2022 IEEE 7th International conference for Convergence in Technology (I2CT)
Pune, India. Apr 07-09, 2022

PCA-LSTM: Deep Learning Approach for the Indian Large-Caps

Chinmay Joshi
Dept. of Data Science
MPSTME, NMIMS
Mumbai, India
joshi.chinmay18@nmims.edu.in

Siba Panda
Dept. of Data Science
MPSTME, NMIMS
Mumbai, India
siba.panda@nmims.edu

***Abstract*— This paper focuses on developing an algorithm for trading in the Indian stock market which outperforms the benchmark strategy. We used technical indicators as our input to predict the price using LSTM which averts multicollinearity within the indicators using PCA. The model was built on the Indian index, NIFTY50 and was selected on mean squared error of the scaled predicted prices, accuracy of signals generated, and annual profits compared to the buy and hold strategy. We also compared the PCA-LSTM model to the classical LSTM model. It was then applied to the 50 stocks which constituted the index to develop an optimized portfolio by maximizing the Sharpe ratio and minimizing the risk using the efficient frontier method. Our objective is to create a model that could answer three major questions - what to trade, how much to trade, and when to trade.**

Keywords*— *Deep learning, Principal Component Analysis, Stock Market, Technical Indicators

I. INTRODUCTION

The stochastic nature of the stock market is what makes it unpredictable, yet profitable. Recently, many researchers and industry experts have defied the random walk theory as the stock market could be exploited by understanding trend and pattern analysis [1]. Fundamental and technical analysis have been evolved to produce promising results, followed by the ideation of modelling and forecasting the information generated from the analysis. This space saw many developments over time. It started with understanding of the relevance of Brownian motion to the capital markets, and now it is moving towards more dynamic deep learning for analyzing the time series. In today's world, not only investors and traders but also regular people will profit from a system that makes investment and trading decisions for them.

The motivation behind developing this algorithm engendered from the idea of simplifying decisions about what to trade and when to trade for people from all walks of life. A large amount of population from India is keen on trading in the stock market irrespective of their primary occupation, and they will benefit from a system which tells them when to buy and sell, as well as guide them about which stocks to trade in and in what proportion of the total cash should one trade the given stock.

Although decision making is the most crucial part of the trading world, it is debased with the investors' and traders' sentiment [2]. Technical analysis came close to resolving this problem. Further, statistical models developed by quants performed much better since it not only provided insight about the present situation in the market but also forecast the future. Lately, researchers have realized the value of Neural networks [3], [4] and a few other machine learning algorithms for price and signal prediction. Neural networks are capable of considering a huge amount of information and dynamically adjust well with time-series data without the need to transfer to locally stationary series. Past research has emphasized the use of recurrent neural networks for forecasting of time series data. But, when it comes to using the technical indicators as input data, multicollinearity becomes a challenge as we go on increasing the number of technical indicators. Another observation is that portfolio optimization is often condoned but needs to be considered in order to minimize the risk while developing a robust algorithm.

II. RELATED WORK

Applicability of machine learning and deep learning techniques in the field of stock trading has gained popularity in recent years, and using technical indicators in combination or as input features to these predictive algorithms is a proven methodology for a more profitable model. Researchers have used historical data of chosen indices as raw data as the field demands the use of secondary data. Mean time span used for extracting historical data is 6 years. Of course, the interpretation is different based on whether the data is collected on a daily basis, hourly basis or high frequency data.

Pothumsetty [5] dives deep into the working of algorithmic trading and its benefits. He mentions that fuzzy logics, decision trees and neural networks are widely used. This work suggests resource allocation to be the focus of future work. Few insights from this work could be used as a starting a point for developing an appropriate framework. Large amount of work in terms of algorithms focuses on neural networks, genetic algorithms, SVM and ensemble methods like random forest.

In [4], various neural networks, both hybrid and simple were studied. Simple ANNs performed better than hybrid neural networks based on MSE and mean absolute deviate. In this work, technical indicators were not computed, whereas daily stock exchange rates were used as input the network. Another researcher [6] uses a novel framework based on neural networks, CEFLANN (computational efficient functional link artificial neural network) which generates higher profits as compared to naïve bayes, SVM, k-NN, and decision trees given the use of stated indicators. Experiment conducted by Ayala [7] also provides superior results using ANN but did not use multiple technical indicators in combination. Instead, machine learning techniques as a validation to technical analysis were used, which signaled the trade only when the two methods corresponded. Hasan [8] compared MLP, linear regression, random forest ad SVM using 97 features computed from

978-1-6654-2168-3/22/$31.00 ©2022 IEEE

different time spans of technical indicators based on Precision, recall, f1 score, compounded returns, average returns per trade and max drawdown. The research concludes MLP to perform the best. Whereas, in [9], trading strategy based on fine-tuned set of indicators and MLP was developed, but did not provide reasonable annualized returns. In [10], different approach was taken by converting the indicators to binary variables using conventional rules and then using them as input features to multiple models. RNNs and LSTMs were considered the best models based on accuracy and f1-scores. Back testing could have presented a more concrete conclusion. Agrawal [11] used O-LSTM, an LSTM framework which first finds an optimal subset of features using correlation analysis and use of tensors. Their model performed better than traditional LSTM, SVM and LR. This research emphasizes on computation of an optimal subset of features.

Haq [12] uses a unique hybrid approach to compute an optimal subset of features using multiple machine learning models as filters, which takes in 44 indicators and provides a subset of 20 significant ones. Their work uses weight vectors for SVM and linear regression, and error of prediction for random forest to rank the indicators, followed by clustering those obtained from multiple filters using affinity propagation. Eventually, they achieve an optimized subset of indicators that work as input features to STOCKNET. Based on classification accuracy and Mathew's correlation coefficient, their model performs better in Comparison to features only from one of the filters (L1-LR, SVM, RF), original dataset with 44 features, pure STOCKNET, classical ARIMA, and RAND. The markets are stochastic in nature and reacts differently to each indicator at a given time. In [13], various algorithms (Naïve Bayes, Random Forest, SVM, k-NN, Softmax) are used to predict the prices of selected stocks. The research realizes that the results vary by varying the number of indicators as well as size of the dataset. While naïve Bayes is claimed to perform best for small dataset (<=1800 entries), Random Forest is best for large dataset (>4500 entries). The accuracy of algorithms also decrease as number of indicators are decreased.

Ayala mentions AMH (Adaptive markets hypothesis) in contrast to EMH (Efficient market hypothesis) which provides an evolutionary perspective to the stock market [7]. Many studies make use of evolutionary algorithms for the model to evolve over time based on the stated objectives. Vora[14] used genetic algorithm along with PCA to identify a subset of indicators. The interesting part of the study was using correlated indicators but getting rid of the correlation using PCA. In [15], similar objective is achieved by preserving 10% of elite chromosomes and generating the rest 90% using roulette- selection method. The optimal set of indicators performs better than using either a single indicator or all the indicators. Bodas-Sagi [16] also uses genetic algorithm used to optimize parameters for both indicators use, with an objective to maximize profit, minimizing risk, volatility and transaction cost. Zhang [17] proposes a dynamic model to optimize the parameters of indicators. Their model uses grid search to optimize the features obtained from applying trading rules to the indicators. These indicators were fed to a genetic algorithm model which dynamically maximizes the proposed ratio (combination of Sharpe and sterling ratio) providing higher profits at a given risk. Compared to the benchmark models on basis of ROI, maximum drawdown and Sharpe ratio, it is a highly optimized and complete model. Choudhry [18] used different approach using genetic algorithm and SVM. While using genetic algorithm to find correlating stocks and computing indicators of those stocks as features, SVM was used for classification. He computed 35 indicators for each stock, extending 35*m features in total, where m is the number of correlated stocks.

In [19], models with and without technical indicators were experimented and concluded that addition of technical indicators in the input data to the model improved returns. This study also covered a crucial gap in the literature – portfolio optimization. Although, out of 500 chosen stocks portfolio consisted of only 4 stocks at a time, the two optimization methods proposed (Monte-Carlo simulation and multi variance optimization) performed significantly better than equally weighted portfolio.

III. METHODOLOGY

The process towards developing the algorithm could be divided in three steps – Finding the right model for the selected set of stocks, applying and evaluating for the model for all the selected stocks, portfolio construction and computing annual returns on the testing data with the forecasted signals. A general approach to the complete process is provided in the fig 1.

The developed methodology is designed for trading in cash; and maximum of only one trade is made at the opening time of the market for each stock in the portfolio. Transaction cost is not considered while ideating the process. It is assumed that the index reflects the information of the securities it comprises and thus, the model selected on the results of the index can be profitable when applied on its securities.

A. Data and Preprocessing

Data was collected and preprocessed twice, once while building the model for which the NIFTY50 index data from 2008 to 2020 was collected for model selection and tuning, and the second time the data for the same time period was collected for all the 50 stocks that constituted the index intended for applying the model and constructing the portfolio. Yahoo finance API was used for collecting the data of close prices.

After collecting the OHLCV data, 84 technical indicators were computed which considered the indicators for price momentum, trend, volume, volatility and daily returns. N/A and infinity values were eliminated from the dataset. The data, which finally consisted of the closing prices and the technical indicators was scaled between in order to prevent any indicator from assuming undesired dominance.

B. Model Selection

The model was selected on two major criteria, mean squared error, performance of the model in terms of annual returns as compared to the buy and hold strategy (it is considered the benchmark strategy). The model parameters were manually tuned to achieve the lowest possible mean squared error, and once consistency in improvement was observed, the signals were computed from the forecasted prices. The accuracy of signals was also computed, based on the predicted trend as compared to the actual trend. The model was finalized based how well it outperforms against the benchmark (buy and hold) strategy results.

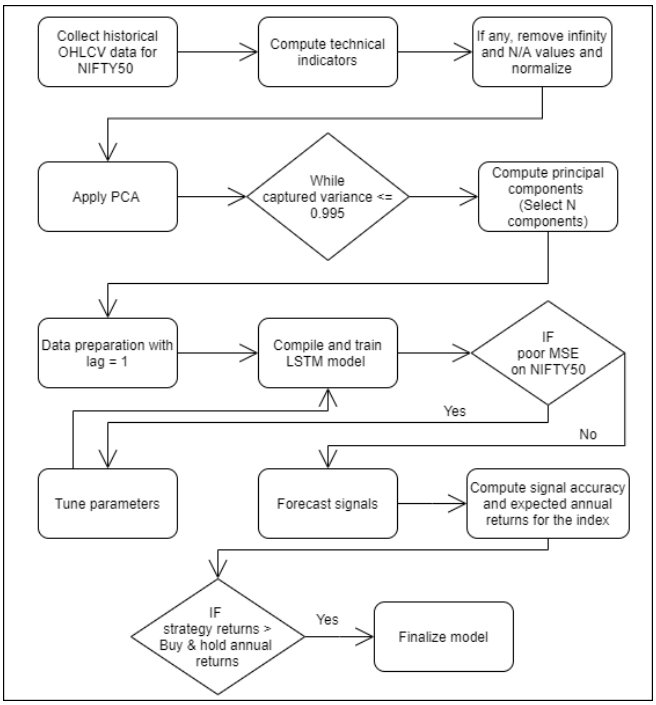

Fig 1. Proposed algorithm for making the model

1) Principal Component analysis

PCA uses eigenvalues and eigenvectors to calculate the components that give us the maximum variability. Since it allows us to use limited components instead of all the features, we can choose only those components which provide us the maximum variability and this is called eigenspace. It lets us to reduce the dimensionality of our features, without much loss information, thus allowing us to build a more efficient algorithm.

Mathematically,

$$S.u = \gamma.u \qquad (1)$$

Where, S is the covariance matrix of all our features and u is the unit vector to determine direction of our vector, we find lambda value for each diagonal element which provided us vectors such that when multiplied by lambda or eigenvalues, it stretches itself in a constant direction. We basically compute the eigenvalues for our set of features and the vector along which most features lie. So, the first few eigenvalues give us the maximum variability,

It serves two purposes, dimension reduction and elimination of high levels of correlation in our data. Dimension reduction plays a significant role due to the high computation cost of intensive neural networks on 50 stocks, while elimination of multicollinearity prevents highly correlated indicators from affecting each other while training on data for forecasting the closing price. In this algorithm, number of components are chosen dynamically based on the variability captured. The threshold for variability is 99.5%, until which principal components are computed. Approximately, it was observed that for each stock in the list, 25 to 35 components are used, thus reducing the dimensions from 84 to the thirties at maximum. The following three figures highlight the number of variables and correlation between indicators before and after PCA.

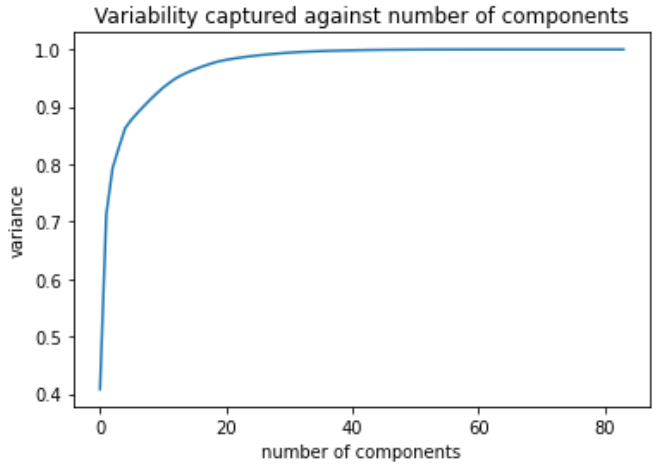

Fig 2. Variance captured against the number of components (NIFTY50)

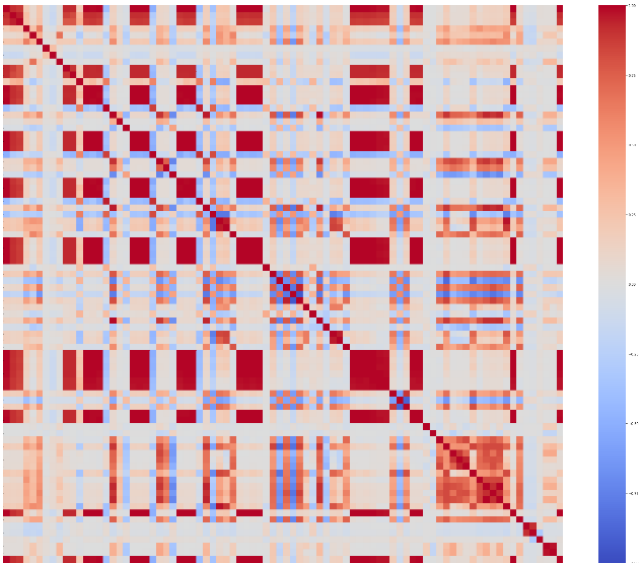

Fig3. Correlation heatmap BEFORE PCA for 84 technical indicators (NIFTY50)

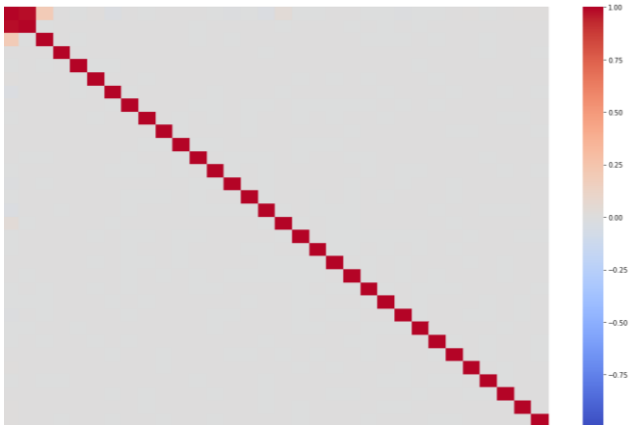

Fig 4. Correlation heatmap AFTER PCA for technical indicators – 30 components (NIFTY50)

2) LSTM (Long-short term memory networks)

As observed in the literature, LSTM is the preferred RNN [11]. One of the major advantages of LSTM is its capability to capture autoregressive structure of random signals [20]. Its gated architecture permits us to use cell state for considering or discarding information.

LSTM considers a complete LSTM block instead of nodes, which include various gates, cell state, weights and biases, and input and output vectors. The crux of LSTM lies in the three types of gates – input gate, which inputs the new information to be stored in the cell state; forget gate, which discards unwanted information; and output gate, which delivers the activation to the eventual output at a particular timestamp. All the gates use sigmoid function, which provides an output between 0 and 1.

$$i_t = \sigma(w_i[h_{t-1}, x_t] + b_i) \quad (2)$$

$$f_t = \sigma(w_f[h_{t-1}, x_t] + b_f) \quad (3)$$

$$o_t = \sigma(w_o[h_{t-1}, x_t] + b_o) \quad (4)$$

Where, i_t, f_t, and o_t are input, forget and output gates respectively; w_i, w_f, w_o and b_i, b_f, b_o are the corresponding weights and biases; h_{t-1} is the output from previous block of t-1 timestamp and x_t is the input at t timestamp.

The input gate i_t, ideally has two components, a sigmoid layer which updates the existing information in the cell and a tanh layer which creates a vector of candidates to be added to the state.

$$cn_t = \tanh(w_c[h_{t-1}, x_t] + b_c) \quad (5)$$

Equation 5 determines the candidate vector, where cn_t is a candidate. As observed from fig 5, this vector along with input vector, is added to the cell state which consists of information which has passed through the forget gate. All the components from input gate and forget gate forms the cell state.

$$c_t = f_t * c_{t-1} + i_t * cn_t \quad (6)$$

Equation 6 explains the cell state, which consists of selected information from the previous LSTM block output, input at the present timestamp t as well as new and updated candidates from both, the previous block and the present input. This cell state, when passed through the tanh function and merged with the result from the previous output gate provides us with our final output for the layer, as presented in the following information.

$$h_t = o_t * \tanh(c_t) \quad (7)$$

Where, h_t is the output of the complete LSTM block. This result, further lets us compute the cost and back-propagate to tune the gates, wieghts and biases to reduce the cost to its minimum possible value.

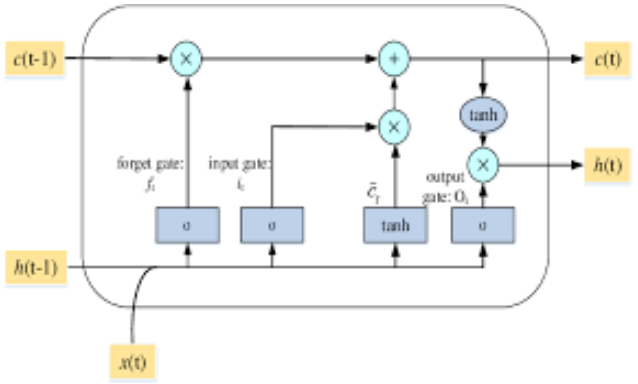

Fig 5. LSTM block

We initially performed LSTM on the data obtained from the NIFTY50 index, using Adam optimizer (adaptive learning rate optimization algorithm) for compilation. Although 100 epochs were set, Early stopping callback with patience of 8 epochs and model checkpoint based on validation loss was used to make the model computationally efficient. Dropout layers were added to prevent overfitting.

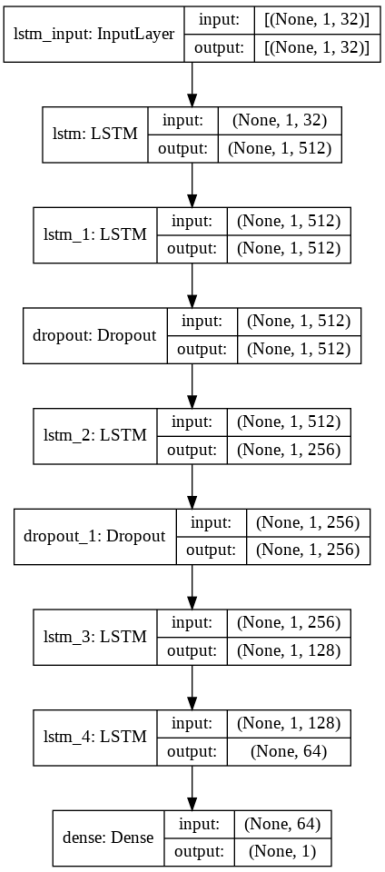

Fig 6. LSTM Model

C. *Portfolio Construction*

Once the PCA-LSTM model displayed satisfactory performance on NIFTY50 index, it was applied on 50 stocks that constituted the index. Similar to previous metrics, mean squared error, signal accuracy and annual returns were computed. The stocks which underperformed as compared to the buy and hold strategy were removed, and for the ones which outperformed the buy and hold strategy, Sharpe ratio and volatility were computed, where annual returns from the strategy for each stock were considered the expected returns. Using efficient frontier, both, the point of maximum Sharpe ratio and lowest volatility were calculated. It provides the proportion of each to trade in accordance to the signals generated by the model for the specific stock, for maximum returns at a minimum risk. We considered 0.02 as the risk-free rate.

$$Sharpe\ Ratio = \frac{Expected\ returns - Risk\ free\ rate}{Standard\ deviation} \quad (8)$$

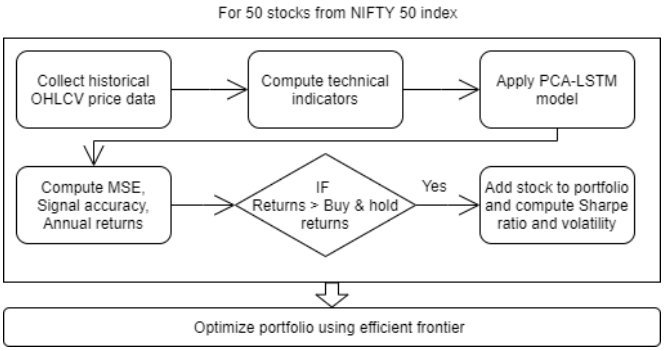

Fig 7. Portfolio construction model

IV. RESULTS

The PCA-LSTM approach is constructed on the fact that it outperforms the buy and hold benchmark strategy, thus outperforming the market. The results were computed for the year 2019 and 2020. It was chosen since it underwent one of the biggest black-swan events, and the steady performance of the model during this time provides insights on the model's robustness.

The initial model outperformed the buy and hold strategy by generating almost 20% more returns on NIFTY50 index, whereas the same LSTM architecture without PCA generated only 4% more returns than the buy and hold strategy. The PCA-LSTM model is superior in capturing the trend in the index.

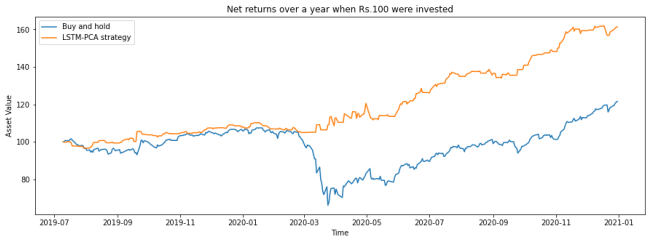

Fig 8. NIFTY 50 trend, PCA-LSTM vs Buy & Hold

When the model was applied to 50 stocks, the average scaled mean squared error was 0.0758 and predicted the trend with average accuracy around 49%. Finally, the constructed portfolio on the first day of 2020 would have resulted in 58% returns at 0.34 standard deviation, provided the signals were executed at the beginning of each trading day, and the stocks were traded in the proportion generated by the algorithm.

TABLE I. RESULTS OF EVALUATION

Stock ticker	Buy & hold returns	PCA-LSTM returns	pattern accuracy	Scaled MSE
ULTRACEMCO.NS	20%	10%	47%	0.028
GRASIM.NS	2%	20%	48%	0.022
SHREECEM.NS	16%	31%	51%	0.048
JSWSTEEL.NS	51%	32%	50%	0.021
SBILIFE.NS	22%	18%	48%	0.167
TATACONSUM.NS	136%	128%	48%	0.384
ICICIBANK.NS	23%	16%	44%	0.061
UPL.NS	-26%	10%	54%	0.026
KOTAKBANK.NS	37%	27%	44%	0.061
MARUTI.NS	28%	-16%	44%	0.020
RELIANCE.NS	58%	84%	49%	0.285
DRREDDY.NS	100%	79%	50%	0.066
TECHM.NS	46%	24%	47%	0.026
TATAMOTORS.NS	21%	59%	48%	0.011
WIPRO.NS	44%	34%	49%	0.051
HCLTECH.NS	85%	48%	50%	0.220
BAJFINANCE.NS	56%	30%	49%	0.050
LT.NS	-14%	7%	48%	0.019
NESTLEIND.NS	59%	50%	49%	0.195
HDFCLIFE.NS	48%	32%	52%	0.117
SUNPHARMA.NS	49%	47%	45%	0.012
BRITANNIA.NS	29%	40%	51%	0.073
TITAN.NS	41%	98%	47%	0.038
CIPLA.NS	51%	6%	45%	0.044
EICHERMOT.NS	34%	34%	47%	0.019
ITC.NS	-23%	-8%	48%	0.030
DIVISLAB.NS	134%	56%	50%	0.210
TCS.NS	38%	10%	45%	0.043
ASIANPAINT.NS	107%	69%	50%	0.117
HINDUNILVR.NS	40%	27%	47%	0.108
HINDALCO.NS	23%	30%	46%	0.022
BAJAJ-AUTO.NS	26%	25%	48%	0.022
HDFCBANK.NS	20%	9%	49%	0.049
INFY.NS	74%	46%	49%	0.111
TATASTEEL.NS	41%	73%	45%	0.016
BPCL.NS	10%	56%	47%	0.027
ADANIPORTS.NS	20%	9%	49%	0.028

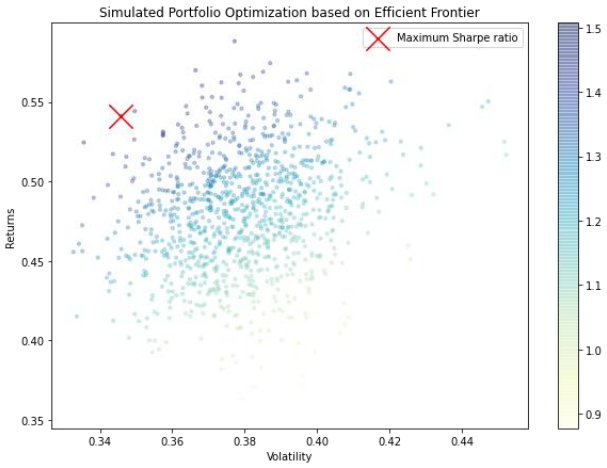

Fig 9. Efficient frontier graph for maximum Sharpe ratio

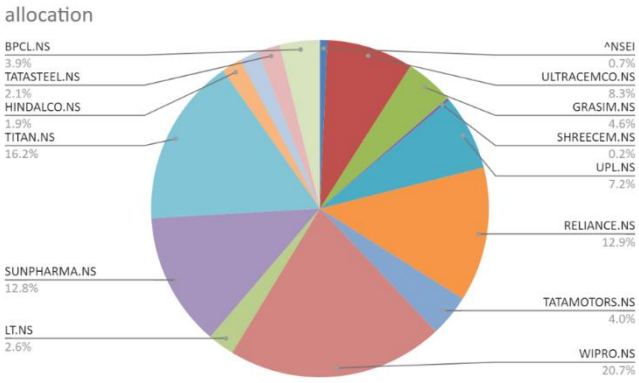

Fig 10. Portfolio allocation for 58% returns at minimum risk on 10-07-2019

V. Discussion

As observed, the proposed model performs effectively to provide high returns at minimum risk for a large-cap portfolio. PCA-LSTM is capable of considering high number of technical indicators as input features without facing challenges due high level of correlation between the indicators. Since the data considers trend, volatility, volume, momentum and daily returns, it is capable of generalizing on different situations. As observed on the testing data, the model leveraged the high volatility in the market to provide promising results.

This approach not only provides an entry and exit signal, but also provides the proportion in which trades should be executed for each stock in the portfolio, in order to maximize the profit while minimizing the risk. Thus, answering the major questions of decision making in a trader's life, it has the potential to scale and achieves the objective of simplifying decisions for anyone willing to trade in the stock market, especially in the large-caps for its capability to generate promising results at low risk.

VI. Future Direction and Conclusion

Observing the models, efficiency in generalizing in volatile environments, we are willing to back-test the strategy on other market indices as well as make it viable for intra-day trading. The model also requires an improvement in the computational efficiency. We believe an addition to the strategy that generates an optimal set of technical indicators post its adaptation to the market situation will decrease the cost and generate more profits. This approach opens up a scope for research in more adaptive and optimized processes for decision making in the finance industry.

The model answers major decision-making questions while trading in the Indian market. It observes gaps from the previous models and literature, and improves on the strategy by using large number of technical indicators and using the correct set of metrics for testing the strategy. There exists a need of such deep learning models in the Indian finance industry, which will eventually take the trading future to a more dynamic and automated approach.

References

[1] M. R. Vargas, "Deep Learning for Stock Market Prediction Using Technical Indicators and Financial News Articles."

[2] B. Huang, Y. Huan, L. Da Xu, L. Zheng, and Z. Zou, "Automated trading systems statistical and machine learning methods and hardware implementation: a survey," *Enterp. Inf. Syst.*, vol. 13, no. 1, pp. 132–144, 2019, doi: 10.1080/17517575.2018.1493145.

[3] P. Gao, R. Zhang, and X. Yang, "The Application of Stock Index Price Prediction with Neural Network," *Math. Comput. Appl.*, vol. 25, no. 3, p. 53, 2020, doi: 10.3390/mca25030053.

[4] E. Guresen, G. Kayakutlu, and T. U. Daim, "Using artificial neural network models in stock market index prediction," *Expert Syst. Appl.*, vol. 38, no. 8, pp. 10389–10397, 2011, doi: 10.1016/j.eswa.2011.02.068.

[5] R. Pothumsetty, "Application of Artificial Intelligence in Algorithmic Trading," *Int. J. Eng. Appl. Sci. Technol.*, vol. 04, no. 12, pp. 140–149, 2020, doi: 10.33564/ijeast.2020.v04i12.019.

[6] R. Dash and P. K. Dash, "A hybrid stock trading framework integrating technical analysis with machine learning techniques," *J. Financ. Data Sci.*, vol. 2, no. 1, pp. 42–57, 2016, doi: 10.1016/j.jfds.2016.03.002.

[7] J. Ayala, M. García-Torres, J. L. V. Noguera, F. Gómez-Vela, and F. Divina, "Technical analysis strategy optimization using a machine learning approach in stock market indices[Formula presented]," *Knowledge-Based Syst.*, vol. 225, p. 107119, 2021, doi: 10.1016/j.knosys.2021.107119.

[8] A. Hasan, O. Kalipsiz, and S. Akyokuş, "Modeling Traders' Behavior with Deep Learning and Machine Learning Methods: Evidence from BIST 100 Index," *Complexity*, vol. 2020, 2020, doi: 10.1155/2020/8285149.

[9] O. B. Sezer, A. M. Ozbayoglu, and E. Dogdu, "An artificial neural network-based stock trading system using technical analysis and big data framework," *Proc. SouthEast Conf. ACMSE 2017*, no. 2, pp. 223–226, 2017, doi: 10.1145/3077286.3077294.

[10] M. Nabipour, P. Nayyeri, H. Jabani, S. Shahab, and A. Mosavi, "Predicting Stock Market Trends Using Machine Learning and Deep Learning Algorithms Via Continuous and Binary Data; A Comparative Analysis," *IEEE Access*, vol. 8, pp. 150199–150212, 2020, doi: 10.1109/ACCESS.2020.3015966.

[11] M. Agrawal, A. U. Khan, and P. K. Shukla, "Stock price prediction using technical indicators: A predictive model using optimal deep learning," *Int. J. Recent Technol. Eng.*, vol. 8, no. 2, pp. 2297–2305, 2019, doi: 10.35940/ijrteB3048.078219.

[12] A. U. Haq, A. Zeb, Z. Lei, and D. Zhang, "Forecasting daily stock trend using multi-filter feature selection and deep learning," *Expert Syst. Appl.*, vol. 168, no. September 2020, p. 114444, 2021, doi: 10.1016/j.eswa.2020.114444.

[13] I. Kumar, K. Dogra, C. Utreja, and P. Yadav, "A Comparative Study of Supervised Machine Learning Algorithms for Stock Market Trend Prediction," *Proc. Int. Conf. Inven. Commun. Comput. Technol. ICICCT 2018*, no. Icicct, pp. 1003–1007, 2018, doi: 10.1109/ICICCT.2018.8473214.

[14] M. N. Vora, "Genetic Algorithm for Trading Signal Generation," *Int. Conf. Bus. Econ.*, vol. 1, pp. 316–320, 2011.

[15] M. Tanaka-Yamawaki and S. Tokuoka, "Adaptive use of technical indicators for the prediction of intra-day stock prices," *Phys. A Stat. Mech. its Appl.*, vol. 383, no. 1 SPEC. ISS., pp. 125–133, 2007, doi: 10.1016/j.physa.2007.04.126.

[16] D. J. Bodas-Sagi, P. Fernández, J. I. Hidalgo, F. J. Soltero, and J. L. Risco-Martín, "Multiobjective optimization of technical market indicators," p. 1999, 2009, doi: 10.1145/1570256.1570266.

[17] Z. Zhang and M. Khushi, "GA-MSSR: Genetic Algorithm Maximizing Sharpe and Sterling Ratio Method for RoboTrading," *Proc. Int. Jt. Conf. Neural Networks*, 2020, doi: 10.1109/IJCNN48605.2020.9206647.

[18] R. Choudhry and K. Garg, "A Hybrid Machine Learning System for Stock Market Forecasting," pp. 315–318, 2008.

[19] V. D. Ta, C. M. Liu, and D. Addis, "Prediction and portfolio optimization in quantitative trading using machine learning techniques," *ACM Int. Conf. Proceeding Ser.*, pp. 98–105, 2018, doi: 10.1145/3287921.3287963.

[20] M. Hansson and B. Nilsson, "On stock return prediction with LSTM networks," *Semin. 1st June 2017*, 2017.