and does not fit perfectly into the graph for NASDAQ stocks; therefore, LSTM results

Table 4 MAPE for the eight most liquid stocks of NSE

	NSE stocks							
Stock name	INC(U)	LSTM(U)	CNN-LSTM(U)	INC	S-LSTM	V-LSTM	B-LSTM	CNN
RELIANCE	0.009	0.017	0.007	0.004	0.003	0.004	0.002	0.003
HDFC	0.007	0.013	0.011	0.005	0.004	0.005	0.002	0.004
HDFCBANK	0.009	0.007	0.009	0.010	0.003	0.004	0.001	0.003
ICICIBANK	0.009	0.009	0.009	0.007	0.002	0.003	0.002	0.004
KOTAKBANK	0.010	0.017	0.011	0.012	0.003	0.004	0.004	0.004
ITC	0.009	0.006	0.005	0.004	0.001	0.001	0.001	0.002
INFY	0.020	0.006	0.007	0.006	0.003	0.004	0.003	0.003
TCS	0.009	0.012	0.008	0.008	0.003	0.002	0.003	0.003
	NASDAQ Stocks							
AAPL	0.012	0.177	0.005	0.004	0.001	0.000	0.000	0.002
MSFT	0.005	0.075	0.006	0.003	0.001	0.001	0.000	0.003
AMZN	0.008	0.421	0.006	0.006	0.005	0.004	0.004	0.003
TSLA	0.007	0.377	0.006	0.004	0.004	0.005	0.003	0.004
GOOGL	0.010	0.344	0.006	0.008	0.004	0.006	0.003	0.004
NVDA	0.011	0.054	0.011	0.796	0.001	0.002	0.001	0.003
BRK-B	0.006	0.184	0.003	0.003	0.001	0.000	0.001	0.004
FB	0.006	1.028	0.004	0.003	0.008	0.005	0.002	0.002

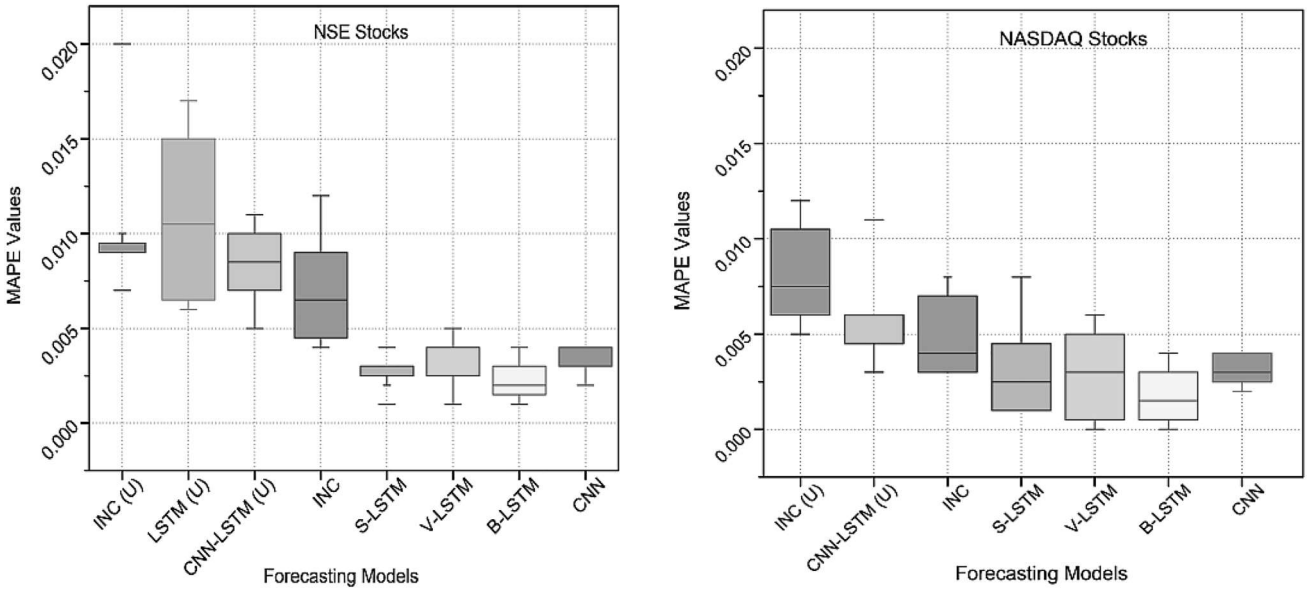

Fig. 12 MAPE of different models on NSE and NASDAQ Stocks

are removed from the NASDAQ graph. The analysis of the outcomes shows that B-LSTM is the most effective model among all, while CNN-LSTM is the most accurate univariate model for both NSE and NASDAQ stocks.

As B-LSTM provides the most accurate forecasts, we have selected it for the comparison of actual versus predicted stock prices for Offline-Online models. Figure 13 and 14 demonstrate the plot of B-LSTM for current stock prices versus the predicted values for the selected NSE and NASDAQ stocks. From the figures, it is clear that the predictions are very close to the actual values for all the selected stocks, which confirms the efficiency of the B-LSTM in forecasting.

Figure 15 shows the comparison of the mean RMSE for all the studied models on NSE and NASDAQ stocks. For all models, the average RMSE of NASDAQ stocks is lower than that of NSE stocks, since NSE stocks are denominated in rupees rather than dollars, so their range is larger. Moreover, the above results also indicate that multivariate models are more accurate than univariate models. The main reason is that the multivariate model considers more than one aspect. Multivariate models consider several independent variables to help forecast stock prices more accurately. Hence, multivariate models outperformed univariate ones even with only two additional indicators: the EMA and VWAP.

The models have been evaluated using real-time trading data during operational trading hours for a 15-min interval. A time difference (latency) was calculated between the actual retrieval time and the forecasting time. The

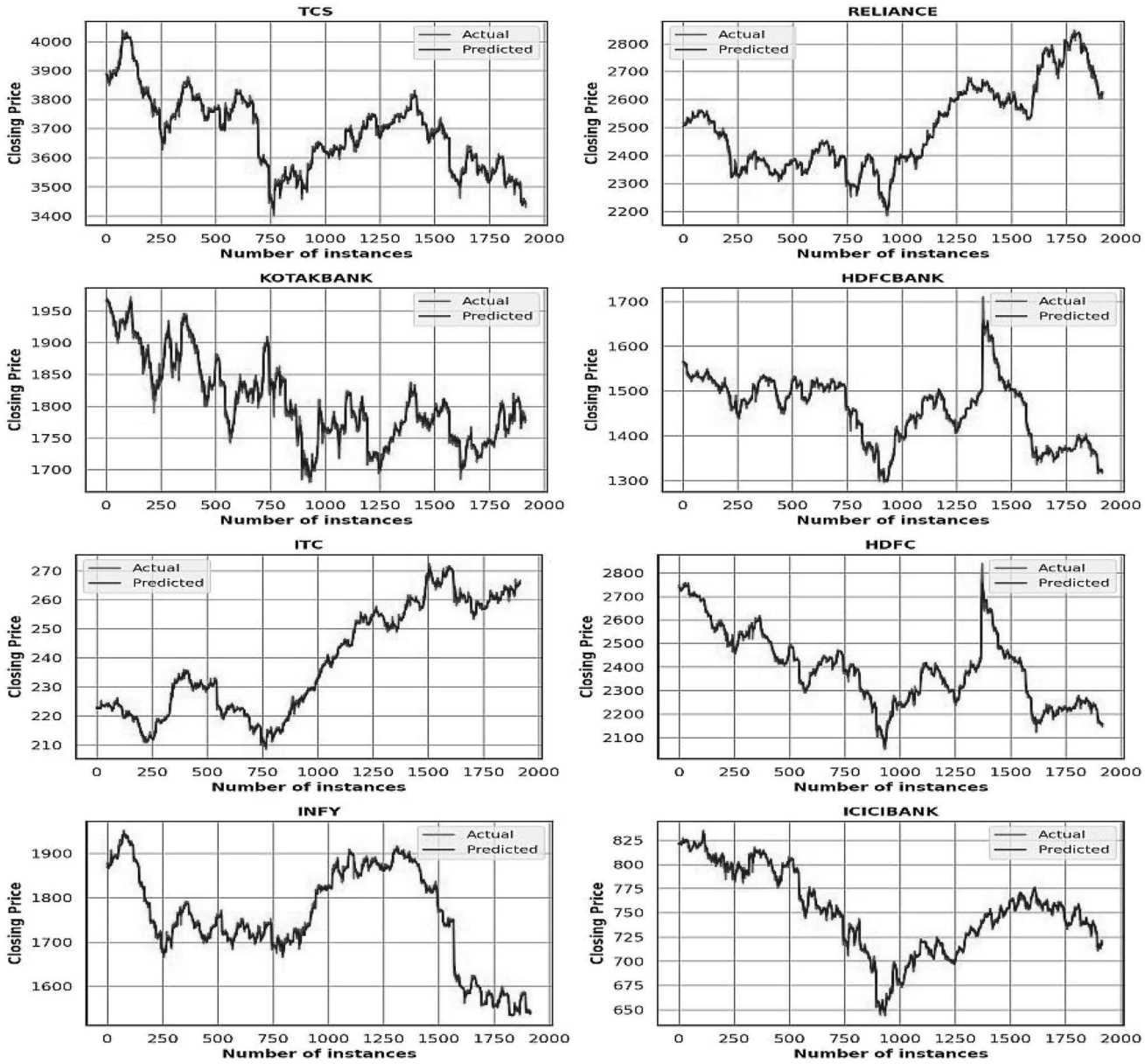

Fig. 13 NSE offline–online CNN results

experimental findings are shown in Fig. 16. As compared to the incremental model, the Offline-Online model has a lower latency. As it does not require retraining during operational hours which results in less forecasting delay. While the incremental model gets updated on the retrieval of new instances from the live feed which requires some additional training time. The average forecasting delay for incremental learning is 940 ms, while that of Offline-Online forecasting is 617 ms. The forecasting delay for both approaches is less than a second, which makes them closer to real-time forecasting. Traders might find these models useful in making short-term trading strategies for effective trading. The Offline-Online model has the limitation that it must be retrained after each trading session to stay up-to-date with current trends. For training purposes, the model requires a significant number of high-frequency historical instances of stock. Thus, it might be less accurate for stocks lacking high-frequency historical data.

7 Conclusion and future work

This study explores incremental and Offline-Online learning techniques for NASDAQ and NSE stock forecasting. The models used for this study were trained on the most recent stock data while the stock's time-series was

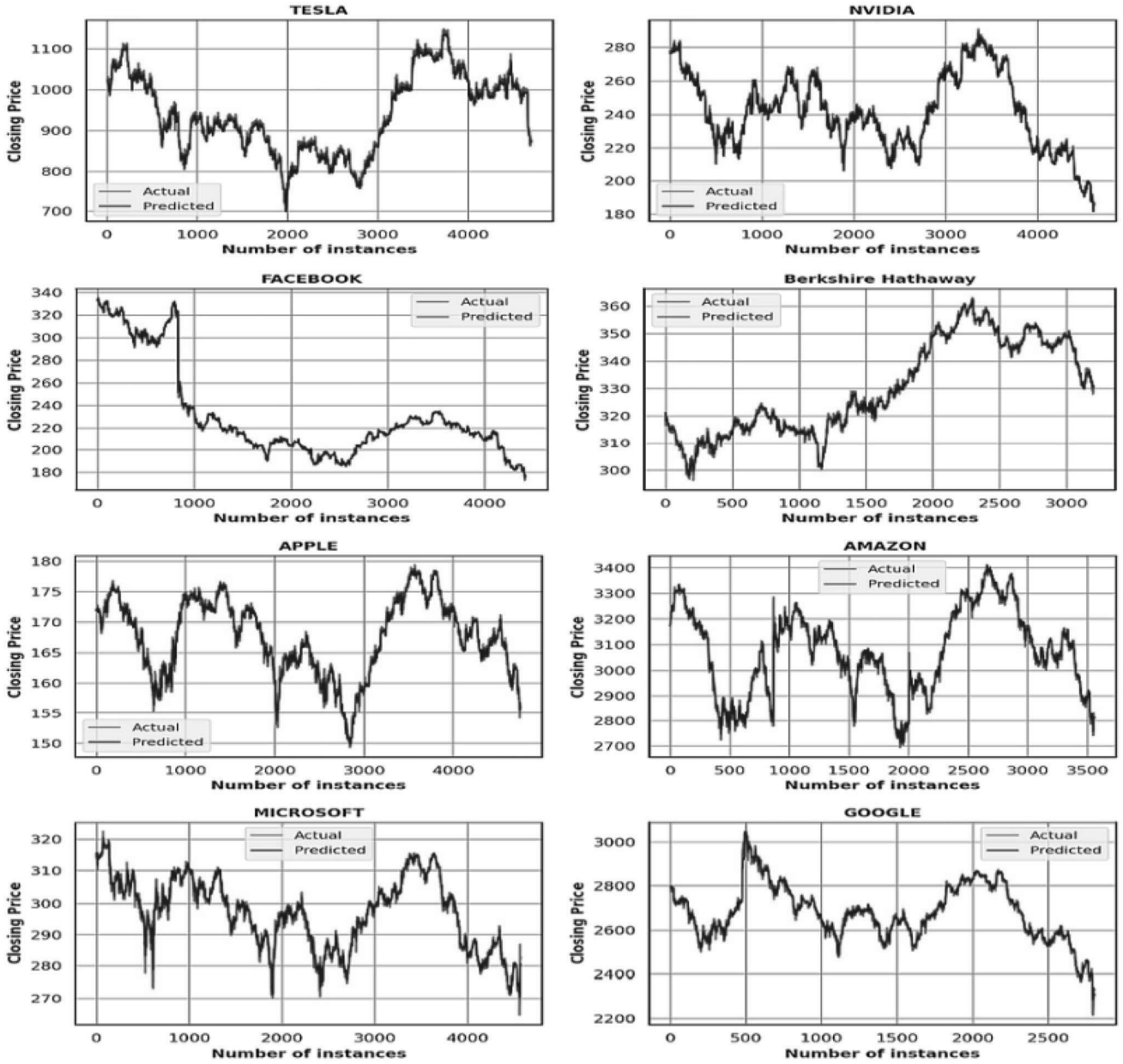

Fig. 14 Nasdaq offline–online CNN results

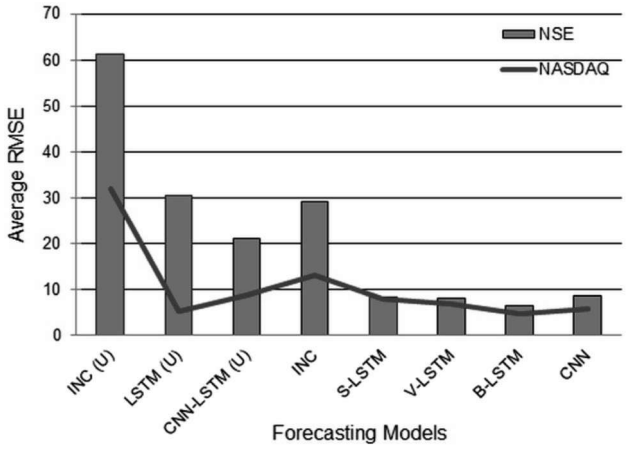

Fig. 15 Average RMSE of NSE and NASDAQ for different models

continuously updated from the live market feed so that these models could fine-tune their hyperparameters based on the changes that occurred in the stock's time-series during the trading sessions. A thorough analysis of various technical indicators that help in better price prediction led us to select the EMA and VWAP as features to consider along with the stock price for creating an effective multivariate time-series dataset. Furthermore, the system was tested on the top 8 stocks listed on the NSE and NASDAQ, respectively, and the performance of models was evaluated through RMSE, MAE, and MAPE. All the models forecasted better on multivariate time-series, showing the utility of the EMA and VWAP in predicting stock prices. B-LSTM was the leading performer among all for both Indian and US stocks, with the MAPE relatively close to zero. It is appropriate for short-term stock price prediction and may act as a helpful resource for the traders' efforts to maximize the returns on intraday trading. The B-LSTM

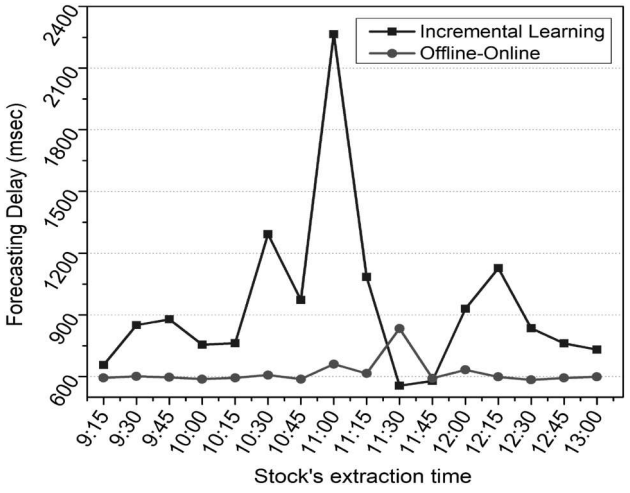

Fig. 16 A comparison of different studied approaches for forecasting delay

approach also provides real experience to anyone conducting research on high-frequency financial time-series. In the future, it may be useful to use deep learning in combination with incremental approaches to avoid Offline-Online model retraining after each trading session. Furthermore, global sentiment can be considered one of the features of multivariate time-series.

Funding No funding was received to assist with the preparation of this manuscript.

Data availibility The datasets analyzed during this study have been web scraped through Zerodha and AlphaVantage APIs, the complete details have been discussed in Sect. 3. It will also be made available on request.

Declarations

Conflict of interest The authors have no relevant financial or non-financial interests to disclose.

References

Aldridge I, Krawciw S (2017) Real-time risk: What investors should know about fintech, high-frequency trading, and flash crashes. John Wiley & Sons

Alves SA, Caarls W, Lima PM (2018) Weightless neural network for high frequency trading. In: 2018 International Joint Conference on Neural Networks (IJCNN), pp. 1–7 . IEEE

Analysis ATS. Partial Autocorrelation. (Accessed on 11/10/2020). https://online.stat.psu.edu/stat510/lesson/2/2.2

Bagheri A, Peyhani HM, Akbari M (2014) Financial forecasting using anfis networks with quantum-behaved particle swarm optimization. Expert Syst Appl 41(14):6235–6250

Behera RK, Das S, Rath SK, Misra S, Damasevicius R (2020) Comparative study of real time machine learning models for stock prediction through streaming data. J Univers Comput Sci 26(9):1128–1147

Cao L, Tay FE (2001) Financial forecasting using support vector machines. Neural Comput Appl 10(2):184–192

Chen CJ, Liu X, Lai KK (2013) Comparisons of strategies on gold algorithmic trading. In: 2013 Sixth International Conference on Business Intelligence and Financial Engineering, pp. 286–290 . https://doi.org/10.1109/BIFE.2013.61

Dan J, Guo W, Shi W, Fang B, Zhang T (2014) Deterministic echo state networks based stock price forecasting. In: Abstract and Applied Analysis, vol. 2014 . Hindawi

Du X, Zhang H, Van Nguyen H, Han Z (2017) Stacked lstm deep learning model for traffic prediction in vehicle-to-vehicle communication. In: 2017 IEEE 86th Vehicular Technology Conference (VTC-Fall), pp. 1–5 . IEEE

González JP, San Roque AMSM, Perez EA (2017) Forecasting functional time series with a new hilbertian armax model: Application to electricity price forecasting. IEEE Trans Power Syst 33(1):545–556

Graves A, Fernández S, Schmidhuber J (2005) Bidirectional lstm networks for improved phoneme classification and recognition.

In: International Conference on Artificial Neural Networks, pp. 799–804 . Springer
Ince H, Trafalis TB (2008) Short term forecasting with support vector machines and application to stock price prediction. Int J Gen Syst 37(6):677–687
Iscen A, Zhang J, Lazebnik S, Schmid C (2020) Memory-efficient incremental learning through feature adaptation. In: European Conference on Computer Vision, pp. 699–715 . Springer
Javed Awan M, Mohd Rahim MS, Nobanee H, Munawar A, Yasin A, Zain AM (2021) Social media and stock market prediction: A big data approach. MJ Awan, M. Shafry, H. Nobanee, A. Munawar, A. Yasin et al.," Social media and stock market prediction: a big data approach," Computers, Materials & Continua 67(2), 2569–2583
Kim K (2003) Financial time series forecasting using support vector machines. Neurocomputing 55(1–2):307–319
Kite Z. Kite Connect trading APIs. kite.trade. [Online; accessed 2022-06-15]
Kumar D, Sarangi PK, Verma R (2021) A systematic review of stock market prediction using machine learning and statistical techniques. Materials Today: Proceedings
Li Y, Zhang M, Chen C (2022) A deep-learning intelligent system incorporating data augmentation for short-term voltage stability assessment of power systems. Appl Energy 308:118347. https://doi.org/10.1016/j.apenergy.2021.118347
Liu FR, Ren MY, Zhai JD, Sui GQ, Zhang XY, Bing XY, Liu YL (2021) Bitcoin transaction strategy construction based on deep reinforcement learning. In: 2021 IEEE 2nd International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE), pp. 180–183 . https://doi.org/10.1109/ICBAIE52039.2021.9389965
Liu Z, Zhu Z, Gao J, Xu C (2021) Forecast methods for time series data: a survey. IEEE Access 9:91896–91912
Lu W, Li J, Li Y, Sun A, Wang J (2020) A cnn-lstm-based model to forecast stock prices. Complexity **2020**
Maguluri L, Ragupathy R (2020) An efficient stock market trend prediction using the real-time stock technical data and stock social media data. Int. J. Intell. Eng. Syst 13:316–332
Menkveld AJ (2013) High frequency trading and the new market makers. Journal of financial Markets 16(4):712–740
Nasdaq (2022) Global Markets Indexes and News | Nasdaq. www.nasdaq.com. [Online; accessed 2022-06-19]
of India NSE (2001) Daily Reports. niftyindices.com. [Online; accessed 2022-06-19]
Pellegrini S, Ruiz E, Espasa A (2011) Prediction intervals in conditionally heteroscedastic time series with stochastic components. Int J Forecast 27(2):308–319
Rathor S, Agrawal S (2021) A robust model for domain recognition of acoustic communication using bidirectional lstm and deep neural network. Neural Comput Appl 33(17):11223–11232
Ren S, He K, Girshick R, Sun J (2015) Faster r-cnn: Towards real-time object detection with region proposal networks. Advances in neural information processing systems **28**
Selvamuthu D, Kumar V, Mishra A (2019) Indian stock market prediction using artificial neural networks on tick data. Financial Innovation 5(1):1–12
Shakva A, Pokhrel A, Bhattarai A, Sitikhu P, Shakva S (2018) Real-time stock prediction using neural network. In: 2018 8th International Conference on Cloud Computing, Data Science & Engineering (Confluence), pp. 1–4 . IEEE
Tuarob S, Wettayakorn P, Phetchai P, Traivijitkhun S, Lim S, Noraset T, Thaipisutikul T (2021) Davis: a unified solution for data collection, analyzation, and visualization in real-time stock market prediction. Financial Innovation 7(1):1–32
Vantage A. Free Stock APIs in JSON & Excel, Alpha Vantage. www.alphavantage.co. [Online; accessed 2022-06-15]
Vijh M, Chandola D, Tikkiwal VA, Kumar A (2020) Stock closing price prediction using machine learning techniques. Procedia computer science 167:599–606
Wen Y, Lin P, Nie X (2020) Research of stock price prediction based on pca-lstm model. In: IOP Conference Series: Materials Science and Engineering, vol. 790, p. 012109 . IOP Publishing
Wu Y, Yuan M, Dong S, Lin L, Liu Y (2018) Remaining useful life estimation of engineered systems using vanilla lstm neural networks. Neurocomputing 275:167–179
Zhang GP (2003) Time series forecasting using a hybrid arima and neural network model. Neurocomputing 50:159–175
Zhou X, Pan Z, Hu G, Tang S, Zhao C (2018) Stock market prediction on high-frequency data using generative adversarial nets. Mathematical Problems in Engineering

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.