Forest-CLAM: A Hybrid Ensemble Approach to Stock Prediction

Project Final Report

University of Wisconsin - Madison

ECE/CS/ME 539 Introduction to Artificial Neural Networks

Abstract

Our goal is to design a machine learning model accessible to everyone and capable of making accurate and informed decisions on the future of specific stocks. By the end, we hope to achieve similar results to current works [1], correctly predict a stock's outlook over a day with at least 60% accuracy, outperform the buy-and-hold strategy (BHS), and potentially test our model during a live market. Our current work has spanned numerous different models, from LSTMs to Random Forests, and our results have varied similarly. With our hybrid LSTM model, we were able to accomplish an accuracy of ~70% on validation data, while random forest achieved results around ~80%.

Introduction

Our objective is to develop a model that accurately predicts upward and downward trends for a given stock or a collection of tickers. By doing so, we aim to address the challenge of making informed investment decisions and provide everyday investors with a leg up. Notably, average investors often lack access to the information and capital resources that are readily available to wealthier market participants, and as a result, see a much lower return on their investments than their counterparts [2][3]. Furthermore, as wealth inequality continues to grow around the world, this topic continues to increase in importance. We hope our model will be able to overcome the hurdles faced by the average investor and propel them to a return they otherwise would not see.

Related work

Finance is one of the most lucrative areas, and many models have already been developed to accurately predict the stock price, with architectures such as RNN, CNN, GNN, Transformer, GAN, and LLM being used [1]. In addition, large hedge funds and private equity firms have invested heavily into AI-powered trading.

A recent study used an LSTM neural network to predict the day-to-day prices of specific stocks, which had an accuracy of 66.9% [4]. Another group utilized high-frequency trading data to achieve a rough approximation of market trends and cut execution costs [5].

The majority of current approaches take a market-general stance, meaning the model is designed on data from numerous different stock tickers [6]. In our work, we aim to make our model stock-specific, allowing it to have a better understanding of an individual stock's (or a small group of stock's) fluctuations and patterns.

Methods & Algorithms

Three distinct models were used in this paper: a CNN-LSTM-Attention hybrid model (CLAM), a Random Forest model, and a MLP model (in a separate report). The first two models are then combined to form our final model, Forest-CLAM.

CLAM

The first model, CLAM, is a hybrid model constructed around the original Long-Short-Term-Memory (LSTM) model. The reason for selecting this model as our backbone, is because the LSTM model is good at processing sequential data by selectively remembering long-term dependencies, while forgetting irrelevant information by its gating mechanisms. We chose this model as it can overcome the vanishing gradient problem that plagues traditional RNNs, while having a better performance on longer sequences, and improving prediction accuracy for time-dependent trends.

The LSTM model first takes a fixed window of time-series features x_t each normalized to a fixed scale. The current input x_t is then combined with the previous hidden state h_{t-1} and fed through the Forget Gate (LSTM.1) to decide how much of the previous cell state c_{t-1} keep. Then through the Input Gate (LSTM.2), the model determines how much the new information should be added to memory. A candidate cell state (LSTM.4) is then generated for updating the cell state based on the current input and previous hidden state, which then was updated by partially keeping the old memory c_{t-1}, while updated with new candidate values i_t (LSTM.5). The Output Gate of the model (LSTM.3) then chooses the parts of the cell that will be exposed as the hidden state, producing a new hidden state (LSTM.6), which also acts as the model's output for the current timestep. Finally, the model outputs the probability distribution for the next movement, in our case, whether the price of the stock to "rise" or "fall".

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \quad \text{(LSTM.1)}$$

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \quad \text{(LSTM.2)}$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \quad \text{(LSTM.3)}$$

$$\tilde{c}_t = tanh(W_c x_t + U_c h_{t-1} + b_c) \quad \text{(LSTM.4)}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \quad \text{(LSTM.5)}$$

$$h_t = o_t \odot tanh(c_t) \quad \text{(LSTM.6)}$$

The LSTM model is then combined with the Convolutional Neural Network (CNN), and Attention Layer Network. The former is implemented to extract local spatial-temporal patterns from the time-series data before feeding into the LSTM. CNN is very effective at short-term feature interactions, in the case of a stock market this is the sudden spike or fall of a stock within fixed-size windows. In addition, the CNN helps reduce noise in the data, overall improving the LSTM's predictability on sequential dependencies. The latter, Attention Layer Network, is used to help the model focus on relevant timesteps in the sequence when making a prediction. It compresses all sequential information into a single hidden state, then adaptively assigns higher weights to important periods and lower weights to less important periods, improving the prediction accuracy as a whole by emphasizing the data points most influential to the decisions.

Random Forest

The second model tested was a random forest classifier. Random forest is an ensemble approach utilizing numerous, independently generated, decision trees. Due to the nature of this design, random forests are theoretical universal approximators, meaning that given enough data and the correct design conditions, a forest can approximate any measurable function. Therefore,

our goal was to test this property on trading data, to see if there exists a "simple" solution to our problem.

Coming from the SKlearn "RandomForestClassifier" package, our forest uses 200 trees with no max depth, balanced class weighting, and all other parameters set to default. The gini and entropy criteria showed similar results, however, gini is marginally more computationally efficient, and so it is used here. The data used is the same as in CLAM, however features are not normalized, as the threshold splits are invariant across scaling.

Forest-CLAM

With the addition of the Random Forest model, our final hybrid model, Forest-CLAM, is constructed, where the Random Forest and CLAM each generate independent weights and predictions, then combined through our ensembling strategy to produce the final trading decisions. The strength of handling non-sequential features by the Random Forest and the ability of capturing complex temporal and spatial-temporal information by the CLAM, all together resulting in a more robust and accurate prediction model.

Experiments

Evaluation Method & Metrics

We evaluate the overall performance and the classification effectiveness of the model using a testing dataset on the measures of accuracy, precision, recall, and F1-score. We then inspect the confusion matrix to analyze the trade-offs between false positives and false negatives.

Additionally, we set up a small trading simulation for the model, letting it decide what trades to make using its predictions. We initially gave the model 1 million dollars worth of capital to trade in a hypothetical stock market, where it has free will to use the amount of money it holds to buy or sell stocks. In this simulation, we measured the potential return of our model trading over a course of ~900 days. Our goal was to see an improvement over the traditional BHS. We also implement multiple "confidence thresholds" for the models, as well as, leveraging the output of both the CLAM and the Random Forest, to determine how different confidence levels relate to gains and losses. In addition, an accuracy of > 60%, and an F1-Score of > 0.6, would indicate our model has a good balance between precision, recall, and accuracy for both of the up and down cases.

Data

The dataset used for both training and validation is the "Yahoo Finance (YF)" dataset, as it is quick and accessible. We utilized Open, Close, High, Low, Volume (OCHLV) data from five major technology stocks: "AAPL", "MSFT", "GOOG", "AMZN", and "META", spanning the period 2018/01/01 to 2025/06/30, with the interval of one day.

To prepare the data, all missing or erroneous data was dropped to assure consistency. Normalisation was performed using *StandardScaler* to transform each feature to have zero mean and a unit variance. We also calculated the logarithmic return feature for the closing price, defined as below:

$$LogRet_t = ln(\frac{Close_t}{Close_{t-1}})$$

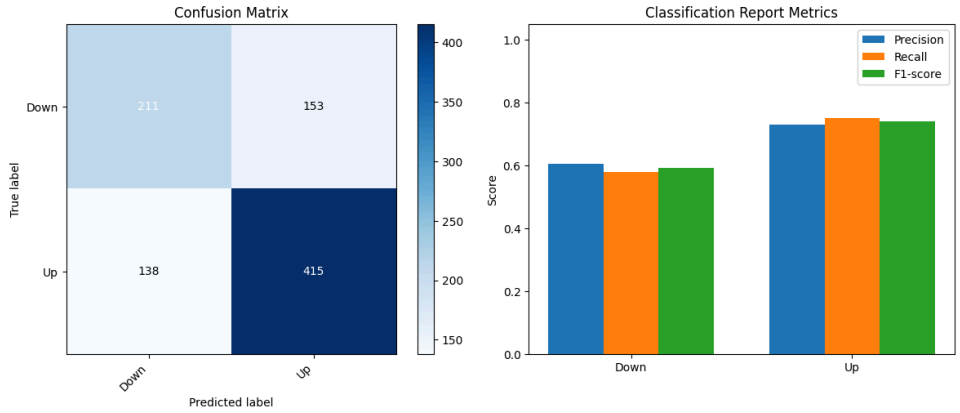
This stabilizes variance and helps the model to capture relative price changes over time. The processed data was then segmented using the *sliding window* approach, with each of the

inputs represented as thirty time steps by features. The dataset was then split with a ratio of 70:15:15 for training, validation, and testing sets, to ensure robustness for the model.

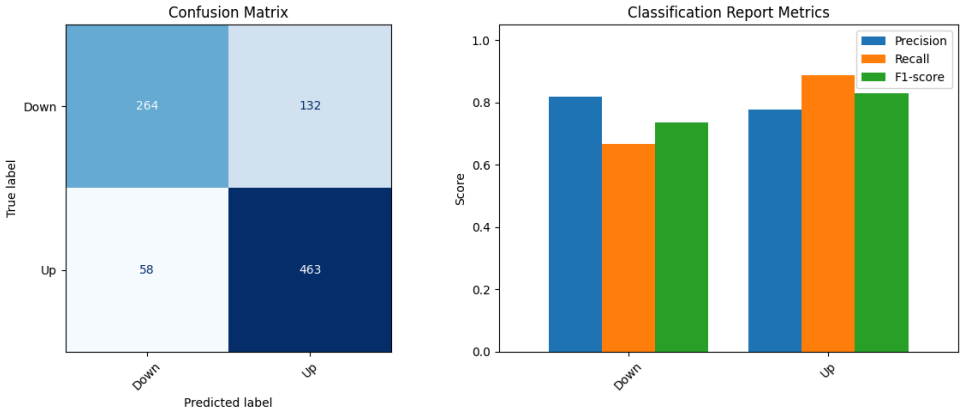
Results & Analysis

Results

The confusion matrix and the classification report metrics are shown below. The accuracy of the CLAM model is ~70% (Fig. 1), and the accuracy of the Random Forest model is ~80% (Fig. 2):

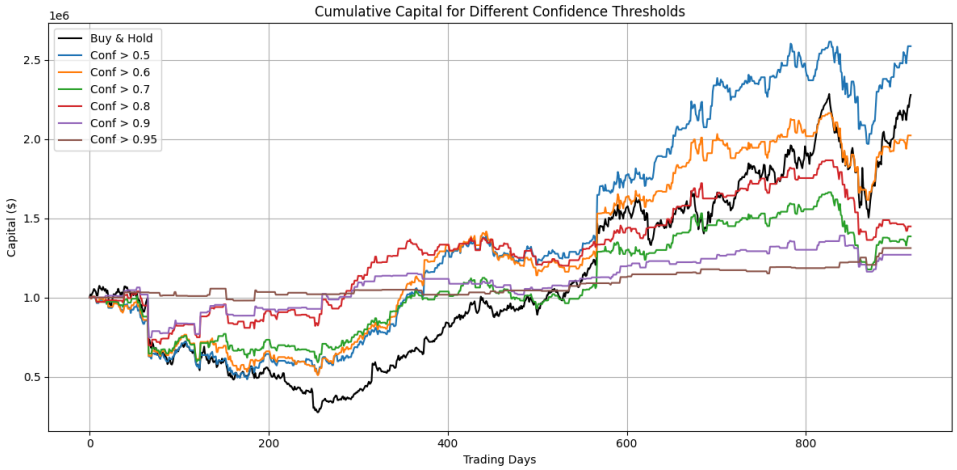


Confusion Matrix & Classification Report Metrics of CLAM (Fig. 1)

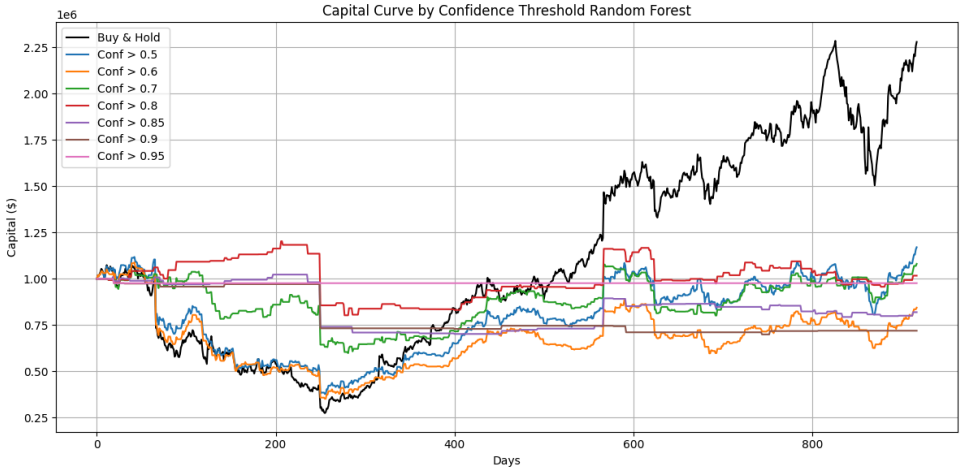


Confusion Matrix & Classification Report Metrics of RandomForest (Fig. 2)

To evaluate the model's practical effectiveness, we tested it in a simulated trading environment. The chart below shows the cumulative capital over time for both the CLAM and the Random Forest model under various confidence thresholds. Each threshold represents the minimum confidence level required for executing a trade. As shown, even with lower accuracy, the CLAM is able to gain more profit than the Random Forest model in our simulation:



Cumulative Capital Gain of CLAM with various Confidence Thresholds (Fig. 3)

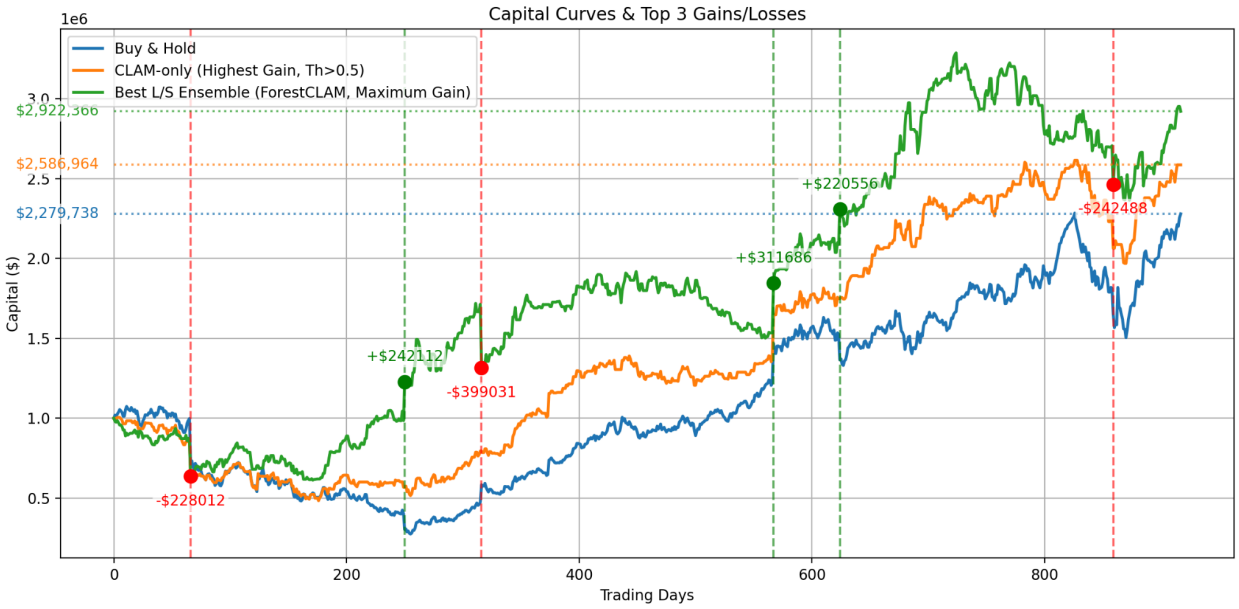


Cumulative Capital Gain of Random Forest with various Confidence Thresholds (Fig. 4)

From (Fig. 3) and (Fig. 4), we observe that lower confidence thresholds, such as 0.5, lead to larger overall returns, but with more price fluctuation, particularly during the early stages of the trading period. In contrast, higher thresholds such as 0.9 may yield lower but more stable returns, which may be preferred by cautious investors who are looking to make a consistent profit with fewer trades.

We can tune the model's behavior to align with individual risk tolerance and turnover preferences of a user, as these thresholds are directly correlated with the number of trades, success rate, and cumulative return, which gives the users the option to choose between high risk, high reward, or low risk, low reward. In general, 0.8 looks to be the best balance between the two, as it loses little compared to the other thresholds at the beginning, and can gain a comparative amount towards the end.

In order to push the gains of the final model, Forest-CLAM, while maintaining minimal loss, we weighted both the predictions of the Random Forest and CLAM together. Combined, the maximum profit (as shown below) saw a total increase of ~200% capital (Fig. 5):



Cumulative Capital Gain of the Hybrid Model Forest-CLAM (Fig. 5)

Analysis

Over ~900 days and with a minimum confidence score set to >0.5, our model was able to achieve a ~200% return on an initial investment of $1 million, ending with a total of ~$3 million. This illustrates our model's ability to detect upward trends in the stock market and convert that info into profitable trades.

In contrast to CLAM alone, the hybrid Forest-CLAM adjusted to the bearish market quickly, and started trading at a profit even before clear bullish patterns were present. Similarly, the random forest model was able to avoid the downward moving market, but when it came to detecting bullish trends, failed much like the CLAM during the bearish market.

An initial baseline can be considered in the traditional BHS, which underperforms our model by ~$0.7 million in the long term. It is interesting to note that our model struggles initially in the downward moving market, but around day 200, when the market starts to flatten, it takes off and doesn't look back.

Furthermore, the individual Forest and CLAM models can be taken as baselines. Our RandomForest model is blown away by both CLAM and Forest-CLAM. Interestingly, Forest-CLAM only outperforms CLAM by ~$400,000, which indicates that the potential to detect upward moving trends comes from CLAM. Despite being outperformed, the forest gives the combined Forest-CLAM an edge, we interpret this as the forest being a sort of "voice of reason" in the ensemble, since there are numerous instances (days ~250, ~620) where instead of losing money like CLAM, Forest-CLAM either avoids the dip or profits. Finally, it is interesting to note that all models struggle between 0-100 days to not lose capital, perhaps more interesting are the points where Forest-CLAM loses money while CLAM gains money (days ~300, ~700), additional work is needed to understand these phenomena.

Conclusion

Overall, we accomplished many of our goals, we achieved testing accuracies ~10% higher than expected, outperformed the BHS, and showed improvements on current open-source work.

We faced a fair amount of challenges, many of our initial attempts showed little promise, and we were stuck being worse than the BHS. Most models were able to detect either downward or upward movement, and failed at recognizing the other, resulting in modest gains to our initial capital. Even after leveraging both the Forest and CLAM to detect both bullish and bearish movements, we were still underperforming BHS. In the end, minor tweaks to model specific thresholds and increasing the patience parameter for CLAM training had major implications. All in all, resulting in present-day Forest-CLAM; capable of truly outperforming the market.

Future Work

Going forward, we believe more rigorous testing is necessary to determine the full capabilities of Forest-CLAM. It would be interesting to test on individual stocks, rather than a theoretical corporation as we have done. Furthermore, analyzing the model's trades in detail will allow us to get a better understanding and address additional shortcomings of Forest–CLAM.

Additionally, our research centered around tech focused stocks, to further assess generalization potential, it would be beneficial to test Forest-CLAM on a diversified portfolio of symbols. For Forest-CLAM's final evaluation, we plan to conduct a month-long run using live market data, starting with $10,000 in capital and trading exclusively on S&P and DOWJ tickers. As J. Robert Oppenheimer once noted, *"... theory will only carry you so far."*, the outcomes of these tests will push our model beyond theory, proving its viability in real-world market conditions.

Aligning with our goal of giving the average investor a leg up, we would like to design an easy-to-use public interface for our model, and give the community unrestricted access to our scripts and model weights. We hope to spur innovation in the open-source trading realm and encourage anyone who takes inspiration from our work to share improvements. As an ending note, the potential for other model architectures, such as DNN + attention, was not explored in this paper. We leave it to anyone curious and brave enough to address these shortcomings. The work for this project can be found here: https://github.com/C-Kianian/CS-539-Project-Stock-Prediction.

References

[1] Bao, W., Cao, Y., Yang, Y., Che, H., Huang, J., & Wen, S. (2025). Data-driven stock forecasting models based on neural networks: A review. Information Fusion, 113, 102616. https://doi.org/10.1016/j.inffus.2024.102616 (clarification: paper listed dozens of different models, such as RNN, CNN, GNN, Transformer, GAN, and LLM.)

[2] IMF Working Paper (2018)
Fagereng, A., Guiso, L., Malacrino, D., & Pistaferri, L. (2018). Heterogeneity and persistence in returns to wealth (IMF Working Paper No. 18/171). International Monetary Fund. https://www.imf.org/en/Publications/WP/Issues/2018/08/06/Heterogeneity-and-Persistence-in-Returns-to-Wealth-46134

[3] Equitable Growth / Norwegian Tax Study Summary
Boushey, H. (2021, May 12). Wealthier individuals receive higher returns to wealth. Washington Center for Equitable Growth. https://equitablegrowth.org/wealthier-individuals-receive-higher-returns-to-wealth/

[4] Radfar, E. Stock market trend prediction using deep neural network via chart analysis: a practical method or a myth?. Humanit Soc Sci Commun 12, 662 (2025). https://doi.org/10.1057/s41599-025-04761-8 (clarification: this is the model that is using the LSTM workflow with the RNN, quite accurate prediction.)

[5] Kearns, Michael, and Yuriy Nevmyvaka. *Machine Learning for Market Microstructure and High Frequency Trading*. [Unpublished manuscript]. Accessed July 12, 2025.

[6] sk0698. (2019). StockPrediction [Computer software]. GitHub. https://github.com/sk0698/StockPrediction

Contributions